Exploration of PCA family for handwritten digit classification

Qichao Li
Department of Mechanical and Aerospace Engineering
The Hong Kong University of Science and Technology
qlich@connect.ust.hk

Haohan Huang
Department of Mathematics
The Hong Kong University of Science and Technology
hhuangbu@connect.ust.hk

1 Introduction

As what we have learnt from the lectures, the principal component analysis (PCA) is usually utilized as a method to undergo dimension reduction, projecting high-dimensional dataset such as image, speech, text and video into a low-dimensional affine space. It was first introduced by Pearson in [2], where a plane (or hyperplane) is used to separate two classes of points in an affine space. Afterward, many variations of this approach, i.e. SparsePCA [1], KernelPCA [3], have been proposed to improve its related performance. In this mini-project, we try to compare the performance of different kinds of PCA, including original PCA, sparse PCA and kernel PCA with different kinds of kernel functions, by utilizing the data reduced through these methods to carry out the classification tasks. Here we select the random forest classifier considering its simplicity.

2 Dataset

We select the popular hand-written digits dataset as the dataset to do the data dimensionality reduction and classification. It is available on the website

`http://www-stat.stanford.edu/~tibs/ElemStatLearn/datasets/zip.digits/`,

containing hundreds of images of 10 handwritten digits $0, 1, 2, \ldots, 9$ (see Table 1). Each digit image is represented as a 16×16 grayscale image, resulting in 256 features (pixel values) for each image. Some examples of images of these hand-written digits are shown in Fig. 1 to illustrate the variety of writing styles and variations in the data.

Table 1: The number of image samples for each digit

Digit	0	1	2	3	4	5	6	7	8	9	Total
# samples	1193	1004	730	657	651	555	663	644	541	643	7281

3 Methodology

3.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a statistical technique used to reduce the dimensionality of a dataset by identifying a set of new variables, called principal components, that capture the

