Signal Processing 90 (2010) 1542–1553

Contents lists available at ScienceDirect

Signal Processing

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

Adaptive kernel principal component analysis

Mingtao Ding [a,*], Zheng Tian [a,b,c], Haixia Xu [b]

[a] School of Science, Northwestern Polytechnical University, Xi'an 710129, China
[b] Department of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an 710129, China
[c] State Key Laboratory of Remote Sensing Science, Institute of Remote Sensing Applications, Chinese Academy of Sciences, Beijing 100101, China

ARTICLE INFO

Article history:
Received 24 June 2009
Received in revised form
30 September 2009
Accepted 1 November 2009
Available online 10 November 2009

Keywords:
Adaptive method
Kernel principal component analysis
Kernel principal component
Non-stationary data
Recursive algorithm

ABSTRACT

An adaptive kernel principal component analysis (AKPCA) method, which has the flexibility to accurately track the kernel principal components (KPC), is presented. The contribution of this paper may be divided into two parts. First, KPC are recursively formulated to overcome the batch nature of standard kernel principal component analysis (KPCA). This formulation is derived from the recursive eigendecomposition of kernel covariance matrix and indicates the KPC variation caused by the new data. Second, kernel covariance matrix is correctly updated to adapt to the changing characteristics of data. In this adaptive method, the KPC is adaptively adjusted without re-eigendecomposing the kernel Gram matrix. The proposed method not only maintains constant update speed and memory usage as the data-size increases, but also alleviates sub-optimality of the KPCA method for non-stationary data. Experiments for simulation data and real applications are detailed to assess the utility of the proposed method. The results demonstrate that our approach yields improvements in terms of both computational speed and approximation accuracy.

1. Introduction

Kernel principal component analysis (KPCA), which is a nonlinear extension of principal component analysis (PCA) [1], has gained significant attention as a learning machine [2] in pattern recognition [3–6], statistical analysis [7,8] and image processing [9,10]. The core idea of KPCA is to first map the input space into a feature space using a nonlinear mapping and then compute the principal components in the feature space. As a result, the extracted kernel principal component (KPC) of the mapped data is nonlinear with regards to the original input space.

Although many issues have been discovered, two key of them remain unresolved in KPCA, i.e., the batch nature [11] and the non-adaptability to the changes in the data characteristics. On one hand, the batch nature hinders KPCA in terms of computation and memory demands as the data-size increases. For KPCA, the kernel Gram matrix has to be wholly available before eigendecomposition can be carried out. The arrival of a new data will require the addition of a new row and column for kernel Gram matrix, and eigendecomposition has to be constantly reevaluated for the ever-growing matrix. In addition, all data must be saved to represent the KPC. This translates into high costs for storage resources and computational load during run-time application of large datasets. On the other hand, the standard KPCA, which relies on a fixed model, cannot be employed for non-stationary data whose input distributions could be temporally changed. According to [12], the non-stationary property in input space will reappear in feature space when the nonlinear mapping is smooth and continuous. However, KPCA spends equivalent modeling power on the mapped data to extract the KPC. This is not desirable for tracking the changing characteristics of the input data.

* Corresponding author. Tel.: +86 13720603654.
E-mail addresses: dingmingtao@tom.com (M. Ding), zhtian@nwpu.edu.cn (Z. Tian), xuhaixia_xhx@163.com (H. Xu).

0165-1684/$ - see front matter Crown Copyright © 2009 Published by Elsevier B.V. All rights reserved.
doi:10.1016/j.sigpro.2009.11.001

In the last decade, several approaches have been proposed to deal with the batch nature of KPCA, and could be mostly grouped into three classes. The first class is to kernelize the generalized Hebbian algorithm which is an iterative self-organizing computation procedure for linear PCA. The idea of using kernel Hebbian algorithm was first explored by Kim et al. [10], where Hebbian algorithm is iteratively implemented to estimate KPCs in the kernel induced feature space. Later, Gunter et al. [13] developed gain adaptation methods to improve convergence of the kernel Hebbian algorithm by incorporating the reciprocal of the current estimated eigenvalues as part of a gain vector. While these methods can potentially lower the time complexity of computing KPC, it is unclear how novel data can be incorporated to update the output. The second class is to compute KPC incrementally [14–16]. Specially, Chin and Suter [15,16] show how to obtain the KPC by incrementally updating singular value decomposition (SVD) of the mapped data in feature space. To maintain constant update speed and memory usage, the mean and KPC representations was compressed by constructing reduced-set expansions, which is computationally expensive. The third class is the greedy KPCA [17,18] which was employed to approximate the KPC by a prior filtering of the training data. However, one drawback is the filtering could be computationally expensive by itself. All the above methods concentrate their study on approximation accuracy and time complexity, with little effort contributed to processing non-stationary data. It is obvious that these methods cannot be directly used to adapt the KPC to recent observations without modification.

