Neurocomputing 73 (2010) 959–967

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A general kernelization framework for learning algorithms based on kernel PCA

Changshui Zhang, Feiping Nie *, Shiming Xiang

The State Key Lab of Intelligent Technologies and Systems, Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Automation, Tsinghua University, Beijing 100084, PR China

ARTICLE INFO

Article history:
Received 24 March 2009
Received in revised form
16 August 2009
Accepted 23 August 2009
Communicated by D. Xu
Available online 6 October 2009

Keywords:
Kernel method
Learning algorithm
Kernel PCA
Two-stage framework

ABSTRACT

In this paper, a general kernelization framework for learning algorithms is proposed via a two-stage procedure, i.e., transforming data by kernel principal component analysis (KPCA), and then directly performing the learning algorithm with the transformed data. It is worth noting that although a very few learning algorithms were also kernelized by this procedure before, why and under what condition this procedure is feasible have not been further studied. In this paper, we explicitly present this kernelization framework, and give a rigorous justification to reveal that under some mild conditions, the kernelization under this framework is equivalent to traditional kernel method. We show that these mild conditions are usually satisfied in most of learning algorithms. Therefore, most of learning algorithms can be kernelized under this framework without having to reformulate it into inner product form, which is a common yet vital step in traditional kernel methods. Enlightened by this framework, we also propose a novel kernel method based on the low-rank KPCA, which could be used to remove the noise in the feature space, speed up the kernel algorithm and improve the numerical stability for the kernel algorithm. Experiments are presented to verify the validity and effectiveness of the proposed methods.

1. Introduction

Kernel methods [1–4] have attracted great interest in the past decades. The reason is largely because that kernel methods show better performance in most real-world applications in pattern recognition, computer vision, data minging, and so on. Many linear learning algorithms have been successfully kernelized [5–14], and most of which are implemented by making use of the kernel trick. In order to use the kernel trick, the output of the learning algorithm should be reformulated into inner product form, and then the nonlinear map from the original space to the high or even infinite dimensional feature space could be implicitly implemented by the kernel function.

Kernel principal component analysis (KPCA) [15], which is a kernel method for principal component analysis (PCA) [16], is one of the earliest kernel methods with the kernel trick. Later, the kernelization extensions for many linear algorithms have been achieved along the same outline of KPCA for PCA. However, when the output of a learning algorithm is difficult to be reformulated into the inner product form, the kernel trick could not be directly used to kernelize the learning algorithm.

In this paper, KPCA is viewed as a data transformation procedure. We propose a general kernelization framework for learning algorithms via this transformation procedure, i.e., transforming data by kernel principal component analysis (KPCA), and then directly performing the learning algorithm with the transformed data. Although a very few learning algorithms were also kernelized by this procedure before [17–19], why and under what condition this procedure is feasible have not been further studied. In this paper, we explicitly present this kernelization framework, and give a rigorous justification to reveal that under some mild conditions, the kernelization under this framework is equivalent to traditional kernel method. We will see that these mild conditions are usually satisfied by most of the learning algorithms, such as a large family of subspace learning related algorithms, distance metric learning, clustering algorithm, etc. Therefore, most of learning algorithms can be kernelized under this framework. Note that this framework introduces a KPCA procedure, so it need additional calculation to perform KPCA. However, as KPCA is a widely used algorithm in many applications, and once the KPCA procedure has been performed, we can directly perform many linear learning algorithms to implement the respective kernel algorithms simultaneously. Therefore, by virtue of this framework, we do not need the extra development of the kernel algorithms for these linear algorithms respectively, which is very convenient especially when we need to test a large number of linear algorithms and their kernel ones at the same time.

* Corresponding author. Tel.: +86 10 627 96 872; fax: +86 10 627 86 911.
E-mail addresses: zcs@mail.tsinghua.edu.cn (C. Zhang), feipingnie@gmail.com (F. Nie).

0925-2312/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2009.08.014

This two-stage kernelization framework provides us with a new perspective on the kernel method for a learning algorithm. Usually, one cannot discern the distribution and behavior of the data in the kernel space due to the implicit map. However, we can turn to see the distribution of the data after the KPCA transformation, since the behavior of the data in the kernel space is equivalent to the behavior of the data after the KPCA transformation. This framework also gives us a mechanism to implement the kernel method for a learning algorithm with more flexibility. Enlightened by this two-stage framework, we propose a new kernel method for learning algorithms based on the low-rank KPCA. In comparison with the full-rank KPCA based kernel method, the low-rank KPCA based kernel method has several advantages. For example, it could remove the noise in the feature space, speed up the kernel algorithm and improve the numerical stability for the kernel algorithm.