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

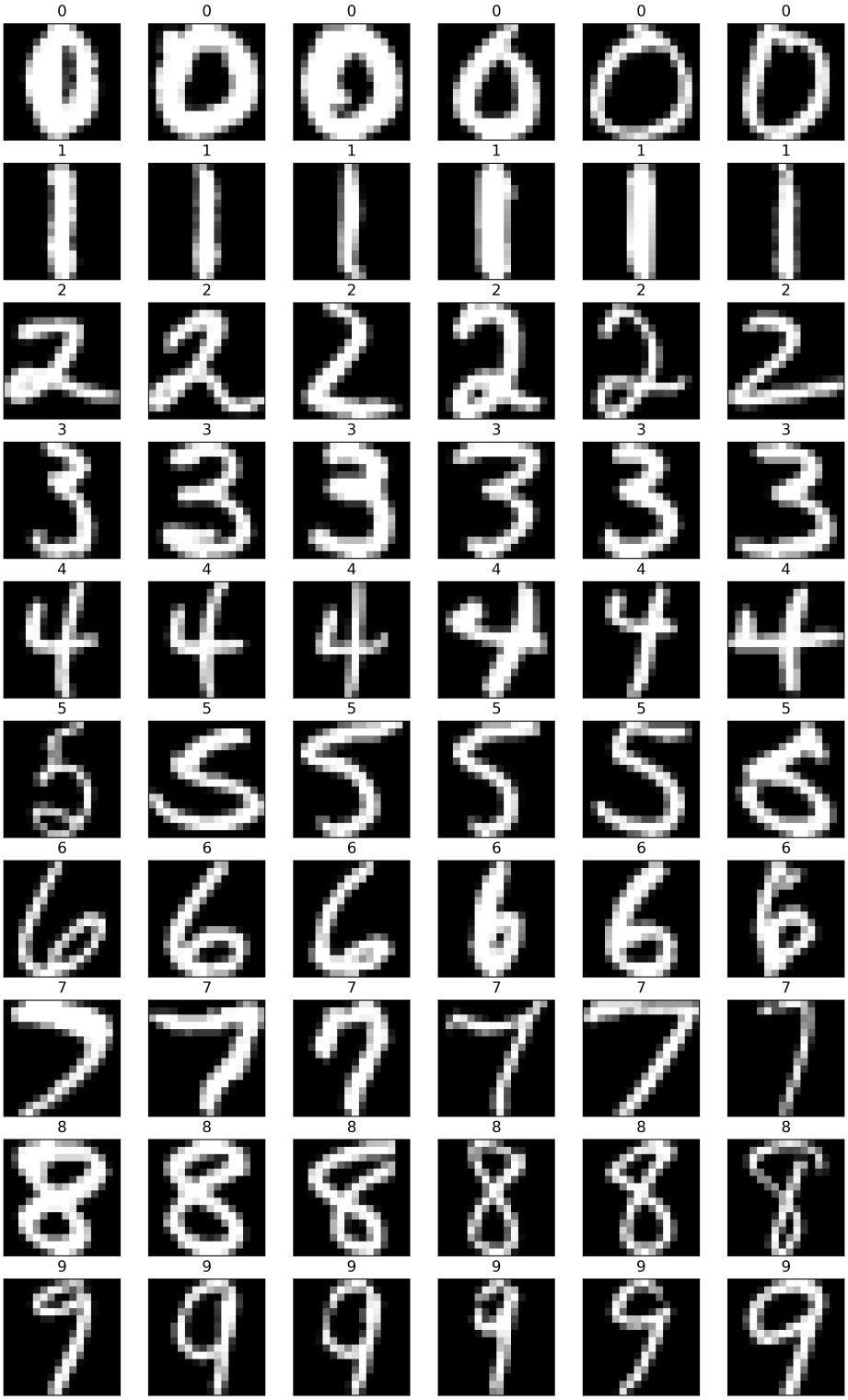

Figure 1: Examples of images for each digit.

most important information in the original data. The goal of PCA is to find a lower-dimensional representation of the data that still retains most of its variability.

PCA works by identifying the directions in which the data varies the most, and projecting the data onto these directions. The first principal component is the direction that captures the largest amount of variability in the data, and subsequent principal components capture the remaining variability in decreasing order. The principal components are orthogonal to each other, meaning that they are uncorrelated.

PCA can be used in a variety of applications, such as data compression, data visualization, and feature extraction. It is commonly used in fields such as image processing, finance, and genetics. PCA is a powerful tool for exploratory data analysis and can help to identify underlying patterns or relationships in complex datasets.

The mathematical formulation of PCA is given by the following optimization problem:

$$\max_{\mathbf{w}} \frac{\mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \tag{1}$$

subject to:

$$|\mathbf{w}|_2 = 1 \tag{2}$$

where $\mathbf{X}$ is the data matrix and $\mathbf{w}$ is the weight vector of the linear combination that defines the principal component.

3.2 Sparse Principal Component Analysis (SPCA)

Sparse Principal Component Analysis (SPCA) is a variation of Principal Component Analysis (PCA) that aims to identify a set of principal components that have sparse loadings, meaning that they are composed of a small number of non-zero weights. The goal of SPCA is to find a lower-dimensional representation of the data that is more interpretable and easier to use for downstream tasks.

SPCA works by adding a sparsity constraint to the optimization problem that is solved in PCA. Specifically, instead of maximizing the variance of the linear combinations subject to a unit-norm constraint as in PCA, SPCA adds a constraint that limits the number of non-zero weights in the linear combinations. This leads to a solution that is sparse and easier to interpret.

