# A Comparative Study of Two Kernel Eigenspace-based Speaker Adaptation Methods on Large Vocabulary Continuous Speech Recognition

Roger Hsiao and Brian Mak

Department of Computer Science
The Hong Kong University of Science and Technology
{hsiao,mak}@cs.ust.hk

## Abstract

Eigenvoice (EV) speaker adaptation has been shown effective for fast speaker adaptation when the amount of adaptation data is scarce. In the past two years, we have been investigating the application of kernel methods to improve EV speaker adaptation by exploiting possible nonlinearity in the speaker space, and two methods were proposed: embedded kernel eigenvoice (eKEV) and kernel eigenspace-based MLLR (KEMLLR). In both methods, kernel PCA is used to derive eigenvoices in the kernel-induced high-dimensional feature space, and they differ mainly in the representation of the speaker models. Both had been shown to outperform all other common adaptation methods when the amount of adaptation data is less than 10s. However, in the past, only small vocabulary speech recognition tasks were tried since we were not familiar with the behaviour of these kernelized methods. As we gain more experience, we are now ready to tackle larger vocabularies. In this paper, we show that both methods continue to outperform MAP, and MLLR when only 5s or 10s of adaptation data are available on the WSJ0 5K-vocabulary task. Compared with the speaker-independent model, the two methods reduce recognition word error rate by 13.4% – 21.1%.

## 1. Introduction

When the amount of adaptation speech is really small, say, a few seconds, eigenspace-based adaptation methods [1, 2] have been shown more effective than the traditionally more popular methods such as the Bayesian-based *maximum a posteriori* (MAP) adaptation [3] and the transformation-based *maximum likelihood linear regression* (MLLR) adaptation [4]. The idea is to derive from a diverse set of speakers a small set of basis vectors called *eigenvoices* (EV) that 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. 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.

Recently, we have been investigating the use of kernel methods [5] to improve the eigenspace-based adaptation methods by exploiting possible nonlinearity in their working space. In [6], we proposed the first kernel version of EV adaptation called *kernel eigenvoice (KEV) speaker adaptation*. The idea is to map input speaker supervectors to a kernel-induced high-dimensional feature space via some nonlinear map $\varphi$, and then apply principal component analysis (PCA) there. During the actual computation, the exact nonlinear map does not need to be known, and the eigenvoices in KEV adaptation are obtained in the feature space using *kernel PCA* [7]. In principle, since the KEV adaptation is a nonlinear generalization of the EV adaptation, the former should be more powerful than the latter, and KEV adaptation is expected to give better performance.

Although KEV is effective, it is slow due to many online kernel evaluations during recognition. We then proposed the fast KEV method which we call the *embedded kernel eigenvoice speaker adaptation* (eKEV) [8, 9]. eKEV adaptation eliminates all online kernel evaluations by finding an approximate pre-image of the adapting speaker model in the kernel-induced feature space. In the mean time, we also developed another variation of KEV called the *kernel eigenspace-based MLLR adaptation* (KEMLLR) [10, 11]. KEMLLR uses the speaker-specific MLLR transforms to represent a speaker, while KEV uses the HMM state mean vectors to do so. Both eKEV and KEMLLR had been shown to perform well on small- to medium-vocabulary tasks like TIDIGITS and Resource Management (RM) [8, 9, 10, 11]. However, their effectiveness on large-vocabulary continuous speech recognition (LVCSR) has yet to be shown. In this paper, we report and compare the adaptation performance of eKEV and KEMLLR on the 5K-vocabulary Wall Street Journal recognition task. We also give an account of our experience in porting the methods to LVCSR.

## 2. Review of the Two Kernel Eigenspace-based Speaker Adaptation Methods

### 2.1. Review of Embedded Kernel Eigenvoice Speaker Adaptation (eKEV)

We will briefly outline the eKEV adaptation procedure step by step using its illustration in Fig. 1 as follows; the details can be found in [6, 9].