The rest of this paper is organized as follows: In Section 2, we revisit KPCA in details. In Section 3, we give the definition of the full-rank PCA and the full-rank KPCA, and then propose the general kernel method for learning algorithms based on the full-rank KPCA. In Section 4, we give some remarks on the general kernel method and propose a new kernel method for learning algorithms based on the low-rank KPCA. Some typical learning algorithm examples which satisfy the mild conditions are given in Section 5. In Section 6, we present the experiments to verify the validity of the general kernel method and the effectiveness of the proposed low-rank kernel method. Finally, we conclude this paper in Section 7.

2. Kernel PCA revisited

Kernel PCA [15] is a nonlinear extension to PCA with the kernel trick. In order to use the kernel trick, the solution to PCA should be reformulated into inner product form first.

Given the training data $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}, \boldsymbol{x}_i \in \mathbb{R}^d$, we denote the mean of the training data by $\overline{\boldsymbol{x}} = (1/n)\sum_i \boldsymbol{x}_i$, the training data matrix by $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]$ and the centralized training data matrix by $\overline{\mathbf{X}} = [\boldsymbol{x}_1 - \overline{\boldsymbol{x}}, \boldsymbol{x}_2 - \overline{\boldsymbol{x}}, \ldots, \boldsymbol{x}_n - \overline{\boldsymbol{x}}]$.

Define a centralization matrix by $\mathbf{L} = \mathbf{I} - 1/n\mathbf{1}\mathbf{1}^T$, where $\mathbf{I}$ is an $n \times n$ identity matrix, and $\mathbf{1} \in \mathbb{R}^n$ is a column vector in which all the elements are equal to one. It can be easily verified that

$$\overline{\mathbf{X}} = \mathbf{X}\mathbf{L} \tag{1}$$

Therefore, the covariance matrix of the training data can be written as

$$\mathbf{C} = \frac{1}{n}\mathbf{X}\mathbf{L}(\mathbf{X}\mathbf{L})^T \tag{2}$$

PCA extracts the principal components by calculating the eigenvectors of the covariance matrix $\mathbf{C}$.

Lemma 2.1 (*Horn and Johnson [20]*). *Given two matrices* $\mathbf{A} \in \mathbb{R}^{n\times d}$ *and* $\mathbf{B} \in \mathbb{R}^{d\times n}$, *then* $\mathbf{AB}$ *and* $\mathbf{BA}$ *have the same nonzero eigenvalues. For each non-zero eigenvalue, if the corresponding eigenvector of* $\mathbf{AB}$ *is* $\boldsymbol{v}$, *then the corresponding eigenvector of* $\mathbf{BA}$ *is* $\boldsymbol{u} = \mathbf{B}\boldsymbol{v}$.

According to Lemma 2.1, the eigenvectors of $\mathbf{C}$ can be calculated from the eigenvectors of $\mathbf{M} = (\mathbf{XL})^T\mathbf{XL} = \mathbf{L}\mathbf{X}^T\mathbf{X}\mathbf{L}$. Denote the k-th largest eigenvalue of $\mathbf{M}$ by λ_k, and the corresponding eigenvector by $\boldsymbol{v}_k$, then the k-th largest eigenvalue of $\mathbf{C}$ is λ_k, and the corresponding eigenvector is $\boldsymbol{u}_k = \mathbf{XL}\boldsymbol{v}_k$. Therefore, the k-th principal direction calculated by PCA is $\tilde{\boldsymbol{u}}_k = \boldsymbol{u}_k/\|\boldsymbol{u}_k\| = \mathbf{XL}\boldsymbol{v}_k/\sqrt{\boldsymbol{v}_k^T\mathbf{M}\boldsymbol{v}_k}$. For any data $\boldsymbol{x} \in \mathbb{R}^d$, the k-th principal component is

$$y_k = \tilde{\boldsymbol{u}}_k^T(\boldsymbol{x} - \overline{\boldsymbol{x}}) = \frac{\boldsymbol{v}_k^T\mathbf{L}\mathbf{X}^T(\boldsymbol{x} - \overline{\boldsymbol{x}})}{\sqrt{\boldsymbol{v}_k^T\mathbf{M}\boldsymbol{v}_k}} \tag{3}$$

