Nonlinear Component Analysis as a Kernel Eigenvalue Problem
Summary Report
IE 529

November 27, 2017

Group Members: Naman Shukla (namans2)
Karthik Venkata (kvn3)
Shubham Bansal (shubham8)
Ziyu Zhou (ziyuz2)
Zhenye Na (zna2)

Instructor: Carolyn Beck

1 Summary of Article

The research, *Nonlinear Component Analysis as a Kernel Eigenvalue Problem*, conducted by Bernhard et al., proposes a new method, kernel PCA, for performing nonlinear principal component analysis, where the principal components of features are nonlinearly related to the input variables. This method can obtain principal components by computing dot products in feature space using kernel functions in input space, instead of computing such dot products explicitly.

The paper first derives the method by generalizing the standard PCA to the nonlinear case, where the input space is mapped to a feature space with some mapping function. Since the map might have a huge or infinite dimensionality, a kernel representation is introduced, in order to compute dot products in feature space efficiently without having to carry out the map. It is proved by Mercer's theorem that kernels can achieve such goal if they are integral operators. Typical examples of this are polynomial kernels, radial basis functions and sigmoid kernels.

Based on the algorithm, the paper then illustrates the properties of kernel PCA. In general, kernel PCA in feature space shares the same properties as standard PCA in input space. Additionally, kernel PCA is unitary invariant. As for the

computational complexity, kernel PCA is computationally comparable to linear PCA, because it can deal with huge dimensionality efficiently by just computing in a small subspace of feature space and avoiding computing dot products explicitly. When extracting principal components, even though kernel PCA is not that fast, it is still much faster than performing linear PCA in feature space directly. It is possible to speed things up by using less samples to approximate eigenvectors while minimizing the squared errors. This method can also be utilized to solve the problems that might occur during variable selection and reconstruction. Besides, nonlinear features can be applied to train linear support vector machines, which will also save running time.

The paper conducts two set of experiments to examine the properties and efficiency of kernel PCA. The first experiment tests three kernel functions on an artificial dataset, showing that kernel PCA can better reflect the structure in the data. The second one performs kernel PCA with polynomial kernel functions in different degrees on the US Postal Service (USPS) dataset to extract nonlinear principal components, which are then used to train linear support vector machines. The experimental results illustrate that such principal components can yield much lower test error rates and produce a better classifier. By using more components than is possible in the input space, kernel PCA can further achieve a better performance.

Based on the experiments, it can be proved that kernel PCA not only performs better than linear PCA, but also outperforms other feature extraction methods. Compared to other methods for nonlinear PCA, kernel PCA exhibits two main advantages. One is that it only needs to solve simple linear algebra problems instead of complicated nonlinear optimization problems. The other lies in the fact that it doesn't need to specify the number of desired components in advance. There might exist some possible drawbacks for kernel PCA if the number of observations is very large and the interpretability is not desirable.

As a conclusion, the kernel method can be applied to construct the nonlinear variants of some classical algorithms. Due to its simplicity and accuracy, kernel PCA has a broad range of applicable domains, such as noise reduction, density estimation, image indexing, etc.

2 Overview of Kernel PCA

2.1 Mathematical Modeling

We are not interested in principal components in input space but in the features (principal components of variables) that are non-linearly related to the input variables. Some of the variables included in them are those that are obtained by taking arbitrary higher-order correlations between input variables. The nonlinearity in the dataset cannot be dealt through linear PCA. In this case we

might want to map the data to a higher dimensional feature space by: $\mathbf{R}^N \rightarrow F$. The PCA is then performed in the new space.

$$\Phi : \mathbf{R}^N \rightarrow F, \mathbf{x} \mapsto \mathbf{X} \tag{1}$$

It can be noted that F (feature space), might have an arbitrarily large, possibly infinite dimensionality. Here and in the following, uppercase characters are used for elements of F, and lowercase characters denote elements of $\mathbf{R}^N$.
In order to keep things simple, the assumption has been made that observations are centered. This is easy to achieve in input space but harder in F, because we cannot explicitly compute the mean of the $\Phi(\mathbf{x}_i)$ in F. Thus, we have

$$\sum_{k=1}^{M} \Phi(\mathbf{x}_k) = 0. \tag{2}$$

The covariance matrix in F space can be found by using the traditional PCA approach,

$$\bar{C} = \frac{1}{M} \sum_{j=1}^{M} \Phi(\mathbf{x}_j)\Phi(\mathbf{x}_j)^T \tag{3}$$

$$\lambda \mathbf{V} = \bar{C}\mathbf{V} \tag{4}$$

As the dimensions of F is very high, the eigenvalue decomposition is computationally extremely expensive. So we modify Eq.4: The eigenvalue problem $\lambda \mathbf{V} = \bar{C}\mathbf{V}$ can also be expressed in terms of a dot product as follows:

$$\lambda(\Phi(\mathbf{x}_k) \cdot \mathbf{V}) = (\Phi(\mathbf{x}_k) \cdot \bar{C}\mathbf{V}) \tag{5}$$

for all $k = 1, \ldots, M$.
Also, we can represent the eigenvectors $\mathbf{V}$ in terms of linear combination of feature vectors:

$$\mathbf{V} = \sum_{j=1}^{M} \alpha_i \Phi(\mathbf{x}_i) \tag{6}$$

Now because we are only interested in the dot product of the transformed feature vectors, we can use kernel functions to our advantage as follows.