STEP 1 : Speaker-dependent (SD) models are first created from a diverse set of speakers. For small vocabulary tasks, when there are many data per speaker, one may build an SD model for each speaker from scratch. However, in general, especially for large vocabulary tasks using triphones, it is unlikely to have sufficient speaker-specific training speech, and one may create SD models by MAP or MLLR adaptation. In any case, all SD models must have the same HMM topology.

STEP 2 : For each speaker, create his *speaker supervector* by concatenating all the Gaussian means in his model. The speaker supervectors are denoted as $\mathbf{x}_i, i = 1, \ldots, N$, where $N$ is the number of speakers in the training set. $N$ should be large, say, more than 100, but for illustration simplicity, it is 5 in Fig. 1.

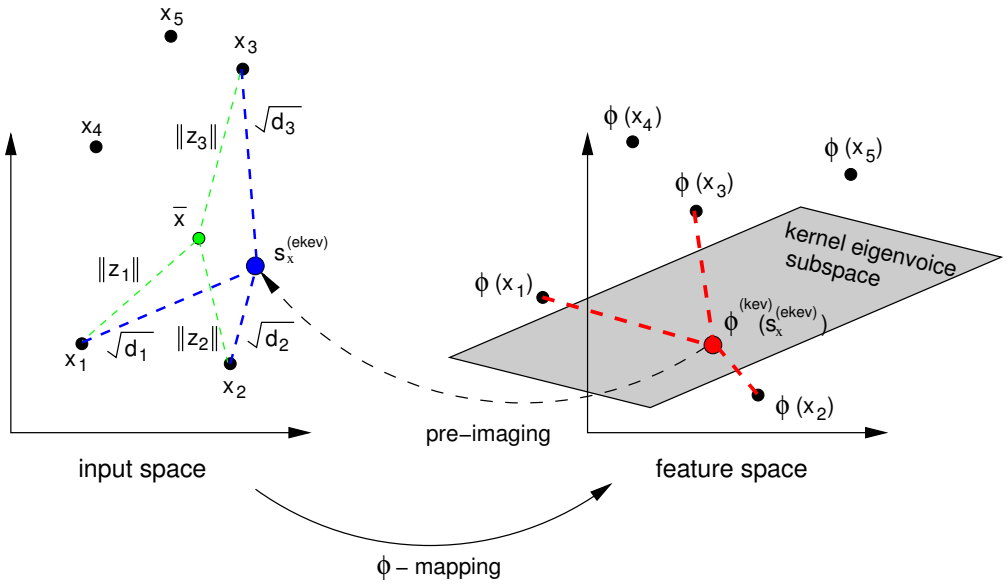


Figure 1: The eKEV adaptation method. (Without the pre-imaging step, it is the KEV adaptation method.)

STEP 3 : Conceptually a speaker supervectors $\mathbf{x}_i$ is then mapped to a high-dimensional feature space by a fictitious function $\varphi$ to $\varphi(\mathbf{x}_i)$. The exact definition of $\varphi$ need not be known. Instead, it is indirectly defined by a kernel function $k(\cdot,\cdot)$ such that $k(\mathbf{x}_i,\mathbf{x}_j) = \varphi(\mathbf{x}_i)'\varphi(\mathbf{x}_j)$ measures the "similarity" between the $i$th and the $j$th speaker supervectors.

STEP 4 : Kernel PCA is carried out using the kernel matrix $\mathbf{K}$ where $K_{ij} = k(\mathbf{x}_i,\mathbf{x}_j)$. The result is a set of basis vectors in the feature space, which we call "*kernel eigenvoices*".

STEP 5 : Using only a subspace constructed by the $M$ leading kernel eigenvoices (the shaded area in Fig. 1), a new speaker is supposed to lie on that kernel eigenvoice subspace. Or, in other words, a new speaker supervector in the feature space, denoted as $\varphi^{(kev)}(\mathbf{s}_x^{(ekev)})$, is a linear combination of the selected $M$ kernel eigenvoices. Usually $M$ is as small as about 10. The adaptation task is to find out the combination weights $\mathbf{w}$. If we stop here, it is the *kernel eigenvoice speaker adaptation* (KEV) method we first proposed in [6].