The mathematical formulation of SPCA is given by the following optimization problem:

$$\max_{\mathbf{w}} \frac{\mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w}}{\mathbf{w}^T \mathbf{w}} - \lambda |\mathbf{w}|_1 \tag{3}$$

subject to:

$$|\mathbf{w}|_2 = 1 \tag{4}$$

where $\mathbf{X}$ is the data matrix, $\mathbf{w}$ is the weight vector of the linear combination that defines the principal component, and λ is a sparsity parameter that controls the number of non-zero weights. The influence of sparsity in SPCA can have several effects on the analysis and interpretation of the results:

- Improved interpretability: Sparse principal components are typically easier to interpret than traditional principal components, as they tend to have fewer contributing variables. This can help to identify the most important features in the data and to gain insights into the underlying patterns and relationships.
- Increased computational efficiency: Sparse principal components can be computed more efficiently than traditional principal components, as the optimization problem in SPCA is simpler and has fewer constraints.
- Reduced overfitting: The sparsity constraint in SPCA can help to prevent overfitting, which occurs when the model fits the noise in the data rather than the underlying signal. By limiting the number of non-zero weights, SPCA can identify a more robust and generalizable solution.

- Trade-off between sparsity and variance: Increasing the sparsity parameter in SPCA can lead to a trade-off between sparsity and variance, as the solution becomes more sparse but may capture less of the variability in the data. Finding an optimal balance between sparsity and variance is a key consideration in using SPCA effectively.

SPCA is a powerful tool for dimensionality reduction and feature extraction, and can help to identify underlying patterns or relationships in complex datasets with a small number of interpretable features.

3.3 Kernel Principal Component Analysis (KPCA)

Kernel Principal Component Analysis (KPCA) is a non-linear extension of Principal Component Analysis (PCA) that uses a kernel function to map the data into a higher-dimensional feature space. KPCA can capture non-linear relationships between variables and is particularly useful when the data is not linearly separable.

KPCA works by first applying a kernel function to the data to obtain a kernel matrix, which measures the similarity between each pair of data points. The kernel matrix is then used in place of the covariance matrix in the PCA algorithm to identify the principal components.

The mathematical formulation of KPCA is similar to that of PCA, but with the addition of a kernel function κ:

$$\max_{\mathbf{w}} \frac{\mathbf{w}^T \mathbf{K} \mathbf{w}}{\mathbf{w}^T \mathbf{K} \mathbf{w}} \tag{5}$$

subject to:

$$|\mathbf{w}|_2 = 1 \tag{6}$$

where $\mathbf{K}$ is the kernel matrix defined by $\mathbf{K}_{i,j} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are the data samples.

KPCA can be used with a variety of different kernel functions, including the linear kernel, polynomial kernel, Gaussian (RBF) kernel, and sigmoid kernel. The choice of kernel function is a crucial part of the algorithm as it determines the non-linear mapping of the data into a higher-dimensional feature space. The kernel function measures the similarity between two data points in the feature space and is used to compute the kernel matrix, which replaces the covariance matrix in the PCA algorithm.

There are several kernel functions that can be used in KPCA, including:

- Linear kernel:
$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j \tag{7}$$
- Polynomial kernel:
$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i^T \mathbf{x}_j + c)^d, \tag{8}$$
where γ, c, and d are kernel parameters.
- Gaussian (RBF) kernel:
$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{2\sigma^2}\right), \tag{9}$$
where σ is a kernel parameter.
- Laplacian kernel:
$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{|\mathbf{x}_i - \mathbf{x}_j|}{\sigma}\right), \tag{10}$$
where σ is a kernel parameter.
- Sigmoid kernel:
$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \mathbf{x}_i^T \mathbf{x}_j + c), \tag{11}$$
where γ and c are kernel parameters.

The choice of kernel function depends on the nature of the data and the specific application. The linear kernel is appropriate when the data is linearly separable, while non-linear kernel functions such as the Gaussian kernel and Laplacian kernel are more suitable for non-linearly separable data.

Once the kernel matrix is computed using the chosen kernel function, KPCA proceeds in the same way as PCA, by computing the eigenvectors and eigenvalues of the kernel matrix and projecting the data onto the principal components. The resulting principal components are non-linear combinations of the original input variables and can capture non-linear relationships between variables in the data.

