7

From Feature Space to Primal Space: KPCA and Its Mixture Model

Haixian Wang
Key Laboratory of Child Development and Learning Science of Ministry of Education, Research Center for Learning Science, Southeast University, Nanjing, Jiangsu 210096, P. R. China

1. Introduction

Kernel principal component analysis (KPCA) (Schölkopf et al., 1998) has proven to be an exceedingly popular technique in the fields of machine learning and pattern recognition, and is discussed at length in literature. KPCA is to perform linear PCA (Hotelling, 1933; Jolliffe, 2002) in a high- (and possibly infinite-) dimensional kernel-defined feature space that is typically induced by a nonlinear mapping. In implementation, the so-called kernel trick is employed. Namely, KPCA is expressed in terms of dot products between the mapped data points, and the dot products are then evaluated by substituting an *a priori* kernel function. KPCA has demonstrated to be an incredibly useful tool for many application areas including handwritten digits recognition and de-noising (Schölkopf et al., 1998; Mika et al., 1999; Schölkopf et al., 1999), nonlinear regression (Rosipal et al., 2001), face recognition (Kim et al., 2002a; Yang, 2002; Kong et al., 2005), and complex image analysis (Kim et al., 2005; Li et al., 2008).

In practice, however, we are often confronted with the situation that needs to process a large number of data points. This raises a problem for KPCA, since KPCA has to store and diagonalize the *kernel matrix* (also known as Gram matrix), whose size is equal to the square of the number of training samples. So, for large scale data set, KPCA would consume large storage space and be computationally intensive (with time complexity $O(n^3)$, a cubic growth with n, where n is the number of the training samples). Then it is impractical for KPCA to be applied in some circumstances. Another attendant problem is that eig-decomposing large matrix directly suffers from the issue of numerical accuracy. Some algorithms have been developed to address the drawbacks associated with KPCA. By considering KPCA from a probabilistic point of view, Rosipal and Girolami (2001) presented an expectation maximization (EM) (Dempster et al., 1977; McLachlan & Krishnan, 1997) method for carrying out KPCA. Their algorithm is of computational complexity $O(pn^2)$ per iteration, where p is the number of extracted components. Whereas the EM algorithm for KPCA does alleviate computational demand, there exists a rotational ambiguity with the algorithm. To remove the obscurity, a constrained EM algorithm for KPCA (and PCA) was formulated based on coupled probability model (Ahn & Oh, 2003). Also, one deficiency of these EM-type algorithms is that the kernel matrix still required to be stored. Kim et al. (2005) then derived the kernel Hebbian algorithm (KHA), which was the counterpart of the generalized Hebbian algorithm (GHA) (Sanger, 1989), to iteratively perform KPCA, where only linear order memory complexity was

involved. However, the price one has to pay for this saving is that the time complexity is not under control. Motivated by the idea "divide and rule", Zheng et el. (2005) proposed another improved algorithm for KPCA as follows. First, the entire data set was divided into some smaller data sets, then the sample covariance matrix of each smaller data set was approximately computed, and finally kernel principal components were extracted by combining these approximate covariance matrices. With their method, the computational demand and memory requirement are effectively relieved. However, the advantages relate with many factors such as the required accuracy of extracted components, the number of the divided smaller data sets (which is usually empirically set), and the data to be processed. As a generic methodology, another thread of speeding up kernel machine learning is to seek a low-rank approximation to the kernel matrix. Since, as noted by several researchers, the spectrum of the kernel matrix tends to decay rapidly, the low-rank approximation often achieves sufficient precision of the requirement. Williams and Seeger (2001) used Nyström method to compute the approximate eigenvalue decomposition of the kernel matrix. Also, Smola and Schölkopf (2000) presented a sparse greedy approximation technique. These two methods yield similar forms and performances.

Another limitation of KPCA is that it defines only a *global* projection of the samples. When the distribution of the data points is complex and non-convex, a global subspace based on KPCA may fail to deliver good performance in terms of feature extraction and recognition. In input space, Tipping and Bishop (1999) and Roweis and Ghahramani (1999) introduced mixture of PCA to remedy the same shortcoming of PCA. Kim et al. (2002b) used mixture-of-eigenfaces for face recognition. There are many other papers on face recognition using mixture method, but as they do not focus on KPCA, references are omitted.

The contributions of this chaper are twofold: Firstly, viewing KPCA as a problem in primal space with the "samples" created by using the incomplete Cholesky decomposition, we show that KPCA is equivalent to performing linear PCA in the primal space using the created samples. So, the same kernel principal components as the standard KPCA are produced. Consequently, all the improved methods dealing with linear PCA (such as the constrained EM algorithm and the GHA method mentioned above), as well as directly diagonalizing the covariance matrix, could be applied to the created samples in the primal space to extract kernel principal components. Theoretical analysis and experimental results on both artificial and real data have shown the superiority of the proposed method for performing KPCA in terms of computational efficiency and storage space, especially when the number of the data points is large. Secondly, we extend KPCA to a mixture of local KPCA models by applying the mixture model of the probabilistic PCA in the primal space. While KPCA uses one set of features to model the data points, the mixture of KPCA uses more than one set of features. Therefore, the mixture of KPCA is expected to represent data more effectively and has better recognition performance than KPCA, which is also confirmed by the experiments.

The remainder of this chaper is organized as follows. The standard KPCA is briefly reviewed in Section 2, and in Section 3, we formulate KPCA in the primal space using the incomplete Cholesky decomposition. Next, we extend KPCA to its mixture model in Section 4. Experimental results are presented in Section 5. In Section 6, we draw the conclusion.

