PAPER Special Section on Corpus-Based Speech Technologies

Applying Sparse KPCA for Feature Extraction in Speech Recognition

Amaro LIMA[†a)], **Heiga ZEN**[†], ***Nonmembers***, **Yoshihiko NANKAKU**[†], **Keiichi TOKUDA**[†], **Tadashi KITAMURA**[†], ***Members***, ***and*** **Fernando G. RESENDE**[††], ***Nonmember***

SUMMARY 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. This data reduction is important to avoid computational unfeasibility and/or an extremely high computational burden related to the feature representation step of the training and the test data evaluations. 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, which nevertheless is not realizable for large training databases. The proposed approach consists in clustering the training data and applying to these clusters a SKPCA like data reduction technique generating the reduced data clusters. These reduced data clusters are merged and reduced in a recursive procedure until just one cluster is obtained, making the SKPCA approach realizable for a large amount of training data. 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.

key words: *kernel, sparsity, principal component analysis, feature extraction, speech recognition*

1. Introduction

Nowadays speech recognition systems have reached a high effectiveness condition, which can be confirmed by the numerous commercial products with recognition applicability available not only for corporate, but also for personal use. The feature extraction is one of the factors that contributed to this effectiveness.

The most commonly used feature extraction techniques 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 feature 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 high 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.

As it is shown in Fig. 1, the SKPCA technique can be separated in two blocks, the reduction of training data 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 SKPCA re-estimation procedure. 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 $\mathbf{t}$ is given by $\mathbf{T}^{skpca}$. In this paper the mel-cepstral coefficients are used in the feature vector representation. Figure 1 also shows where the SKPCA is located in the speech recognition system as a whole.

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 re-

Manuscript received July 7, 2004.
Manuscript revised October 4, 2004.

[†]The authors are with the Department of Computer Science and Engineering, Nagoya Institute of Technology, Nagoya-shi, 466–8555 Japan.

[††]The author is with the Department of Electronics and Computer Science/EPoli, and with the Electrical Engineering Program/COPPE, Federal University of Rio de Janeiro, Rio de Janeiro, PO Box 68504, 21941–972, Brazil.

a) E-mail: amaro@ics.nitech.ac.jp

DOI: 10.1093/ietisy/e88–d.3.401

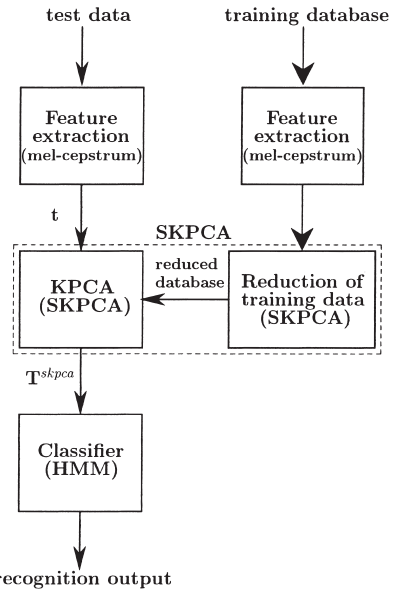


Fig. 1 A simplified representation of the recognition system with the SKPCA block.

sults that SKPCA could surpass the performance of KPCA and standard feature extraction techniques.

The paper is structured as follows. In Sect. 2, a detailed evaluation of PCA, KPCA and SKPCA techniques are described concluding the section with the SKPCA weights re-estimation and feature space representation. In Sect. 3, the proposed approach is explained considering the training data reduction using a SKPCA like approach. In Sect. 4, experiments are presented assuring the efficiency of SKPCA. Finally, Sect. 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 $\mathbf{x}_k \in \mathcal{R}^n$, $k = 1, \ldots, M$, 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.

Table 1 Some examples of kernel functions.

Kernel function	Equation
Polynomial function	$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p$
Gaussian RBF (GRBF)	$k(\mathbf{x}, \mathbf{y}) = e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}}$
Sigmoid function	$k(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x} \cdot \mathbf{y} - \delta)$

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, $M^{-1}\mathbf{X}^T\mathbf{XU} = \mathbf{U\Lambda}$. Multiplying it by $\mathbf{X}$ on both sides, $M^{-1}\mathbf{X}\mathbf{X}^T(\mathbf{XU}) = (\mathbf{XU})\mathbf{\Lambda}$, is obtained and can be noticed that $\mathbf{XU}$ are proportional to the eigenvectors of $M^{-1}\mathbf{X}\mathbf{X}^T = \mathbf{S}$, i.e., $\mathbf{XU} \propto \mathbf{V}$ and $\mathbf{\Lambda}$ are the corresponding eigenvalues. The previous statement was observed because the column vectors of $\mathbf{XU}$ are not normalized, which is clearly noticed by, $\mathbf{U}^T\mathbf{X}^T\mathbf{XU} = \mathbf{\Lambda}$. Hence the normalized eigenvectors of $M^{-1}\mathbf{X}\mathbf{X}^T$ are given by

$$\mathbf{V} = \mathbf{XU\Lambda}^{-\frac{1}{2}}. \tag{2}$$

This approach is generally used when $n \gg M$, i.e., when the dimensionality of $\mathbf{x}$ is greater than the number of training samples. However this is also essential to the KPCA development.

