Kernel Eigenspace-based MLLR Adaptation

Brian Mak and Roger Hsiao

Abstract— **Recently, we have been investigating the application of kernel methods for fast speaker adaptation by exploiting possible non-linearity in the input speaker space. In this paper, we propose another solution based on kernelizing the *eigenspace-based MLLR adaptation* (EMLLR) method. We call our new method "*kernel eigenspace-based MLLR adaptation*" (KEMLLR). In KEMLLR, speaker-dependent (SD) models are estimated from a common speaker-independent (SI) model using MLLR adaptation, and the SD MLLR transformation matrices are mapped to a kernel-induced high-dimensional feature space, and kernel principal component analysis is used to derive a set of eigenmatrices in the feature space. In addition, composite kernel is used to preserve the row information in the transformation matrices. A new speaker's MLLR transformation matrix is then represented as a linear combination of the leading kernel eigenmatrices, which, though exists only in the feature space, still allows the speaker's mean vectors to be found explicitly. As a result, at the end of KEMLLR adaptation, a regular HMM is obtained for the new speaker and subsequent speech recognition is as fast as normal HMM decoding. KEMLLR adaptation was tested and compared with other adaptation methods (MAP, MLLR, EV, EMLLR, and eKEV) on the Resource Management and Wall Street Journal tasks using 5s or 10s of adaptation speech. It is found that in both cases, KEMLLR adaptation gives the greatest improvement over the SI model with 11–20% word error rate reduction.**

Keywords— **Eigenvoice speaker adaptation, eigenspace-based MLLR adaptation, kernel PCA, composite kernels, kernel eigenvoice adaptation, embedded kernel eigenvoice adaptation, BFGS optimization.**

I. Introduction

When the amount of adaptation speech is really small, say, a few seconds, eigenspace-based adaptation methods [1], [2], [3], [4] have been shown more effective than the traditionally popular methods such as the Bayesian-based *maximum a posteriori* (MAP) adaptation [5] and the transformation-based *maximum likelihood linear regression* (MLLR) adaptation [6]. The eigenvoice (EV) adaptation method [1] was motivated by the eigenface approach in face recognition [7]. The idea is to derive from a diverse set of speakers a small set of basis vectors called *eigenvoices* using principal component analysis (PCA) (or other basis-deriving algorithms). These eigenvoices are believed to represent different voice characteristics (e.g. gender, age, accent, etc.), and any training/new speaker is then a point in the eigenspace. That is, a new speaker vector is represented as a linear combination of the eigenvoices, and the combination weights may be obtained by maximizing the likelihood of his/her adaptation data. In practice, a few to a few tens of eigenvoices are found adequate for fast speaker adaptation. Since the number of estimation parameters is greatly reduced, fast adaptation using EV is possible with a few seconds of speech [1].

In Kuhn's original eigenvoice adaptation [1], a speaker is represented by a supervector that is composed by splicing the mean vectors of his hidden Markov models (HMMs) together. Chen *et al.* [2] later suggested using a speaker's MLLR transforms instead of his HMM mean vectors to represent the speaker in their *eigenspace-based MLLR* (EMLLR) adaptation method; a speedup was also proposed in [8]. Since the dimension of the speaker MLLR transforms is usually much smaller than the total dimension of HMM mean vectors, EMLLR requires less memory and computation resources. EMLLR also naturally solves the problem of aligning mixture density components across different speakers in eigenvoice adaptation. Both EV and EMLLR approaches have been successfully applied to large vocabulary continuous speech recognition (LVCSR) [9], [10], [11]. A shortcoming of deriving the speaker eigenspace by PCA is that it only minimizes the residual in the least-square sense, and is inconsistent with the maximum likelihood (ML) criterion usually used to find a speaker's location in the eigenspace. Furthermore, orthogonality of the eigenspace is not necessarily required. In light of this, Nguyen later proposed finding the speaker eigenspace by the ML approach in his *maximum likelihood eigenspace* (MLES) method [12]. Interestingly, at the same time that eigenvoice was first proposed, Gales, in his *cluster adaptive training* (CAT) [13] — a separate and independent effort — already proposed the ML estimation of the speaker eigenspace within the speaker-adaptive training (SAT) [14] framework. Under the CAT scheme, both the eigenspace, cluster weights, and model parameters can be jointly optimized. Both Gales and Nguyen showed in LVCSR tasks that speaker eigenspace found by the ML approach outperformed speaker eigenspace found by PCA.

Recently, we have been investigating another way to improve the original eigenspace-based adaptation methods by exploiting possible non-linearity in their working space with the use of kernel methods [15], [16], [17]. In [18], we proposed the first kernel version of EV adaptation called the *kernel eigenvoice (KEV) speaker adaptation* method. The idea is to map input speaker supervectors to a kernel-induced high-dimensional feature space[1] via some nonlinear map φ, and then apply *kernel PCA* [19] there to extract

Brian Mak is with the Department of Computer Science, the Hong Kong University of Science and Technology (HKUST), Clear Water Bay, Hong Kong. E-mail: mak@cs.ust.hk.

Roger Hsiao finished this paper when he was a graduate student at the Department of Computer Science of HKUST, Hong Kong. Currently, he is a graduate student at the Language Technologies Institute, School of Computer Science of Carnegie Mellon University, Pittsburgh, Pennsylvania, USA. E-mail: wrhsiao@cs.cmu.edu.

[1]In the kernel methods terminology, the original space where raw data reside is called the *input space* and the space to which raw data are mapped is called the *feature space*. Readers are cautioned not to confuse this *feature space* with the acoustic feature space in speech. Sometimes, we will call the feature space in kernel methods as "*kernel-induced feature space*" when additional clarity is necessary.

the eigenvoices. During the actual computation, the exact nonlinear map need not be known. Instead, kernel PCA only requires the similarities between any two mapped inputs (in the form of their dot products) in the feature space, which is captured by a kernel function; this is known as the *kernel trick*[2] [16]. In principle, since the KEV adaptation method is a nonlinear generalization of the EV adaptation method, the former should be more powerful than the latter, and KEV adaptation is expected to give better performance. In fact, KEV adaptation is reduced to the traditional EV adaptation method if linear kernel is employed. In a TIDIGITS adaptation task, it was shown that KEV adaptation outperformed the SI model by about 30% using only 2.1, 4.1, or 9.6 seconds of adaptation speech [20], and was better than EV, MAP, and MLLR adaptation [21].

Although KEV adaptation performs well, the performance gain is obtained at the expense of many online kernel evaluations during both adaptation and recognition. One solution is the *embedded kernel eigenvoice (eKEV) speaker adaptation* method [22], [23]. eKEV adaptation eliminates all online kernel evaluations during recognition by finding an approximate *pre-image*[3] [24] of the new speaker[4] model obtained by KEV adaptation. Unlike KEV adaptation in which the new adapted speaker model resides only in the "*fictitious*" feature space, at the end of eKEV adaptation, a "real" HMM is obtained for the new speaker in the input space. However, eKEV adaptation has a limitation: the new speaker's supervector must lie on the span of a set of reference (training) speakers; it can be argued that the new speaker model is sub-optimal.

In this paper, we investigate another solution to the computation problem of KEV by applying kernel methods on eigenspace-based MLLR (EMLLR) adaptation instead. We will show that one may kernelize the EMLLR adaptation method in such a way that although the MLLR transformation matrix for the new speaker is not explicitly found, the mean vectors of his new model still can be directly computed using kernel methods — in other words, a real speaker HMM is obtained at the end of the adaptation. As a result, subsequent recognition with the new speaker is as fast as usual HMM decoding without any online kernel evaluations. We call our new method *kernel eigenspace-based MLLR adaptation* (KEMLLR). The basic idea of KEMLLR adaptation has already been reported and evaluated on the Resource Management (RM) task with 1000 words in [25], [26]. This paper further improves our previous work by generalizing the Gaussian kernel function to use Mahalanobis distance (instead of Euclidean distance) so as to normalize the MLLR transformation matrix components before kernel PCA is performed. It will be shown that such normalization gives better adaptation performance. In addition, the improved KEMLLR method was tested and compared with other adaptation methods on both the RM task as well as the Wall Street Journal (WSJ0) task with 5000 words.

The paper is organized as follows. We first review the eigenspace-based MLLR adaptation method in Section II. Our new kernel eigenspace-based MLLR speaker adaptation method is then detailed in Section III. This is followed by experimental evaluation in Section IV, and an analysis of eigenmatrix weights in Section V. Finally, the paper ends with concluding remarks and pointers for future directions in Section VI.

II. Eigenspace-based MLLR (EMLLR) Adaptation

Suppose there are some speech data from N speakers, and a speaker-independent (SI) model that is a hidden Markov model (HMM) with totally N_g Gaussians. These speakers should come from a diverse population so that there is a good coverage of different speaker characteristics such as speaking style, accent, age, gender, etc. Then N speaker-dependent (SD) models of the same HMM topology are estimated from the SI model by MLLR speaker adaptation. Let's further assume that all of the N_g Gaussians in the SI model are grouped into L regression classes, and let H be the mapping function that maps the gth Gaussian to its regression class $h = H(g)$, where $g = 1, \ldots, N_g$, and $h = 1, \ldots, L$. The estimation of each SD model consists in finding L MLLR transformation matrices for each speaker. That is, the gth Gaussian mean vector $\boldsymbol{\mu}_g^{(i)} \in \mathbb{R}^d$ of the ith speaker is given by

$$\boldsymbol{\mu}_g^{(i)} = \mathbf{Y}_{H(g)}^{(i)'} \boldsymbol{\xi}_g^{(si)} \tag{1}$$

where $\mathbf{Y}_{H(g)}^{(i)'} \in \mathbb{R}^{d \times (d+1)}$ is his MLLR transformation for the $H(g)$-th regression class, and $\boldsymbol{\xi}_g^{(si)} = [\boldsymbol{\mu}_g^{(si)'}, 1]'$ is the augmented mean vector of the corresponding Gaussian in the SI model[5]. Notice that the mean vector is augmented with an extra element of 1 to allow an affine MLLR transformation — a rotation followed by a translation.