2. Kernel Principal Component Analysis

Suppose $\mathbf{x}_i \in \mathbb{R}^l, i = 1, \ldots, n$, are n observations. The basic idea of KPCA is as follows. First, the samples are mapped into some potentially high- (and possibly infinite-) dimensional *feature*

space $\mathcal{F}$

$$\phi : \mathbb{R}^l \to \mathcal{F},\ \mathbf{x}_i \mapsto \phi(\mathbf{x}_i),\ (i = 1,\ldots,n) \tag{1}$$

where ϕ is a typically nonlinear function. Then a standard linear PCA is performed in $\mathcal{F}$ using the mapped samples. In evaluation, we don't have to compute the mapping ϕ explicitly. The mapped samples occur in the forms of dot products, say between $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$, which are computed by choosing a *kernel function* k:

$$k(\mathbf{x}_i,\mathbf{x}_j) = (\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)). \tag{2}$$

The mapping ϕ into $\mathcal{F}$ such that (2) stands exists if k is a positive definite kernel, thanks to Mercer's theorem of functional analysis. So, the mapping ϕ and thus $\mathcal{F}$ are fixed implicitly via the function k. The dth-order polynomial kernel, $k(\mathbf{x}_i,\mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)^d$, Gaussian kernel with width $\sigma > 0$, $k(\mathbf{x}_i,\mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2)$, and sigmoid kernel $k(\mathbf{x}_i,\mathbf{x}_j) = \tanh(a(\mathbf{x}_i \cdot \mathbf{x}_j) + b)$ are commonly used Mercer kernels.

For notation simplicity, the mapped samples are assumed to be centered, i.e. $\sum_{i=1}^n \phi(\mathbf{x}_i) = 0$. We wish to find eigenvalues $\lambda > 0$ and associated eigenvectors $\mathbf{v} \in \mathcal{F} \setminus \{0\}$ of the covariance matrix of the mapped samples $\phi(\mathbf{x}_i)$, given by

$$\mathbf{C}^\phi = \frac{1}{n}\sum_{i=1}^n \phi(\mathbf{x}_i)\phi(\mathbf{x}_i)^\mathsf{T}, \tag{3}$$

where T denotes the transpose of a vector or matrix. Since the mapping ϕ is implicit or $\mathbf{C}^\phi$ is very high dimensional, direct eigenvalue decomposition will be intractable. The difficulty is circumvented by using the so-called kernel trick; that is, linear PCA in $\mathcal{F}$ is formulated such that all the occurrences of ϕ are in the forms of dot products. And the dot products are then replaced by the kernel function k. So, dealing with the ϕ-mapped data explicitly is avoided. Specifically, since $\lambda\mathbf{v} = \mathbf{C}^\phi\mathbf{v}$, all solutions $\mathbf{v}$ with $\lambda \neq 0$ fall in the subspace spanned by $\{\phi(\mathbf{x}_1),\ldots,\phi(\mathbf{x}_n)\}$. Therefore, $\mathbf{v}$ could be linearly represented by $\phi(\mathbf{x}_i)$:

$$\mathbf{v} = \sum_{i=1}^n \alpha_i\phi(\mathbf{x}_i), \tag{4}$$

where $\alpha_i\ (i = 1,\ldots,n)$ are coefficients. The eigenvalue problem is then reduced as the following equivalent problem

$$\lambda(\phi(\mathbf{x}_j) \cdot \mathbf{v}) = (\phi(\mathbf{x}_j) \cdot \mathbf{C}^\phi\mathbf{v}) \text{ for all } j = 1,\ldots,n. \tag{5}$$

Substituting (3) and (4) into (5), we arrive at the eigenvalue equation

$$n\lambda\mathbf{K}\alpha = \mathbf{K}^2\alpha \Rightarrow n\lambda\alpha = \mathbf{K}\alpha, \tag{6}$$

where α denotes a column vector with entries $\alpha_1,\ldots,\alpha_n$, and $\mathbf{K}$, called kernel matrix, is an $n \times n$ matrix with elements defined as

$$\mathbf{K}_{ij} = k(\mathbf{x}_i,\mathbf{x}_j) = (\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)). \tag{7}$$

Assume that $\mathbf{t}$ is a testing point whose ϕ-image is $\phi(\mathbf{t})$ in $\mathcal{F}$. We calculate its kernel principal components by projecting $\phi(\mathbf{t})$ onto the kth eigenvectors $\mathbf{v}^k$. Specifically, the kth kernel principal components corresponding ϕ are

$$(\mathbf{v}^k \cdot \phi(\mathbf{t})) = \sum_{i=1}^n \alpha_i^k(\phi(\mathbf{x}_i) \cdot \phi(\mathbf{t})) = \sum_{i=1}^n \alpha_i^k k(\mathbf{x}_i,\mathbf{t}), \tag{8}$$

where α^k has been normalized such that $\lambda_k(\alpha^k \cdot \alpha^k) = 1$. Note that centering the vectors $\phi(\mathbf{x}_i)$ and $\mathbf{t}$ in $\mathcal{F}$ is realized by centering the corresponding kernel matrices (Schölkopf et al., 1998).

