Research Article

Incremental Matrix-Based Subspace Method for Matrix-Based Feature Extraction

Zhaoyang Zhang , Shijie Sun , and Wei Wang

School of Information Engineering, Chang'an University, Xi'an 710068, China

Correspondence should be addressed to Shijie Sun; shijiesun@chd.edu.cn

Received 29 August 2020; Revised 17 September 2020; Accepted 6 October 2020; Published 29 October 2020

Academic Editor: Tony Chen

Copyright © 2020 Zhaoyang Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The matrix-based features can provide valid and interpretable information for matrix-based data such as image. Matrix-based kernel principal component analysis (MKPCA) is a way for extracting matrix-based features. The extracted matrix-based feature is useful to both dimension reduction and spatial statistics analysis for an image. In contrast, the efficiency of MKPCA is highly restricted by the dimension of the given matrix data and the size of the training set. In this paper, an incremental method to extract features of a matrix-based dataset is proposed. The method is methodologically consistent with MKPCA and can improve efficiency through incrementally selecting the proper projection matrix of the MKPCA by rotating the current subspace. The performance of the proposed method is evaluated by performing several experiments on both point and image datasets.

1. Introduction

Subspace analysis is helpful for a network in computer vision [1, 2] and data modeling problems [3] and social network [4]. AlexNet [5] is a pioneer in using principal component analysis (PCA), a basic subspace analysis method, to help complex networks improve their performance. It performs PCA on the set of RGB pixel values throughout the ImageNet training set to reduce overfitting on image data by data augmentation. PCANet [6] employs PCA to learn multistage filter banks in its architecture. The PCA makes the architecture be extremely easily and efficiently designed and learned. The PCA based analysis method can be flexibly applied to designing the architecture of neural networks [7–9]. In complex networks, the training image, convolution induced feature maps, and some channel splicing features can be considered as matrix-based data. Improving the matrix-based subspace method and studying the relationship between matrix-based PCA and the basic vector-based one may provide a reference for networks.

The regular PCA is a linear method in the sense that it only utilizes the first-order and second-order statistics. Therefore, its modeling capability is limited when

confronted with highly nonlinear data structures. To enhance its modeling capability, the vector-based kernel PCA (KPCA) [10, 11] is proposed. It improves the performance of PCA in modeling by nonlinearly mapping the data from the original space to a very high-dimensional feature space, the so-called reproducing kernel Hilbert space (RKHS). The nonlinear mapping enables an implicit characterization of high-order statistics and can be flexibly combined with other learning methods [12–14]. The key idea of kernel methods is to avoid the explicit knowledge of the mapping function by evaluating the dot product in the feature space using a kernel function. The vector-based KPCA takes vectors as input. The matrix data, such as a two-dimensional image, is vectorized into a vector before being fed into kernel methods. Such vectorization ruins the spatial structure of the pixels that define the two-dimensional image. To bring back the matrix structure, matrix-based KPCA [15] is proposed by combining two-dimensional PCA (2DPCA) [16] with the kernel approach. The matrix-based KPCA can generalize the vector-based KPCA and can provide richer representations than vector-based KPCA.

The advantage of the matrix-based KPCA methods is that they enable us to study the spatial statistics in the matrix.

However, the size of Gram matrix of matrix-based KPCA is much bigger than that of vector-based KPCA, which means that the batch problem of the matrix-based KPCA is much more serious than that of vector-based KPCA. For matrix-based KPCA, the Gram matrix has to be provided before the eigendecomposition process can be conducted. For extra data added, additional new rows and columns are required for a new Gram matrix, and the eigendecomposition has to be performed for the size grown matrix. In addition, the principal component vectors must be supported by all the matrices in input dataset. This induces a high cost for storage resources and computing workloads during conducting applications with large datasets and input matrix with big size.

In the last decade, several strategies and approaches have been proposed to improve the batch nature for the vector-based KPCA, such as the methods that kernelize the generalized Hebbian algorithm [17, 18], the methods that compute the KPC incrementally [19–21], the greedy KPCA method [22, 23], and the adaptive KPCA method [24]. In recent years, the KPCA based method still has great vitality [25–27], while the methods that counteract the huge batch nature of matrix-based KPCA are rare. The matrix-based KPCA can be speeded up using the idea proposed in improved KPCA [28]. However, this does not imply that the matrix-based KPCA computation is solved, since the size of eigendecomposition still depends on the size of the datasets. Consequently, an approach that adapts to the batch nature of matrix data with efficient computations is required.

This paper proposes an incremental matrix-based KPCA (IMKPCA) method to approximating the traditional one with less computation time and memory usage for extracting kernel principal vectors under a certain accuracy. The contributions of this paper are in three points:

- (1) The proposed method is implemented through the total scatter matrix to avoid directly operating on the Gram matrix of matrix-based KPCA. That is what inspires us the idea of the proposed IMKPCA.
- (2) The basis of our solution is incrementally adjusting the current eigenvector matrix to keep the total scatter of the matrix dataset. This work can be completed by decomposing the added matrix data into a component parallel to the previous eigenvector matrix and a component orthogonal to that. Meanwhile, the standard method computes the scatter matrix directly.
- (3) The proposed matrix-based feature can be used to study the spatial statistics in the matrix under an acceptable computational complexity and memory usage. However, the vector-based one cannot.

The rest of this paper is organized as follows. The preliminaries of matrix-based KPCA are briefly introduced in Section 2. Then, an incremental matrix-based kernel subspace method is presented in Section 3, followed by the experiment results shown in Section 4. Finally, the conclusion is presented in Section 5.

2. Preliminaries

$\Rightarrow$ and $\Downarrow$ mean horizontal and vertical concatenations, respectively. The expressions of matrix and matrix dataset can be simplified by using $\Rightarrow$ and $\Downarrow$ as in [15].

