Pattern Recognition 46 (2013) 2169–2174

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

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

Computational and space complexity analysis of SubXPCA

Vijayakumar Kadappa [a,*], Atul Negi [b,1]

[a] Department of Computer Applications, BMS College of Engineering, Bangalore 560019, India
[b] School of Computer and Information Sciences, University of Hyderabad, Hyderabad 500046, India

ARTICLE INFO

Article history:
Received 11 April 2012
Received in revised form
1 December 2012
Accepted 12 January 2013
Available online 18 January 2013

Keywords:
Dimensionality reduction
Feature extraction
Principal component analysis
Feature partitioning
Space complexity
Time complexity

ABSTRACT

Principal Component Analysis (PCA) is one of the well-known linear dimensionality reduction techniques in the literature. Large computational requirements of PCA and its insensitivity to 'local' variations in patterns motivated to propose partitional based PCA approaches. It is also observed that these partitioning methods are incapable of extracting 'global' information in patterns thus showing lower dimensionality reduction. To alleviate the problems faced by PCA and the partitioning based PCA methods, SubXPCA was proposed to extract principal components with global and local information. In this paper, we prove analytically that (i) SubXPCA shows its computational efficiency up to a factor of k ($k \geq 2$) as compared to PCA and competitive to an existing partitioning based PCA method (SubPCA), (ii) SubXPCA shows much lower classification time as compared to SubPCA method, (iii) SubXPCA and SubPCA outperform PCA by a factor up to k ($k \geq 2$) in terms of space complexity. The effectiveness of SubXPCA is demonstrated upon a UCI data set and ORL face data.

1. Introduction

Principal Component Analysis (PCA) is one of the widely used methods for dimensionality reduction. Its applications include multi-modal biometric systems [1], web page watermarking [2], vehicle traffic monitoring [3], face recognition [4], image processing [5], to name a few. The PCA method is concerned with summarizing the *variance–covariance structure* using a few linear combinations of the original set of d features [6]. PCA is an optimal linear dimensionality reduction scheme in terms of mean squared error (MSE). However, PCA suffers from a large time complexity [$O(N \cdot d^2)$, where N is the number of training patterns, d is the dimensionality of patterns] to compute the covariance matrix. Many approaches such as neural network based PCA methods [7], 2DPCA based methods [8] show reduced computational complexity as compared to PCA method. However, these methods are based on whole-patterns, which are suitable for global feature extraction like PCA, and they do not extract local features if local variations are prominent [9]. It is known that PCA performs global feature extraction that retains information based upon covariances between every pair of original features. However, global features are ineffective when variations are confined to subpatterns (i.e. local variations). For example, a smile is a local variation confined to a sub-region in a face and is distributed over several patterns.

To overcome the difficulties with global feature extraction methods, *Feature Partitioning* based PCA (FP-PCA) methods (Block based methods) were proposed which exploit the local variations within subpatterns with improved computational efficiency. The FP-PCA methods are based on the principle of *partitioning a pattern into subpatterns and extract principal components* (*PCs*) *from these subpatterns*. However, these FP-PCA methods do not exhibit good summarization of variance like PCA. Some of the FP-PCA methods include SubPCA [10], Modular PCA [11], Eigen-regions [12]. Recently, block based idea was extended to kernel PCA methods [13] and Liang et al. [15] showed that FP-PCA methods have larger reconstruction error as compared to PCA methods [14]. To overcome the problems faced by the FP-PCA and PCA methods, SubXPCA [9] was proposed. SubXPCA exploits the merits of FP-PCA methods (that is, local feature extraction and computational efficiency), and merits of PCA methods (that is, global feature extraction and good summarization of variance).

We observed that there is no detailed analytical study on space utilization and computational analysis of the existing FP-PCA methods in comparison to PCA. Space and Computational complexities are vital parameters to assess the merit of a technique which is used in data intensive applications such as data mining in particular. Here, we would like to bring out the space and time complexity related properties of SubXPCA, SubPCA, and PCA methods. A comparison is also made with another FP-PCA method, SubPCA [10], upon ORL Face data, and UCI Musk data sets.

* Corresponding author. Tel.: +91 80 26622130-35x4064.
E-mail addresses: kadappakumar@gmail.com, kvkumar28@yahoo.com (V. Kadappa), atulcs@uohyd.ernet.in, atul.negi@gmail.com (A. Negi).
[1] Tel.: +91 40 23134031; fax: +91 40 23010780.