3. Kernel Principal Component Analysis in Primal Space

In this section, we will derive KPCA in the primal space with the samples created by using the incomplete Cholesky decomposition. Let $\phi(\mathbf{X}) = [\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n)]$ be the data matrix containing all the ϕ-mapped training samples as columns. By partial Gram-Schmidt orthonormalization (Cristianini et al., 2002), we factorize the data matrix $\phi(\mathbf{X})$ as

$$\phi(\mathbf{X}) = \mathbf{QR}, \tag{9}$$

where $\mathbf{Q}$ has m orthonormal columns, $\mathbf{R} \in \mathbb{R}^{m \times n}$ is an upper triangular matrix with positive diagonal elements, and m is the rank of $\phi(\mathbf{X})$. Note that the matrix $\mathbf{R}$ could be evaluated row by row without computing ϕ explicitly. The partial Gram-Schmidt procedure pivots the samples and selects the linearly independent samples in the feature space $\mathcal{F}$. The orthogonalized version of the selected independent samples, i.e. the columns of $\mathbf{Q}$, is thus used as a set of basis. All ϕ-mapped data points could be linearly represented using the basis. Specifically, the ith column of the matrix $\mathbf{R}$ are the coefficients for the linear representation of $\phi(\mathbf{x}_i)$ using the columns of $\mathbf{Q}$ as the basis. So, the columns of $\mathbf{R}$ are, in fact, the new coordinates in the feature space of the corresponding data points of $\phi(\mathbf{X})$ using the basis. By (9), the following decomposition of the kernel matrix is yielded:

$$\mathbf{K} = \phi(\mathbf{X})^\mathsf{T}\phi(\mathbf{X}) = \mathbf{R}^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{QR} = \mathbf{R}^\mathsf{T}\mathbf{R}, \tag{10}$$

which is the incomplete Cholesky decomposition (Fine & Scheinberg, 2001; Bach & Jordan, 2002).

From (10), if defining a new mapping

$$\tilde{\phi} : \mathcal{F} \to \mathbb{R}^m,\ \phi(\mathbf{x}_i) \mapsto \mathbf{r}_i,\ (i = 1, \ldots, n) \tag{11}$$

where $\mathbf{r}_i$ is the ith column of $\mathbf{R}$, then the n vectors $\{\mathbf{r}_1, \ldots, \mathbf{r}_n\}$ give rise to the same Gram matrix $\mathbf{K}$ (Shawe-Taylor & Cristianini, 2004); that is

$$(\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)) = k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{r}_i \cdot \mathbf{r}_j). \tag{12}$$

The space $\mathbb{R}^m$ is referred to as the *primal space*, and $\mathbf{r}_i$ $(i = 1, \ldots, n)$ are viewed as "samples". From (9), we see that, if $\phi(\mathbf{x}_i)$ are centered, then $\mathbf{r}_i$ are also centered. In other words, $\mathbf{r}_i$ could be centered by centering the kernel matrix $\mathbf{K}$. By using the samples $\mathbf{r}_i$ created in the primal space $\mathbb{R}^m$, we have the following

Theorem 1. *Given observations $\boldsymbol{x}_i$ $(i = 1, \ldots, n)$ and kernel function k, KPCA is equivalent to performing linear PCA in the primal space $\mathbb{R}^m$ using the created samples $\boldsymbol{r}_i$ $(i = 1, \ldots, n)$, both of which produce the same kernel principal components.*

Proof. It suffices to note that the dot products between the ϕ-mapped samples in the feature space $\mathcal{F}$ are the same with that between the corresponding $\tilde{\phi}$-mapped samples in the primal space $\mathbb{R}^m$, and linear PCA in both the feature space $\mathcal{F}$ (i.e., KPCA) and the primal space $\mathbb{R}^m$ could be represented through the forms of dot products between samples. The equivalence between KPCA and linear PCA in the primal space is schematically illustrated as follows:

$$\boxed{\begin{array}{c}\text{diagonalize Gram matrix } \mathbf{K} = \phi(\mathbf{X})^{\mathsf{T}}\phi(\mathbf{X}) \\ \Updownarrow \text{ through dot products} \\ \text{KPCA, i.e., linear PCA in } \mathcal{F} \text{ with samples } \phi(\mathbf{x}_i)\end{array}} \quad \equiv \quad \boxed{\begin{array}{c}\text{diagonalize Gram matrix } \mathbf{K} = \mathbf{R}^{\mathsf{T}}\mathbf{R} \\ \Updownarrow \text{ through dot products} \\ \text{linear PCA in } \mathbb{R}^m \text{ with samples } \mathbf{r}_i\end{array}}$$

The theorem is thus established. □

In the primal space, we could, of course, carry out linear PCA by using the dual expression of dot products; but this obviously makes the motivation for creating the new samples in the primal space useless. Considering the dimension of the primal space, m, is small, we perform linear PCA by directly diagonalizing the $m \times m$ covariance matrix of the $\tilde{\phi}$-mapped data points $\mathbf{r}_i$, given by

$$\mathbf{C}^{\tilde{\phi}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{r}_i \mathbf{r}_i^{\mathsf{T}} = \frac{1}{n}\mathbf{R}\mathbf{R}^{\mathsf{T}}. \tag{13}$$

Let $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_m$ be the eigenvalues of $\mathbf{C}^{\tilde{\phi}}$, and $\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_m$ the corresponding eigenvectors. We proceed to compute the kernel principal components of the testing point $\mathbf{t}$. Firstly, we need to compute its $\tilde{\phi}$-image. We carry out the projections of $\phi(\mathbf{t})$ onto the basis vectors, i.e., the columns of $\mathbf{Q}$. This is achieved by calculating an extensional column of the matrix $\mathbf{R}$ in the partial Gram-Schmidt procedure (Shawe-Taylor & Cristianini, 2004). The kernel principal components corresponding to ϕ are then computed as

$$(\tilde{\mathbf{v}}_k \cdot \bar{\mathbf{r}}) \tag{14}$$

for $k = 1, \ldots, p$, where $\bar{\mathbf{r}}$ is the $\tilde{\phi}$-image of $\phi(\mathbf{t})$ and p ($\leq m$) is the number of components.