For a matrix $\mathbf{X} \triangleq [\Rightarrow_{i=1}^q \mathbf{x}_i] \in \mathbb{R}^{p \times q}$, where p and q are the numbers of rows and columns, respectively, we define its kernelized matrix $\phi(\mathbf{X}) \in \mathbb{R}^{f \times q}$ as $\phi(\mathbf{X}) = [\Rightarrow_{i=1}^q \phi(\mathbf{x}_i)]$, where $\mathbf{x}_i$ is the i -th column of $\mathbf{X}$ and ϕ is a nonlinear mapping from $\mathbb{R}^p$ to $\mathbb{R}^f$.

The dot product matrix $\mathbf{K} \in \mathbb{R}^{q \times q}$ of two kernelized matrices $\mathbf{X}$ and $\mathbf{Y}$ is defined as

$$\mathbf{K}(\mathbf{X}, \mathbf{Y}) \triangleq \phi(\mathbf{X})^\top \phi(\mathbf{Y}) = [\Downarrow_{i=1}^q [\Rightarrow_{j=1}^q k(\mathbf{x}_i, \mathbf{y}_j)]], \quad (1)$$

where $k(\mathbf{x}_i, \mathbf{x}_j) \triangleq \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$.

We begin by observing a matrix dataset: block matrix $\mathcal{X}_n \triangleq [\Rightarrow_{i=1}^n \mathbf{x}_i] \in \mathbb{R}^{p \times nq}$; the idea of matrix-based KPCA is performing two-dimensional PCA [16] in feature space, where $\mathbf{x}_i \in \mathbb{R}^{p \times q}$ is the i -th matrix data in the dataset.

Assuming that the dataset $\mathcal{X}_n$ is kernelized to $\phi(\mathcal{X}_n) \triangleq [\Rightarrow_{i=1}^n \phi(\mathbf{x}_i)] \in \mathbb{R}^{f \times nq}$ and that it is centered in feature space (we shall return to this point later), its total scatter matrix is estimated by

$$\Sigma_n \triangleq \frac{1}{n} \sum_{i=1}^n (\phi(\mathbf{x}_i) \phi(\mathbf{x}_i)^\top). \quad (2)$$

Given that the mapping function ϕ is implicit, the KPC cannot be computed by performing eigendecomposition on Σ_n directly. The reason is that the nonlinear feature space's cardinality is extremely large, and it usually makes the matrix Σ_n rank deficient. The matrix-based KPCA circumvents the KPC by following the standard method in [15] and an eigendecomposition problem for the Gram matrix

$$\mathcal{K}_n \triangleq \phi(\mathcal{X}_n)^\top \phi(\mathcal{X}_n) = [\Downarrow_{i=1}^n [\Rightarrow_{j=1}^n \mathbf{K}(\mathbf{x}_i, \mathbf{x}_j)]] \in \mathbb{R}^{nq \times nq} \quad (3)$$

is done in practice to calculate the leading eigenvectors for Σ_n . Suppose that $(\lambda_w, \mathbf{v}_w)$ is an eigenpair of the matrix $\mathcal{K}_n$; that is, $\mathbf{v}_w \in \mathbb{R}^{nq \times 1}$ is a unit eigenvector with the corresponding eigenvalue λ_w as the w -th largest eigenvalue or $\mathcal{K}_n \mathbf{v}_w = \lambda_w \mathbf{v}_w$. Then the r most significant matrix-based KPC in feature space takes the matrix form of

$$\begin{aligned} \mathbf{U}_n &\triangleq [\Rightarrow_{w=1}^r \mathbf{U}_w] = \left[\Rightarrow_{w=1}^r \frac{1}{\sqrt{\lambda_w}} \phi(\mathcal{X}_n) \mathbf{v}_w \right] \\ &= \phi(\mathcal{X}_n) \mathbf{V}_n \Lambda_n^{-(1/2)} \in \mathbb{R}^{f \times r}, \end{aligned} \quad (4)$$

where $\mathbf{U}_w \in \mathbb{R}^f$ are the w -th columns of $\mathbf{U}_n$, $\mathbf{V}_n \in \mathbb{R}^{nq \times r}$ is the matrix with columns of $\mathbf{v}_w$, and $\Lambda_n \in \mathbb{R}^{r \times r}$ is a diagonal matrix whose diagonal elements are $\{\lambda_w: w = 1, 2, \dots, r\}$. Then the total scatter matrix has the form $\Sigma_n \triangleq \frac{1}{n} \mathbf{U}_n \Lambda_n \mathbf{U}_n^\top$. For a given matrix $\mathbf{Y} \triangleq [\Rightarrow_{i=1}^q \mathbf{y}_i] \in \mathbb{R}^{p \times q}$ as a test data, with a kernelized form $\phi(\mathbf{Y}) \in \mathbb{R}^{f \times q}$, its w -th principal component vector corresponding to ϕ is computed as

$$\begin{aligned}
\mathbf{p}_Y^w &\triangleq (\mathbf{U}_w)^\top \phi(\mathbf{Y}) \\
&= \frac{1}{\sqrt{\lambda_w}} (\mathbf{v}_w)^\top \phi(\mathcal{X}_n)^\top \phi(\mathcal{X}_n) \phi(\mathbf{Y}) \\
&= \frac{1}{\sqrt{\lambda_w}} (\mathbf{v}_w)^\top [\mathbb{J}_{i=1}^n \mathbf{K}(\mathbf{x}_i, \mathbf{Y})] \in \mathbb{R}^q.
\end{aligned} \tag{5}$$

Equation (5) is a critical factor for the proposed IMKPCA because an explicit form of the function ϕ is not required. That means the principal component vectors of a given matrix data onto the matrix-based KPC can be solved entirely by kernel functions. Based on equation (5), for one projection direction, the IMKPCA outputs a principal component vector, and the dimensionality of the vector is equal to the number of the columns of the input matrix. In comparison, the vector-based KPCA outputs a principal component value for one projection direction [11]. In the next section, we show how to update matrix-based KPC in the feature space based on the new data's projections.