2.2 Kernel Functions

These functions are widely used to solve the problem of nonlinear mapping (ϕ) to a higher dimensional space, without using explicit mapping, which would be computationally unfeasible.

If the function $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ is a symmetric positive function that obeys the Mercer's condition [4], then it can be shown that this function represents the dot product of the variables $\mathbf{x}_i$ and $\mathbf{x}_j$ in the feature space. Thus the nonlinear mapping can be 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 indices are $(\mathbf{K})_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Some examples of kernel functions are shown in Table 1.

2.3 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].

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

$$\mathbf{S}_{\mathcal{F}} = \frac{1}{M} \sum_{j=1}^{M} \phi_j \phi_j^T = M^{-1} \mathbf{\Phi}\mathbf{\Phi}^T, \tag{3}$$

and consequently the representation of the eigenvectors V in

the feature space is

$$\mathbf{V}_{\mathcal{F}} = \mathbf{\Phi}\mathbf{U}_K\mathbf{\Lambda}_K^{-\frac{1}{2}}, \tag{4}$$

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 $\phi(\mathbf{t})$. Even without obtaining $\mathbf{V}_{\mathcal{F}}$, $\mathbf{\Phi}$ and $\phi(\mathbf{t})$ explicitly, the final KPCA representation can be achieved by computing the dot product of the "implicit" variables through the application of a kernel function, which is shown as follows:

$$\mathbf{V}_{\mathcal{F}}^T\phi(\mathbf{t}) = \left(\mathbf{\Lambda}_K^{-\frac{1}{2}}\right)^T\mathbf{U}_K^T\mathbf{\Phi}^T\phi(\mathbf{t}) = \mathbf{\Lambda}_K^{-\frac{1}{2}}\mathbf{U}_K^T\mathbf{k_t}^T, \tag{5}$$

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

All the steps evaluated in this section are better visualized in Fig. 2. It shows the ϕ mapping of the training and test data ($\mathbf{t}_1$, $\mathbf{t}_2$ and $\mathbf{t}_3$), as well as, the principal components (dashed line) obtained from the training data, and finally the KPCA representation of the test data ($\mathbf{T}_1^{kpca}$, $\mathbf{T}_2^{kpca}$ and $\mathbf{T}_3^{kpca}$). In practice, the individual mapping of the data does not occur, the mapping is performed implicitly based on the dot products of the data, however this way of presenting the KPCA technique makes the procedure easier to be understood.

Although the KPCA is a powerful technique, it has the disadvantage of requiring the full training data to calculate the kernel matrix $\mathbf{K}$ and $\mathbf{k_t}$ in (5). The calculation of $\mathbf{K}$ is constrained by computational resources due to the fact that it is desired to use as much data as possible in the training step to generate a good estimation of the data space, and $\mathbf{K}$ is a $M \times M$ matrix, where M is the number of samples of the training data. In practice, the amount of training data can not be excessively large, otherwise the eigenvectors of $\mathbf{K}$ in the KPCA will be computationally unfeasible, and even if the amount of data is not excessively large for the eigenvectors calculation, it could be big enough to cause a certain computational burden for $\mathbf{k_t}$ calculation in the final KPCA representation.

A common solution to the problem mentioned before is to reduce the number of frames (full training data) to N frames, which are randomly picked up 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, i.e., when frames are randomly picked up, the training data is reduced without necessarily keeping any statistics of the original data set.

2.4 SKPCA

In order to provide a solution for the previous mentioned

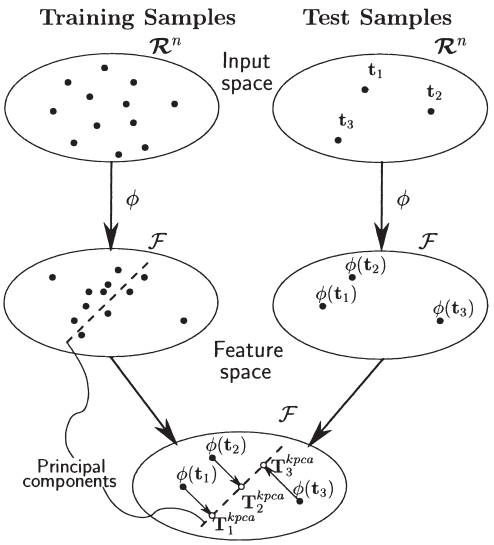


Fig. 2 Visualization of KPCA procedure.

