SPARSE KPCA FOR FEATURE EXTRACTION IN SPEECH RECOGNITION

A. Lima, H. Zen, Y. Nankaku, K. Tokuda, T. Kitamura

Nagoya Institute of Technology
Department of Computer Science and Engineering
Nagoya, 466-8555, Japan

F. G. Resende

Federal University of Rio de Janeiro
Dept. of Electronics and Computer Engineering/ EPoli
and Program of Electrical Engineering/ COPPE
Rio de Janeiro, 21945-970, Brazil

ABSTRACT

This paper presents an analysis of the applicability of Sparse Kernel Principal Component Analysis (SKPCA) for feature extraction in speech recognition, as well as, a proposed approach to make the SKPCA technique realizable for a large amount of training data, which is an usual context in speech recognition systems. Although the KPCA (Kernel Principal Component Analysis) has proved to be an efficient technique for being applied to speech recognition, it has the disadvantage of requiring training data reduction, when its amount is excessively large. The standard approach to perform this data reduction is to randomly choose frames from the original data set, which does not necessarily provide a good statistical representation of the original data set. In order to solve this problem a likelihood related re-estimation procedure was applied to the KPCA framework, thus creating the SKPCA. The experimental results show the efficiency of SKPCA technique with the proposed approach over the KPCA with the standard sparse solution using randomly chosen frames and the standard feature extraction techniques.

1. INTRODUCTION

The most commonly used feature extraction techniques used in speech recognition are mel frequency cepstral coefficients (MFCC) [1], linear prediction coefficients-cepstral (LPC-cepstral) coefficients [1] and perceptual linear prediction (PLP) coefficients [2]. They have already been very well analyzed and their efficiency widely proved. However the development of a kernel-based approach to "manipulate" data in a feature space (a non-linear higher dimensional space) came up with new concepts, in which the main idea is to express the speech data in a higher dimensional space to generate what would possibly be more discriminative speech features.

This approach was firstly applied to Support Vector Machines (SVMs) [3]–[4]. Some other examples of kernel-based learning machines are Kernel Discriminant Analysis (KDA) [5], Kernel Principal Component Analysis (KPCA) [6]–[12] and Sparse KPCA [13]. The KPCA is a non-linear approach to PCA. It depends on the training data to evaluate the higher dimensional principal components and also to represent a certain input data in the feature space. Depending on the training data amount these evaluations could be unfeasible and/or cause a huge computational burden. Considering this, the training data reduction is fundamental to the KPCA realization. The standard frame reduction is performed by choosing frames randomly, however these choices do not guarantee that the reduced data well represent the original data set. The SKPCA was developed to solve this problem by generating the reduced data set through a likelihood maximization criterion.

The SKPCA technique can be separated into two blocks, the re-estimation and the KPCA block. The covariance matrix used in SKPCA approach is modeled as the weighted outer-product of the training speech feature vectors plus an isotropic noise component, and these weights are updated by the re-estimation block. These weights generate the sparse solution for the KPCA, because they represent a measure of how well a specific training vector contribute to the likelihood maximization. Once obtained the reduced data, the common KPCA technique is applied and the representation of a feature test vector can be generated.

Although the SKPCA generates a reduced training data, it requires the full original training data to evaluate the maximization step, which could be computationally unfeasible, depending on the training data amount. In order to solve it, an approach is proposed, where the original training data is clustered and the SKPCA is applied to these clusters. Despite this approach does not guarantee that the overall data maximum is reached, it will be shown by experimental results that SKPCA could overcome the performance of KPCA and standard feature extraction techniques.

The paper is structured as follows. In Section 2, a detailed evaluation of PCA and KPCA techniques are described, emphasizing the main points to obtain SKPCA. In Section 3, the SKPCA is explained, and it comprises the weights re-estimation, the feature space representation and the proposed approach. In Section 4, experiments are presented assuring the efficiency of SKPCA. Finally, Section 5 presents the conclusions of this work and ideas for future work, as well.

2. FEATURE EXTRACTION USING KERNEL PCA

2.1. PCA

PCA is a well-established technique for dimensionality reduction. It represents a linear transformation where the data is expressed in a new coordinate basis that corresponds to the maximum variance "direction."