KPCA has a wide range of applications, including in areas such as image and speech recognition, bioinformatics, and finance. In image and speech recognition, KPCA can be used to extract features that capture the non-linear relationships between pixels or audio signals. In bioinformatics, KPCA can be used to identify patterns in gene expression data. In finance, KPCA can be used to identify a small number of factors that explain the variation in asset returns.

Overall, KPCA is a powerful technique for dimensionality reduction and feature extraction that can capture non-linear relationships between variables and improve the interpretability of complex data.

In this project, we choose the linear kernel, Gaussian kernel and sigmoid kernel to compare their perfomance.

3.4 Random Forest (RF)

Unlike the above traditional methods, Random Forest is a machine learning algorithm that uses an ensemble of decision trees to predict the output variable. Each decision tree in the ensemble is trained on a random subset of the training data and a random subset of the input features. The predicted output of the random forest is the mode of the predicted outputs of the individual trees.

Random Forest is a powerful and widely used algorithm that can handle complex datasets with high-dimensional input variables. It has several advantages over other machine learning algorithms, such as robustness to noise and outliers, non-parametric, feature importance, and ensemble learning.

Random Forest has a wide range of applications, including in areas such as image and speech recognition, bioinformatics, and finance. In image and speech recognition, Random Forest can be used to classify images or audio signals based on their features. In bioinformatics, Random Forest can be used to classify gene expression data or to predict drug efficacy. In finance, Random Forest can be used to predict stock prices or to identify fraudulent transactions.

Overall, Random Forest is a powerful and versatile algorithm that can be used for a wide range of prediction tasks and can handle complex datasets with high-dimensional input variables.

4 Experiments

We split the dataset into training and testing sets, which respectively account for 80% and 20% of the data. Then the original PCA, sparse PCA with sparity $1, 10, 30$, and kernel PCA with linear, Gaussian and sigmoid kernel functions are selected as the dimensionality reduction methods for comparison. The reduced data in training set are put into the random forest classifier with 100 estimators to fit the model. Finally, we make predictions on reduced data on testing set and calculate their predictive accuracy as well as total time cost to evaluate their performance.

4.1 Original PCA

The data distribution with top 2 principal components (PC) is shown in Fig. 2, where digits 1, 6 and 0 can be well classified from other digits while the others are just mixed together, meaning that we can not just use the top 2 PC to classify all the digits. The contributions of the top 8 PC can also be seen in Fig. 3, where we can see the top 8 PC contribute around 55% to the original data.

4.2 SPCA

In this experiment, we compare the SPCA with different sparsity. As can be seen in Fig. 4, with sparsity equal to 1, digits 1 and 6 can be clearly separated from the other ones. However, as the sparsity becomes larger, there seems to be no important information we can get from only top two PC, which indicates that for the handwritten digits dataset, it is more suitable to implement SPCA with lower sparsity.

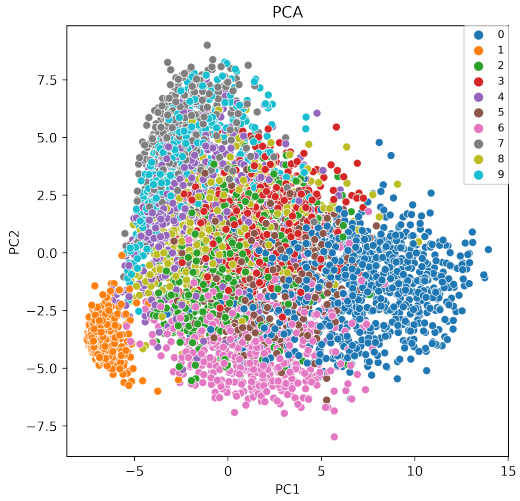

Figure 2: The projections on to the top two principal components of original PCA.