disadvantage of the KPCA technique, an approach where the frame reduction obeys the output probability maximization criterion was developed, and it is called SKPCA. 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 solution 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

$$\mathbf{C}_{\mathcal{F}} = \sigma^2\mathbf{I} + \sum_{i=1}^{M} w_i\phi_i\phi_i^T = \sigma^2\mathbf{I} + \mathbf{\Phi}\mathbf{W}\mathbf{\Phi}^T, \tag{6}$$

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. It was observed that fixing σ^2, and maximizing the likelihood under the weighting factors w_i, the estimates of several w_i are zero, thus realizing a reduced (sparse) representation of the covariance matrix. 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 (6), is denoted by

$$\mathcal{L} = -\frac{M}{2}\left[D\log(2\pi) + \log|\mathbf{C}_{\mathcal{F}}| + \mathrm{tr}(\mathbf{C}_{\mathcal{F}}^{-1}\mathbf{S}_{\mathcal{F}})\right], \tag{7}$$

ignoring the terms independent of the weights, it is expressed by

$$\mathcal{L} = -\frac{1}{2}\left[M\log|\mathbf{C}_{\mathcal{F}}| + \mathrm{tr}(\mathbf{C}_{\mathcal{F}}^{-1}\mathbf{\Phi}\mathbf{\Phi}^T)\right]. \tag{8}$$

Differentiating (8) 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, (8) should be decomposed. The first term of (8), $M\log|\mathbf{C}_{\mathcal{F}}|$ can be decomposed in

$$M \log|\mathbf{C}_{\mathcal{F}}| = M\left(\log|\sigma^2\mathbf{I} + \mathbf{\Phi}\mathbf{W}\mathbf{\Phi}^T|\right)$$
$$= M\left(D\log\sigma^2 + \log|\mathbf{W}| + \log|\mathbf{W}^{-1} + \sigma^{-2}\mathbf{\Phi}^T\mathbf{\Phi}|\right)$$
$$= M\left(D\log\sigma^2 + \log|\mathbf{W}| + \log|\mathbf{W}^{-1} + \sigma^{-2}\mathbf{K}|\right), \quad (9)$$

and the second term, $\mathrm{tr}(\mathbf{C}_{\mathcal{F}}^{-1}\mathbf{\Phi}\mathbf{\Phi}^T)$ is rewritten as:

$$\mathrm{tr}(\mathbf{C}_{\mathcal{F}}^{-1}\mathbf{\Phi}\mathbf{\Phi}^T) = \sum_{i=1}^{M}\phi_i^T\left(\mathbf{C}_{\mathcal{F}}^{-1}\right)\phi_i$$
$$= \sum_{i=1}^{M}\phi_i^T\left(\sigma^2\mathbf{I} + \mathbf{\Phi}^T\mathbf{W}\mathbf{\Phi}\right)^{-1}\phi_i$$
$$= \sum_{i=1}^{M}\phi_i^T\left[\sigma^{-2}\mathbf{I} - \sigma^{-4}\mathbf{\Phi}\left(\mathbf{W}^{-1} + \sigma^{-2}\mathbf{\Phi}^T\mathbf{\Phi}\right)^{-1}\mathbf{\Phi}^T\right]\phi_i$$
$$= \sum_{i=1}^{M}\sigma^{-2}k_{ii} - \sigma^{-4}\mathbf{k}_i^T\left(\mathbf{W}^{-1} + \sigma^{-2}\mathbf{K}\right)^{-1}\mathbf{k}_i, \quad (10)$$

where $k_{ii} = k(\mathbf{x}_i, \mathbf{x}_i)$ and $\mathbf{k}_i$ is a column vector, which corresponds to the i-th column of the matrix $\mathbf{K}$.

Now evaluating $\frac{\partial\mathcal{L}}{\partial w_i}$ by differentiating the terms obtained in (9) and (10) with respect to w_i, the following expression is obtained,

$$\frac{\partial\mathcal{L}}{\partial w_i} = -\frac{1}{2}\left[M\left(w_i^{-1} - w_i^{-2}\mathbf{\Sigma}_{ii}\right) - w_i^{-2}\sum_{j=1}^{M}\mu_{ji}^2\right]$$
$$= \frac{1}{2w_i^2}\left[M\mathbf{\Sigma}_{ii} - Mw_i + \sum_{j=1}^{M}\mu_{ji}^2\right], \quad (11)$$