Assuming that the data set consists of M centered observations $\mathbf{x}_k \in \boldsymbol{\mathcal{R}}^n$, $k = 1, \ldots, M$, and $\sum_{k=1}^{M} \mathbf{x}_k = \mathbf{0}$, the sample covariance matrix corresponding to this data set is given by

$$\mathbf{S} = \frac{1}{M} \sum_{j=1}^{M} \mathbf{x}_j \mathbf{x}_j^T = M^{-1} \mathbf{X}\mathbf{X}^T, \tag{1}$$

where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_M]$ represents the matrix of data.

The principal components are obtained by solving the following eigenvalue system of equations, $\mathbf{SV} = \mathbf{V\Omega}$, where $\mathbf{\Omega}$ is the diagonal matrix with the eigenvalues and $\mathbf{V}$ is an orthogonal matrix of column eigenvectors of $\mathbf{S}$. It is well-known that the eigenvectors $\mathbf{V}$ can be obtained from the eigenvectors of the matrix $\mathbf{X}^T\mathbf{X}$ of inner-products.

Having $\mathbf{U}$ as the orthogonal matrix of column eigenvectors and $\mathbf{\Lambda}$ as the diagonal matrix of eigenvalues of $M^{-1}\mathbf{X}^T\mathbf{X}$, the following expression can be obtained, $M^{-1}\mathbf{X}^T\mathbf{XU} = \mathbf{U\Lambda}$. After a few algebraic procedures an expression where $\mathbf{V}$ is a function of $\mathbf{U}$ can be obtained and it is given by $\mathbf{V} = \mathbf{XU\Lambda}^{-\frac{1}{2}}$. This approach is generally used when $n \gg M$, i.e., when the dimensionality of $\mathbf{x}_k$ is greater than the number of training samples. However this is also essential to the KPCA development.

2.2. KPCA

The Kernel PCA is the technique which applies the kernel function to the PCA technique, in order to obtain the representation of PCA in a higher dimensional space [7]. The kernel functions are widely used to solve the problem of nonlinear mapping (ϕ) to a higher dimensional space, without using explicit mapping. This nonlinear mapping is performed by using kernel functions as the dot product of the mapped variables: $\phi : \mathbf{x} \cdot \mathbf{y} \rightarrow \phi(\mathbf{x}) \cdot \phi(\mathbf{y}) = k(\mathbf{x}, \mathbf{y})$. The kernel matrix $\mathbf{K}$ is defined as the matrix whose indexes are $(\mathbf{K})_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$.

Defining $\phi(\mathbf{x}_i) = \phi_i$, it can be said that $\phi_i^T\phi_j = k(\mathbf{x}_i, \mathbf{x}_j)$, and the mapping of the full data matrix $\mathbf{X}$ can be defined by a $(D \times M)$ matrix $\mathbf{\Phi} = [\phi_1 \ldots \phi_i \ldots \phi_M]$, where ϕ_i represents the mapping of $\mathbf{x}_i$ in a higher dimension D.

Analogous to equation (1), the covariance matrix in a feature space is given by

$$\mathbf{S}_{\mathcal{F}} = \frac{1}{M}\sum_{j=1}^{M}\phi_j\phi_j^T = M^{-1}\mathbf{\Phi\Phi}^T, \tag{2}$$

and consequently the representation of the eigenvectors $\mathbf{V}$ in the feature space is $\mathbf{V}_{\mathcal{F}} = \mathbf{\Phi U}_K\mathbf{\Lambda}_K^{-\frac{1}{2}}$, where $\mathbf{U}_K$ and $\mathbf{\Lambda}_K$ contain the eigenvectors and eigenvalues of the kernel matrix $\mathbf{K}$.

Finally, the KPCA representation of a test vector $\mathbf{t}$ is given by the projection of the mapped vector $\phi(\mathbf{t})$ onto the eigenvectors $\mathbf{V}_{\mathcal{F}}$. It is mathematically expressed as $\mathbf{T}^{kpca} = \mathbf{V}_{\mathcal{F}}^T\phi(\mathbf{t})$, where $\mathbf{T}^{kpca}$ is a D dimensional column vector, which gives the KPCA representation of $\phi(\mathbf{t})$. The final representation is shown as follows:

$$\mathbf{V}_{\mathcal{F}}^T\phi(\mathbf{t}) = \left(\mathbf{\Lambda}_K^{-\frac{1}{2}}\right)^T\mathbf{U}_K^T\mathbf{\Phi}^T\phi(\mathbf{t}) = \mathbf{\Lambda}_K^{-\frac{1}{2}}\mathbf{U}_K^T\mathbf{k}_{\mathbf{t}}^T, \tag{3}$$

where $\mathbf{k_t}$ represents a M dimensional column vector formed by $k(\mathbf{t}, \mathbf{x}_i)$, for $i = 1, \ldots, M$.