3. Incremental Matrix-Based Kernel Principal Component Analysis

In this section, a recursive matrix-based KPC formulation is selected to improve the batch nature of matrix-based KPCA. We describe how the original matrix-based KPC can be updated in an incremental way. The detailed procedures and their superiority compared with the standard counterparts are shown in the following sections.

3.1. Recursive Formulation for Matrix-Based KPC. Assume that the current data has been given as a block matrix $\mathcal{X}_n \triangleq [\Rightarrow_{i=1}^n \mathbf{x}_i] \in \mathbb{R}^{p \times nq}$ and the new data $\mathbf{x}_{n+1} \in \mathbb{R}^{p \times q}$ is given for update. We commence with the recursion of the total scatter matrix:

$$\begin{aligned}
\Sigma_{n+1} &\triangleq \frac{1}{n+1} \sum_{i=1}^{n+1} (\phi(\mathbf{x}_i) \phi(\mathbf{x}_i)^\top) = \frac{n}{n+1} \Sigma_n \\
&\quad + \frac{1}{n+1} (\phi(\mathbf{x}_{n+1}) \phi(\mathbf{x}_{n+1})^\top).
\end{aligned} \tag{6}$$

Based on equation (4), $\phi(\mathbf{x}_{n+1})$ can be decomposed into a component parallel and a component orthogonal to $\mathbf{U}_n$; i.e.,

$$\phi(\mathbf{x}_{n+1}) = \mathbf{U}_n \mathbf{P}^{\mathbf{U}_n}(\phi(\mathbf{x}_{n+1})) + \mathbf{U}_n^\top \mathbf{P}^{\mathbf{U}_n^\top}(\phi(\mathbf{x}_{n+1})), \tag{7}$$

where $\mathbf{P}^{\mathbf{U}_n}(\phi(\mathbf{x}_{n+1}))$ is the matrix of $\phi(\mathbf{x}_{n+1})$ projecting onto the matrix-based KPC $\mathbf{U}_n$ and $\mathbf{P}^{\mathbf{U}_n^\top}(\phi(\mathbf{x}_{n+1}))$ is the matrix of $\phi(\mathbf{x}_{n+1})$ projecting onto the orthogonal complementary space of $\mathbf{U}_n$.

Based on the idea of KPCA [6, 18, 29], the principal component can be expanded approximately in terms of some training samples in the feature space. This means that $\phi(\mathbf{x}_{n+1})$ can be reconstructed by the subspace spanned by $\mathbf{U}_n$,

the previous matrix-based KPC, and $\mathbf{U}_n^\top \mathbf{P}^{\mathbf{U}_n^\top}(\phi(\mathbf{x}_{n+1}))$ can be modeled as noises. However, we have

$$\mathbf{P}_{\mathbf{x}_{n+1}}^{\mathbf{U}_n} \triangleq \mathbf{P}^{\mathbf{U}_n}(\phi(\mathbf{x}_{n+1})) = [\mathbb{J}_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w] \in \mathbb{R}^{r \times q}, \tag{8}$$

where $\mathbf{p}_{\mathbf{x}_{n+1}}^w$ can be calculated by replacing $\mathbf{Y}$ with $\mathbf{x}_{n+1}$ in equation (5). Then, the current matrix-based KPC can be obtained by rotating the previous matrix-based KPC to preserve the total scatter of $\mathcal{X}_{n+1} \triangleq [\Rightarrow_{i=1}^{n+1} \mathbf{x}_i] \in \mathbb{R}^{f \times (n+1)q}$ most faithfully.

3.2. Rotation of Matrix-Based KPC. The key observation is that the effect of $\phi(\mathbf{x}_{n+1})$ on the previous matrix-based KPC can be presented by a rotation. Then, based on equations (7) and (8), we have

$$\phi(\mathbf{x}_{n+1}) = \mathbf{U}_n \mathbf{P}_{\mathbf{x}_{n+1}}^{\mathbf{U}_n} = \mathbf{U}_n [\mathbb{J}_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w]. \tag{9}$$

Substituting the previous matrix-based KPCA result $\Sigma_n = (1/n) \mathbf{U}_n \Lambda_n \mathbf{U}_n^\top$ and equation (9) into equation (6), we get

$$\begin{aligned}
\Sigma_{n+1} &= \frac{1}{n+1} \mathbf{U}_n \Lambda_n \mathbf{U}_n^\top + \frac{1}{n+1} \mathbf{U}_n [\mathbb{J}_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w] [\mathbb{J}_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w]^\top \mathbf{U}_n^\top \\
&= \frac{1}{n+1} \mathbf{U}_n \{ \Lambda_n + [\mathbb{J}_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w] [\mathbb{J}_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w]^\top \} \mathbf{U}_n^\top.
\end{aligned} \tag{10}$$

Denote the eigendecomposition of the matrix $\Lambda_n + [\mathbb{J}_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w] [\mathbb{J}_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w]^\top$ as $\mathbf{W}_1 \mathbf{D}_1 \mathbf{W}_1^\top$, where $\mathbf{W}_1 \in \mathbb{R}^{r \times r}$ is an orthonormal matrix and $\mathbf{D}_1 \in \mathbb{R}^{r \times r}$ is a diagonal matrix; equation (10) becomes

$$\Sigma_{n+1} = \frac{1}{n+1} \mathbf{U}_n \mathbf{W}_1 \mathbf{D}_1 \mathbf{W}_1^\top \mathbf{U}_n^\top. \tag{11}$$