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 $\mu_j = \sigma^{-2}\mathbf{\Sigma}\mathbf{k}_j$. Setting (11) to zero, which means to find the maximum of the function represented by the equation, generates the re-estimation update functions for the weights,

$$w_i^{new} = M^{-1}\sum_{j=1}^{M}\mu_{ji}^2 + \mathbf{\Sigma}_{ii}. \quad (12)$$

According to [13], an equation for re-estimation update that converges faster than (12), can be obtained by rewriting (11) equal to zero such as

$$w_i^{new} = \frac{\sum_{j=1}^{M}\mu_{ji}^2}{M\left(1 + \mathbf{\Sigma}_{ii}/w_i\right)}. \quad (13)$$

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

$$\mathbf{T}^{skpca} = \mathbf{V}_{\mathcal{F}}^T\phi(\mathbf{t}) = \tilde{\mathbf{\Lambda}}_K^{-\frac{1}{2}}\tilde{\mathbf{U}}_K^T\hat{\mathbf{k}}_{\mathbf{t}}^T, \quad (14)$$

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 $\mathbf{X}$.

3. 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 (13) 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 Fig. 3.

The following explanation describes how the frame reduction from $2L$ to L using SKPCA is performed. Firstly, the σ^2 is chosen to be equal to the *L-th* eigenvalue obtained from the cluster of $2L$ frames, then the re-estimation (13) is performed generating the weights w_i for the cluster of $2L$ frames. In practice, the weights describe the order of the most representative vectors that maximize the likelihood considering σ^2. Hence, the L highest values of w_i are selected in order to reduce the cluster size, and consequently the feature vectors corresponding to these weights are saved to the following step, which could be the cluster merging step to continue the full training data reduction or the SKPCA (KPCA block) representation step. The weights are used to select the most representative feature vectors, however they are not included in the following frame reduction step and neither in the SKPCA feature representation

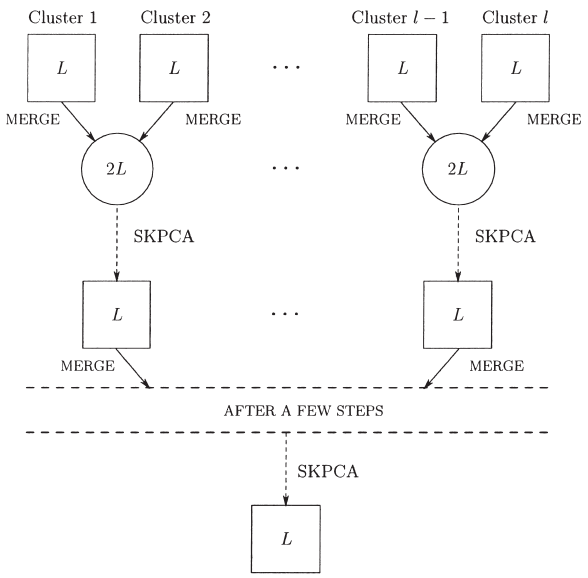


Fig. 3 Proposed approach.

procedure.

The total number of steps necessary to reduce the l clusters of L frames to 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 (13) over the full training database, 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 Sect. 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.

4.1 Settings

The speech signals were obtained with 10 KHz of sampling rate, and 13 mel-cepstral coefficients (12th order mel-cepstral analysis plus the zero-th mel-cepstral term) were extracted from each frame by using 25.6 ms Hamming windows with 10 ms shifts.

The reduced number of training frames N applied to KPCA experiment were 256, 512 and 1024 frames randomly chosen, which is the standard technique to solve the sparsity problem of KPCA [11]. The same numbers of reduced frames (256, 512 and 1024) were applied to the SKPCA with the proposed approach. Beside the two mentioned experiments, it was evaluated the full training data PCA to be compared with the KPCA with first degree polynomial kernel.

All the experiments were based on input vectors formed by the 13 mel-cepstral coefficients plus their Δ and $\Delta\Delta$ features. Each word was modeled by using a 12 state HMM (hidden Markov model) with single mixture of diagonal covariance and the numbers of frames N (256, 512 and 1024) were chosen to reduce the computational burden.

The baseline features were obtained by applying 13 mel-cepstral plus their Δ and $\Delta\Delta$ coefficients to the previously mentioned recognition system. Its performance was 8.36% of error rate.

4.2 PCA Performance

Experiments using PCA extracted features were evaluated in order to make a comparative analysis against KPCA and SKPCA performances.

The results shown in Table 2 agree with the conclusion stated in [11], that the dynamic and acceleration features applied to the output of the feature extraction block are essential to reach higher recognition performance. Although this conclusion was based on KPCA experiments, the PCA results show a similar characteristic. The best PCA performance, 7.70% of error rate was obtained by using dimensionality 13 and wDA (with dynamic and acceleration characteristics). It represents higher recognition accuracy than the best performance without dynamic and acceleration features, 10.05% of error rate (32 dimensions and w/DA column), and the baseline performance (8.36% of error rate). The columns w/DA (without dynamic and acceleration characteristics) and wDA represent the use PCA feature vectors and the PCA feature vectors plus their Δ and $\Delta\Delta$ coefficients, respectively.