Although the KPCA is a powerful technique, it has the disadvantage of requiring the full training data to calculate $\mathbf{K}$ and $\mathbf{k_t}$ in (3), which could cause computational problems, as it was mentioned in Section 1. A common solution to this problem is to reduce the number of frames (full training data) by picking up N frames randomly from the training data, as it was cited in [14]. Although this approach has shown an efficient performance in a speech recognition task [12], it does not use the overall information provided by the training data.

3. THE PROPOSED APPROACH TO SKPCA FEATURE EXTRACTION

3.1. SKPCA

The SKPCA technique was developed in order to provide a solution for the previous mentioned disadvantage of the KPCA technique. It consists in estimating the feature space sample covariance for a noise component and the sum of the weighted outer products of the original feature vectors, which generate a sparse solutions to KPCA. This is obtained by maximizing the likelihood of the feature vectors under a Gaussian density model $\phi \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_{\mathcal{F}})$, where the covariance $\mathbf{C}_{\mathcal{F}}$ is defined by

$$\mathbf{C}_{\mathcal{F}} = \sigma^2\mathbf{I} + \sum_{i=1}^{M} w_i\phi_i\phi_i^T = \sigma^2\mathbf{I} + \mathbf{\Phi W\Phi}^T, \tag{4}$$

where $\mathbf{W}$ is a diagonal matrix composed by the adjustable weights $w_1, \ldots, w_M$, and σ^2 is an isotropic noise component, $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, common to all dimensions of feature space. This approach was based on the probabilistic PCA (PPCA) formulation [15].

The log-likelihood under the Gaussian model with covariance $\mathbf{C}_{\mathcal{F}}$ given by (4), ignoring the terms independent of the weights, is denoted by

$$\mathcal{L} = -\frac{1}{2}\left[M\log|\mathbf{C}_{\mathcal{F}}| + \mathrm{tr}(\mathbf{C}_{\mathcal{F}}^{-1}\mathbf{\Phi\Phi}^T)\right]. \tag{5}$$

Differentiating (5) under the weights w_i and making it equal to zero, means to maximize the log-likelihood with respect to w_i. However, in order to reach better mathematical representation, (5) should be decomposed. The first term of (5) can be decomposed in $M\log|\mathbf{C}_{\mathcal{F}}| = M(D\log\sigma^2 + \log|\mathbf{W}| + \log|\mathbf{W}^{-1} + \sigma^{-2}\mathbf{K}|)$ and the second term in $\mathrm{tr}(\mathbf{C}_{\mathcal{F}}^{-1}\mathbf{\Phi\Phi}^T) = \sum_{i=1}^{M}\sigma^{-2}k_{ii} - \sigma^{-4}\mathbf{k}_i^T\left(\mathbf{W}^{-1} + \sigma^{-2}\mathbf{K}\right)^{-1}\mathbf{k}_i$, where $k_{ii} = k(\mathbf{x}_i, \mathbf{x}_i)$.

Now evaluating $\frac{\partial\mathcal{L}}{\partial w_i}$ by differentiating the two terms of (5) obtained above with respect to w_i, the following expression is achieved,

$$\frac{\partial\mathcal{L}}{\partial w_i} = \frac{1}{2w_i^2}\left[M\mathbf{\Sigma}_{ii} - Mw_i + \sum_{j=1}^{M}\mu_{ji}^2\right], \tag{6}$$

where $\mathbf{\Sigma}_{ii}$ and μ_{ji} are respectively the diagonal components of the matrix $\mathbf{\Sigma} = (\mathbf{W}^{-1} + \sigma^{-2}\mathbf{K})^{-1}$ and the elements of the column vector $\mu_j = \sigma^{-2}\mathbf{\Sigma k}_j$. Setting (6) to zero, which means to find the maximum of the function represented by the equation (5), generates the re-estimation update functions for the weights, $w_i^{new} = M^{-1}\sum_{j=1}^{M}\mu_{ji}^2 + \mathbf{\Sigma}_{ii}$. According to [13], an equation for re-estimation update that converges faster than the one previously mentioned, can be obtained by rewriting (6) equal to zero as

$$w_i^{new} = \frac{\sum_{j=1}^{M}\mu_{ji}^2}{M\left(1 + \mathbf{\Sigma}_{ii}/w_i\right)}. \tag{7}$$