Instead of doing PCA in the input space, kernel PCA performs PCA in a mapped high-dimensional inner product space $\mathcal{F}$. The map $\phi : \mathbb{R}^d \to \mathcal{F}$ is nonlinear and is implicitly implemented via kernel function

$$\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}') = \phi(\boldsymbol{x})^T\phi(\boldsymbol{x}') \tag{4}$$

The kernel function, $\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ may be any positive kernel satisfying Mercer's condition [21,5]. For instance, the frequently used one is the radial basis function (RBF) kernel defined by

$$\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}') = \exp\left\{-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{\sigma^2}\right\} \tag{5}$$

The mapped space $\mathcal{F}$ is also called feature space. For an algorithm that can be expressed in terms of inner product, the algorithm can be also performed in the feature space using the kernel trick. Fortunately, PCA is such an algorithm. From Eq. (3) we can see the output of PCA can be calculated solely by inner product. Therefore, in KPCA, for any data $\boldsymbol{x} \in \mathbb{R}^d$, the k-th principal component is

$$y_k = \frac{\boldsymbol{v}_k^T\mathbf{L}\phi(\mathbf{X})^T(\phi(\boldsymbol{x}) - \overline{\phi})}{\sqrt{\boldsymbol{v}_k^T\mathbf{K}\boldsymbol{v}_k}} \tag{6}$$

where $\overline{\phi} = (1/n)\sum_i \phi(\boldsymbol{x}_i)$, $\phi(\mathbf{X}) = [\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_n)]$, $\mathbf{K} = \mathbf{L}\phi(\mathbf{X})^T\phi(\mathbf{X})\mathbf{L}$ and $\boldsymbol{v}_k$ is the k-th largest eigenvector of $\mathbf{K}$.

3. Kernel method for learning algorithms based on full-rank KPCA

In this section, we propose a general kernel method for learning algorithms based on full-rank KPCA. First we give the definition of the full-rank PCA and the definition of the full-rank KPCA.

Suppose the training data for a learning algorithm are $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}, \boldsymbol{x}_i \in \mathbb{R}^d$. We denote the training data matrix by $\mathbf{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{d\times n}$, the mean of the training data by $\overline{\boldsymbol{x}} = (1/n)\sum_i \boldsymbol{x}_i$, and the centralized inner product matrix by $\mathbf{M} = \mathbf{L}\mathbf{X}^T\mathbf{X}\mathbf{L}$. Corresponding, in the feature space, we denote the training data matrix by $\phi(\mathbf{X}) = [\phi(\boldsymbol{x}_1), \phi(\boldsymbol{x}_2), \ldots, \phi(\boldsymbol{x}_n)]$, the mean of the training data by $\overline{\phi} = (1/n)\sum_i \phi(\boldsymbol{x}_i)$, and the centralized kernel matrix by $\mathbf{K} = \mathbf{L}\phi(\mathbf{X})^T\phi(\mathbf{X})\mathbf{L}$.

Definition 3.1 (*full-rank PCA*). For the training data matrix $\mathbf{X}$, suppose the rank of the centralized inner product matrix $\mathbf{M}$ is r. If we extract the first r principal components of PCA, we say we have performed the full-rank PCA.

Suppose the eigen-decomposition of matrix $\mathbf{M}$ is

$$\mathbf{M} = [\boldsymbol{\alpha}, \boldsymbol{\beta}]\begin{bmatrix} \boldsymbol{\Lambda} & 0 \\ 0 & 0 \end{bmatrix}[\boldsymbol{\alpha}, \boldsymbol{\beta}]^T \tag{7}$$

where $\boldsymbol{\Lambda}$ is the diagonal matrix with the diagonal elements being the non-zero eigenvalues of $\mathbf{M}$, the columns of $\boldsymbol{\alpha}$ is the corresponding unit eigenvectors of the non-zero eigenvalues, and the columns of $\boldsymbol{\beta}$ is the corresponding unit eigenvectors of the zero eigenvalues. We call $\mathbf{M} = \boldsymbol{\alpha}\boldsymbol{\Lambda}\boldsymbol{\alpha}^T$ is the full-rank eigen-decomposition of matrix $\mathbf{M}$.

It can be derived that the projection matrix of the full-rank PCA is

$$\mathbf{W} = \mathbf{XL}\boldsymbol{\alpha}\boldsymbol{\Lambda}^{-1/2} \tag{8}$$