Table 2 Error rate (%) of the system using PCA features as the input of the classification block. The PCA output dimensionality is given by the PCA feature vectors only and the PCA feature vectors plus their Δ and $\Delta\Delta$ coefficients, which are represented by the columns w/DA (without dynamic and acceleration features) and wDA (with dynamic and acceleration features), respectively.

PCA		
dim	w/DA	wDA
8	23.33	8.76
13	22.69	**7.70**
16	17.29	7.77
32	**10.05**	10.84
39	11.12	12.68

4.3 KPCA Performance

The KPCA experiment is required due to comparison matters against PCA and SKPCA, because there is no way of evaluating the efficiency of the SKPCA technique, except comparing it with other related approaches.

Table 3 presents the results due to the use of KPCA with first degree polynomial kernel function and N equal to 256, 512 and 1024 frames. The KPCA with first degree polynomial kernel function represents the use of PCA for the reduced training data N. Analyzing the results, it is clearly noticed that the columns wDA show better performances than the columns w/DA, which again shown the importance of applying the dynamic and acceleration features. Also in accordance with the conclusions presented in [12], as the number of frames N increases, the recognition accuracy tends to be better. The best performances for N equal to 256, 512 and 1024 are 7.99%, 8.07% and 7.45% of error rate, respectively, with dimensionality 13 and wDA. All of them overcome the baseline result.

Comparing the results of Table 3 to the ones of Table 2, it is observed that the best PCA performances of both columns w/DA and wDA reached lower error percentages than the best KPCA performances of Table 3, except for the KPCA case with N=1024, 13 dimensions and column wDA. Observing each cell individually and comparing them to the PCA corresponding ones, it is noticed that the KPCA & p=1 overcome the PCA results in 37% of the cases. As

Table 3 Error rate (%) of the system using KPCA with a 1st degree polynomial kernel function and N equal to 256, 512 and 1024. The columns w/DA and wDA represent respectively the feature vectors not using and using Δ and $\Delta\Delta$.

KPCA & p=1						
	N=256		N=512		N=1024	
dim	w/DA	wDA	w/DA	wDA	w/DA	wDA
8	22.16	8.61	22.61	8.43	24.01	8.83
13	22.96	**7.99**	23.19	**8.07**	23.50	**7.45**
16	19.79	8.72	21.09	8.32	17.77	8.19
32	**11.81**	9.94	**11.48**	10.06	**10.16**	10.38
39	12.88	11.77	13.28	12.29	11.26	10.26

Table 4 Error rate (%) of the system using KPCA with a 2nd degree polynomial kernel function and N=256, 512 and 1024. The columns w/DA and wDA represent respectively the feature vectors not using and using Δ and $\Delta\Delta$.

KPCA & p=2						
	N=256		N=512		N=1024	
dim	w/DA	wDA	w/DA	wDA	w/DA	wDA
8	25.91	7.92	24.60	7.88	22.30	7.64
13	25.62	7.48	24.54	7.35	21.72	6.70
16	25.13	7.70	23.44	7.59	19.46	6.85
32	18.44	**7.43**	**17.61**	**7.29**	14.34	**6.53**
39	**18.22**	7.77	17.62	7.85	**11.26**	6.79
64	20.44	9.45	22.01	9.95	17.42	8.96
128	28.60	14.92	31.93	17.66	26.17	16.44

it was expected, the PCA results using the full training data tend to reach higher accuracy than the PCA representatives for the reduced data (KPCA & p=1), because the reduced data does not necessarily hold enough statistic information to well represent the training data set.

Table 4 presents the results due to the use of KPCA with second degree polynomial kernel function and N equal to 256, 512 and 1024 frames. It is noticed that the lowest error rates presented in column wDA are 7.43%, 7.29% and 6.53% for 256, 512 and 1024 randomly chosen frames, respectively, and dimensionality 32. These results overcome the best performances presented in the columns wDA of Table 3 and Table 2, besides the baseline. These results were expected because the KPCA with second degree polynomial kernel function represents the real use of KPCA, and the KPCA with first degree polynomial kernel function is a reduced data representation of PCA. It is clearly observed by comparing the best performances of PCA (Table 2), KPCA with p=1 (Table 3) and baseline against the best performances of KPCA with p=2 (Table 4), that effectively the application of KPCA technique caused a certain improvement in the system accuracy. In addition to the previous analysis, it is expected that the lowest error rates obtained with KPCA with p=2 are not beaten by the KPCA with p=1, the PCA nor the baseline error rates. It also suggests that the following analyses for SKPCA should be focused on the second degree polynomial kernel function.