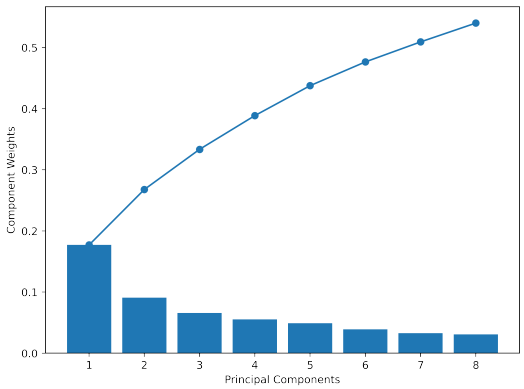

Figure 3: The explained variance of the top 8 principal components of original PCA.

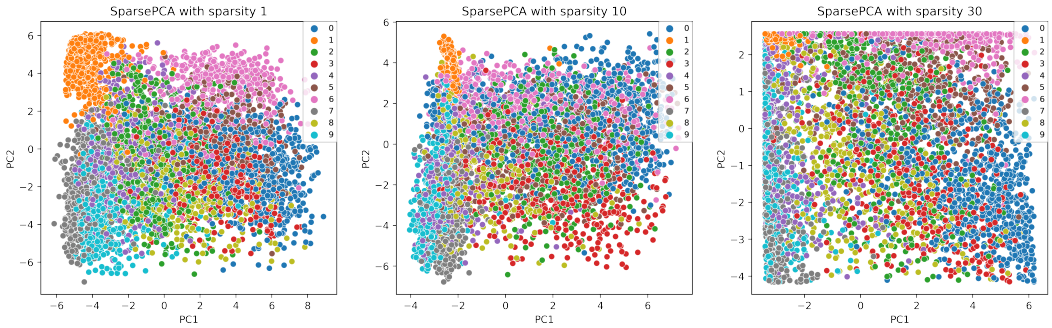

Figure 4: The projections on to the top two principal components of SPCA with different sparsity 1, 10 and 30.

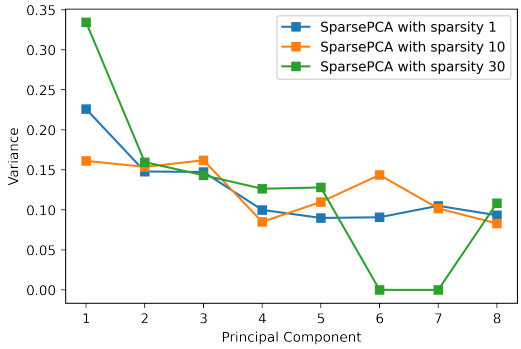

Figure 5: The explained variance of the top 8 principal components of SPCA with different sparsity 1, 10 and 30.

Fig. 5 shows the contributions of the top 8 PC of SPCA with different sparsity. It can be found that the contribution of each PC is not decreasing. This is because SPCA introduces a sparsity constraint, where the coefficient vector of the PC has fewer non-zero elements. This constraint can make some smaller PC have larger contributions, as they may contain fewer but more important information. Therefore, the ranking of PC in SPCA depends not only on the size of the variance, but also on the importance of the information captured by each PC.

4.3 KPCA

we compare the KPCA with kernel functions including linear, Gaussian and sigmoid. As can be seen in Fig. 6, all the kernel functions can do well in separating the digits 1, 6 and 0, especially the Gaussian kernel function. The contributions of top 8 PC can be seen in Fig. 7, where we can see the PC1 and PC2 contribute more with Gaussian kernel function, which is consistent to the finding in Fig. 6.