Alternatively, all improved methods dealing with linear PCA could be applied to $\mathbf{R}$ in the primal space. For example, with the constrained EM algorithm for PCA (Ahn & Oh, 2003), we obtain iterative formula

$$\text{E step:} \qquad \mathbf{Z} = (\mathcal{L}(\Gamma^{\mathsf{T}}\Gamma))^{-1}\Gamma^{\mathsf{T}}\mathbf{R}, \tag{15}$$

$$\text{M step:} \qquad \Gamma^{\text{new}} = \mathbf{R}\mathbf{Z}^{\mathsf{T}}(\mathcal{U}(\mathbf{Z}\mathbf{Z}^{\mathsf{T}}))^{-1}, \tag{16}$$

where the element-wise lower operator $\mathcal{L}$ is defined such as $\mathcal{L}(w_{st}) = w_{st}$ for $s \geq t$ and is zero otherwise, the upper operator $\mathcal{U}$ is defined such as $\mathcal{U}(w_{st}) = w_{st}$ for $s \leq t$ and is zero otherwise, and $\mathbf{Z}$ denotes the $p \times n$ matrix of latent variables. The matrix Γ at convergence is equal to $\Gamma = \mathbf{V}\Lambda$, where the columns of $\mathbf{V} = [\tilde{\mathbf{v}}_1, \ldots, \tilde{\mathbf{v}}_p]$ are the first p eigenvectors of $\mathbf{C}^{\tilde{\phi}}$, with corresponding eigenvalues $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_p$ forming the diagonal matrix Λ. Another extensively used iterative method for PCA is the generalized Hebbian algorithm (GHA) (Sanger, 1989). Based on GHA, the $m \times p$ eigenvectors matrix $\mathbf{V}$ corresponding to the p largest eigenvalues is updated according to the rule

$$\mathbf{V}(t+1) = \mathbf{V}(t) + \delta(t)\left(\mathbf{r}(t)\mathbf{y}(t)^{\mathsf{T}} - \mathbf{V}(t)\mathcal{U}(\mathbf{y}(t)\mathbf{y}(t)^{\mathsf{T}})\right), \tag{17}$$

where $\mathbf{y} = \mathbf{V}^{\mathsf{T}}\mathbf{r}$ is the principal component of $\mathbf{r}$, $\delta(t)$ is a learning rate parameter. Here, the argument t denotes a discrete time when a sample $\mathbf{r}(t)$ is selected randomly from all the samples $\mathbf{r}_i$. It has been shown by Sanger (1989) that, for proper setting of learning rate $\delta(t)$ and initialization $\mathbf{V}(0)$, the columns of $\mathbf{V}$ converge to the eigenvectors of $\mathbf{C}^{\tilde{\phi}}$ as t tends to infinity.

In summary, the procedure of the proposed algorithm for performing KPCA is outlined as follows:

1. Perform the incomplete Cholesky decomposition for the training points as well as testing points to obtain $\mathbf{R}$ and $\bar{\mathbf{r}}$;
2. Compute the p leading eigenvectors corresponding to the first p largest eigenvalues of the covariance matrix $\mathbf{C}^{\tilde{\phi}}$ (defined in (13)) by (a) directly diagonalizing $\mathbf{C}^{\tilde{\phi}}$, (b) using the constrained EM algorithm (according to (15) and (16)), or (c) using the GHA algorithm (according to (17));
3. Extract kernel principal components by projecting each testing point onto the eigenvectors (according to (14)).

Complexity analysis. The computational complexity of performing the incomplete Cholesky decomposition is of the order $O(m^2n)$, and the storage requirement is $O(mn)$. Next, if one explicitly evaluates the covariance matrix $\mathbf{C}^{\tilde{\phi}}$ followed by diagonalization to obtain the eigenvectors, the computational and storage complexity are $O(m^2n + m^3)$ and $O((p+m)m)$, respectively. When $p \ll m$, it is possible to obtain computational savings by using the constrained EM algorithm, the time and storage complexity of which are respectively $O(pmn)$ per iteration and $O(p(m+n))$. If using the GHA method, the time complexity is $O(p^2m)$ per iteration and storage complexity $O(pm)$. The potential efficiency gains, nevertheless, depend on the number of iterations needed to reach the required precision and the ratio of m to p. As one has seen, the proposed method do not need to store the kernel matrix $\mathbf{K}$, the storage of which is $O(n^2)$, and its computational complexity compares favorably with that of the traditional KPCA, which scales as $O(n^3)$.

4. Mixture of Kernel Principal Component Analysis Models

Single KPCA provides only a globally linear model for data representation in a low dimensional subspace. It may be insufficient to model (heterogeneous) data with large variation. One remedy method is to model the complex data with a mixture of local linear sub-models. Considering KPCA in its equivalent form in the primal space, and using the mixture model in the primal space (Tipping & Bishop, 1999), we extend KPCA to a mixture of local KPCA models. While KPCA uses one set of features for the data points, the mixture of KPCA uses more than one set of features. Mathematically, in the primal space, we suppose that $\mathbf{r}_1, \ldots, \mathbf{r}_n$ are generated independently from a mixture of g underlying populations with unknown proportion $\pi_1, \ldots, \pi_g$

$$\mathbf{r}_i = \mu_j + \Gamma_j \mathbf{z}_{ij} + \varepsilon_{ij}, \text{with probability } \pi_j, (j = 1, \ldots, g; i = 1, \ldots, m) \tag{18}$$