4.4 SKPCA Performance

The SKPCA is performed by re-estimating w_1^{new} in (13),

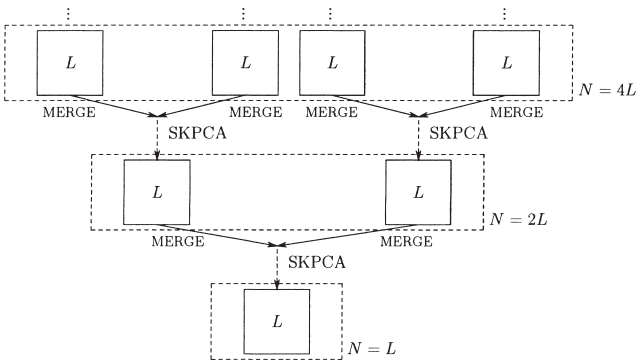


Fig. 4 Proposed approach focusing on the number of frames N.

Table 5 Error rate (%) of the system using SKPCA with a 2nd degree polynomial kernel function and N=256, 512 and 1024. The columns w/DA and wDA represent respectively the feature vectors not using and using Δ and $\Delta\Delta$.

SKPCA & p=2						
	N=256		N=512		N=1024	
dim	w/DA	wDA	w/DA	wDA	w/DA	wDA
8	25.21	8.36	22.55	7.51	22.96	7.23
13	22.05	**6.37**	21.31	**6.30**	22.71	**6.44**
16	21.73	6.42	19.12	6.39	19.90	6.53
32	17.04	7.15	14.00	6.97	14.28	6.59
39	**16.68**	7.42	**13.88**	7.05	**13.90**	7.08
64	19.62	9.02	17.72	8.74	17.21	8.85
128	25.04	13.15	23.98	13.90	25.45	15.91

while $\mathbf{\Sigma}$ and μ_j are updated. The number of re-estimations are constrained to the maximum permitted number of iterations. Then, this process is evaluated successively for all i in w_i^{new}. This re-estimation was motivated by noticing that several w_i^{new} tend to converge to one single value, what could be used to reduce the computational cost associated to the usual SKPCA re-estimation.

The proposed approach in Sect. 3 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 sets for N equal to 512 and 1024 were obtained by merging the final clusters achieved in $(step-1)$ and $(step-2)$ steps. In other words, the data set when N=256 is a subset of the data set when N=512 and both are subsets of the data set when N=1024, as it is shown in Fig. 4, which is a simplified version of Fig. 3 focusing on the number of frames N.

Table 5 shows the results related to the use of SKPCA with the proposed approach and second degree polynomial kernel function for N equal to 256, 512 and 1024 frames. The best performances were 6.37%, 6.30% and 6.44% of error rate for dimensionality 13 and N equal to 256, 512 and 1024, respectively. The best result when N=1024 was worse than the best results when N was equal to 512 and 256. This could be explained by considering the data set with 1024 frames having redundant data information, the performance degradation from N=512 to N=256 could be explained by the elimination of important data information from the data set with 512 frames. The previous explanation is corrobo-

rated by knowing that in the proposed approach a reduced data set with a certain number of frames is a subset of a reduced data set with a greater number of frames. The best performances were significantly superior when compared to the ones presented in Tables 2, 3 and 4, and also when compared to the baseline result. Comparing individually the results of the cells from both columns in the KPCA & p=2 and SKPCA & p=2 cases, it is noticed that the SKPCA overcome the KPCA performances in 81% of the cases.

Considering the computational time, the experiments were performed on an Intel Xeon, 2.8 GHz with 1024 MB of RAM and using Linux operating system and all the routines were implemented in C. The proposed approach can not be compared to the original SKPCA with full training data due to the fact that the SKPCA is not realizable using a large amount of training data, which in this work corresponds to 1,200,000 frames, approximately. However a comparison of the computational cost for the proposed approach using L equal to 256, 512 and 1024 could show the impact of using the clustered training data approach. The mean computational times for 10 clusters considering exclusively the training data reduction routine were approximately 5 min, 70 min and 400 min for L equal to 256, 512 and 1024, respectively. Although the cluster size reduction causes an increase in the number of clusters, the computational time reduction due to this cluster size reduction is more significant than the time increase related to the increment in the number of clusters, which consequently results in an overall computational time reduction, thus showing the advantage of the proposed approach in reducing the computational cost.

4.5 Cross-Validation Experiments

The cross-validation experiment consisted in dividing the test database of the previous experiments into two different data sets, estimation data and test data. The estimation data was used to establish the best features, which are related to the highest recognition performance, and the test data was used to confirm the reliability of these best features.