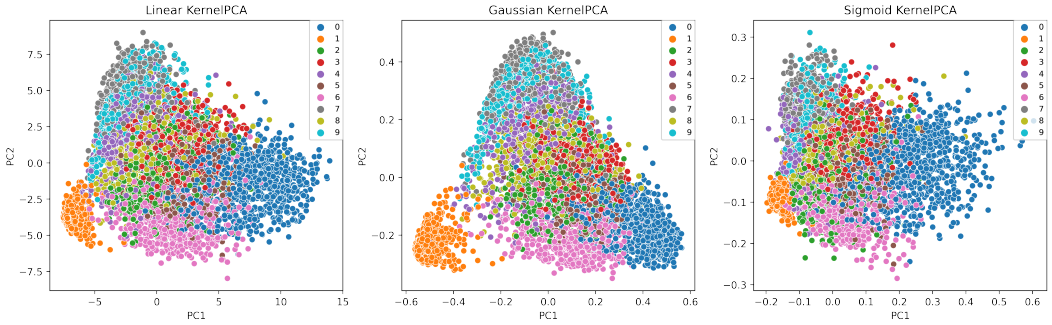

Figure 6: The projections on to the top two principal components of KPCA with linear, Gaussian and sigmoid kernel functions.

4.4 Performance

The elapsed time and the test accuracy using RF classifier with different dimension reduction methods can be seen in Table 4.4. We find that the elapsed time of SPCA increases as the sparsity decreases, while its test accuracy increases instead. This is because as the sparsity decreases, there are more non-zero elements in the PC coefficient vector, which means that the algorithm needs to compute more PC coefficients. For KPCA, the linear kernel function seems more suitable for this dataset considering both the elapsed time and test accuracy, compared with the Gaussian and sigmoid ones. The classification is more accurate with more PC, while the effect is decreasing. In general, all these dimension reduction approaches are effective, while the higheset accuracy is achieved by KPCA with linear kernel function.

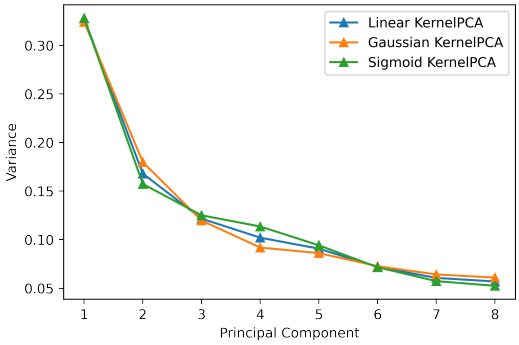

Figure 7: The explained variance of the top 8 principal components of KPCA with linear, Gaussian and sigmoid kernel functions.

Table 2: Comparison on elapsed time (the first number) and test accuracy (the second number)

	8 PC	16 PC	32 PC
RF	$2.33/95.68\%$	$2.33/95.68\%$	$2.33/95.68\%$
PCA + RF	$1.10/89.09\%$	$1.79/93.55\%$	$2.42/94.78\%$
SPCA (1) + RF	$126.01/89.16\%$	$248.92/94.30\%$	$620.04/95.40\%$
SPCA (10) + RF	$10.29/88.68\%$	$27.83/94.23\%$	$20.47/94.78\%$
SPCA (30) + RF	$5.04/80.99\%$	$4.23/83.25\%$	$6.15/84.15\%$
KPCA (linear) + RF	$1.93/88.54\%$	$10.75/93.41\%$	$11.47/95.26\%$
KPCA (Gaussian) + RF	$1.92/88.19\%$	$11.18/92.72\%$	$11.99/94.78\%$
KPCA (Sigmoid) + RF	$2.27/89.29\%$	$11.37/93.75\%$	$12.64/94.30\%$

5 Conclusion

In this mini-project, we investigate the performance of different kinds of PCA and our experiments show that PCA is an effective tool for dimension reduction. As an application of PCA, we perform RF classifier on the data reduced by different PCA and conclude that KPCA with linear kernel function shows the best performance in handwritten digits dataset.

Contribution

- Qichao Li is responsible for coding, conducting experiments and comparing the results.
- Haohan Huang is responsible for the methodology and other left parts of the paper.

References

[1] Iain M Johnstone and Arthur Yu Lu. On consistency and sparsity for principal components analysis in high dimensions. *Journal of the American Statistical Association*, 104(486):682–693, 2009.

[2] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901.

[3] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.