where μ_j is a m-dimensional non-random vector, the $m \times p$ matrix Γ_j is known as the *factor loading matrix*, $\mathbf{z}_{ij}$ are p-dimensional latent (unobservable) variables (also known as *common factors*), ε_{ij} are m-dimensional error variables, and π_j is the corresponding mixing proportion with $\pi_j > 0$ and $\sum_{j=1}^{g} \pi_j = 1$. The generative model (18) assumes that $\mathbf{z}_{1j}, \ldots, \mathbf{z}_{nj}$ are independently and identically distributed (i.i.d.) as Gaussian with zero mean and identity covariance, $\varepsilon_{1j}, \ldots, \varepsilon_{nj}$ i.i.d. as Gaussian with mean zero and covariance matrix $\sigma_j^2 I_m$, and $\mathbf{z}_{ij}$ is independent with ε_{ij} and their joint distribution is Gaussian.
In the case of $g = 1$, it was shown by Tipping and Bishop (1999) and Roweis and Ghahramani (1999) that, as the noise level σ_1^2 becomes infinitesimal, the PCA model in the primal space was recovered, which just is KPCA as shown in Section 3. So, model (18) is an extension to KPCA. We refer to (18) as mixture of KPCA (MKPCA). Note that a separate mean vector μ_j is

associated with each component of the g mixture model, therefore allowing each component to model the data covariance structure in different regions of the primal space.
Since the true classification of $\mathbf{r}_i$ into components are unknown, we use the marginal probability density function (p.d.f.) that is a g-component Gaussian mixture p.d.f.

$$f(\mathbf{r}_i;\Theta) = \sum_{j=1}^{g} \pi_j \varphi(\mathbf{r}_i;\mu_j,\Sigma_j) \tag{19}$$

for the observations, where $\varphi(\mathbf{r}_i;\mu_j,\Sigma_j)$ is the Gaussian p.d.f. with mean μ_j and variance $\Sigma_j = \Gamma_j\Gamma_j^{\mathsf{T}} + \sigma_j^2 I_m$, the model parameters are given by $\Theta = (\mu_1,\ldots,\mu_g,\Gamma_1,\ldots,\Gamma_g,\sigma_1^2,\ldots,\sigma_g^2,\pi_1,\ldots,\pi_g)$. For log-likelihood maximization, the model parameters could be estimated via the EM algorithm as follows (Tipping & Bishop, 1999). The E-step is

$$\hat{z}_{ij}^{(k)} = \frac{\hat{\pi}_j^{(k)} \varphi(\mathbf{r}_i;\hat{\mu}_j^{(k)},\hat{\Sigma}_j^{(k)})}{\sum_{i=1}^{g} \hat{\pi}_i^{(k)} \varphi(\mathbf{r}_i;\hat{\mu}_i^{(k)},\hat{\Sigma}_i^{(k)})}. \tag{20}$$

In the M-step, the parameters are updated according to

$$\hat{\pi}_j^{(k+1)} = \frac{1}{n}\sum_{i=1}^{n} \hat{z}_{ij}^{(k)}, \tag{21}$$

$$\hat{\mu}_j^{(k+1)} = \frac{\sum_{i=1}^{n} \hat{z}_{ij}^{(k)} \mathbf{r}_i}{\sum_{i=1}^{n} \hat{z}_{ij}^{(k)}}, \tag{22}$$

$$\hat{\Gamma}_j^{(k+1)} = \hat{\mathbf{S}}_j^{(k+1/2)} \hat{\Gamma}_j^{(k)} \left(\hat{\sigma}_j^{2(k)} I_m + (\hat{\sigma}_j^{2(k)} I_p + \hat{\Gamma}_j^{(k)\mathsf{T}} \hat{\Gamma}_j^{(k)})^{-1} \hat{\Gamma}_j^{(k)\mathsf{T}} \hat{\mathbf{S}}_j^{(k+1/2)} \hat{\Gamma}_j^{(k)} \right)^{-1}, \tag{23}$$

$$\hat{\sigma}_j^{2(k+1)} = \frac{1}{m} \mathrm{tr}\left(\hat{\mathbf{S}}_j^{(k+1/2)} - \hat{\mathbf{S}}_j^{(k+1/2)} \hat{\Gamma}_j^{(k)} (\hat{\sigma}_j^{2(k)} I_p + \hat{\Gamma}_j^{(k)\mathsf{T}} \hat{\Gamma}_j^{(k)})^{-1} \hat{\Gamma}_j^{(k+1)\mathsf{T}} \right), \tag{24}$$

where

$$\hat{\mathbf{S}}_j^{(k+1/2)} = \frac{1}{n\hat{\pi}_j^{(k+1)}} \sum_{i=1}^{n} \hat{z}_{ij}^{(k)} (\mathbf{r}_i - \hat{\mu}_j^{(k+1)})(\mathbf{r}_i - \hat{\mu}_j^{(k+1)})^{\mathsf{T}}. \tag{25}$$