To track the changing characteristics of data, it is desirable to focus more on recently-acquired data and less on earlier data. For example, when tracking a target with a changes appearing in computer version, it is likely that recent observations will be more indicative of its appearance than distant ones. The most common way to moderate the balance between old and new observations is to incorporate an exponential forgetting window to the observations [19]. However, the exponential window may have several drawbacks: the influence of old data can last for a very long time, and the exponential forgetting factor must be determined appropriately. In time-varying environment, another approach is to use sliding window [20] where data within the window is assumed to be stationary. Sliding window corresponds to rank-2 update, i.e., adding a column to and deleting a column from the data matrix simultaneously. Hoegaerts et al. [21] have developed a fast updating and downdating procedure for dominant eigenspace of a growing symmetrical kernel Gram matrix. However, this does not imply the KPC computation is solved, since the dominant eigenspace of symmetrical kernel Gram matrix can be used to represent the KPC only when the mapped date is zero-mean in the feature space. Consequently, an approach which adapts to the changing characteristics with efficient computation is required.

This paper proposes an adaptive KPCA (AKPCA) method with rapid and accurate computation for extracting KPC. Rather than directly operating on mapped kernel data or kernel Gram matrix, we commence with the kernel covariance matrix, from which the AKPCA arises. The basis of our solution lies in decomposing the new data into a component orthogonal and a component parallel to the previous KPC, and adaptively adjusting KPC based on the rank-2 update of kernel covariance matrix. The contribution made in this paper is twofold: (1) recursively formalizing KPC to adapt to the changing characteristics of non-stationary data and (2) reducing the computational complexity and memory usage of KPCA to $O(N)$.

The rest of the paper is organized as follows. Section 2 presents a brief description of the KPCA method. Section 3 elucidates the proposed AKPCA approach, including the recursive decomposition of kernel covariance matrix, recursive KPC formulation and the update of kernel covariance matrix which adapts to the changing characteristics of data. Section 4 discusses the methodological and computational issues of the proposed AKPCA. Section 5 details the experiments and analysis aiming at assessing the performance of the proposed method. We derive the conclusion in Section 6.

2. Preliminaries

We begin by obtaining a data matrix $\mathbf{x}=(x_1,x_2,\ \ldots,\ x_n)\in\mathbb{R}^{m\times n}$, with $x_i\in\mathbb{R}^m$ being the data vector at time $i\geq 1$. KPCA nonlinearly maps $\mathbf{x}$ into a higher dimensional space $\mathcal{F}$, and subsequently performs linear PCA in $\mathcal{F}$. Assuming that the mapped data is centered in feature space (we shall return to this point later), its covariance matrix is given by

$$\mathbf{C}_n=\frac{1}{n}\sum_{i=1}^{n}\phi(x_i)\phi(x_i)^T, \tag{1}$$

where ϕ is the nonlinear mapping function. The map ϕ is induced by a kernel function $k(\cdot,\cdot)$ that allows us to evaluate inner products in $\mathcal{F}: k(x_i,x_j)=\phi(x_i)\cdot\phi(x_j)$, for $i,j=1,\ldots,n$. Given that the mapping function ϕ is implicit, eigendecomposition cannot be performed on $\mathbf{C}_n$ to compute the KPC. KPCA circumvents the KPC by a dual eigendecomposition problem for kernel Gram matrix $K_n\mathbf{a}^w=n\lambda_w\mathbf{a}^w$, in which $\mathbf{a}^w=(a_1^w,a_2^w,\ \ldots,\ a_n^w)^T$ is the normalized eigenvector[1] associated with the w-th largest eigenvalues. Then, the r most significant KPC in feature space take the form of

$$\mathbf{V}_n=[\mathbf{v}^1,\mathbf{v}^2,\ \ldots,\ \mathbf{v}^r]=[\phi(x_1),\phi(x_2),\ \ldots,\ \phi(x_n)]\mathbf{A}_n, \tag{2}$$

where $\mathbf{v}^w$ is the w-th columns of $\mathbf{V}_n$, and $\mathbf{A}_n$ is the matrix with the columns of $\mathbf{a}^w$, $w=1,2,\ldots,r$. Let $z\in\mathbb{R}^m$ be a test data, with an image $\phi(z)$, then its w-th principal component corresponding to ϕ is

$$\mathbf{p}_z(w)=\mathbf{v}^w\cdot\phi(z)=\sum_{i=1}^{n}a_i^w k(x_i,z). \tag{3}$$

The Eq. (3) is important for the proposed AKPCA since it does not require ϕ in explicit form and the projections of arbitrary data onto the KPC can be solved entirely in

[1] $\mathbf{a}^w$ should be scaled to ensure $\mathbf{a}^w\cdot\mathbf{a}^w=1/n\lambda_w$.