In eigenspace-based MLLR (EMLLR) adaptation, a speaker is indirectly represented by a *speaker transformation supervector* (STSV) which is obtained by stacking up the L vectorized MLLR transformation matrices, $\{\mathbf{Y}_1^{(i)'}, \ldots, \mathbf{Y}_L^{(i)'}\}$, of the speaker. Let's denote the STSV of the ith speaker by $\mathbf{y}^{(i)} = [vec(\mathbf{Y}_1^{(i)})', \ldots, vec(\mathbf{Y}_L^{(i)})']'$. Principal component analysis (PCA) is performed using the correlation matrix of the N STSVs, $\{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(N)}\}$, to obtain the eigenvectors, $\mathbf{v}_m^{(emllr)}, m = 1, 2, \ldots, N$, which are the vectorized eigenmatrices[6]. To do that, each STSV

[2]Under the Mercer's condition, any positive semi-definite kernel can be represented as a dot product in a high-dimensional space.

[3]Finding an exact or a good approximate vector in the input space from its image in the feature space is known as the pre-image problem in kernel methods.

[4]In this paper, we will refer to the speaker to whom the system is adapting as the "new speaker."

[5]In this paper, vector quantities are written in bold letters, and matrices are in capital bold letters. Scalar quantities are not bold. The transpose of a vector or matrix is denoted by the superscript $'$. In addition, for any quantity $\mathbf{x}$, various accents will be used to indicate its mean, centered version, and centered and variance-normalized version respectively as follows: $\bar{\mathbf{x}}$, $\tilde{\mathbf{x}}$, and $\hat{\mathbf{x}}$.

[6]The formulation of EMLLR in [2] used the covariance matrix of the speaker transformation supervectors to perform PCA, but the original eigenvoice paper [1] suggests that PCA using the correlation matrix gives better results; our own experience agrees with the latter.

is mean-zeroed and then normalized by its variance. Let's further denote the centered and variance-normalized STSV of the ith speaker by $\hat{\mathbf{y}}^{(i)}$. Then, we have

$$\hat{\mathbf{y}}^{(i)} = \mathbf{C}_y^{-\frac{1}{2}}(\mathbf{y}^{(i)} - \bar{\mathbf{y}}) \tag{2}$$

where $\bar{\mathbf{y}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{y}^{(i)}$ is the mean STSV among the N speakers, and $\mathbf{C}_y$ is the diagonal covariance of the STSVs. For a new speaker, his centered and normalized STSV $\hat{\mathbf{y}}$ is approximated as a linear combination of the M leading vectorized eigenmatrices as

$$\hat{\mathbf{y}} \simeq \hat{\mathbf{y}}^{(emllr)} = \sum_{m=1}^{M} w_m \mathbf{v}_m^{(emllr)} \tag{3}$$

where $\mathbf{w} = [w_1, \ldots, w_M]'$ will be called the eigenmatrix weight vector. Finally, the speaker's STSV $\mathbf{y}$ is given by

$$\mathbf{y} \simeq \mathbf{y}^{(emllr)} = \bar{\mathbf{y}} + \mathbf{C}_y^{\frac{1}{2}}\hat{\mathbf{y}}^{(emllr)} \ . \tag{4}$$

To get each HMM Gaussian component of the new speaker model, let's first add the subscript hr to any quantity to represent the part of the quantity that is related to the rth row of the hth MLLR transformation matrix. Thus, the new speaker STSV can be written as $\mathbf{y} = [\ldots, \mathbf{y}'_{h1}, \ldots, \mathbf{y}'_{hd}, \ldots]'$, where $\mathbf{y}_{hr} \in \mathbb{R}^{(d+1)}$ for $r = 1, \ldots, d$ and $h = 1, \ldots, L$ is the rth row of his hth MLLR transformation matrix. Then, $\mathbf{y}_{hr} = \bar{\mathbf{y}}_{hr} + \mathbf{C}_{yhr}^{\frac{1}{2}}\hat{\mathbf{y}}_{hr}$, and according to Eqn. (3),

$$\hat{\mathbf{y}}_{hr} = \sum_{m=1}^{M} w_m \mathbf{v}_{mhr}^{(emllr)} \ . \tag{5}$$

Hence, the rth component μ_{gr} of the gth Gaussian mean $\boldsymbol{\mu}_g$ of the new speaker model (that belongs to the hth regression class as $h = H(g)$) can be found by combining Eqns.(1,4,5) as follows:

$$\begin{aligned}
\boldsymbol{\mu}_g &= \mathbf{Y}'_h \boldsymbol{\xi}_g^{(si)} \\
\Rightarrow \mu_{gr} &= \mathbf{y}'_{hr}\boldsymbol{\xi}_g^{(si)} \\
&= (\bar{\mathbf{y}}_{hr} + \mathbf{C}_{yhr}^{\frac{1}{2}}\hat{\mathbf{y}}_{hr})'\boldsymbol{\xi}_g^{(si)} \\
&= \bar{\mathbf{y}}'_{hr}\boldsymbol{\xi}_g^{(si)} + \sum_{m=1}^{M} w_m (\mathbf{v}_{mhr}^{(emllr)'}\mathbf{C}_{yhr}^{\frac{1}{2}}\boldsymbol{\xi}_g^{(si)}) \ .
\end{aligned} \tag{6}$$

Given the adaptation data $\boldsymbol{O} = \{\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T\}$ from the new speaker, his eigenmatrix weights can be estimated by maximizing the likelihood of $\boldsymbol{O}$, or, equivalently the following $Q(\mathbf{w})$ function (where irrelevant terms are dropped for simplicity):

$$Q(\mathbf{w}) = -\sum_{g=1}^{N_g}\sum_{t=1}^{T}\gamma_t(g)(\mathbf{o}_t - \boldsymbol{\mu}_g(\mathbf{w}))'\mathbf{C}_g^{-1}(\mathbf{o}_t - \boldsymbol{\mu}_g(\mathbf{w})) \tag{7}$$

where $\gamma_t(g)$ is the posterior probability of the observation sequence being at the gth Gaussian at time t, and $\mathbf{C}_g$ is the covariance matrix of the gth Gaussian. Differentiating $Q(\mathbf{w})$ w.r.t. each weight, $w_m, m = 1, \ldots, M$, we get

$$\frac{\partial Q(\mathbf{w})}{\partial w_m} = 2\sum_{g=1}^{N_g}\sum_{t=1}^{T}\gamma_t(g)(\mathbf{o}_t - \boldsymbol{\mu}_g(\mathbf{w}))'\mathbf{C}_g^{-1}\frac{\partial \boldsymbol{\mu}_g(\mathbf{w})}{\partial w_m} \ . \tag{8}$$

By setting the M derivatives to zero, the optimal weights are obtained by solving a system of M linear equations [1], [2].

III. Kernel Eigenspace-based MLLR (KEMLLR) Adaptation

In KEMLLR adaptation, we try to improve the performance of eigenspace-based MLLR (EMLLR) adaptation by exploiting possible non-linearity in the speaker transformation supervector space. This is achieved by kernelizing EMLLR in a way analogous to the use of kernel methods in *kernel eigenvoice* (KEV) adaptation [18] to improve the performance of eigenvoice adaptation [1]. That is, linear PCA used in EV or EMLLR is replaced by kernel PCA to derive a nonlinear eigenbasis. However, while KEV adaptation only results in an implicit speaker supervector in the kernel-induced feature space for the new speaker and thus suffers from slow recognition speed, explicit HMM Gaussian mean vectors can be obtained using KEMLLR adaptation.

To help readers, who are not familiar with kernel methods, understand the theoretical formulation of KEMLLR adaptation, its basic procedure is first summarized below:
Step 1: Map the speaker transformation supervectors (STSVs) to a high-dimensional feature space. Let's assume that the mapping function is φ.
Step 2: Find out the principal components (eigenmatrices in our case) in the kernel-induced feature space assuming that we know the kernel function $k(\cdot,\cdot)$. As in all kernel methods, the eigenmatrices are expressed in terms of the mapped training data.
Step 3: Express the new speaker's transformation supervector in the *feature space* in terms of the unknown eigenmatrix weights $\mathbf{w}$.
Step 4: Express the similarity between the new speaker's STSV and any Gaussian mean vector of the SI model in the feature space, again, in terms of $\mathbf{w}$.
Step 5: Design a kernel function so that the result of Step 4 may be used to compute the adapted mean vectors for the new speaker, and hence the Q function of Eqn. (7).
Step 6: Define an optimization criterion, and estimate the eigenmatrix weights $\mathbf{w}$. Maximum-likelihood approach is used in this paper.

Details of the formulation are elaborated below.

A. Kernel Eigenmatrices in the Feature Space

The basic idea of kernel methods is to map data in the input space to a high-dimensional feature space via some nonlinear map φ, and then apply a linear method there. It is now well-known that the computational procedure depends only on the inner products in the feature space, which can be obtained efficiently with a suitable kernel function [16].