The previous results show that the best performances depend on the appropriate choices of the dimensionality (with or without dynamic and acceleration characteristics) and the N number of frames concerning the SKPCA with the proposed approach. This experiment is intended to show that these appropriate choices can be obtained by analyzing a certain amount of data (estimation data) and then applying the features of the best performances obtained from the estimation data to the unseen new data (test data). In order to carry out these experiments, the original test database of 60 speakers (31200 utterances) was divided into two data sets of 20 (10400 utterances) and 40 (20800 utterances) speakers, which represent the estimation and the test data, respectively. The results are composed by the average of three different divisions.

The three divisions mentioned above, in fact, mean three different cross-validation like experiments. One experiment consists in using the estimation data of one specific division and observe all the possibilities of N (1024, 512 and 256), all dimensions (8, 13, 16, 32, 39, 64 and 128), searching for the overall higher performance. The features (N and dimensionality) for the best result will be the optimal features obtained by using the estimation data. Applying these same features to the test data, it is obtained the test data result according to the optimal features from the estimation data. This procedure is carried out for the three divisions (three experiments) and the obtained recognition error rates are averaged, in order to add more reliability to the final result.

Considering the estimation data for all the three divisions, the best performances were obtained by using dimensionality 13 and wDA. However two divisions had their best results associated to N=512 frames and one division to N=256 frames.

Concerning the test data, the overall best results were associated to N=512 frames, p=2 and dimensionality 13, for all three divisions. It means that using N=256 for one of the test data sets according to one of the estimation data optimal features does not correspond to the test data best performance, thus this estimation data result could not determine the optimal features for the test data. The average of the test data performances using the optimal features obtained from the estimation data was 6.36% of error rate, and the average of the test data performances using the optimal features obtained from the test data itself was 6.30% of error rate, which is equal to the SKPCA experiment overall best performance, because the division average comprises the full 60 speakers data. A comparison of the average of the test data results obtained from the estimation data optimal features and from optimal features for the test data itself against the baseline, the best PCA result and the best KPCA result, which are shown in Sects. 4.1, 4.2 and 4.3, is presented in Fig. 5. Although in one of the experiments the estimation data could not select the best features for the test data, its performance was quite similar to the best test data performance, which can be confirmed by the slight difference in

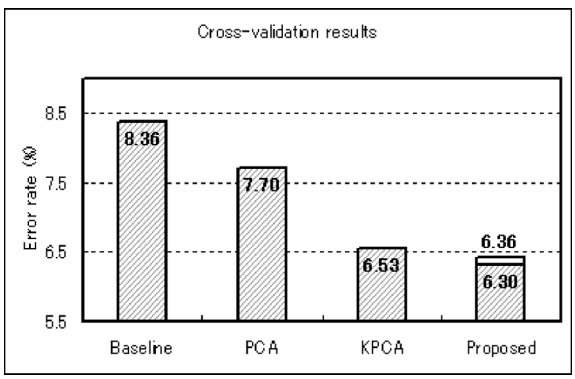


Fig. 5 Summary of the best performances for the cross-validation experiment. The error rates (upper and lower numbers) characterizing the "Proposed" experiment performances are the test data results with the features selected by the optimal features for the estimation data and the test data itself, respectively.

accuracy of the results shown in Fig. 5, column "Proposed."

5. Conclusions

Kernel based techniques applied to a 520 Japanese words recognition task were presented in this paper, and according to the experimental results (Sects. 4.2, 4.3 and 4.4) the effectiveness of SKPCA has been confirmed over the others techniques presented in this work.

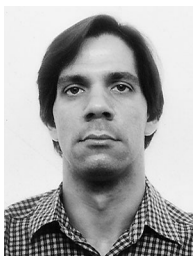
Results showed in Sect. 4.2 and Sect. 4.3 (Table 3) have confirmed the superiority of the PCA using full training data over the reduced data PCA representation obtained from KPCA & p=1. On the other hand, Sect. 4.3 (Table 4) and Sect. 4.4 have demonstrated the importance of using SKPCA technique to improve the recognition accuracy of the system. Also the proposed approach has shown its efficiency through the experimental results.

The error reduction of the overall best performance 6.30% (SKPCA) against the best performances of the others techniques were 3.52%, 15.43%, 18.18% and 24.64%, respectively to the KPCA & p=2 (6.53%), KPCA & p=1 (7.45%), PCA (7.70%) and baseline (8.36%). It is also noted that this work was based on empirical analyses, once there is no theory that supports the superiority of one method over the others.

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 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.