Consequently, based on $\Sigma_n = (1/n) \mathbf{U}_n \Lambda_n \mathbf{U}_n^\top$ and equation (11), the matrix-based KPCA system of the total scatter matrix equation (6) is recursively given by

$$\begin{cases} \mathbf{U}_{n+1} = \mathbf{U}_n \mathbf{W}_1 = \phi(\mathcal{X}_n) \mathbf{v}_n \Lambda_n^{-(1/2)} \mathbf{W}_1, & \mathbf{U}_{n+1} \in \mathbb{R}^{f \times r}, \\ \Lambda_{n+1} = \mathbf{D}_1, & \Lambda_{n+1} \in \mathbb{R}^{r \times r}. \end{cases} \tag{12}$$

In equation (12), $\mathbf{W}_1$ represents the directional variation of matrix-based KPC caused by $\phi(\mathbf{x}_{n+1})$ and $\mathbf{D}_1$ represents the component ratio change of the updated matrix-based KPC. From equation (12), we rotate matrix-based KPC to most faithfully preserve the total scatter of the given data when $\phi(\mathbf{x}_{n+1})$ is linearly independent on the previous matrix-based KPC.

3.3. Recursive Formulation with Mean Updating. For the sake of simplicity, the centralized assumption in feature space is provided in former analysis. However, the assumption is often invalid. The reason is that the mean matrix of the mapped data in the feature space always changes when new

data is added for updating. For discarding this assumption in feature space, we get

$$\begin{aligned}\Sigma_n &= \frac{1}{n} \sum_{i=1}^n \{\phi(\mathbf{x}_i) - \mathbf{M}_n\} \{\phi(\mathbf{x}_i) - \mathbf{M}_n\}^\top, \\ \Sigma_{n+1} &= \frac{1}{n+1} \sum_{i=1}^{n+1} \{\phi(\mathbf{x}_i) - \mathbf{M}_{n+1}\} \{\phi(\mathbf{x}_i) - \mathbf{M}_{n+1}\}^\top,\end{aligned}\quad (13)$$

where $\mathbf{M}_n \in \mathbb{R}^{f \times q}$ and $\mathbf{M}_{n+1} \in \mathbb{R}^{f \times q}$ denote the mean matrices of the previous mapped data and the current mapped data, respectively.

In the total scatter matrix Σ_{n+1} , the mapped matrices data $\phi(\mathcal{X}_{n+1}) \triangleq [\Rightarrow_{i=1}^{n+1} \phi(\mathbf{x}_i)] \in \mathbb{R}^{f \times (n+1)q}$ are centered with the current mean matrix $\mathbf{M}_{n+1}$ rather than the previous mean matrix $\mathbf{M}_n$ that is used in the total scatter matrix Σ_n . Hence, the recursion formulation equation (6) is invalid for application. Fortunately, we have

$$\mathbf{M}_{n+1} = \frac{n}{n+1} \mathbf{M}_n + \frac{1}{n+1} \phi(\mathbf{x}_{n+1}). \quad (14)$$

Following the idea of the scatter matrix update in incremental learning [30], it can be easily achieved that the recursive formulation with the mean update for the total scatter matrix of matrix-based KPC has the following form:

$$\Sigma_{n+1} = \frac{n}{n+1} \Sigma_n + \frac{1}{(n+1)^2} \{\phi(\mathbf{x}_{n+1}) - \mathbf{M}_n\} \{\phi(\mathbf{x}_{n+1}) - \mathbf{M}_n\}^\top. \quad (15)$$

The recursive matrix-based KPC formulation of equation (15) can be computed following the approach in the previous subsection, and the update formulations can be obtained completely by imitating equation (7) ~ equation (12). The mapped matrix should be projected onto equations (7) and (8), i.e., the eigendecomposition of matrix $\Lambda_n + [\Downarrow_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w] [\Downarrow_{w=1}^r \mathbf{p}_{\mathbf{x}_{n+1}}^w]^\top$ in equation (10) should be replaced by the eigendecomposition of matrix $\Lambda_n + [\Downarrow_{w=1}^r (\mathbf{p}_{\mathbf{x}_{n+1}}^w - \mathbf{p}_{\mathbf{M}_n}^w)] [\Downarrow_{w=1}^r (\mathbf{p}_{\mathbf{x}_{n+1}}^w - \mathbf{p}_{\mathbf{M}_n}^w)]^\top$ for updating, where $\mathbf{p}_{\mathbf{M}_n}^w \in \mathbb{R}^q$ is the w -th principal component vector of $\mathbf{M}_n$ and can be calculated by equation (5) as follows:

$$\mathbf{p}_{\mathbf{M}_n}^w = \frac{1}{\sqrt{\lambda_w}} (\mathbf{v}^w)^\top [\Downarrow_{i=1}^n \mathbf{K}(\mathbf{x}_i, \mathbf{M}_n)] = \frac{1}{n} \sum_{i=1}^n \mathbf{p}_{\mathbf{x}_i}^w. \quad (16)$$

The recursive formulation equation (12) comes to one with the mean update for the proposed matrix-based KPC.

In the subsequent experiment section, for a given matrix-based dataset, a portion of matrix data are chosen as original data $\mathcal{X}_n \triangleq [\Rightarrow_{i=1}^n \mathbf{x}_i] \in \mathbb{R}^{p \times nq}$ and the remaining matrix data are used as new data $\mathbf{x}_{n+1} \in \mathbb{R}^{p \times q}$ successively to update the current principal components.

4. Experiments

Several experiments were conducted to examine four properties of the proposed method: (1) the effectiveness of approximating KPC under stationary data; (2) the influence

of parameter on the proposed method under MNIST and Fashion MNIST databases; (3) the superiority compared with several reference methods under some well-known databases, the Fashion MNIST [31], ORL [32], YaleA [33], Extended [34], PF01 [35], and COIL 100 [36]; and (4) the efficiency compared with reference methods for fashion products from the Fashion MNIST database.