Thus, the use of kernels provides elegant nonlinear generalizations of many existing linear algorithms. A well-known example in supervised learning is the support vector machines (SVMs) [15]. In unsupervised learning, the kernel idea has also led to methods such as kernel-based clustering algorithms [27], kernel principal component analysis [19], kernel independent component analysis [28], and kernel linear discriminant analysis [29].

Suppose there are a set of N speaker transformation supervectors (STSVs), $\{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(N)}\}$. To apply kernel PCA in KEMLLR adaptation, all STSVs are first centered and variance-normalized by Eqn. (2) as $\{\hat{\mathbf{y}}^{(1)}, \hat{\mathbf{y}}^{(2)}, \ldots, \hat{\mathbf{y}}^{(N)}\}$. Then a suitable kernel function $k(\cdot,\cdot)$ is chosen which, conceptually, is associated with a mapping φ that maps a speaker's transformation supervector $\hat{\mathbf{y}}^{(i)}, i = 1, \ldots, N$, in the input STSV space to $\varphi(\hat{\mathbf{y}}^{(i)})$ in the kernel-induced high-dimensional feature space. Let $\tilde{\mathbf{K}}$ be the $N \times N$ centered kernel matrix with $\tilde{\mathbf{K}}_{ij} \equiv \tilde{k}(\hat{\mathbf{y}}^{(i)}, \hat{\mathbf{y}}^{(j)}) = \tilde{\varphi}(\hat{\mathbf{y}}^{(i)})'\tilde{\varphi}(\hat{\mathbf{y}}^{(j)})$, where $\tilde{\varphi}(\hat{\mathbf{y}}) = \varphi(\hat{\mathbf{y}}) - \bar{\varphi}$ and $\bar{\varphi} = \frac{1}{N}\sum_{i=1}^{N}\varphi(\hat{\mathbf{y}}^{(i)})$. Notice that $\tilde{\mathbf{K}}$ is related to the non-centered kernel matrix $\mathbf{K}$ by $\tilde{\mathbf{K}} = \mathbf{HKH}$, where $\mathbf{H} = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}'$ is the centering matrix, $\mathbf{I}$ is the $N \times N$ identity matrix, and $\mathbf{1} = [1, \ldots, 1]'$ is an N-dimensional vector.

To perform kernel PCA, instead of directly working on the covariance matrix in the feature space, one may carry out eigendecomposition on the centered kernel matrix $\tilde{\mathbf{K}}$ as

$$\tilde{\mathbf{K}} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}',$$

where $\mathbf{U} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_N]$ with $\boldsymbol{\alpha}_i = [\alpha_{i1}, \ldots, \alpha_{iN}]'$ are the eigenvectors, and $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$ are the eigenvalues of the centered kernel matrix. The mth orthonormal eigenvector of the covariance matrix in the feature space is then given by ([19])

$$\mathbf{v}_m^{(kemllr)} = \sum_{i=1}^{N} \frac{\alpha_{mi}}{\sqrt{\lambda_m}} \tilde{\varphi}(\hat{\mathbf{y}}^{(i)}) \ . \tag{9}$$

B. Composite Kernel

Eqn. (6) shows that in order to compute the mean vectors of a new speaker, one will need to access each row of his transformation matrix. However, the row information, in general, is lost during the φ-mapping of the transformation vectors to the kernel-induced feature space. To preserve the row information, a composite kernel is used: a possibly different kernel function, k_{hr}, $h = 1, \ldots, L$ and $r = 1, \ldots, d$, and thus different mapping, φ_{hr}, is applied to each row vector of the transformation matrices, and a composite function is then used to combine the dL constituent inner products. This is analogous to the use of composite kernels to preserve the state information in kernel eigenvoice adaptation [20]. The following direct-sum composite kernel (which has been shown to give good performance in KEV adaptation) is adopted in this paper:

$$\begin{aligned} k(\hat{\mathbf{y}}^{(i)}, \hat{\mathbf{y}}^{(j)}) &= \varphi(\hat{\mathbf{y}}^{(i)})'\varphi(\hat{\mathbf{y}}^{(j)}) \\ &= \sum_{h=1}^{L}\sum_{r=1}^{d} \varphi_{hr}(\hat{\mathbf{y}}_{hr}^{(i)})'\varphi_{hr}(\hat{\mathbf{y}}_{hr}^{(j)}) \\ &= \sum_{h=1}^{L}\sum_{r=1}^{d} k_{hr}(\hat{\mathbf{y}}_{hr}^{(i)}, \hat{\mathbf{y}}_{hr}^{(j)}) \end{aligned} \tag{10}$$

where $\hat{\mathbf{y}}_{hr}^{(i)}$ is the (centered and normalized) rth row of the MLLR transformation matrix of the hth regression class.

C. New Speaker's Transformation Supervector in the Feature Space

Analogous to the formulation of a new speaker STSV in the original EMLLR method (Eqn. (3)), the centered STSV of the new speaker in the kernel-induced feature space[7] $\tilde{\varphi}^{(kemllr)}(\hat{\mathbf{y}})$ is assumed to be a linear combination of the leading M eigenvectors (or eigenmatrices in our case as given by Eqn. (9)):

$$\tilde{\varphi}^{(kemllr)}(\hat{\mathbf{y}}) = \sum_{m=1}^{M} w_m \mathbf{v}_m^{(kemllr)} = \sum_{m=1}^{M}\sum_{i=1}^{N} \frac{w_m \alpha_{mi}}{\sqrt{\lambda_m}} \tilde{\varphi}(\hat{\mathbf{y}}^{(i)}) \ . \tag{11}$$

With the use of the composite kernel of Eqn. (10), the $\tilde{\varphi}$-mapping of the rth row of the MLLR transform of the hth regression class for the new speaker's STSV $\hat{\mathbf{y}}_{hr}$ is given by

$$\tilde{\varphi}_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}) = \sum_{m=1}^{M}\sum_{i=1}^{N} \frac{w_m \alpha_{mi}}{\sqrt{\lambda_m}} \tilde{\varphi}_{hr}(\hat{\mathbf{y}}_{hr}^{(i)}) \ . \tag{12}$$

D. Similarity and Kernel Evaluation

The similarity between the new speaker's STSV and the gth Gaussian mean vector of the SI model in the feature space, $\varphi_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr})'\varphi_{hr}(\boldsymbol{\xi}_g^{(si)})$, can be computed using

[7]The notation of the speaker transformation supervector in the feature space requires some explanation. In kernel methods, the existence of an object in the feature space does not necessarily imply the existence of its pre-image in the input space. Here, we use $\tilde{\varphi}^{(kemllr)}(\hat{\mathbf{y}})$ to represent the image even if the pre-image $\hat{\mathbf{y}}$ may not exist due to the intuitiveness of the notation. Notice that our KEMLLR adaptation does not require the existence of the pre-image $\hat{\mathbf{y}}$ in the input speaker transformation supervector space.

Eqn. (12) as follows:

$$\begin{aligned}
& k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \boldsymbol{\xi}_g^{(si)}) \\
\equiv\ & \varphi_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr})' \varphi_{hr}(\boldsymbol{\xi}_g^{(si)}) \\
=\ & \left[\bar{\varphi}_{hr} + \left(\sum_{m=1}^{M}\sum_{i=1}^{N} \frac{w_m \alpha_{mi}}{\sqrt{\lambda_m}} \tilde{\varphi}_{hr}(\hat{\mathbf{y}}_{hr}^{(i)})\right)\right]' \varphi_{hr}(\boldsymbol{\xi}_g^{(si)}) \\
=\ & \left[\bar{\varphi}_{hr} + \left(\sum_{m=1}^{M}\sum_{i=1}^{N} \frac{w_m \alpha_{mi}}{\sqrt{\lambda_m}} (\varphi_{hr}(\hat{\mathbf{y}}_{hr}^{(i)}) - \bar{\varphi}_{hr})\right)\right]' \varphi_{hr}(\boldsymbol{\xi}_g^{(si)}) \\
=\ & \bar{\varphi}'_{hr}\varphi_{hr}(\boldsymbol{\xi}_g^{(si)}) + \\
& \left[\sum_{m=1}^{M}\sum_{i=1}^{N} \frac{w_m \alpha_{mi}}{\sqrt{\lambda_m}} \left(k_{hr}(\hat{\mathbf{y}}_{hr}^{(i)}, \boldsymbol{\xi}_g^{(si)}) - \bar{\varphi}'_{hr}\varphi_{hr}(\boldsymbol{\xi}_g^{(si)})\right)\right] \\
=\ & A_{hr}(g) + \sum_{m=1}^{M} \frac{w_m}{\sqrt{\lambda_m}} B_{hr}(m,g) \ , \qquad (13)
\end{aligned}$$

where

$$A_{hr}(g) = \bar{\varphi}'_{hr}\varphi_{hr}(\boldsymbol{\xi}_g^{(si)}) = \frac{1}{N}\sum_{i=1}^{N} k_{hr}(\hat{\mathbf{y}}_{hr}^{(i)}, \boldsymbol{\xi}_g^{(si)}) \ , \qquad (14)$$

$$B_{hr}(m,g) = \sum_{i=1}^{N} \alpha_{mi} \left(k_{hr}(\hat{\mathbf{y}}_{hr}^{(i)}, \boldsymbol{\xi}_g^{(si)}) - A_{hr}(g)\right) \ , \qquad (15)$$

and $\bar{\varphi}_{hr} = \frac{1}{N}\sum_{i=1}^{N} \varphi_{hr}(\hat{\mathbf{y}}_{hr}^{(i)})$. Notice that all the kernel values in Eqns. (14, 15) may be computed offline prior to adaptation.