In the case $\sigma_j^2 \to 0$, the converged $\hat{\Gamma}_j$ contains the (scaled) eigenvectors of the local covariance matrix $\hat{\mathbf{S}}_j$. Each component performs a local PCA weighted by the mixing proportion. And the $\hat{\Gamma}_j$ in the limit case are updated as

$$\hat{\Gamma}_j^{(k+1)} = \hat{\mathbf{R}}_j^{(k+1/2)} \hat{\mathbf{Z}}_j^{(k)\mathsf{T}} (\mathcal{U}(\hat{\mathbf{Z}}_j^{(k)} \hat{\mathbf{Z}}_j^{(k)\mathsf{T}}))^{-1}, \tag{26}$$

where

$$\hat{\mathbf{Z}}_j^{(k)} = (\mathcal{L}(\hat{\Gamma}_j^{(k)\mathsf{T}} \hat{\Gamma}_j^{(k)}))^{-1} \hat{\Gamma}_j^{(k)\mathsf{T}} \hat{\mathbf{R}}_j^{(k+1/2)},$$

and

$$\hat{\mathbf{R}}_j^{(k+1/2)} = \left(\sqrt{\frac{\hat{z}_{1j}^{(k)}}{n\hat{\pi}_j^{(k+1)}}} (\mathbf{r}_1 - \hat{\mu}_j^{(k+1)}), \ldots, \sqrt{\frac{\hat{z}_{nj}^{(k)}}{n\hat{\pi}_j^{(k+1)}}} (\mathbf{r}_n - \hat{\mu}_j^{(k+1)}) \right).$$

Methods for performing KPCA	Size of matrix needed to be diagonalized	Training time (seconds)	Storage space (M bytes)
Standard KPCA	2000×2000	293	90
Proposed KPCA	233×233	36	2
MKPCA	N.A.	63	5

Table 1. Comparison of training time and storage space on the toy example with 2000 data points.

Number of features	Standard KPCA			Proposed KPCA			MKPCA		
	d=2	3	4	d=2	3	4	d=2	3	4
32	93.6	93.6	92.5	93.5	93.6	92.4	94.5	94.3	93.5
64	94.3	93.1	93.0	94.2	93.0	93.0	**95.2**	94.2	**94.1**
128	94.4	93.7	**93.2**	**94.4**	93.5	**93.1**	95.0	94.1	94.0
256	**94.5**	93.5	92.9	94.4	**93.8**	92.9	94.9	**94.4**	94.0
512	94.3	**93.9**	92.8	N.A.	93.8	92.8	N.A.	94.1	93.6
1024	94.5	93.9	92.4	N.A.	N.A.	92.4	N.A.	N.A.	93.1

Table 2. Recognition rates of the 2007 testing points of the USPS handwritten digit database using the standard KPCA, proposed KPCA and MKPCA methods with polynomial kernel of degree two through four.

When the noise level becomes infinitesimal, the component p.d.f. $\varphi(\mathbf{r}_i; \hat{\mu}_i^{(k)}, \hat{\Sigma}_i^{(k)})$ in (20) is singular. It, in probability 1, falls in the p-dimensional subspace $span\{\hat{\Gamma}_j^{(k)}\}$, i.e.,

$$\varphi(\mathbf{r}_i; \hat{\mu}_i^{(k)}, \hat{\Sigma}_i^{(k)}) = (2\pi)^{-p/2} \left(\det(\hat{\Gamma}_j^{(k)\mathsf{T}} \hat{\Gamma}_j^{(k)}) \right)^{-1/2} \exp\left(-\hat{\mathbf{a}}^{(k)\mathsf{T}} \hat{\mathbf{a}}^{(k)} / 2 \right), \tag{27}$$

where $\hat{\mathbf{a}}^{(k)} = (\hat{\Gamma}_j^{(k)\mathsf{T}} \hat{\Gamma}_j^{(k)})^{-1} \hat{\Gamma}_j^{(k)\mathsf{T}} (\mathbf{r}_i - \hat{\mu}_j^{(k)})$.
As can be seen, by applying the KPCA mixture model, all the observations are softly divided into g clusters each modelled by a local KPCA. We use the most appropriate local KPCA for a given observation. Based on the probabilistic framework, a natural choice is to assign the observation to the cluster belong to which its posterior probability is the largest.

5. Experiments

In this section, we will use both artificial and real data sets to compare the performance of the proposed method with that of the standard KPCA (Schölkopf et al., 1998). In the first example, we use toy data to visually compare the results by projecting testing points onto extracted principal axes, and to show that the proposed method is superior to the standard KPCA in terms of time and storage complexity. In the second example, we perform the experiment of handwritten digital character recognition to further illustrate the effectiveness of the proposed method. All these experiments are run with the settings of 3.06GHz CPU and 3.62GB RAM using *Matlab* software.

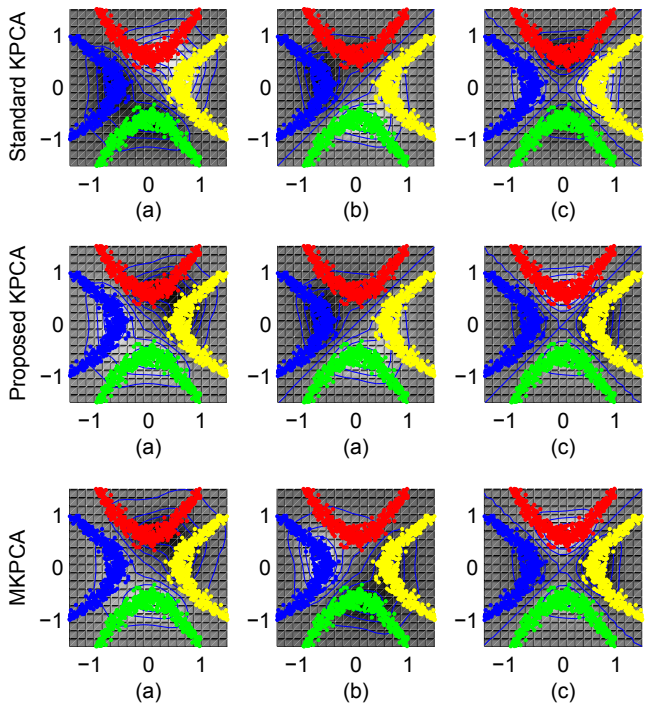


Fig. 1. From left to right, the first three kernel principal components extracted by the standard KPCA (top), proposed KPCA (middle), and MKPCA method with $g = 2$ (bottom), respectively, using the Gaussian kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/0.1)$. The feature values are illustrated by shading and constant values are connected by contour lines.