Equivalently to the KPCA representation, the projection of a test vector $\phi(\mathbf{t})$ onto the principal axes $\mathbf{V}_{\mathcal{F}}$ is calculated by

$$\mathbf{T}^{skpca} = \mathbf{V}_{\mathcal{F}}^T\phi(\mathbf{t}) = \tilde{\mathbf{\Lambda}}_K^{-\frac{1}{2}}\tilde{\mathbf{U}}_K^T\hat{\mathbf{k}}_{\mathbf{t}}^T, \tag{8}$$

where $\tilde{\mathbf{U}}_K$ and $\tilde{\mathbf{\Lambda}}_K$ are defined, respectively, as the eigenvectors and eigenvalues of $\mathbf{W}^{\frac{1}{2}}\mathbf{K}\mathbf{W}^{\frac{1}{2}}$, and $\hat{\mathbf{k}}_{\mathbf{t}}$ represents the vector calculated by $k(\mathbf{t}, \mathbf{x}_i)$, where $\mathbf{x}_i$ corresponds to the non-zero weighted vectors represented in $\mathbf{X}$.

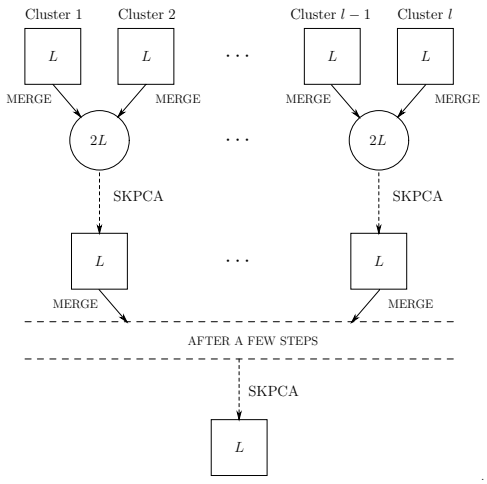

Fig. 1. *Proposed approach.*

3.2. Proposed approach

The proposed approach consists in making the SKPCA technique computationally feasible for a data set with a great number of samples, which is an usual situation in speech recognition.

Generally, in speech recognition the amount of training data tends to be large, for example, in this work the training data comprises about 1,200,000 frames, and with this number of frames the SKPCA re-estimation in equation (7) is computationally unfeasible, once it depends on the kernel matrix $\mathbf{K}$ to calculate $\mathbf{\Sigma}$. In order to overcome this limitation, it is proposed to divide the full training data into clusters of L frames, then merge the clusters forming new clusters of $2L$ frames, which are reduced to L frames by using SKPCA. The process is repeated successively until obtaining just one cluster of L frames, which is the final number of frames desired to represent the full training data, as shown in Figure 1. The weights w_i are just used to select the most representative training vectors considering the likelihood maximization. Thus, they are not included in the following frame reduction step and neither in the SKPCA feature representation procedure.

The total number of steps necessary to reduce the l clusters of L frames to just one cluster, is given by $step = \log_2 l$, where $step$ is the number of steps. The "ideal" approach is to perform the re-estimation in (7) over the training database as a whole, however it is not realizable due to the computational reasons mentioned before. Considering this, the proposed approach does not guarantee to reach the overall data maximization, just individual cluster maximization. However as it will be shown in Section 4, its performance overcomes the standard randomly chosen frames approach used in KPCA.

4. EXPERIMENTAL WORK

In order to evaluate the efficiency of this technique, a speaker-independent isolated word recognition experiment was conducted. The experiment consisted in using a larger database, a 520 Japanese words vocabulary with 80 speakers (40 males and 40 females) extracted from the C set of the ATR Japanese database. The training data was composed of 10400 utterances and the remaining 31200 utterances were used as test data. This configuration was used due to the original database characteristics (label files) and also to provide enough statistical reliability to the experimental results.

4.1. Settings

The sampling rate of speech signal was 10 kHz. Mel-cepstral coefficients were extracted through a 12-th order mel-cepstral analysis using 25.6 ms Hamming windows with 10 ms shifts. The feature vectors were obtained from the 13 mel-cepstral coefficients and their delta (Δ) and acceleration ($\Delta\Delta$) coefficients, which correspond to a vector of 39 coefficients.