Furthermore, the derivative of $k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \boldsymbol{\xi}_g^{(si)})$ w.r.t. each eigenmatrix weight w_m, $m = 1, \ldots, M$, is given by

$$\frac{\partial}{\partial w_m}\left(k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \boldsymbol{\xi}_g^{(si)})\right) = \frac{B_{hr}(m,g)}{\sqrt{\lambda_m}} \ . \qquad (16)$$

E. Adapted Gaussian Mean Vectors

The computation of the $Q(\mathbf{w})$ function of Eqn. (7) requires the knowledge of the Gaussian mean vectors, $\boldsymbol{\mu}_g, g = 1, \ldots, N_g$, of the new speaker in terms of the eigenmatrix weights $\mathbf{w}$. From Eqn. (6), we know that a Gaussian mean vector $\boldsymbol{\mu}_g$ can be computed component-wise from the normalized inner product $\hat{\mathbf{y}}'_{hr}\mathbf{C}_{yhr}^{\frac{1}{2}}\boldsymbol{\xi}_g^{(si)}$ if we may obtain the latter from the kernel values $k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \boldsymbol{\xi}_g^{(si)})$. This can be done if we have an invertible kernel function that is a function of the inner product of its inputs. That is, we need a kernel function $k_{hr}(\cdot,\cdot)$ of the following form: $k_{hr}(\mathbf{u},\mathbf{v}) = F(\mathbf{u}'\mathbf{A}\mathbf{v})$, where $\mathbf{A}$ is a normalizing matrix, and F is invertible. Then,

$$\hat{\mathbf{y}}'_{hr}\mathbf{C}_{yhr}^{\frac{1}{2}}\boldsymbol{\xi}_g^{(si)} = F^{-1}(k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \boldsymbol{\xi}_g^{(si)})) \qquad (17)$$

which, in turn, is a function of $\mathbf{w}$ as given by Eqn. (13).

Quite a few common kernels have such properties. For example, the polynomial kernel $k(\mathbf{u},\mathbf{v}) = (1+\mathbf{u}'\mathbf{v})^p$ where p is the polynomial order, or the sigmoid kernel $k(\mathbf{u},\mathbf{v}) = \tanh(a\mathbf{u}'\mathbf{v} - b)$ where $a, b \in \mathbb{R}$. Below, we explore the use of the isotropic Gaussian kernel for KEMLLR adaptation as it has been proven to be successful for other kernel-based adaptation methods [18], [23].

E.1 Isotropic Gaussian Kernels

Let's consider the following isotropic Gaussian kernel

$$k_{hr}(\mathbf{u},\mathbf{v}) = \exp(-\beta_{hr}\|\mathbf{u}-\mathbf{v}\|_{\mathbf{A}}^2) \ , \qquad (18)$$

where β_{hr} controls the width of a Gaussian kernel, and $\|\mathbf{u}-\mathbf{v}\|_{\mathbf{A}}^2 = (\mathbf{u}-\mathbf{v})'\mathbf{A}^{-1}(\mathbf{u}-\mathbf{v})$ is the Mahalanobis distance between $\mathbf{u}$ and $\mathbf{v}$ normalized by $\mathbf{A}$, which is any symmetric and positive-definite matrix. Thus, we have

$$\|\mathbf{u}-\mathbf{v}\|_{\mathbf{A}}^2 = -\frac{1}{\beta_{hr}} \log k_{hr}(\mathbf{u},\mathbf{v}) \ . \qquad (19)$$

Because of the following identity:

$$\begin{aligned}
& \|\mathbf{u}-\mathbf{v}\|_{\mathbf{A}}^2 = \|\mathbf{u}\|_{\mathbf{A}}^2 + \|\mathbf{v}\|_{\mathbf{A}}^2 - 2\mathbf{u}'\mathbf{A}^{-1}\mathbf{v} \\
\Rightarrow\ & 2\mathbf{u}'\mathbf{A}^{-1}\mathbf{v} = \|\mathbf{u}\|_{\mathbf{A}}^2 + \|\mathbf{v}\|_{\mathbf{A}}^2 - \|\mathbf{u}-\mathbf{v}\|_{\mathbf{A}}^2 \ , \qquad (20)
\end{aligned}$$

if we substitute $\mathbf{u} = \hat{\mathbf{y}}_{hr}$, $\mathbf{v} = \boldsymbol{\xi}_g^{(si)}$, and $\mathbf{A} = \mathbf{C}_{yhr}^{-\frac{1}{2}}$ into Eqn. (20), and make use of Eqns. (17, 19), we will have

$$\begin{aligned}
2\hat{\mathbf{y}}'_{hr}\mathbf{C}_{yhr}^{\frac{1}{2}}\boldsymbol{\xi}_g^{(si)} =\ & -\frac{1}{\beta_{hr}} \log\left(k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \mathbf{0})\right) \\
& + \|\boldsymbol{\xi}_g^{(si)}\|_{\mathbf{C}_{yhr}^{-\frac{1}{2}}}^2 \\
& + \frac{1}{\beta_{hr}} \log\left(k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \boldsymbol{\xi}_g^{(si)})\right) \qquad (21)
\end{aligned}$$

Hence, from Eqn. (6), each Gaussian component is given by

$$\begin{aligned}
\mu_{gr}^{(kemllr)} =\ & \bar{\mathbf{y}}'_{hr}\boldsymbol{\xi}_g^{(si)} + \hat{\mathbf{y}}'_{hr}\mathbf{C}_{yhr}^{\frac{1}{2}}\boldsymbol{\xi}_g^{(si)} \\
=\ & \bar{\mathbf{y}}'_{hr}\boldsymbol{\xi}_g^{(si)} + \frac{1}{2}\left[\|\boldsymbol{\xi}_g^{(si)}\|_{\mathbf{C}_{yhr}^{-\frac{1}{2}}}^2 + \right. \\
& \left. \frac{1}{\beta_{hr}} \log\left(\frac{k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \boldsymbol{\xi}_g^{(si)})}{k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \mathbf{0})}\right)\right] \ . \qquad (22)
\end{aligned}$$

F. ML Estimation of Eigenmatrix Weights

Substituting Eqns. (13, 14, 15) into Eqn. (22), differentiating the result w.r.t. each eigenmatrix weight $w_m, m = 1, \ldots, M$, and making use of the kernel function gradient of Eqn.(16), we get

$$\begin{aligned}
\frac{\partial \mu_{gr}^{(kemllr)}}{\partial w_m} =\ & \frac{1}{2\beta_{hr}\sqrt{\lambda_m}}\left[\frac{B_{hr}(m,g)}{k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \boldsymbol{\xi}_g^{(si)})} - \right. \\
& \left. \frac{B_{hr}(m,-1)}{k_{hr}^{(kemllr)}(\hat{\mathbf{y}}_{hr}, \mathbf{0})}\right] \qquad (23)
\end{aligned}$$

where we use the index $g = -1$ to represent a special augmented vector $\boldsymbol{\xi}_{-1}^{(si)}$ which is the zero vector $\mathbf{0}$. Using Eqn. (23), the derivatives of $Q(\mathbf{w})$ of Eqn. (8) w.r.t. each eigenmatrix weights w_m, can be easily obtained.

Due to the non-linearity of the kernel functions, there is no closed form solution for the optimal $\mathbf{w}$. Instead, generalized EM algorithm [30] is used in which, gradient-based numerical methods, such as the gradient ascent method,

is employed to improve the Q function during each maximization step. In order to improve convergence of the estimation, quasi-Newton optimization algorithm is employed in this paper. The quasi-Newton method is similar to the traditional Newton's method and makes use of the Hessian to retrieve the Newton's direction. However, it approximates the Hessian with an estimate that can be derived solely from the gradient. As a result, it is more efficient and it can enforce the Hessian estimate to be strictly positive-definite.

In the quasi-Newton method, the inverse of the Hessian matrix $\mathbf{A}^{-1}$ is approximated by $\mathbf{H}_i$ in an iterative procedure so that $\lim_{i\to\infty} \mathbf{H}_i = \mathbf{A}^{-1}$, where $\mathbf{H}_i$ is the Hessian inverse in the ith iteration, and it has to be positive definite and symmetric. We update $\mathbf{H}_i$ by the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm as follows:

$$\mathbf{H}_{i+1} = (\mathbf{I} - \frac{\mathbf{a}_i \mathbf{b}_i^{'}}{\mathbf{b}_i^{'} \mathbf{a}_i}) \mathbf{H}_i (\mathbf{I} - \frac{\mathbf{b}_i \mathbf{a}_i^{'}}{\mathbf{b}_i^{'} \mathbf{a}_i}) + \frac{\mathbf{a}_i \mathbf{a}_i^{'}}{\mathbf{b}_i^{'} \mathbf{a}_i} \ ,$$

where

$$\begin{aligned} \mathbf{a}_i &= \mathbf{w}_{i+1} - \mathbf{w}_i \ , \\ \text{and} \quad \mathbf{b}_i &= \bigtriangledown Q(\mathbf{w}_{i+1}) - \bigtriangledown Q(\mathbf{w}_i) \end{aligned}$$

Detailed description and proof of these formulas can be found in [31].

Finally, the optimal eigenmatrix weights can be optimized iteratively by the following updating formula:

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \eta_i \mathbf{H}_i \bigtriangledown Q(\mathbf{w})|_{\mathbf{w}_i} \ ,$$

where i is the iteration index, η_i is a learning rate to be determined by a line search algorithm at the ith iteration, and the gradient can be computed from Eqns. (8, 23).