5.1 Toy Example

The data are generated from four two-dimensional parabolic shape clusters that are vertically and horizontally mirrored by the function $y = x^2 + \epsilon$, where x-values are uniformly distributed within $[-1, 1]$ and ϵ is Gaussian distributed noise having mean 0.6 and standard deviation 0.1. We generate 500 data points for each parabolic shape, and use the Gaussian kernel with $\sigma^2 = 0.05$ through the experiment. The standard KPCA, proposed KPCA (by, in this experiment, using the incomplete Cholesky decomposition followed by directly diagonalizing $\mathbf{C}^{\tilde{\phi}}$), and MKPCA methods are adopted to calculate the leading principal components of the data set. In using the two proposed methods, we choose 233 linearly independent samples from the entire 2000 samples, i.e., $m = 233$, during the incomplete Cholesky decomposition, and set $g = 2$ for performing the MKPCA. Therefore, the proposed KPCA method consumes much less time and storage space than that of the standard KPCA (to see table 1 for detailed comparison). In Fig. 1, we depict the first three kernel principal components extracted by the three methods. The features are indicated by shading and constant feature values are con-

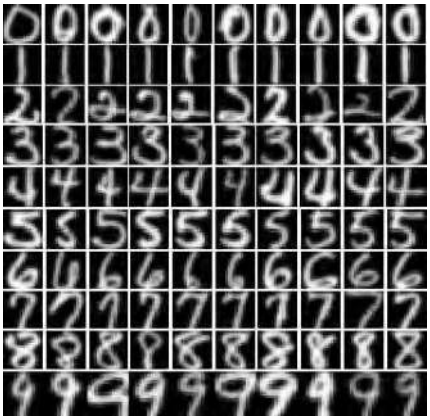


Fig. 2. Some digit images from the USPS database.

Methods for performing KPCA	Size of matrix needed to be diagonalized	Training time (seconds)	Storage space (M bytes)
Standard KPCA	5000×5000	6112	457
Proposed KPCA ($d = 2$)	371×371	114	41
Proposed KPCA ($d = 3$)	956×956	342	69
Proposed KPCA ($d = 4$)	1639×1639	634	118
MKPCA ($d = 2$)	N.A.	321	68
MKPCA ($d = 3$)	N.A.	901	115
MKPCA ($d = 4$)	N.A.	2213	304

Table 3. Comparison of training time and storage space on the USPS handwritten digit database with 5000 training points.

nected by contour lines. From Fig. 1, we see that the proposed KPCA method obtains almost the same results with that of the standard KPCA method (ignoring the sign difference), and both methods nicely separate the four clusters. For the data points, the average relative deviation of the principal components found by the standard KPCA (by diagonalizing the kernel matrix $\mathbf{K}$) and by the proposed KPCA is less than 0.01. In this simple simulated experiment, the toy data are compact in the feature space, and are well modelled by the KPCA method. So, the MKPCA method doesn't show its advantage; in fact, most of the toy data belong to one component of the MKPCA model, since the estimated mixing proportions $\hat{\pi}_1 = 0.9645$ and $\hat{\pi}_2 = 0.0355$. As a result, the MKPCA method produces the similar result with that of the KPCA method.

5.2 Handwritten Digit Character Recognition

In the second example, we consider the recognition problem of handwritten digital character. The experiment is performed on the US Postal Service (USPS) handwritten digits database that are collected from mail envelopes in Buffalo, New York. The database contains 7291 training samples and 2007 testing samples for 10 numeral classes with dimensionality 256 (Schölkopf et al., 1998). Some digit samples are shown in Fig. 2. With this database, 5000 training points

are chosen as training data and all the 2007 testing points are used as testing data. The polynomial kernel with various degree d is utilized in each trial to compute the kernel function. We employ the *nearest neighbor classifier* for classification role. In using MKPCA, we set $g = 2$. The recognition rates obtained by the three approaches are reported in Table 2, while the training times and storage spaces consumed are listed in Table 3. From Table 2, we see that the MKPCA method achieves the best recognition rate among the three systems. The standard KPCA and the proposed approach to performing KPCA have similar recognition rates. Nevertheless, the proposed KPCA reduces the time and storage complexity significantly.

6. Conclusion

We have presented an improved algorithm for performing KPCA especially when the size of training samples is large. This is achieved by viewing KPCA as a primal space problem with the "samples" produced via the incomplete Cholesky decomposition. Since the spectrum of the kernel matrix tends to decay rapidly, the incomplete Cholesky decomposition, as an elegant low-rank approximation to the kernel matrix, arrives at sufficient accuracy. Compared with the standard KPCA method, the proposed KPCA method reduces the time and storage requirement significantly for the case of large scale data set.
In order to provide a locally linear model for the data projection onto a low dimensional subspace, we extend KPCA to a mixture of local KPCA models by applying mixture model of PCA in the primal space. MKPCA supplies an alternative choice to model data with large variation. The mixture model outperforms the standard KPCA in terms of recognition rate. The methodology introduced in this chaper could be applied to other kernel-based algorithms, provided the algorithm could be expressed through dot products.