STEP 6 : $\varphi^{(kev)}(\mathbf{s}_x^{(ekev)})$ is actually not explicitly computed. Instead, it is mapped back to the input space by a procedure called *pre-imaging*. Theoretically speaking, an exact pre-image is generally not possible, and eKEV only computes an approximate pre-image $\mathbf{s}_x^{(ekev)}$ of the new speaker supervector $\varphi^{(kev)}(\mathbf{s}_x^{(ekev)})$ found by KEV.

STEP 7 : $\mathbf{s}_x^{(ekev)}$ is found by the principle of multi-dimensional scaling (MDS), making use of the distance relationship between the new speaker and a set of $n$ reference speakers in the input space and in the feature space. In [9], we suggested to use those who have the highest likelihoods of the adaptation data as the reference speakers.

STEP 8 : In Fig. 1, $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$ are the reference speakers. To find the pre-image, we have to first find their centroid $\bar{\mathbf{x}}$, and its distances to them $||\mathbf{z}_i||$. $\mathbf{z}_i$'s can be found by *singular value decomposition* using the reference speaker matrix:

$$[\mathbf{x}_1\ \mathbf{x}_2\ \cdots \mathbf{x}_n] = \mathbf{U}_2\mathbf{\Lambda}_2\mathbf{V}' = \mathbf{U}_2\,[\mathbf{z}_1\ \mathbf{z}_2\ \cdots \mathbf{z}_n]\ . \quad (1)$$

STEP 9 : The pre-imaging algorithm also requires the distances $d_i$ between the pre-image and the reference speakers in the input space. This can be deduced from their corresponding distances in the kernel-induced feature space when appropriate kernel functions are used. For instance, Gaussian kernel is used in eKEV.

STEP 10 : Finally, using the two set of distances: $||\mathbf{z}_i||$ and $d_i, i = 1,\ldots,n$, and the principle of MDS, there is a closed-form solution for the pre-image of the new kernel speaker supervector which is expressed in terms of the eigenvoice weights. The weights can be found by maximizing the likelihood of the adaptation data from the new speaker.

STEP 11 : Due to the nonlinear nature of the kernel function, the eigenvoice weights have to be determined by numerical methods such as gradient-based iterative methods.

Notice that steps 1–6 as well as all kernel evaluations between any two speaker supervectors are done offline, which are then used in the online execution of steps 8–11.

### 2.2. Review of Kernel Eigenspace-based MLLR Speaker Adaptation (KEMLLR)

The KEMLLR procedure is simpler and similar to that of eKEV's STEP 1–5 and STEP 11. The exception is that in STEP 2, an SD model is always estimated by MLLR of the SI model, and all SD models use the same number of regression classes. Then instead of using the concatenated Gaussian means to represent a speaker as in eKEV, the MLLR transforms of each speaker are vectorized and the set of corresponding vectorized MLLR transforms of the same regression class is used to perform kernel PCA to derive the kernel eigenvoices. Alternatively, one may concatenate together all vectorized MLLR

transforms for each speaker to form a speaker transformation supervector and perform kernel PCA on the transformation supervectors. However, our experience shows that better performance is obtained with separate kernel PCA on each MLLR transform class. When more training and adaptation data are available, more regression classes certainly will give better performance.

Readers are referred to [10, 11] for details.

### 2.3. Remarks

In our experience, successful application of eKEV and KEMLLR requires the following additional work:

- **Normalization**. We found that normalization of the various dimensions of the speaker supervectors by their variances in both methods gives more stable behaviour. Usually better adaptation performance is also obtained.
- **Composite Kernels**. In KEMLLR, when a transformation supervector is mapped to the kernel-induced feature space, the row information in the original MLLR transform will be lost. To preserve the row information, a composite kernel that defines on each row segment of the MLLR transforms must be employed. For eKEV, the use of composite kernels is not necessary. However, since KEV also requires them, for historical reasons, we also employ them in eKEV. Thus, in eKEV, composite kernels are defined on each state mean vector segment.
- **Interpolation with SI**. When the amount of adaptation is really small, it is usually better to improve the reliability of adaptation by incorporating some prior information. It is found that by simply interpolating the model computed by KEMLLR with the SI model, more stable and better speaker-adapted models are obtained. However, such interpolation is not found helpful for eKEV adaptation.