In order to calculate the matrix $\mathbf{K}$ for the KPCA case, it was used N equal to 256 and 512 frames, which were randomly picked up from the full training data, and for the SKPCA case, it was used L=256 frames and N equal to 256 and 512 frames. The number N was chosen such that the system was computationally feasible.

Each word was modeled using 12 state HMMs (Hidden Markov Models) with single mixture of diagonal covariance, and as for the kernel function, it was used a polynomial kernel function such as $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p$.

4.2. Results

The baseline error rate of the standard features was 8.36%, when 13 mel-cepstral coefficients and their Δ and $\Delta\Delta$ were used as a feature vector, and the PCA best result with full training data was 7.70% of error rate for dimensionality 32 and column wDA (with Delta and Acceleration characteristics).

The proposed approach in Section 3.2 was applied using L=256. The reduced number of frames N was obtained according to L, i.e., the training data set obtained by using L=256 was used in N=256, and the training data set for N equal to 512 was obtained by merging the final clusters achieved in $(step - 1)$ steps. In other words, the data set when N=256 is a subset of the data set when N=512, as it is shown in Figure 2, which is a simplified version of Figure 1 focusing on the number of frames N.

Table 1 shows the performances using N=256 of KPCA & p=1, which represents the PCA with a reduced training data, KPCA & p=2 and SKPCA & p=2, where the columns wDA and w/DA (without Delta and Acceleration) refer to the feature vector representation with and without the Δ and $\Delta\Delta$ coefficients applied after the feature extraction technique, respectively. The best perfor-

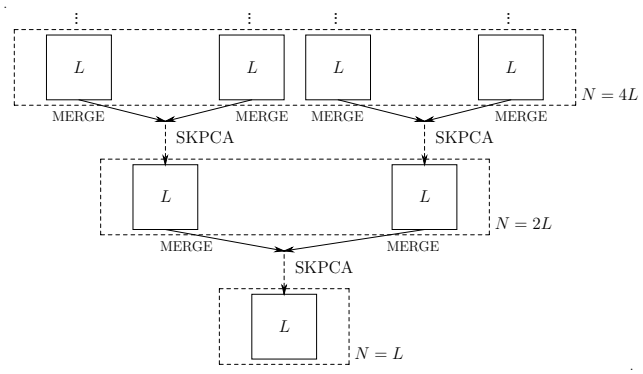

Fig. 2. *Proposed approach focusing on the number of frames N.*

Table 1. Error rate (%) of the system using KPCA with a 1^{st} and 2^{nd} degree polynomial kernel function, SKPCA with 2^{nd} degree polynomial kernel function and N equal to 256. The columns w/DA and wDA represent respectively the feature vectors not using and using Δ and $\Delta\Delta$. The symbol "$*$" means "it does not apply."

N=256						
	KPCA & p=1		KPCA & p=2		SKPCA & p=2	
dim	w/DA	wDA	w/DA	wDA	w/DA	wDA
8	22.16	8.61	25.91	7.92	25.21	8.36
13	22.96	**7.99**	25.62	7.48	22.05	**6.37**
16	19.79	8.72	25.13	7.70	21.73	6.42
32	**11.81**	9.94	18.44	**7.43**	17.04	7.15
39	12.88	11.77	**18.22**	7.77	**16.68**	7.42
64	$*$	$*$	20.44	9.45	19.62	9.02
128	$*$	$*$	28.60	14.92	25.04	13.15

Table 2. Error rate (%) of the system using KPCA with a 1^{st} and 2^{nd} degree polynomial kernel function, SKPCA with 2^{nd} degree polynomial kernel function and N equal to 512.

N=512						
	KPCA & p=1		KPCA & p=2		SKPCA & p=2	
dim	w/DA	wDA	w/DA	wDA	w/DA	wDA
8	22.61	8.43	24.60	7.88	22.55	7.51
13	23.19	**8.07**	24.54	7.35	21.31	**6.30**
16	21.09	8.32	23.44	7.59	19.12	6.29
32	**11.48**	10.06	**17.61**	**7.29**	14.00	6.97
39	13.28	12.29	17.62	7.85	**13.88**	7.05
64	$*$	$*$	22.01	9.95	17.72	8.74
128	$*$	$*$	31.93	17.66	23.87	13.90

mances were respectively, 7.99% (dimension 13), 7.43% (dimension 32) and 6.37% (dimension 13) of error rate for column wDA. It is observed that the SKPCA overcame all the others techniques, and as it was expected the PCA with full training data reached a higher performance than KPCA & p=1, which possibly was due to the data reduction for the KPCA case. Thus, this work was focused on 2^{nd} degree polynomial kernel function.