In order to measure the accuracy of the proposed method in approximating MKPCA and to assess the quality of the solution objectively, a distance measure based on the angles between principal kernel components is employed in this section as follows:

$$\theta_w = \arccos(|\mathbf{v}_w \cdot \mathbf{v}_w^*|), \quad (17)$$

where $\mathbf{v}_w^*$ is the ground truth of the w -th principal kernel component computed by a standard MKPCA method and $\mathbf{v}_w$ is the w -th extracted kernel principal component computed by the proposed method.

For visualizing the capacity of IMKPCA in describing the construction of given data, two-dimensional stationary set and matrix-based dataset both are necessary because a two-dimensional datum is a special matrix datum. In Sections 4.1 and 4.2, the experiments are mainly based on two generated two-dimensional datasets. In Sections 4.3 and 4.4, the experiments are based on matrix-based datasets. The Gaussian kernel function $k(\mathbf{x}, \mathbf{y}) = \exp\{-\sigma^{-1} \|\mathbf{x} - \mathbf{y}\|^2\}$ is used in this section. For two types of matrix data $\mathbf{X} \in \mathbb{R}^{p \times q}$ and $\mathbf{Y} \in \mathbb{R}^{p \times q}$, the dot product matrix $\mathbf{K} \in \mathbb{R}^{q \times q}$ of them is $\mathbf{K}(\mathbf{X}, \mathbf{Y}) = [\Downarrow_{i=1}^q [\Rightarrow_{j=1}^q \exp\{-\sigma^{-1} \|\mathbf{x}_i - \mathbf{y}_j\|^2\}]]$ as that mentioned in Section 2.

4.1. Stationary Data Approximation Accuracy. The experiment is carried out on the 90 toy data [10] to serve in testing the effectiveness of the proposed method in approximation accuracy. The data are generated in stationary environment in the following way: x -values have uniform distribution in $[-1, 1]$, $x_i \sim U[-1, 1]$, and y -values are generated from $y_i = x_i^2 + \xi$, $i = 1, 2, \dots, 90$, where ξ is normal noise with standard deviation 0.2. We calculate the principal kernel components by standard MKPCA [15], vector-based KPCA [6], and the proposed method, respectively. The values of the test data projection onto the extracted kernel principal components are given in Figures 1 and 2. Figure 1 contains lines of constant principal component value (contour lines). Those contour lines indicate the structure of the data in the feature space. The red data are the total sample, and the constant projection values onto the first five KPC computing by the three methods are shown. Figure 2 illustrates the results of the proposed method in the computation process. The green dots are the original data, red dots are the appended data, and blue contours depict the constant projection values onto the first three KPC. It can be visually seen that the KPC exacted by IMKPCA converges to the ground truth KPC, even beginning with an unsatisfactory initialization. Figures 1 and 2 show that the proposed method is with equal authentic accuracy but a wider range of original data

FIGURE 1: Two-dimensional examples. From top to bottom, the results of vector-based KPCA, standard MKPCA, and the proposed method, respectively. From left to right, projection contour lines onto the first five kernels principal component (red points denote input data).

choice compared with MKPCA and vector-based KPCA for computing the principal kernel components of the generated stationary data.

4.2. Influence of Parameters. In this experiment, we investigate how the number of original data affects the effectiveness of IMKPCA. The measure equation (17) is used as a distance between subspace computed by IMKPCA and the ground truth computed by MKPCA. Two matrix-based stationary data are used. One consists of the first 200 training images of “T-shirt/top” in Fashion MNIST, and the other contains the first 120 sets of training data of digit “0” from the MNIST dataset. For each stationary data, two results are shown with $n = 30$ and $n = 50$ as original data capacity, respectively. Equation (12) is used to update the subspace in feature space, and the distance between the updated subspace and the standard one is recorded with radian. Figure 3 shows the result of each iteration under different n , in which the

distances between the first three kernel principal components are computed as a distance between two subspaces (represented by “MKPC-1”, “MKPC-2,” and “MKPC-3,” respectively).

Figures 3(a) and 3(b) show the results for image data of “T-shirt/top.” Figures 3(c) and 3(d) show the results for matrix data of digit “0.” In each subfigure, the result shows that the KPC extracted by IMKPCA converges to the ground truth when iteration process equation (12) continues. Figure 3 shows that setting n to a large value helps IMKPCA improve the accuracy of approximating the ground truth.

4.3. Study of the Spatial Statistics in the Matrix Data. The experiment concerns the advantage of IMKPCA in studying the spatial statistics in the matrix data. We conduct the proposed method on digits from the MNIST dataset and extract the matrix-based principal component on both column-lifted and row-lifted stages using the Gaussian

FIGURE 2: Two-dimensional examples by IMKPCA. The first and sixth columns: original input data and the total sample. From the second to the fourth column: the result of update processes. From top to bottom: the projection value onto the first three kernels principal component.