## 3. Porting to LVCSR

When eKEV and KEMLLR adaptation methods are ported to LVCSR using triphones, since there are not sufficient data to build speaker-dependent (SD) models from scratch, they are created by MLLR adaptation of the speaker-independent (SI) model using multiple regression classes. Below are some additional porting remarks:

- When SD models are created by adaptation, some of the eigenvalues in $\mathbf{\Lambda}_2$ of Eqn.(1) are very small. As the pre-image formula requires the inverse of $\mathbf{\Lambda}_2$, small eigenvalues will lead to serious numerical problem. The small eigenvalues should indicate that the span of the $n$ reference speakers is smaller than $n-1$. We simply discard those small eigenvalues and the eKEV algorithm then runs fine.
- Both kernel eigenspace-based methods basically computes the new speaker-adapted model as some non-linear combination of the SD models. Since speaker models are big in LVCSR, their storage can be a problem. Thus, MLLR adaptation is preferred to create the SD models so that one may simply store the MLLR transforms for each SD model instead of the SD hidden Markov models (HMM). On adaptation, the SD HMM mean vectors are then computed on-the-fly.
- One appealing feature of both eKEV and KEMLLR is that the kernel evaluations that both methods require can be pre-computed *before* adaptation. In eKEV, the composite kernel values between any two training speakers $k_r(\mathbf{x}_{ir}, \mathbf{x}_{jr})$ are pre-computed, where $\mathbf{x}_{ir}$ is the mean vector of the $r$th mixture of the $i$th speaker. Similarly, in KEMLLR, the composite kernel values $k_r(\mathbf{y}_{hr}, \xi_j)$ between any MLLR transformation row vector $\mathbf{y}_{hr}$ and any augmented SI mean vector $\xi_j$ are pre-computed.

Since both methods produce a regular SD HMM at the end, subsequent recognition does not involve any kernel evaluations and runs as fast as normal HMM.

## 4. Experimental Evaluation

The two kernelized eigenspace-based speaker adaptation methods were tested on the Wall Street Journal speech corpus WSJ0 [12]. 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.

### 4.1. Acoustic Modeling

The traditional 39-dimensional MFCC vectors were extracted at every 10ms over a window of 25ms.

The speaker-independent (SI) model consists of 15,449 cross-word triphones based on 39 base phonemes. Each of them was modeled as a continuous density HMM (CDHMM) which is strictly left-to-right and has three states with a Gaussian mixture density of 8 components per state.

The SD models were created by MLLR adaptation. In eKEV, a regression class tree of 32 classes were used, while in KEMLLR, only a global MLLR transform was used because from our past experience, KEMLLR requires many more eigenvoices and since adaptation of each regression class will be done separately, there may not be enough adaptation data to estimate the eigenvoice weights.

### 4.2. Parameter Setting

There are several free system parameters in eKEV and KEMLLR, such as the Gaussian kernel parameters. These system parameters were empirically determined using the Resource Management (RM) corpus, and they were then applied to WSJ0 task without any change. The recognition performance is not very sensitive to these parameters setting once they are in the right order of magnitude. The parameter values are listed below for readers' reference:

For eKEV,

- Number of eigenvoices to use: 10.
- Number of maximum-likelihood reference speakers: 5.
- Gaussian composite kernels of the form: $k_r(\mathbf{u}, \mathbf{v}) = \exp(-\beta\|\mathbf{u}-\mathbf{v}\|^2)$ were adopted; and $\beta = 0.005$.

For KEMLLR,

- The number of eigenvoices to use equals to the number of speakers. That is, for RM, 109 eigenvoices were used and for WSJ0, 83 eigenvoices were used.
- Again Gaussian composite kernels were adopted, and $\beta = 0.001$.

In both case, the quasi-Newton BFGS algorithm was used to search for the optimal eigenvoice weights.