G. Robust KEMLLR

When the amount of adaptation data is really small, the MLLR transformation found by KEMLLR adaptation may not be reliable. Here, we linearly interpolate the transformations found by KEMLLR with the identity matrix to get a more robust estimate of the transformations[8]. Equivalently, a mean vector found by KEMLLR $\mu_{gr}^{(kemllr)}$ is interpolated with the corresponding SI mean vector $\mu_{gr}^{(si)}$ to get its robust estimate $\mu_{gr}^{(rkemllr)}$ in our *robust KEMLLR* adaptation as follows:

$$\mu_{gr}^{(rkemllr)} = w_0 \mu_{gr}^{(si)} + (1 - w_0) \mu_{gr}^{(kemllr)}, \ 0.0 \le w_0 \le 1.0 \ . \quad (24)$$

The gradient of the Gaussian means is modified accordingly as below:

$$\frac{\partial \mu_{gr}^{(rkemllr)}}{\partial w_0} = \mu_{gr}^{(si)} - \mu_{gr}^{(kemllr)} \quad (25)$$

and

$$\frac{\partial \mu_{gr}^{(rkemllr)}}{\partial w_m} = (1 - w_0) \frac{\partial \mu_{gr}^{(kemllr)}}{\partial w_m} \ , \quad m = 1, \ldots, M. \quad (26)$$

And w_0 can be optimized jointly with the other M eigenmatrix weights $w_m, m = 1, \ldots, M$.

IV. Experimental Evaluation

The proposed kernel eigenspace-based MLLR (KEMLLR) speaker adaptation method was first evaluated using context-independent acoustic models on the DARPA Resource Management (RM) continuous speech recognition task [33], which has a vocabulary of 1,000 words. The simpler database allows us to conduct many experiments to investigate various aspects of our new adaptation method. Then, KEMLLR adaptation was tested again using context-dependent acoustic models on the Wall Street Journal (WSJ) task [34] with a larger vocabulary of 5,000 words. In each task, some or all of the following models or adaptation methods were compared:

SI*:* the baseline speaker-independent model.
GD*:* the gender-dependent model.
MAP*:* the speaker-adapted (SA) model found by MAP adaptation [5].
MLLR*:* the SA model found by MLLR adaptation [6].
(robust) EV*:* the SA model found by eigenvoice adaptation [1].
(robust) eKEV*:* the SA model found by embedded kernel eigenvoice adaptation [35].
(robust) EMLLR*:* the SA model found by eigenspace-based MLLR adaptation [2].
(robust) KEMLLR*:* the SA model found by kernel EMLLR adaptation.

For each adaptation method, we tried to find the best setup for the method so as to obtain its best results for comparison. Both MAP and MLLR adaptation were done using the HTK software; thus, only their basic algorithms were employed[9]. For MAP adaptation, scaling factors in the range of 3–30 were tried. For MLLR adaptation, it was performed with a regression tree of 32 classes; the minimum occupation count threshold for a regression class was also adjusted. It was found that only 1 regression class was actually used in 5s adaptation, and 1 or 2 regression classes were used in 10s adaptation. In addition, the better results of MLLR adaptation using diagonal and full transformation matrix are reported. For EV, eKEV, and EMLLR adaptation, the SD models for the training speakers were created by MLLR adaptation using the same 32-class regression tree. SA models were created by interpolation with the SI model in the same way as robust KEMLLR adaptation. For EV, simple linear PCA was used to derive the eigenspace; for eKEV and KEMLLR, kernel PCA was used instead. Furthermore, EMLLR adaptation was implemented as a special case of KEMLLR adaptation using linear kernel.

[8]The technique is commonly used to smooth a detailed model that is less well trained with a well-trained general model, and it has been proven to be successful in KEV and eKEV adaptation.

[9]One may perhaps get better performance with their variants such as structural MAP [36] and MAPLR [37].

Lastly, all adaptations were carried out in the supervised mode in both tasks: the contents of all adaptation utterances were assumed to be known, and the SI model was used to compute the initial Gaussian mixture posteriors. Several adaptation iterations were attempted for each method, but, except for EV adaptation, one or two iterations were usually found enough. In particular, it was found that KEMLLR adaptation converged in one adaptation iteration[10]. When several adaptation iterations were run, the posteriors were updated with the new adapted model found at each iteration.

Parameter initialization and settings

System parameters of the various adaptation methods (other than MLLR and MAP adaptation) were tuned and set using the RM corpus; they were then simply adopted by the WSJ evaluation without modification. Their values were listed below:

- Parameters for KEMLLR adaptation:
  - initial learning rate = 0.00001.
  - $\beta_r = \beta = 0.001$ for $r = 1, \ldots, R$. That is, all constituent Gaussian kernels have the same global β value.
  - The eigenmatrix weights $w_m, m = 1, \ldots, M$, were initialized by projecting the following transformation,

$$\mathbf{Y}^{(si)} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{27}$$

  onto each of the M kernel eigenmatrices after it was normalized and φ-mapped to the kernel-induced feature space.
- Parameters for eKEV adaptation:
  - initial learning rate = 0.1.
  - $\beta_r = \beta = 0.005$ for $r = 1, \ldots, R$.
  - number of maximum-likelihood reference speakers = 5.
  - number of kernel eigenvoices = 7.
- Parameters for the quasi-Newton BFGS optimization algorithm:
  - It stopped when either the relative improvement on the likelihood of the adaptation data was smaller than 0.00015, or 30 iterations was reached.
  - Learning rates were changed dynamically during the heuristic line search as mentioned in Section III-F.

Furthermore, for all robust adaptation methods that interpolate the adapted model with the SI model, the interpolation weight w_0 was initialized to 0.5. Rigorously speaking, one has to enforce the constraint that $0.0 \leq w_0 \leq 1.0$, but that will turn those nonlinear adaptation algorithms, namely, eKEV and KEMLLR, into nonlinear programming problems. In our current implementation of eKEV and KEMLLR, we did not do that. In our experience, if we started with $w_0 = 0.5$, we never got into the trouble that w_0 gets out of the required range. In case that this becomes an issue, a simple (though sub-optimal) solution is to optimize w_0 and the other weights separately as follows: first find the non-robust solution, then interpolate it with the SI solution as described in Eqn.(24). The ensuing Q function is quadratic in w_0 and can be easily solved.

[10]Note that within one KEMLLR iteration, there can be many BFGS iterations, during which the posteriors are not modified.

A. Evaluation on Medium-vocabulary Continuous Speech Recognition

In this part, we would use simple context-independent models to investigate the behavior of KEMLLR adaptation on the simpler RM1 corpus. In addition, the acoustic vector consists of the static cepstra only. The simpler task allows us to run many experiments to investigate the behaviour of KEMLLR adaptation which was new to us.

A.1 RM Corpus

The Resource Management corpus RM1 consists of clean read speech that represents queries about the naval resources. The utterances were recorded using a headset microphone at 16kHz with 16-bit resolution. The corpus comprises a speaker-independent (SI) section and a speaker-dependent (SD) section. The SI section consists of 3990 training utterances from 109 speakers. On the other hand, there are 12 speakers in the SD section, each having 600 utterances for training, 100 utterances for development, and 100 utterances for evaluation. The corpus has a vocabulary size of 1000 words.

A.2 Feature Extraction and Acoustic Modeling

All training and testing data were processed to extract 12 static mel-frequency cepstral coefficients (MFCCs) and the normalized frame energy from each speech frame of 25 ms at every 10 ms. Thus, the dimension of acoustic vectors in RM1 is $d = 13$. Forty-seven context-independent and speaker-independent (SI) phoneme models were trained using only the acoustic observations from the SI training set. Each phoneme model is a strictly left-to-right, 3-state hidden Markov model (HMM) with a mixture of 10 Gaussian components per state. All Gaussians have diagonal covariances. In addition, there are a 3-state "sil" model to capture silence and a 1-state "sp" model to capture short pauses.

A.3 Experimental Procedure

From the SI model, an SD model was constructed for each of the 109 speakers in the SI training set using MLLR adaptation with one or two regression classes that were determined a priori. As a result, we obtained a set of $N = 109$ speaker transformation supervectors (STSVs) for deriving the kernel eigenmatrices. Experiments were performed with about 5s or 10s adaptation data (or, about 4.6s and 9.2s if we exclude the leading and ending silence.) Only a single GEM iteration was run. To improve the statistical reliability of the results, for each test speaker, 3 sets of adaptation data were randomly chosen from his 100 development utterances. All reported results are the averages of experiments over the 3 adaptation sets of the 12 test speakers. The adapted models were tested on their

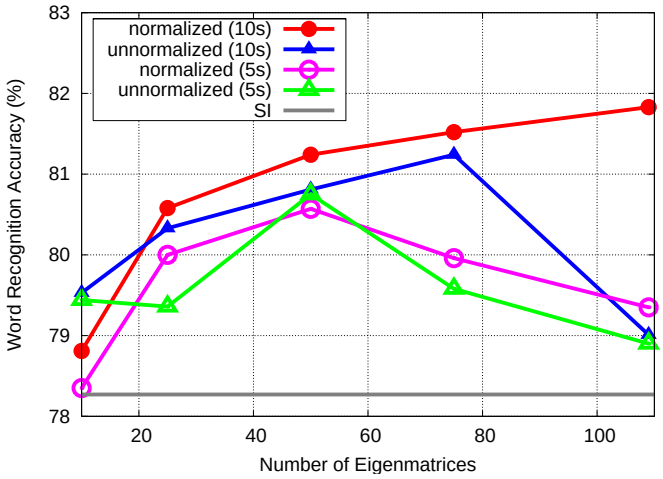

Fig. 1. Effect of STSV normalization on KEMLLR adaptation on RM1 using a single regression class.