kernel function. The column-lifted stage means each column of the matrix data is kernelized as $\phi(\mathbf{X}) = [\Rightarrow_{i=1}^q \phi(\mathbf{x}_i)]$ as that mentioned in Section 2. The row-lifted stage means processing the feature matrix extracted by the proposed method in the column-lifted stage as input. The result of the row-lifted stage represents the final extracted spatial statistics of the given dataset. Each digit is a 28×28 matrix, while we kept only the first sixteen principal components that are displayed as a 16×16 matrix. For comparing the proposed method with the vector-based KPCA, the vector should be an input. However, an image of the digit is represented as a matrix. Hence the images are vectorized into vectors before being fed into vector-based KPCA, and the feature of each digit, a sixteen-dimension vector, consists of the first sixteen principal component values. Figure 4 shows ten example images of digit “3” and its matrix-based principal components computed by the proposed IMKPCA. Figure 5 shows the result of vector-based KPCA for

vectorized digit “3,” in which the vertical axis shows the projection values of the vectorized digit onto the principal component, and the horizontal axis is the index of the ten samples in Figure 4. The projection value of digit “3” onto the i -th principal component is represented by “VPC $_i$.” Figure 6 shows ten example images of digits “1” to “5”; two samples for each digit and their matrix-based principal components are computed by the proposed methods. Figures 4–6 show that the matrix-based features can provide more significant information than that of vector-based ones. The vector-based KPCA outputs a vector as a feature, while the IMKPCA outputs a matrix. Typically, the matrix-based features of digit “3” present as a distinctive white line in matrix in Figure 4; the digits “1” to “5” are present as lines in Figure 6, while the vector-based ones are numerical values in Figure 5. In particular, digit “4” is present with two lines as it has two distinct line structures; digits “2” and “5” are similar in column-lifted stage (the second row in Figure 6) and

FIGURE 3: Angles between the first three kernel principal components (red, green, and blue) computed by the proposed method and the standard MKPCA. (a, b) Results of two-dimensional example for data capacity 200; n is 30 and 50, respectively; $\sigma = 1$ for Gauss function. (c, d) Results of digit “0” matrix set for data capacity 120, n is 30 and 50, respectively; $\sigma = 1$ for Gauss function.

different in row-lifted stage (the third row in Figure 6). The reason is that they are two different digits, but the vertical flip of digit “5” is similar to that of digit “2.”

In order to show the advantage of the spatial statistics for the matrix data as quantitative analysis, two image tests, a digit recognition test and an image database classification test, are conducted. For digits test, the first 100 training

images and the first 500 test images of each digit in MNIST serve to test the proposed method’s capability in digits recognition. The experiment is repeated for the dataset with different training data ratios as original data for computing matrix-based KPC in the feature space. The extracted features in the column-lifted and row-lifted stages are used to digitize the digit recognition under the nearest distance

FIGURE 4: The spatial feature extracted for digit “3” in MNIST data. The first row: ten random samples; the second row: results of the proposed method by column-lifted IMKPCA; the third row: results of row-lifted IMKPCA for the second row; input: image matrix of size 28×28 ; output: feature matrix of size 16×16 .

FIGURE 5: Feature extracting for digit “3” in MNIST data by vector-based KPCA. The vertical axis shows projection values of 10 samples onto the kernel principal component.

classifier. For Gauss function, $\sigma = 100$ in column-lifted stage and $\sigma = 1$ in row-lifted stage for the proposed method, and $\sigma = 100$ for vector-based KPCA are chosen to retain the total scatter of the data in the feature space.

Table 1 presents the average digit recognition rates and the variances (the value follow $\pm$) for the chosen dataset. The ratios $r = 0.1$, $r = 0.2$, $r = 0.3$, and $r = 0.4$ correspond to $n = 10$, $n = 20$, $n = 30$, and $n = 40$ as the capacity of original data, respectively. In Table 1, the result of the vector-based method is unique because it does not need to choose original data for updating. Table 1 shows that IMKPCA has a prominent advantage compared with vector-based KPCA. The reason is that the vectorization ruins the pixels’ spatial

structure that defines the image of digit from the MNIST dataset. Meanwhile, IMKPCA brings back the structure. In particular, the results obtained by features in the row-lifted stage are better than those of column-lifted one due to the extracted spatial statistics for matrix data. Additionally, using more samples as original data in the training period increases the recognition performance. For image database tests, the Fashion MNIST, ORL, YaleA, Extended YaleB, and COIL 100 databases are chosen to evaluate the classification. Figure 7 shows some image samples from each database. The Fashion MNIST database consists of 60,000 training images and 10,000 test images of 10 fashion products, such as “T-shirt/top,” “Trouser,” and “Dress”. In

FIGURE 6: Feature extracting for digits “1” to “5” in MNIST data. The first row: digit samples; the second row: results of the proposed method by column-lifted IMKPCA; the third row: results of row-lifted IMKPCA for the second row; input: image matrix of size 28×28 ; output: feature matrix of size 16×16 .

TABLE 1: Digits recognition rates for MNIST dataset.

Capacity of training data	$r = 0.1$ $n = 10$	$r = 0.2$ $n = 20$	$r = 0.3$ $n = 30$	$r = 0.4$ $n = 40$
Matrix-based method (column-lifted, $\sigma = 100$)	74.3 ± 2.2	78.4 ± 2.0	82.1 ± 3.4	82.4 ± 2.6
Matrix-based method (row-lifted, $\sigma = 1$)	82.6 ± 3.0	94.0 ± 2.1	96.7 ± 1.5	98.2 ± 1.1
Vector-based method ($\sigma=100$)		81.7 ± 2.3		

the results with this database, the first 200 training images of each fashion product, 2,000 training images, are chosen. The test images are all used for the test. The ORL database contains 400 frontal images of 40 subjects, and ten images were taken with a few movements. The YaleA database consists of 165 frontal images of 15 subjects with different emotions. Half of each subject’s ORL and Yale images are chosen for training and the remaining for the test in the experiment. YaleB face database consists of over 2,200 frontal images of 38 subjects with different illuminations and the same position. Ten images of each subject are randomly chosen for training and the others for testing. The COIL 100 database consists of 7200 images of 72 objects. In the experiment, ten images of each object are chosen for training, and the remaining images are used for testing. Each image from COIL 100 is resized to a 28×28 matrix in the experiment for computer resource restriction. Table 2 shows the average database recognition rates and variances under ten different training sample sequences for the given six databases, in which “NC” indicates “Noncomputable.” The proposed IMKPCA performs better than the vector-based PCA, vector-based KPCA, and 2D PCA to recognition rate for the five databases, as shown in Table 2. The proposed IMKPCA and the MKPCA show similar performances for the test databases. The reason is that the proposed method is methodologically consistent with MKPCA, and it is incrementally approximating the ground truth of MKPCA. In comparison, the proposed method is more efficient and more computable than MKPCA. The major reason is that the size of the eigendecomposition of MKPCA still depends on the datasets’ size, such as the beyond computation for the COIL 100 database and Fashion MNIST database in Table 2.