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," J. Acoust. Soc. Am., vol.87, no.4, 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, pp.121–167, 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 Comput., vol.10, no.5, pp.1299–1319, 1998.
[7] B. Schölkopf, A. Smola, and K.R. Müller, "Kernel principal component analysis," Advances in Kernel Methods - Support Vector Learning, ed. B. Schölkopf, C. Burges, and A. Smola, pp.327–352, MIT Press, Cambridge, MA, 1999.
[8] K.I. Kim, S.H. Park, and H.J. Kim, "Kernel principal component analysis for texture classification," IEEE Signal Process. Lett., vol.8, no.2, pp.39–41, 2001.
[9] K.I. Kim, K. Jung, and H.J. Kim, "Face recognition using kernel principal component analysis," IEEE Signal Process. Lett., vol.9, no.2, pp.40–42, 2002.
[10] A. Kocsor, A. Kuba, 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 KPCA for feature extraction in speech recognition," Proc. EuroSpeech, pp.2625–2628, Sept. 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 13, ed. T.K. Leen, T.G. Dietterich, and V. Tresp, pp.633–639, MIT Press, 2001.
[14] A.J. Smola, O.L. Mangasarian, and B. Schölkopf, "Sparse kernel feature analysis," Technical Report, University of Wisconsin, 1999.
[15] M.E. Tipping and C.M. Bishop, "Probabilistic principal component analysis," J. Royal Statistical Society, pp.611–622, 1999.



Amaro Lima received the B.Sc. degree in electrical and electronic engineering and M.Sc. degree in electrical engineering from Federal University of Rio de Janeiro, Rio de Janeiro, Brazil, in 1999 and 2001, respectively. Currently he is a candidate for the Dr.Eng. degree in electrical and computer engineering at Nagoya Institute of Technology, Nagoya, Japan. His research interests include automatic speech recognition and Kernel learning machines.

Heiga Zen received the B.E. degree in computer science and M.E. degree in electrical and computer engineering from Nagoya Institute of Technology, Nagoya, Japan, in 2001 and 2003, respectively. Currently he is a Ph.D student of the Department of Computer Science at the Nagoya Institute of Technology. His research interests include automatic speech recognition and text-to-speech synthesis. He is a member of ASJ.

Yoshihiko Nankaku received the B.E. degree in Computer Science, and the M.E. and Dr.Eng. degrees in the Department of Electrical and Electronic Engineering from the Nagoya Institute of Technology, Nagoya Japan, in 1999, 2001, and 2004 respectively. He is currently a postdoctoral fellow at the Nagoya Institute of Technology. His research interests include image recognition, speech recognition and synthesis and multimodal interface. He is a member of ASJ.

Keiichi Tokuda received the B.E. degree in electrical and electronic engineering from the Nagoya Institute of Technology, Nagoya, Japan, the M.E. and Dr.Eng. degrees in information processing from the Tokyo Institute of Technology, Tokyo, Japan, in 1984, 1986, and 1989, respectively. From 1989 to 1996 he was a Research Associate at the Department of Electronic and Electric Engineering, Tokyo Institute of Technology. From 1996 to 2004 he was a Associate Professor at the Department of Computer Science, Nagoya Institute of Technology as Associate Professor, and now he is a Professor at the same institute. He is also an Invited Researcher at ATR Spoken Language Translation Research Laboratories, Japan and was a Visiting Researcher at Carnegie Mellon University from 2001 to 2002. He is a co-recipient of the Paper Award and the Inose Award both from the IEICE in 2001, and the TELECOM System Technology Prize from the Telecommunications Advancement Foundation Award, Japan, in 2001. He was a member of the Speech Technical Committee of the IEEE Signal Processing Society. His research interests include speech coding, speech synthesis and recognition, and statistical machine learning.

Tadashi Kitamura received the B.E. degree in Electrical and Electronic Engineering from the Nagoya Institute of Technology, and the M.E. and Dr.Eng. degrees from the Tokyo Institute of Technology, in 1973, 1975, and 1978, respectively. He is currently a Professor at Interdisciplinary Graduate School of Science and Engineering, Nagoya Institute of Technology, Nagoya, Japan. His research interests include digital signal processing, speech analysis and synthesis, image synthesis, and multimodal speech recognition. He is a member of IEEE, ISCA, IPSJ and ASJ.

Fernando G. Resende received the B.E. degree from the Military Institute of Engineering, Brazil, in 1990, and the M.Sc. and Ph.D. degrees from Tokyo Institute of Technology, Japan, in 1994 and 1997, respectively, all in eletrical and electronic engineering. Since 1998 he has been with the Department of Electronics and Computer Engineering, Polytechnic School, and since 2003 he has also been with the Program of Electrical Engineering, COPPE, both as Associate Professor at the Federal University of Rio de Janeiro, Brazil. His research interest include speech coding, synthesis and recognition.