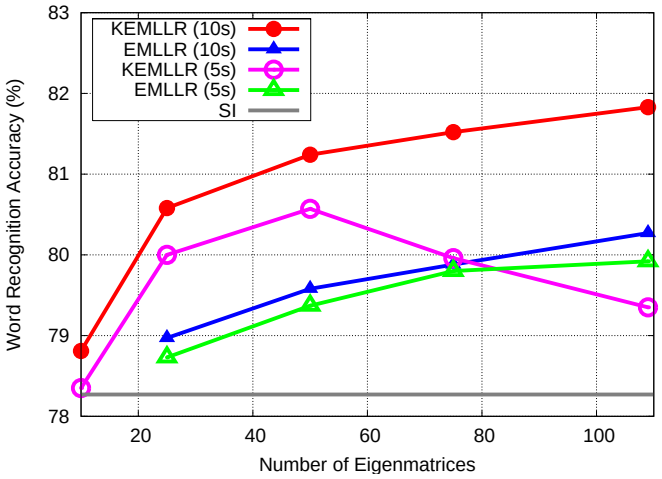

Fig. 2. Effect of kernelization on EMLLR adaptation on RM1 using a single regression class.

100 evaluation utterances using word-pair grammar which has a perplexity of 60.

A.4 Experiment 1: Effect of Normalization on Speaker Transformation Supervectors

We first investigated if normalizing the MLLR transformation matrices in the input space was helpful for KEMLLR adaptation. We implemented a version of KEMLLR which used the speaker transformation supervectors (STSVs) $\{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(N)}\}$ directly instead of the centered and variance-normalized STSVs, $\{\hat{\mathbf{y}}^{(1)}, \hat{\mathbf{y}}^{(2)}, \ldots, \hat{\mathbf{y}}^{(N)}\}$. Both schemes were tested on RM1 using SD models created by global MLLR adaptation, and the results are shown in Fig. 1.

From Fig. 1, it is observed that, when the same number of eigenmatrices are used, KEMLLR adaptation using normalized speaker transformation supervectors (STSVs) generally gives better performance. The same phenomenon is observed for EMLLR adaptation as well, confirming that it is better to use the correlation matrix to derive the eigenspace as suggested by Kuhn [1]. Furthermore, the adaptation performance generally improves with more eigenmatrices until there are insufficient amount of adaptation data to estimate the eigenmatrix weights reliably. Thus, there is an optimal number of eigenmatrices to use for a given amount of adaptation speech. In this particular experiment, the optimal number of eigenmatrices is found to be 50 and 109 for 5s and 10s of adaptation speech respectively. In practice, one may determine the optimal number of eigenmatrices to use by cross-validation. Also notice that even with 25 eigenmatrices, KEMLLR adaptation reduces the word error rate (WER) of the SI model by about 8 to 11% with only 5s or 10s of adaptation data.

Hereafter, all experiments were run with normalized STSVs.

A.5 Experiment 2: EMLLR vs. KEMLLR Adaptation

The performance of EMLLR and KEMLLR adaptation using various numbers of eigenmatrices is compared

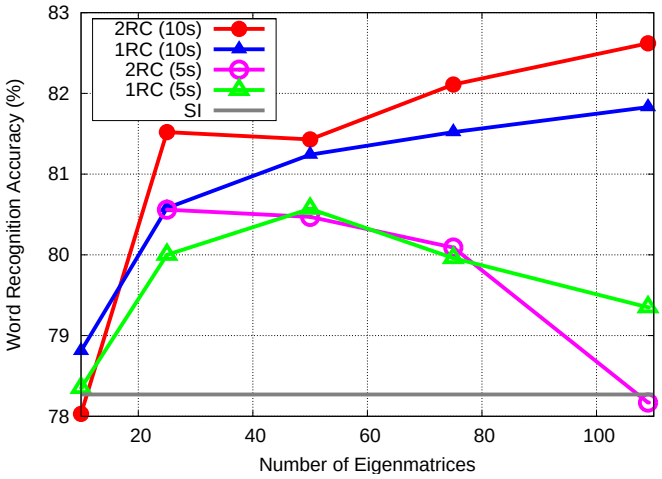

Fig. 3. Performance of KEMLLR adaptation on RM1 using one or two regression classes.

in Fig. 2. In both adaptation methods, the speaker-dependent models were created by global MLLR transformation. From Fig. 2, we observe that KEMLLR generally outperforms EMLLR adaptation when the same number of eigenmatrices are employed using the same amount of adaptation speech. For example, when there were sufficient adaptation data, KEMLLR outperforms EMLLR by ∼1.5% for the 10s adaptation, and ∼1.0% for the 5s adaptation. Or, equivalently speaking, KEMLLR may achieve the same adaptation performance of EMLLR by using fewer eigenmatrices. This shows that the KEMLLR can derive more effective leading eigenmatrices using kernel PCA to capture useful speaker characteristics than EMLLR. Moreover, the adaptation performance of EMLLR saturates quickly and there is little difference between its adaptation performance using 5s or 10s adaptation speech. On the contrary, although KEMLLR already performs well with only 5s of adaptation speech, there is substantial gain when another 5s of adaptation speech is available.

A.6 Experiment 3: Effect of Multiple Regression Classes

EMLLR and KEMLLR require MLLR adaptation to create the speaker-dependent (SD) models. Depending on the amount of speaker-specific data, one may generally improve the SD models by using multiple MLLR transforms, one for each regression class. On the other hand, in [26], we found that when multiple regression classes are used, better adaptation performance may be obtained when each regression class has its own separate eigenmatrices and weights. Hence, the use of multiple regression classes will increase the number of eigenmatrix weights to estimate, and more adaptation speech will be needed. Fig. 3 shows the adaptation performance of KEMLLR when the SD models are created using 1 or 2 regression classes determined a priori. We may see that when there are enough adaptation speech to estimate the eigenmatrix weights, more regression classes will improve the adaptation performance as evidenced in the 10s case. Or, in other words, when there are enough adaptation data, fewer eigenmatrices are required with multiple regression classes. For instance, in the 5s case, KEMLLR adaptation with one regression class reaches its best performance using 50 kernel eigenmatrices, whereas KEMLLR adaptation with two regression classes requires only 25 kernel eigenmatrices to do so with very comparable results (80.5%).

TABLE I
PERFORMANCE COMPARISON BETWEEN KEMLLR ADAPTATION AND OTHER ADAPTATION METHODS ON RM1. (FIGURES IN PARENTHESES ARE THE WER REDUCTIONS IN %.)

Model/Adaptation	Word Accuracy (%)	
	5s	10s
SI	78.27	78.27
MLLR	78.43 (0.74)	82.10 (17.6)
EMLLR	79.92 (7.59)	80.51 (10.3)
eKEV	80.58 (10.6)	80.70 (11.2)
KEMLLR	80.57 (10.6)	82.62 (20.0)

A.7 Experiment 4: Comparison with Other Adaptation Methods

In this experiment, the performance of KEMLLR adaptation is compared with that of the SI model, eKEV adaptation, MLLR adaptation, and EMLLR adaptation at their best settings. The results are listed in Table I in the order of ascending performance. We have the following observations:

- MLLR adaptation barely improves over the SI model.
- Both EMLLR adaptation and eKEV adaptation saturate quickly with 5s of adaptation data.
- The best results of EMLLR and KEMLLR both used one regression class for 5s adaptation, and two regression classes for 10s adaptation.
- All the eigenspace-based adaptation methods — namely, EMLLR, eKEV, and KEMLLR — perform well even with only 5s of adaptation speech.
- The two kernel-method-based adaptation methods perform equally well and the best when there are only 5s of adaptation data. However, KEMLLR adaptation seems to be more scalable and its performance improves with more adaptation data. In our experiments, it doubles the WER reduction when the amount of adaptation speech doubles from 5s to 10s.
- In summary, the order of adaptation performance of the various methods using 5s adaptation speech is: SI $\simeq$ MLLR $<$ EMLLR $<$ eKEV $<$ KEMLLR.

B. Evaluation on Large-vocabulary Continuous Speech Recognition (LVCSR)

In the previous medium-vocabulary evaluation, only simple context-independent models were tried. The simple task allows us to get familiarized more easily with the new KEMLLR adaptation method. In this section, we would like to check if KEMLLR adaptation is also effective on a relatively large-vocabulary recognition task using triphone HMMs with the common 39-dimensional MFCC acoustic vectors.

B.1 WSJ0 Corpus

The Wall Street Journal corpus WSJ0 [34] with 5K vocabulary was chosen. The standard SI-84 training set was used for training the speaker-independent (SI) model. It consists of 83 speakers and 7138 utterances for a total of about 14 hours of training speech (after discarding the problematic data from one speaker as in the Aurora4 corpus [38]). The standard nov'92 5K non-verbalized test set was used for evaluation. It consists of 8 speakers, each with about 40 utterances.

B.2 Feature Extraction and Acoustic Modeling

The traditional 39-dimensional MFCC vectors were extracted at every 10ms over a window of 25ms from the training and testing data. The speaker-independent (SI) model consists of 15,449 cross-word triphones based on 39 base phonemes. Each triphone was modeled as a continuous density HMM which is strictly left-to-right and has three states with a Gaussian mixture density of 16 components per state. State tying was performed to give 3131 tied states in the final SI model. In addition, the same type of "sil" and "sp" models were trained as in the last RM evaluation.