4.4. Computational Efficiency Comparison. This section compares the computation time and resource consumption between the MKPCA, vector-based KPCA, and IMKPCA. The Fashion MNIST database is employed to compare the proposed method’s feature extraction efficiency with that of reference ones. For exact and reasonable comparison, the computer operating environment is the same, and the three methods are executed on a computer with 4G RAM and 3.2 G CPU. The input dataset is extracted from the training samples of “T-shirt/top.” MKPCA, vector-based KPCA, and IMKPCA are executed for datasets with different capacities. Since the resource consumption of MKPCA is large, we consider the case that computing time is longer than 400 seconds and memory overflow with 4G RAM as “Non-computable.” For the proposed method, the computing time in this example mainly depends on equation (12), in which $\mathbf{M}_1$ and $\mathbf{D}_1$ are computed with computational complexity $O(r^3)$. The computing times of the standard MKPCA and the vector-based KPCA depend on eigendecomposition with computational complexity $O(n^2)$ and $O(n^2 pq)$, respectively, where n is the number of pieces of training data and p and q are the row size and column size of the given matrix data, respectively. Figure 8 shows the maximum value, minimum value, and mean value of angles between the extracted kernel principal component and the ground truth of MKPCA for images of 10 fashion products. It indicates that the proposed method’s result can approximate the real projection matrix well for Fashion MNIST. In particular, the minimum values and mean values for ten fashion products are mostly smaller than 0.05. Figure 9(a) shows the computational time for Fashion MNIST database under different capacities. It indicates that the vector-based method and the proposed

FIGURE 7: Image samples from chosen databases. From top to bottom, images of two objects from (a) Fashion MNIST, (b) YaleA, (c) ORL, (d) COIL, and (e) Extended YaleB, respectively; each object shows ten samples only.

TABLE 2: Recognition rates for different databases.

Algorithm	Fashion	ORL	YaleA	YaleB	COIL
V-PCA	72.4 ± 3.7	92.3 ± 1.3	84.1 ± 2.4	25.6 ± 4.3	83.5 ± 3.5
V-KPCA	76.5 ± 2.2	92.6 ± 1.1	86.7 ± 2.6	40.4 ± 3.8	83.7 ± 3.3
2D PCA	82.6 ± 2.1	92.8 ± 3.4	84.3 ± 3.7	67.9 ± 2.6	82.5 ± 3.4
MKPCA	NC	93.7 ± 2.3	84.6 ± 3.4	90.1 ± 1.9	NC
IMKPCA	94.1 ± 3.3	93.5 ± 2.6	85.2 ± 3.1	89.9 ± 2.7	86.4 ± 2.3

method need much less computational time and resource consumption than MKPCA under the same computer system with software Matlab2014. Figure 9(b) shows the computational times for the proposed method under

different ratios of original data to the computed data. For each curve, the percentage of 1000 fashion products image is used to perform the proposed IMKPCA method. Figures 8 and 9 show that the proposed method needs more time than

FIGURE 8: Angles between the proposed method and the standard MKPCA for 10 fashion products in Fashion MNIST (metric: radian).

FIGURE 9: Computing time for Fashion MNIST database. (a) Computational time under different capacities. Red triangle denotes situations where computing time is longer than 400 seconds in addition to memory overflow on 4G RAM, 3.2 GCPU. (b) Computational time under different ratio of original data.

the vector-based KPCA method but the proposed method occupies some advantages as a whole; for example, it can retain the spatial structure of the data in the matrix and it can provide more information than the vectored-based method with a better property. The proposed IMKPCA method nearly linearly increases, while the standard MKPCA increased sharply according to the training samples.

5. Conclusion

In this paper, an incremental matrix-based kernel subspace method is proposed. The main innovation made by the proposed method is the idea of rotating matrix-based kernel principal components to most faithfully preserve the total scatter of the given data when new data are appended. The proposed method is still subject to the KPCA methodology, which makes it more reliable than most of the existing algorithms for improvement. For the proposed method, the feature extractor should also be associated with the eigen-decomposition problem, while the eigenvalue has a different dimension compared with that of the vector-based KPCA method. The experiment results show that feature extraction based on the proposed method is more efficient than the standard MKPCA and the vector-based KPCA. On the other hand, the proposed method obtains efficient feature extraction relying on the linear relationship between the current principal component and new appended data. It means that the proposed method is efficient in processing stationary dataset. Future work will concentrate on how to adapt the principal components to the nonstationary matrix dataset.

Data Availability

The data that support the findings of this study are available upon request from the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key Research and Development Program (no: 2019YBF1600700).