0031-3203/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2013.01.018

In the following, Section 2 reviews the major FP-PCA methods and their time and space complexities are discussed in Section 3. The computational and space complexity properties of SubXPCA are investigated in Section 4. Experimental results are presented in Section 5 and concluding remarks are given in Section 6.

2. An overview of SubXPCA and SubPCA methods

In this section, we review two feature partitioning based PCA methods, SubXPCA [9] and SubPCA [10], in brief so that the properties presented in Section 4 can be clearly understood.

2.1. Cross-subpattern correlation based PCA (SubXPCA) method

We shall use the notation as described below. The set, $\mathbf{X}$, of given patterns is denoted by using a matrix, $[\mathbf{X}]_{N\times d} = [\mathbf{X}_1\mathbf{X}_2 \ldots\ldots \mathbf{X}_N]^T$, where N is the cardinality of $\mathbf{X}$. Each pattern, $\mathbf{X}_i$, is a feature vector of dimension d, that is, $[\mathbf{X}_i]_{d\times 1} = [x_{i1}x_{i2}\ldots x_{id}]^T$ where $x_{i1}, x_{i2}, \ldots, x_{id}$ are the feature values that represent $\mathbf{X}_i$. $[\mathbf{C}]_{d\times d}$ is the covariance matrix of patterns, $[\mathbf{X}]_{N\times d}$. We represent by k as the number of subpatterns, u as the subpattern size, and r as the number of local features (local PCs).

1. *Partitioning step*: Divide every d-dimensional pattern, $\mathbf{X}_i$ $(i=1,2,\ldots,N)$ into k (≥ 2) equally-sized subpatterns, $\mathbf{X}_i^1, \mathbf{X}_i^2, \ldots, \mathbf{X}_i^k$. Each subpattern is of size u, where $u = \lfloor d/k \rfloor$. For a given pattern, $\mathbf{X}_i$, the jth subpattern, $\mathbf{X}_i^j$, is given by $[\mathbf{X}_i^j]_{u\times 1} = [x_{il}\, x_{i(l+1)}\ldots x_{i(l+u-1)}]^T$, where $l=(j-1)\cdot u+1$, $1\leq j\leq k$.

2. *Grouping step*: We pick-up jth subpattern corresponding to every pattern, $\mathbf{X}_i$; $i=1,2,\ldots,N$, and form jth subpattern group (partition), $\mathbf{P}^j$, which is given by the matrix, $[\mathbf{P}^j]_{N\times u} = [\mathbf{X}_1^j\mathbf{X}_2^j\ldots\mathbf{X}_N^j]^T$. Let $[\overline{\mathbf{X}}^j]_{u\times 1}$ be the mean of N subpatterns, $[\mathbf{P}^j]_{N\times u}$. Let $\mathbf{P}_M^j$ be the mean-subtracted version of $\mathbf{P}^j$.

3. *Local feature extraction step*: For every subpattern group, $\mathbf{P}^j$, where $j=1,2,\ldots,k$, repeat the following steps (a)–(d): (a) Compute local covariance matrix, $[\mathbf{C}^j]_{u\times u}$. (b) Compute eigenvalues (λ_p^j) and eigenvectors $(\mathbf{e}_p^j)$, where $p=1,2,\ldots,u$, of $\mathbf{C}^j$. (c) Select r $(<u)$ local eigenvectors corresponding to the first r largest eigenvalues obtained in the preceding step. Let $[\mathbf{E}^j]_{u\times r}$ be the matrix of r local eigenvectors selected in this step. (d) Extract r local features (local PCs) by projecting $\mathbf{P}_M^j$ onto $\mathbf{E}^j$.

4. *Combining locally extracted features step*: (a) Form locally-reduced pattern, $\mathbf{Y}_i$ by concatenating locally-reduced subpatterns, $[\mathbf{Y}_i^j]_{r\times 1}$, $\forall j=1,2,\ldots,k$, as shown by $[\mathbf{Y}_i]_{k\cdot r\times 1} = [[\mathbf{Y}_i^1]^T, [\mathbf{Y}_i^2]^T, \ldots\ldots, [\mathbf{Y}_i^k]^T]^T$ and $\mathbf{Y}_i^j$ is the locally-reduced form of subpattern, $\mathbf{X}_i^j$. Let $[\mathbf{Y}]_{N\times k\cdot r} = [\mathbf{Y}_1\mathbf{Y}_2,\ldots\mathbf{Y}_N]^T$ denotes the matrix of locally-reduced patterns. (b) Perform global feature extraction using cross-subpattern covariances of $\mathbf{Y}$, obtained in the preceding step: (i) Compute global covariance matrix, $[\mathbf{C}^g]_{(k\cdot r)\times(k\cdot r)}$ for the data $\mathbf{Y}$. (ii) Compute eigenvalues (λ_s) and eigenvectors $(\mathbf{e}_s)$ of $\mathbf{C}^g$, where $s=1,2,\ldots,(k\cdot r)$. (iii) Select w $(<k\cdot r)$ eigenvectors corresponding to first w largest eigenvalues obtained in the preceding step. Let $[\mathbf{E}^g]$ be the matrix of w eigenvectors selected in this step. (iv) Extract w global PCs by projecting $\mathbf{Y}$ onto $\mathbf{E}^g$. Let $\mathbf{Z}$ be the data obtained after projection in this step, then, $[\mathbf{Z}]_{N\times w} = [\mathbf{Y}]_{N\times k\cdot r}\cdot[\mathbf{E}^g]_{k\cdot r\times w}$. The reduced data, $\mathbf{Z}$, is used for subsequent tasks such as classification.