Gender-dependent (GD) models were also trained by performing MAP adaptation on the SI models with the gender-specific data from the training speakers.

B.3 Experimental Procedure

For each of the 8 testing speakers, 1–3 utterances of his speech were randomly selected so that the amount of adaptation speech is about 5s or 10s (or, about 4s and 8s respectively if one excludes the silence portions). His adapted model was tested on his remaining speech in the test set using a bigram language model of perplexity 147. The adaptation procedure was repeated three times and the three adaptation results are averaged before they are reported.

Learning from the results of the RM evaluation, all EMLLR and KEMLLR adaptation experiments on WSJ0 employed the normalized STSVs, and a global regression class for 5s adaptation and two regression classes for 10s adaptation. In addition, both EMLLR and KEMLLR adaptations used all 83 eigenmatrices as they gave the best performance.

TABLE II
PERFORMANCE COMPARISON BETWEEN KEMLLR ADAPTATION AND OTHER ADAPTATION METHODS ON WSJ0. (FIGURES IN PARENTHESES ARE THE WER REDUCTIONS IN %.)

Model/Adaptation Method	5s	10s
SI	92.19	92.19
GD	92.60 (5.25)	92.60 (5.25)
MLLR	92.32 (1.66)	92.98 (10.1)
EV	92.46 (3.46)	92.51 (4.10)
MAP	92.48 (3.71)	92.47 (3.59)
EMLLR	92.73 (6.91)	92.81 (7.94)
eKEV	92.86 (8.58)	92.92 (9.35)
KEMLLR	93.18 (12.7)	93.41 (15.6)

B.4 Experiment 5: Comparison with Other Adaptation Methods

KEMLLR adaptation was compared with traditional adaptation methods (MAP and MLLR) and eigenspace-based methods (EV, EMLLR, and eKEV) as well as the GD models on the WSJ0 corpus. Again efforts were made to find the best setup for each method as mentioned in the beginning of Section IV, and the system parameters for the various methods were simply adopted from those tuned in the RM evaluation without modification. The evaluation of GD models assumes perfect gender detection. Thus, utterances from a test speaker was decoded with a GD model of the speaker's gender.

Table II summarizes the performance of the various adaptation methods. Compared with the results in the RM evaluation, we have the following observations:

- The performance gains (in terms of WER reduction) of various methods on WSJ0 5s-adaptation are similar to those on RM 5s-adaptation, but were smaller in the 10s adaptations. For example, MLLR may obtain 17.6% WER reduction on the RM1 10s-adaptation, but only 10.1% on the WSJ0 10s-adaptation; the gain is cut almost by half. The corresponding figures for KEMLLR are 20.0% and 15.6%; the gain reduction for KEMLLR adaptation is smaller.
- MAP and EV have similar performance and give the least improvement.
- Between the two basic eigenspace-based methods, EMLLR is more effective than EV.
- Both EV and EMLLR are outperformed by their kernelized counterparts, namely, eKEV and KEMLLR respectively.
- MLLR again does not work well with only 5s of adaptation data even when a single global transformation is used. However, it is still proved to be effective when sufficient adaptation data, as little as 10s of speech, are available.
- The adaptation performance of MLLR (10s adaptation), EMLLR, eKEV and KEMLLR is better than the GD model. This shows that these adaptation methods should be doing more than simple gender adaptation.
- All in all, both kernelized eigenspace-based adaptation methods, eKEV and KEMLLR, perform very well in both 5s and 10s adaptations. In fact, KEMLLR performs the best among all the tested methods. On the other hand, eKEV adaptation requires only 7 kernel eigenvoices to obtain the good performance whereas it requires KEMLLR to use all (83) kernel eigenmatrices to beat the other adaptation methods. Since both kernel-based adaptation methods find the eigenvoice/eigenmatrix weights by the same gradient-based BFGS algorithm, and their computation mainly consists of evaluating the gradients of each weight, their algorithmic complexities are roughly proportional to the number of weights they use[11]. Consequently, eKEV runs faster than KEMLLR.

V. Analysis of Eigenmatrix Weights

In this section, we would like to compare EMLLR and KEMLLR adaptation methods by examining the eigenspaces that they derive. Since there are too few test speakers in both RM1 and WSJ0 corpora, we instead turn to the TIDIGITS corpus [39] for the analysis. The corpus is consisted of 163 speakers in both of its training and testing sets. The whole training set is used to derive the eigenspace spanned by the leading 40 or 163 eigenmatrices found by EMLLR or KEMLLR. Then EMLLR or KEMLLR adaptation is performed on the 56 men and 57 women in the testing set to find their coordinates in the eigenspace. To visualize their locations in the high-dimensional eigenspace, they are projected onto a 2-dimensional plot by the technique of *classical (metric) multi-dimensional scaling* provided by the Matlab software (which is also known as *principal coordinates analysis*) as shown in Fig. 4–7. From the plots, it is clear that the kernel eigenspace derived by KEMLLR is more effective than EMLLR in separating the men speakers from the women speakers, even using as little as 40 kernel eigenmatrices. Although our task at hand is adaptation and not gender detection, the more discriminative eigenspace derived by KEMLLR should indicate that more informative eigenvectors are extracted by kernel PCA used in KEMLLR, which lead to better performance of the adaptation method.

VI. Conclusions

In this paper, we proposed another application of kernel methods to improve the performance of eigenspace-based

[11]It is hard to make a vigorous comparison between the speed of eKEV and KEMLLR algorithms as their actual computational requirements depends on the amount of computations involved in each BFGS iteration and how many BFGS iterations are needed to reach convergence.

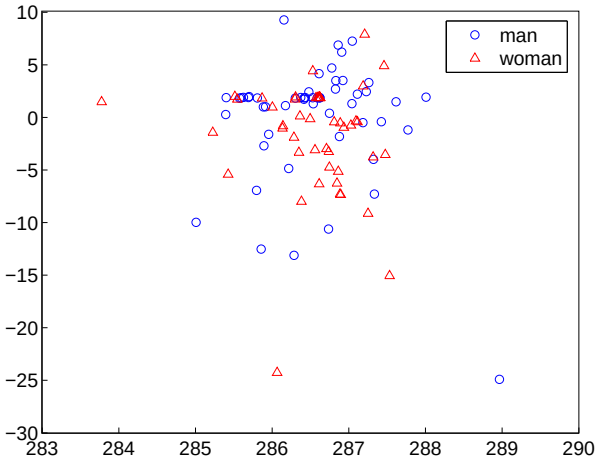

Fig. 4. Distribution of the adult TIDIGITS test speakers on the subspace spanned by the top 40 eigenmatrices found by EMLLR adaptation.

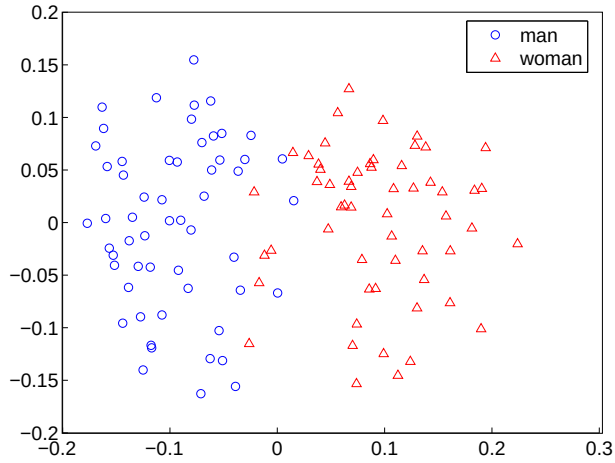

Fig. 6. Distribution of the adult TIDIGITS test speakers on the subspace spanned by the top 40 kernel eigenmatrices found by KEMLLR adaptation.

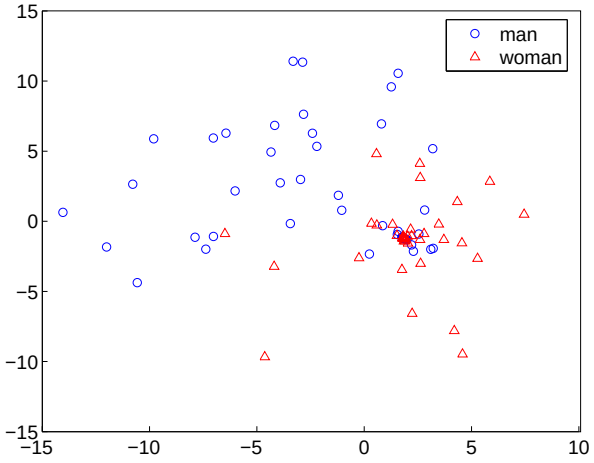

Fig. 5. Distribution of the adult TIDIGITS test speakers on the subspace spanned by the top 163 eigenmatrices found by EMLLR adaptation.

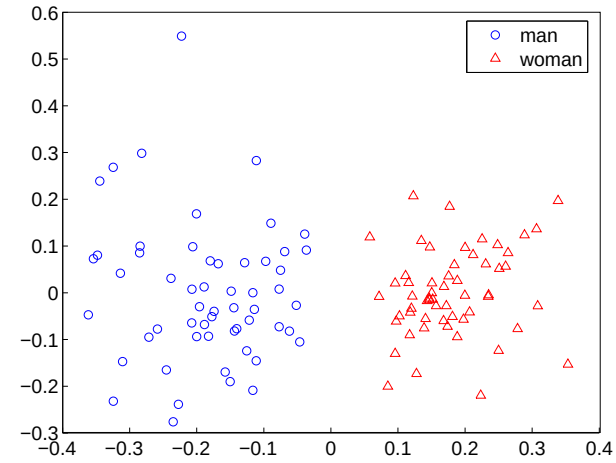

Fig. 7. Distribution of the adult TIDIGITS test speakers on the subspace spanned by the top 163 kernel eigenmatrices found by KEMLLR adaptation.