References

- [1] J. Li, S. Sun, H. Yin, P. Dawson, and R. Doss, "SEPN: a sequential engagement based academic performance prediction model," *IEEE Intelligent Systems*, vol. 99, p. 1, 2020.
- [2] S. Sun, N. Akhtar, H. Song, C. Zhang, J. Li, and A. Mian, "Benchmark data and method for real-time people counting in cluttered scenes using depth sensors," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3599–3612, 2019.
- [3] H.-S. Song, S.-N. Lu, X. Ma, Y. Yang, X.-Q. Liu, and P. Zhang, "Vehicle behavior analysis using target motion trajectories," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 8, pp. 3580–3591, 2014.
- [4] J. Li, T. Cai, K. Deng, X. Wang, T. Sellis, and F. Xia, "Community-diversified influence maximization in social networks," *Information Systems*, vol. 92, Article ID 101522, 2020.
- [5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of The ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [6] T.-H. Chan, K. Jia, S. Gao et al., "PCANet: a simple deep learning baseline for image classification?" *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5017–5032, 2015.
- [7] S. Wang, Y. Jiang, F.-L. Chung, and P. Qian, "Feedforward kernel neural networks, generalized least learning machine, and its deep learning with application to image classification," *Applied Soft Computing*, vol. 37, pp. 125–141, 2015.
- [8] Y. Cho and L. K. Saul, "Kernel methods for deep learning," *Advances in Neural Information Processing Systems*, vol. 22, pp. 342–350, 2009.
- [9] D. K. Hu, A. S. Ye, L. Li, and L. Zhang, "Recognition of facial expression via kernel PCA network," *Applied Mechanics and Materials*, vol. 632, pp. 498–501, 2014.
- [10] S. Bernhard, S. Alexander, and M. Klaus-Robert, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 44, 1998.
- [11] P. P. Pokharell, W. Liu, and J. C. Principe, "Kernel least mean square algorithm with constrained growth," *Signal Processing*, vol. 89, no. 3, pp. 257–265, 2009.
- [12] Q. Jiang and X. Yan, "Parallel PCA-KPCA for nonlinear process monitoring," *Control Engineering Practice*, vol. 80, pp. 17–25, 2018.
- [13] J. Huang and X. Yan, "Related and independent variable fault detection based on KPCA and SVDD," *Journal of Process Control*, vol. 39, pp. 88–99, 2016.
- [14] Z. Liu, W. Guo, J. Hu, and W. Ma, "A hybrid intelligent multi-fault detection method for rotating machinery based on RSGWPT, KPCA and twin SVM," *Isa Transactions*, vol. 66, pp. 249–261, 2017.
- [15] V. Verdult and M. Verhaegen, "Kernel methods for subspace identification of multivariable ltv and bilinear systems," *Automatica*, vol. 41, no. 9, pp. 1557–1565, 2005.
- [16] J. Yang, D. Zhang, A. Frangi, and J. Yang, "Two-dimensional PCA: a new approach to appearance-based face representation and recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 131–137, 2004.
- [17] K. I. Kim, M. Franz, and B. Schölkopf, "Iterative kernel principal component analysis for image modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 9, pp. 1351–1366, 2005.
- [18] S. Gunter, N. N. Schraudolph, and S. V. N. Vishwanathan, "Fast iterative kernel principal component analysis," *Journal of Machine Learning Research*, vol. 8, pp. 1893–1918, 2007.
- [19] T. J. Chin, K. Schindler, and D. Suter, "Incremental kernel SVD for face recognition with image sets," in *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition 2006*, pp. 461–466, Southampton, UK, April 2006.
- [20] T. J. Chin and D. Suter, "Incremental kernel PCA for efficient non-linear feature extraction," in *Proceedings of the British Machine Vision Conference 2006*, pp. 939–948, Coventry, UK, September 2006.
- [21] T.-J. Chin and D. Suter, "Incremental kernel principal component analysis," *IEEE Transactions on Image Processing*, vol. 16, no. 6, pp. 1662–1674, 2007.
- [22] V. Franc and V. Hlavac, "greedy algorithm for a training set reduction in the kernel methods," in *Proceedings of the*

- Computer Analysis of Images and Patterns*, pp. 426–433, Groningen, The Netherlands, August 2003.
- [23] S. Das, A. Abraham, and A. Konar, “Automatic kernel clustering with a multi-elitist particle swarm optimization algorithm,” *Pattern Recognition Letters*, vol. 29, no. 5, pp. 688–699, 2008.
 - [24] M. Ding, Z. Tian, and H. Xu, “Adaptive kernel principal component analysis,” *Signal Processing*, vol. 90, no. 5, pp. 1542–1553, 2010.
 - [25] Y. Choi, S. Ozawa, and M. Lee, “Incremental two-dimensional kernel principal component analysis,” *Neurocomputing*, vol. 134, pp. 280–288, 2014.
 - [26] Q. Wang, Q. Gao, X. Gao, and F. Nie, “Optimal mean two-dimensional principal component analysis with f-norm minimization,” *Pattern Recognition*, vol. 68, Article ID 286294, 2017.
 - [27] L. Khedher, J. Ramírez, J. M. Górriz, A. Brahim, and F. Segovia, “Early diagnosis of Alzheimer’s disease based on partial least squares, principal component analysis and support vector machine using segmented MRI images,” *Neurocomputing*, vol. 151, pp. 139–150, 2015.
 - [28] Y. Xu, D. Zhang, F. Song, J.-Y. Yang, Z. Jing, and M. Li, “A method for speeding up feature extraction based on KPCA,” *Neurocomputing*, vol. 70, no. 4–6, pp. 1056–1061, 2007.
 - [29] X. Zhang, L. Hu, L. Zhang, and X. Fu, “An efficient multiple kernel computation method for regression analysis of economic data,” *Neurocomputing*, vol. 118, pp. 58–64, 2013.
 - [30] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, “Incremental learning for robust visual tracking,” *International Journal of Computer Vision*, vol. 77, no. 1–3, pp. 125–141, 2008.
 - [31] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” 2017, <https://arxiv.org/abs/1708.07747v2>.
 - [32] AT&T, “The ORL database of faces,” 2020, <http://cam-orl.co.uk/facedatabase.html>.
 - [33] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Yi Ma, “Robust face recognition via sparse representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.
 - [34] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, “From few to many: illumination cone models for face recognition under variable lighting and pose,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 643–660, 2001.
 - [35] H. Dong and N. Gu, “Asian face image database Pf01,” 2001.
 - [36] S. Nayar, “Columbia object image library (Coil100),” 1996.