Table 2 shows the equivalent characteristics of Table 1, except that N=512 frames. The best performances were respectively, 8.07% (dimension 13), 7.29% (dimension 32) and 6.30% (dimension 13) of error rate for column wDA. The results presented in this table confirmed the efficiency of SKPCA over the PCA with full training data, KPCA & p=1 and the baseline, as mentioned previously.

Comparing the best performances for KPCA and SKPCA in both tables, it is noticed that the performances degrade when the number of frames is reduced, except for the KPCA & p=1 with wDA. This could be explained by the elimination of important data information from the data set with 512 frames when it is reduced to 256 frames, once the data set for N=256 is a subset of the data set for N=512. The overall best performance was 6.30% of error rate using SKPCA with 13 dimensions and wDA for N=512.

5. CONCLUSIONS

In this paper, the SKPCA technique with the proposed approach was applied for feature extraction in speech recognition. As it was expected the SKPCA provided a better representation of the reduced training data than the one obtained by the standard randomly chosen frames approach used in KPCA. The overall best performance 6.37% of error rate (ER) (dimensionality 13, wDA, SKPCA and N=512) generated error rates reduction of 24.6%, 21.2% (dimensionality 13, wDA, KPCA & p=1 and N=256), 18.2% and 13.6% (dimensionality 32, wDA, KPCA & p=2 and N=512) over the baseline (8.36% ER) and the best performances of KPCA & p=1 (7.99% ER), PCA (7.70% ER) and KPCA & p=2 (7.29% ER), respectively. The results confirmed the efficiency of SKPCA and the proposed approach for this task.

Despite the technique presented in this paper has considerably improved the recognition performance of the analyzed task, it is required further research in order to observe carefully the effects of using different techniques to cluster the full training data to make the SKPCA re-estimation realizable. Besides the previous mentioned topic, further study on other kernel-based sparse approaches and different kernel-based learning machines are the natural future steps of this work.

6. REFERENCES

[1] L. R. Rabiner and B. Juang, *Fundamentals on Speech Recognition*, Prentice Hall, New Jersey, 1996.

[2] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *Journal of the Acoustical Society of America*, , no. 87, pp. 1738–1752, 1990.

[3] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, 1995.

[4] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, 1998.

[5] V. Roth and V. Steinhage, "Nonlinear discriminant analysis using kernel functions," *Advances in Neural Information Processing Systems*, pp. 568–574, 1999.

[6] B. Schölkopf, A. Smola, and K. R. Müller, "Nonliner component analysis as a kernel eigenvalue problem," *Neural Computation*, , no. 10, pp. 1299–1319, 1998.

[7] B. Schölkopf, A. Smola, and K. R. Müller, "Kernel principal component analysis," *Advances in Kernel Methods*, pp. 327–352, 1998.

[8] K. I. Kim, S. H. Park, and H. J. Kim, "Kernel principal component analysis for texture classification," *IEEE Signal Processing Letters*, vol. 8, no. 2, 2001.

[9] K. I. Kim, K. Jung, and H. J. Kim, "Face recogniton using kernel principal component analysis," *IEEE Signal Processing Letters*, vol. 9, no. 2, 2002.

[10] A. Kocsor, A. Kuba Jr., and L. Tóth, "Phoneme classification using kernel principal component analysis," *Periodica Polytechnica*, vol. 44, no. 1, pp. 77–90, 2000.

[11] A. Lima, H. Zen, Y. Nankaku, C. Miyajima, K. Tokuda, and T. Kitamura, "On the use of Kernel PCA for feature extraction in speech recognition," *Proc. of EuroSpeech*, pp. 2625–2628, Sep. 2003.

[12] A. Lima, H. Zen, Y. Nankaku, C. Miyajima, K. Tokuda, and T. Kitamura, "On the use of KPCA for feature extraction in speech recognition," *IEICE Trans. Inf. & Syst.*, vol. E87-D, no. 12, pp. 2802–2811, Dec. 2004.

[13] M. E. Tipping, "Sparse kernel principal component analysis," in *Advances in Neural Information Processing Systems*, 2000.

[14] A. J. Smola, O. L. Mangasarian, and B. Schölkopf, "Sparse kernel feature analysis," Tech. Rep., University of Wisconsin, 1999.

[15] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society*, pp. 611–622, 1999.