MLLR (EMLLR) adaptation. The new method, which we call "*kernel eigenspace-based MLLR*" (KEMLLR) adaptation performs kernel PCA on speaker (MLLR) transformation supervectors, and extracts eigenmatrices on the kernel-induced high-dimensional feature space. Unlike our first proposed kernel eigenvoice (KEV) adaptation method, at the end of KEMLLR adaptation, one gets back a real model for the new speaker; subsequent recognition is as fast as normal HMM decoding.

The new adaptation method was tested and compared with eKEV, EV, EMLLR, MLLR, and MAP adaptation on the medium-vocabulary RM task as well as the large-vocabulary WSJ task using only 5s and 10s of adaptation data. In both tasks and for both adaptation data lengths, KEMLLR outperforms EMLLR. Since the two methods only differ in the use of kernel methods, we believe that kernel PCA in KEMLLR helps extract more effective non-linear eigenvoices (or eigenmatrices) from the speakers. The analysis of the eigenspaces derived by EMLLR and KEMLLR on TIDIGITS also suggests that the (kernel) eigenspace found by KEMLLR is more informative. In fact, KEMLLR outperforms all the other methods in all cases (when they are run in their optimal settings). More specifically, for 5s adaptation, the performance of the various adaptation methods on WSJ0 are in the following order: SI $\simeq$ MLLR $<$ EV $\simeq$ MAP $<$ GD $<$ EMLLR $<$ eKEV $<$ KEMLLR; and for 10s adaptation, the order is: SI $<$ MAP $\simeq$ EV $<$ GD $<$ EMLLR $<$ eKEV $\simeq$ MLLR $<$ KEMLLR. In terms of computation, KEMLLR is more complex than the traditional methods like MLLR and MAP, but fortunately, all kernel evaluations can be pre-computed offline. More importantly, subsequent recognition speed using KEMLLR adaptation is as fast as normal HMM decoding.

In our current work, only the derivation of the eigenba-

sis is kernelized; subsequent estimation of the eigenmatrix weights still makes use of linear regression of the maximum-likelihood means. In the future, we would like to kernelize *both* the eigenbasis derivation process as well as the regression process.

VII. Acknowledgments

This research is partially supported by the Research Grants Council of the Hong Kong SAR under the grant numbers CA02/03.EG04 and DAG04/05.EG09.

References

[1] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski, "Rapid speaker adaptation in eigenvoice space," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 4, pp. 695–707, Nov 2000.

[2] K. T. Chen, W. W. Liau, H. M. Wang, and L. S. Lee, "Fast speaker adaptation using eigenspace-based maximum likelihood linear regression," in *Proceedings of the International Conference on Spoken Language Processing*, 2000, vol. 3, pp. 742–745.

[3] R. Kuhn, F. Perronnin, P. Nguyen, J.-C. Junqua, and L. Rigazio, "Very fast adaptation with a compact context-dependent eigenvoice model," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, May 2001, vol. 1, pp. 373–376.

[4] B. Zhou and J. Hansen, "Rapid discriminative acoustic model based on eigenspace mapping for fast speaker adaptation," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 4, pp. 554–564, July 2005.

[5] J. L. Gauvain and C. H. Lee, "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 291–298, April 1994.

[6] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Journal of Computer Speech and Language*, vol. 9, pp. 171–185, 1995.

[7] M. Turk and A. Pentland, "Face recognition using eigenfaces," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 1991, pp. 586–591.

[8] N. Wang, S. Lee, F. Seide, and L. S. Lee, "Rapid speaker adaptation using a priori knowledge by eigenspace analysis of MLLR parameters," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2001, pp. 345–348.

[9] H. Botterweck, "Very fast adaptation for large vocabulary continuous speech recognition using eigenvoices," in *Proceedings of the International Conference on Spoken Language Processing*, 2000, vol. 4, pp. 354–357.

[10] X. L. Aubert, "Eigen-MLLRs applied to unsupervised speaker enrollment for large vocabulary continuous speech recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2004, vol. I, pp. 349–352.

[11] V. Doumpiotis and Y. Deng, "Eigenspace-based MLLR with speaker adaptive training in large vocabulary conversational speech recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2004, vol. I, pp. 357–360.

[12] P. Nguyen, C. Wellekens, and J.-C. Junqua, "Maximum likelihood eigenspace and MLLR for speech recognition in noisy environments," in *Proceedings of the European Conference on Speech Communication and Technology*, 1999, pp. 2519–2522.

[13] M. F. J. Gales, "Cluster adaptive training of hidden Markov models," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 4, pp. 417–428, July 2000.

[14] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul, "A compact model for speaker-adaptive training," in *Proceedings of the International Conference on Spoken Language Processing*, 1996, pp. 1137–1140.

[15] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*, Cambridge University Press, 2000.

[16] Bernhard Schölkopf and Alexander J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, Cambridge, MA, 2002.

[17] V. Vapnik, *Statistical Learning Theory*, Wiley, New York, 1998.

[18] B. Mak, J. T. Kwok, and S. Ho, "Kernel eigenvoice speaker adaptation," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, pp. 984–992, September 2005.

[19] B. Schölkopf, A. Smola, and K. R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, pp. 1299–1319, 1998.

[20] J. T. Kwok, B. Mak, and S. Ho, "Eigenvoice speaker adaptation via composite kernel PCA," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Schölkopf, Eds. MIT Press, Cambridge, MA, 2004.

[21] B. Mak, J. T. Kwok, and S. Ho, "A study of various composite kernels for kernel eigenvoice speaker adaptation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Montreal, Canada, May 2004, vol. I, pp. 325–328.

[22] B. Mak, S. Ho, and J. T. Kwok, "Speedup of kernel eigenvoice speaker adaptation by embedded kernel PCA," in *Proceedings of the International Conference on Spoken Language Processing*, Jeju Island, South Korea, October 14–18 2004, vol. IV, pp. 2913–2916.

[23] B. Mak and S. Ho, "Various reference speakers determination methods for embedded kernel eigenvoice speaker adaptation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Philadelphia, USA, March 18–23 2005, vol. 1, pp. 981–984.

[24] S. Mika, B. Schölkopf, A. Smola, K.R. Müller, M. Scholz, and G. Rätsch, "Kernel PCA and de-noising in feature spaces," in *Advances in Neural Information Processing Systems 11*, M.S. Kearns, S.A. Solla, and D.A. Cohn, Eds., San Mateo, CA, 1998, Morgan Kaufmann.

[25] B. Mak and R. Hsiao, "Improving eigenspace-based MLLR adaptation by kernel PCA," in *Proceedings of the International Conference on Spoken Language Processing*, Jeju Island, South Korea, October 14–18 2004, vol. I, pp. 13–16.

[26] R. Hsiao and B. Mak, "Kernel eigenspace-based MLLR adaptation using multiple regression classes," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, Philadelphia, USA, March 18–23 2005, vol. 1, pp. 985–988.

[27] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik, "Support vector clustering," *Journal of Machine Learning Research*, vol. 2, pp. 125–137, 2001.

[28] F. R. Bach and M. I. Jordan, "Kernel independent component analysis," *Journal of Machine Learning Research*, vol. 3, pp. 1–48, 2002.

[29] S. Mika, G. Rätsch, J. Weston, B. Schoölkopf, and K.-R. Müller, "Fisher discriminant analysis with kernels," in *Neural Networks for Signal Processing IX*, Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, Eds., 1999, pp. 41–48.

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B*, vol. 39, no. 1, pp. 1–38, 1977.

[31] J. Frédéric Bonnans, J. Charles Gilbert, Claude Lemaréchal, and Claudia A. Sagastizábal, *Numerical Optimization: Theoretical and Practical Aspects*, Springer-Verlag, Berlin Heidelberg, 2003.

[32] K. T. Chen and H. M. Wang, "Eigenspace-based maximum a posteriori linear regression for rapid speaker adaptation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2001, vol. 1, pp. 317–320.

[33] P. Price, W. M. Fisher, J. Bernstein, and D. S. Pallett, "The DARPA 1000-word Resource Management database for continuous speech recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1988, vol. 1, pp. 651–654.

[34] D. B. Paul and J. M. Baker, "The design of the Wall Street Journal-based CSR corpus," in *Proceedings of the DARPA Speech and Natural Language Workshop*, Feb. 1992.

[35] B. Mak, S. Ho, R. Hsiao, and J. T. Kwok, "Embedded kernel eigenvoice speaker adaptation and its implication to reference speaker weighting," *IEEE Transactions on Speech and Audio Processing*, 2006, (Scheduled to be published in June 2006).

[36] K. Shinoda and C. H. Lee, "Unsupervised adaptation using structural Bayes approach," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1998.

[37] C. Chesta, O. Siohan, and C. H. Lee, "Maximum a posteriori

linear regression for hidden Markov model adaptation," in *Proceedings of the European Conference on Speech Communication and Technology*, 1999, vol. 1, pp. 211–214.

[38] N. Parihar and J. Picone, "DSR front end LVCSR evaluation," *AU/384/02, Aurora Working Group*, Dec. 2002, (http://www.isip.msstate.edu/projects/aurora).

[39] R. G. Leonard, "A database for speaker-independent digit recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1984, vol. 3, pp. 4211–4214.