2.2. Subpattern based PCA (SubPCA) method

The SubPCA [10] method is constituted by following the Steps 1–3 and Step-4(a) of SubXPCA method given in Section 2.1. The SubPCA extracts local features (local PCs) which yield locally-reduced subpatterns, $(\mathbf{Y}_i^j)_{r\times 1}$, $\forall j=1,2,\ldots,k$ as given in the preceding Section 2.1.

3. Computational time and space complexities of PCA, SubPCA, and SubXPCA methods

In this section, we present time and space complexities of PCA, SubPCA, and SubXPCA, which will be used subsequently in Section 4.

3.1. Time complexities for feature extraction

The time complexity for feature extraction of PCA method (which includes computation of covariance matrix $(O(N\cdot d^2))$, eigenvectors/eigenvalues $(O(d^3))$, etc.) is given by

$$T_C = O(N\cdot d^2 + d^3) \tag{1}$$

On the similar lines, the time complexity for feature extraction of SubPCA method with k subpatterns of size u is given by

$$T_S = O[k\cdot(N\cdot u^2 + u^3)] \tag{2}$$

Now, we focus on computing the time complexity of SubXPCA, T_F, which is the sum of (i) time complexity of processing all k subpatterns (same as SubPCA) (Steps 1–3 of Section 2.1) and (ii) time complexity of extracting features using inter-subpattern correlations (Step-4 of Section 2.1). The time complexity for feature extraction of SubXPCA, T_F, is given by

$$T_F = T_S + O[N\cdot(k\cdot r)^2 + (k\cdot r)^3] \tag{3}$$

3.2. Time complexities for classification based on nearest neighbour (NN) rule

Here we find nearest neighbour (NN) of each of the given M test patterns from the N training patterns used for computing PCs. It is known that NN rule takes $O(N\cdot w)$ time for finding a nearest neighbour of a test pattern from N patterns with w PCs. Therefore, the time complexity for NN based classification of M test patterns with w PCs obtained by PCA method, T_{CC}, is given by

$$T_{CC} = O(M\cdot N\cdot w) \tag{4}$$

We know that SubPCA (Section 2.2) extracts $k\cdot r$ local PCs (r from each of the k subpatterns). The time complexity for NN based classification of M test patterns with $k\cdot r$ PCs obtained by SubPCA method, T_{CS}, is given by

$$T_{CS} = O(M\cdot N\cdot k\cdot r) \tag{5}$$

From Section 2.1, we know that SubXPCA extracts $w(<k\cdot r)$ global PCs from $k\cdot r$ local PCs. The time complexity for NN based classification of M test patterns with w PCs obtained by SubXPCA, T_{CF}, is given by

$$T_{CF} = O(M\cdot N\cdot w) \tag{6}$$

3.3. Space complexities of PCA, SubPCA and SubXPCA methods

The space complexities to store (i) $[\mathbf{X}]_{N\times d}$, the patterns, (ii) covariance matrix of size $d\times d$, (iii) eigenvalues and eigenvectors, and (iv) local PCs, $[\mathbf{Y}]_{N\times w}$ are given as $O(N\cdot d), O(d^2), O(d^2), O(N\cdot w)$ respectively. Thus, the space complexity of PCA method is given by

$$S_C = O(N\cdot d + d^2) \tag{7}$$

We know that SubPCA computes local PCs by storing *a single partition* of subpatterns $([\mathbf{P}^j]_{N\times u})$ at any point of time. On the similar lines of PCA, the space complexity of SubPCA method with subpatterns of size u is given by

$$S_S = O(N\cdot u + u^2) \tag{8}$$