Acknowledgement

This work was supported by Specialized Research Fund for the Doctoral Program of Higher Education of China under grant 20070286030, and National Natural Science Foundation of China under grants 60803059 and 10871001.

7. References

Ahn, J. H. & Oh, J. H. (2003). A constrained EM algorithm for principal component analysis, *Neural Computation* 15: 57–65.

Bach, F. R. & Jordan, M. I. (2002). Kernel independent component analysis, *Journal of Machine Learning Research* 3: 1–48.

Cristianini, N., Lodhi, H. & Shawe-Taylor, J. (2002). Latent semantic kernels, *Journal of Intelligent Information System* 18: 127–152.

Dempster, A. P., Laird, N. M. & Rubin, D. B. (1977). Maximum likelihood from incomplete data using the EM algorithm (with discussion), *J. R. Statist. Soc. Ser. B.* 39: 1–38.

Fine, S. & Scheinberg, K. (2001). Efficient SVM training using low-rank kernel representation, *Technical Report RC 21911*, IBM T.J. Watson Research Center.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components, *J. Educat. Psychology* 24: 417–441.

Jolliffe I.T. (2002). *Principal Component Analysis,* Second edition, Springer-Verlag, New York.

Kim, K. I., Franz, M. O. & Schölkopf, B. (2005). Iterative kernel principal component analysis for image modelling, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27: 1351–1366.

Kim, K. I., Jung, K. & Kim, H. J. (2002a). Face recognition using kernel principal component analysis, *IEEE Signal Processing Letters* 19: 40–42.

Kim, H. C., Kim, D. & Bang, S. Y. (2002b). Face recognition using the mixture-of-eigenfaces method, *Pattern Recognition Letters* 23: 1549–1558.

Kong, H., Wang, L., Teoh, E. K., Li, X., Wang, J. G. & Venkateswarlu, R. (2005). Generalized 2D principal component analysis for face image representation and recognition, *Neural Networks* 18: 585–594.

Li, J., Li, M. L. & Tao, D. C. (2008). KPCA for semantic object extraction in images, *Pattern Recognition* 41: 3244-3250.

McLachlan, G. J. & Krishnan, T. (1997). *The EM Algorithm and Extensions*, Wiley, New York.

Mika, S., Schölkopf, B., Smola, A. J., Müller, K. R., Scholz, M. & Rätsch, G. (1999). Kernel PCA and de-Noising in feature spaces, *in* M. S. Kearns, S. A. Solla & D. A. Cohn (eds.), *Advances in Neural Information Processing Systems*, Vol. 11, MIT Press, Cambridge, Mass., pp. 536–542.

Rosipal, R. & Girolami, M. (2001). An expectation-maximization approach to nonlinear component analysis, *Neural Computation* 13: 505–510.

Rosipal, R., Girolami, M., Trejo, L. J. & Cichocki, A. (2001). Kernel PCA for feature extraction and de-noising in nonlinear regression, *Neural Computing & Applications* 10: 231–243.

Roweis, S. & Ghahramani, Z. (1999). A unifying review of linear gaussian models, *Neural Computation* 11: 305–345.

Sanger, T. D. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network, *Neural Networks* 2: 459–473.

Schölkopf, B., Mika, S., Burges, C., Knirsch, P., Müller, K.-R., Rätsch, G. & Smola, A. J. (1999). Input space vs. feature space in kernel-based methods, *IEEE Transactions on Neural Networks* 10: 1000–1017.

Schölkopf, B., Smola, A. & Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue prolem, *Neural Computation* 10: 1299–1319.

Shawe-Taylor, J. & Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*, Cambridge University Press, England.

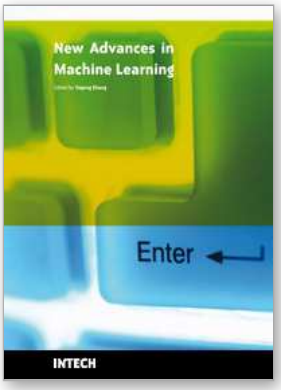
Smola, A. J. & Schölkopf, B. (2000). Sparse greedy matrix approximation for machine learning, *Proceeding of the Seventeenth International Conference on Mchine Learning*, Morgan Kaufmann.

Tipping, M. E. & Bishop, C. M. (1999). Mixtures of probabilistic principal component analysers, *Neural Computation* 11: 443–482.

Williams C. K. I. & Seeger, M. (2001). Using the Nyström method to speed up kernel machines, *Advances in Neural Information Processing Systems*, Vol. 13, MIT Press.

Yang, M. H. (2002). Kernel eigenfaces vs. kernel fisherfaces: face recognition using kernel methods, *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, Washington, DC, pp. 215–220.

Zheng, W., Zou, C. & Zhao, L. (2005). An improved algorithm for kernel principal component analysis, *Neural Processing Letters* 22: 49–56.



New Advances in Machine Learning

Edited by Yagang Zhang

ISBN 978-953-307-034-6
Hard cover, 366 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

The purpose of this book is to provide an up-to-date and systematical introduction to the principles and algorithms of machine learning. The definition of learning is broad enough to include most tasks that we commonly call “learning” tasks, as we use the word in daily life. It is also broad enough to encompass computers that improve from experience in quite straightforward ways. The book will be of interest to industrial engineers and scientists as well as academics who wish to pursue machine learning. The book is intended for both graduate and postgraduate students in fields such as computer science, cybernetics, system sciences, engineering, statistics, and social sciences, and as a reference for software professionals and practitioners. The wide scope of the book provides a good introduction to many approaches of machine learning, and it is also the source of useful bibliographical information.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821