Table 1: Adaptation performance on WSJ0 using 5s of speech.

| Model/Method | Word Accuracy (%) | WERR (%) |
|---|---|---|
| SI | 91.14 | – |
| MAP | 91.23 | 1.00 |
| MLLR (diagonal) | 91.45 | 3.50 |
| MLLR (full) | – | – |
| eKEV | 92.33 | 13.43 |
| KEMLLR | 92.58 | 16.25 |

Table 2: Adaptation performance on WSJ0 using 10s of speech.

| Model/Method | Word Accuracy (%) | WERR (%) |
|---|---|---|
| SI | 91.14 | – |
| MAP | 91.29 | 1.69 |
| MLLR (diagonal) | 91.37 | 2.60 |
| MLLR (full) | 92.36 | 13.77 |
| eKEV | 92.39 | 14.11 |
| KEMLLR | 93.01 | 21.11 |

### 4.3. Adaptation Performance

The following adaptation methods are compared:

**SI** : speaker-independent model.

**MLLR (diagonal)** : MLLR adaptation with diagonal transform(s).

**MLLR (full)** : MLLR adaptation with full transform(s).

**eKEV** : embedded kernel eigenvoice adaptation.

**KEMLLR** : kernel eigenspace-based MLLR adaptation.

All acoustic model training, MAP, and MLLR adaptation were carried out using the HTK software.

#### *4.3.1. Procedure and Results*

For each of the 8 speakers in the standard nov'92 5K nonverbalized test set, 1–3 utterances of his speech were randomly selected so that the amount of adaptation speech is about 5s or 10s, and his adapted model was tested on his remaining speech in the test set. A bigram language model of perplexity 147 was employed in the recognition. This was repeated three times and the three adaptation results were averaged before they were reported in Table 1 and Table 2.

Table 1 and Table 2 show that eKEV and KEMLLR continue to work better than MAP and MLLR on fast speaker adaptation with 5s or 10s in LVCSR. Their performance in LVCSR is consistent with their performance on small- and medium-vocabulary CSR which we reported in [6, 9, 10, 11]. For instance, with 5s of adaptation data, MAP and MLLR barely work, but eKEV and KEMLLR give 13.4% and 16.3% word error rate reduction (WERR). With 10s, MLLR using full transform gives comparable performance as eKEV's, and KEMLLR really outperforms both.

## 5. Discussions

An obvious question is: which of the two kernel eigenspace-based adaptation to choose? eKEV or KEMLLR?

In terms of accuracy, KEMLLR outperforms eKEV. That is especially true for 10s as eKEV's performance saturates quickly. KEMLLR also uses fewer transforms to achieve the better performance. Actually, KEMLLR's performance may further be improved with the use of more MLLR transforms. However, KEMLLR needs to use many more eigenvoices than eKEV. In the WSJ0 task, eKEV uses only 10 eigenvoices but KEMLLR uses 83. As a result, the adaptation speed of eKEV is much faster than KEMLLR: eKEV's adaptation time is only about $1/3$–$1/2$ of KEMLLR's.

## 6. Acknowledgments

This research is partially supported by the Research Grants Council of the Hong Kong SAR under the grant numbers HKUST6201/02E, and CA02/03.EG04.

## 7. References

[1] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski, "Rapid speaker adaptation in eigenvoice space," *IEEE Trans. on SAP*, 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 *Proc. of ICSLP*, 2000, vol. 3, pp. 742–745.

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

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

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

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

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

[8] B. Mak, S. Ho, and J. T. Kwok, "Speedup of kernel eigenvoice speaker adaptation by embedded kernel PCA," in *Proc. of ICSLP*, Jeju Island, South Korea, October 2004, vol. IV, pp. 2913–2916.

[9] B. Mak and S. Ho, "Various reference speakers determination methods for embedded kernel eigenvoice speaker adaptation," in *Proc. of ICASSP*, Philadelphia, USA, March 2005.

[10] B. Mak and R. Hsiao, "Improving eigenspace-based MLLR adaptation by kernel PCA," in *Proc. of ICSLP*, Jeju Island, South Korea, October 2004, vol. I, pp. 13–16.

[11] R. Hsiao and B. Mak, "Kernel eigenspace-based MLLR adaptation using multiple regression classes," in *Proc. of ICASSP*, Philadelphia, USA, March 2005.

[12] 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.