$$\lambda \sum_{i=1}^{M} \alpha_i(\Phi(\mathbf{x}_k) \cdot \Phi(\mathbf{x}_i)) = \frac{1}{M} \sum_{i=1}^{M} \alpha_i(\Phi(\mathbf{x}_k) \cdot \sum_{j=1}^{M} \Phi(\mathbf{x}_j))(\Phi(\mathbf{x}_j) \cdot \Phi(\mathbf{x}_i)) \tag{7}$$

for all $k = 1, \ldots, M$.
Dot product in new space is represented as a kernel:

$$\kappa(x, y) = (\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})) \tag{8}$$

Then the eigenvalue problem can be further simplified as:

$$M\lambda\boldsymbol{\alpha} = K\boldsymbol{\alpha} \tag{9}$$

Where the kernel matrix K is defined as:

$$\mathbf{K}_{ij} = (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)) \tag{10}$$

The eigenvectors and eigenvalues may be found by solving this equation. We see that, in kernel PCA, a non-trivial arbitrary function $\Phi(\cdot)$ is chosen, but is usually never calculated explicitly, allowing the possibility for using a very high dimensional $\Phi(\cdot)$. Hence, the computation of the eigenvectors and eigenvalues of the covariance matrix in the high dimensional feature space is avoided. Instead, the kernel matrix is found and its eigenvectors and eigenvalues are computed, which in some cases can be much easier.
The projection of feature vectors onto principal components can be calculated in the following way:

$$(\mathbf{V}^n \cdot \Phi(\mathbf{x})) = \sum_{i=1}^{M} \alpha_i^n \kappa(\mathbf{x}_i, \mathbf{x}) \tag{11}$$

Where $\boldsymbol{\alpha_i}$ is the eigenvector of K, and $\lambda_1, \lambda_2, \ldots, \lambda_M$, are the eigenvalues of K.

2.2 The Algorithm for Kernel PCA

To perform kernel-based PCA, the following steps have to be carried out:

Algorithm 1 Kernel PCA Algorithm

procedure K - PCA(X)
  Given Input: $X_{N\times M} \leftarrow [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_M]$
  Centralize : $X_{centered} \leftarrow X_{N\times M}$
  Kernel Matrix : $K_{M\times M} : k_{ij} \leftarrow k(\mathbf{x}_i, \mathbf{x}_j)$
  Centralization in F space :
    $K : K \leftarrow K - I_M K/M - K I_M/M + I_M K I_M/M^2$
  Extracting eigenvectors : $M\lambda\boldsymbol{\alpha} = K\boldsymbol{\alpha}$
  Normalization :
  $\boldsymbol{\alpha} \leftarrow \frac{\boldsymbol{\alpha}}{mod(\boldsymbol{\alpha})\sqrt{M\lambda}}$
  loop: $i \leftarrow 1 : p$
    $P_i(x) = \sum_{i=1}^{M} \alpha_{ij} \kappa(\mathbf{x}_i, \mathbf{x})$
  goto *top.*

2.3 Discussion: Choosing a Kernel function

Gaussian kernel PCA with a properly selected parameter σ can perfectly separate the two classes in an unsupervised manner, which is impossible for standard

PCA. With a well selected σ, Gaussian kernel PCA will have a proper capture range, which will enhance the connection between data points that are closer to each other in the original feature space. Then by applying eigenvector analysis, the eigenvectors will describe the directions in a high-dimensional space in which different clusters of data are scattered to the greatest extent.

2.3.1 Discussion: Choosing the number of principal eigenvalues

Traditionally, one of the most well-known criteria for this decision is the scree criterion (Fabrigar et al., 1999), which involves a scree plot with the components identified on the x-axis and their associated eigenvalues on the y-axis. If such a plot shows a break, or an elbow, identifying the last component that accounts for a considerable amount of variance in the data. The location of this elbow (the x coordinate) indicates the appropriate number of components to be included in the solution.

Typically, choose k (number of principal components) to be the smallest value so that the proportion of the sum of the significant diagonal elements (Obtained after diagonalization) to the sum of all elements is obtained and if this ratio typically lies in the range of 0.95-0.99, the corresponding values are considered as the principal components that need to be taken into consideration.

2.4 Kernel PCA : Pseudocode

a. Loading Test data

b. Centering Test data

c. Defining Kernel function

d. Creating Kernel K matrix

e. Centering of Kernel K matrix in F space

f. Eigenvalue Decomposition of K centered Matrix

g. Sorting Eigenvalues in descending order thus, sorting the corresponding eigenvectors as well using bubble sort.

h. Selecting the smallest number of these sorted eigenvalues which account for maximum variation(99%) in the data, thus, selecting the significant eigenvectors corresponding to these eigenvalues.

i. Normalizing all significant sorted eigenvectors of K

j. Projecting data in the principal component coordinate system

3 Discussion of the application areas

The purpose of nonlinear PCA is to identify and to extract nonlinear components from a given data set. The extracted components span a component space which is supposed to cover the most important information of the data.

Linear PCA is being used in numerous technical and scientific applications, including noise reduction, density estimation, image indexing and retrieval systems, and the analysis of natural image statistics. Kernel PCA can be applied to all domains where traditional PCA has so far been used for feature extraction and where a nonlinear extension would make sense. The method of nonlinear kernel PCA analysis can be used to classify big and high dimensional data with lots of features. One of the fundamental areas where this can be used would be in document classification which has a lot of features in terms of thousands of words and contextual information. Other forms of implementation would involve domains like MEG brain imaging and conducting and classifying information obtained from surveys on platforms like Netflix.

A primary avenue of utilization of non linear PCA is in image analysis where it is extensively used for image compression, recognition of faces (Eigenfaces), image segmentation with respect to intensity (black-white) and texture. A wide array of challenging segmentation problems in dynamic vision like that of rigid-body motions, video and dynamic textures can also be solved.

NLPCA (Nonlinear Principal Component Analysis) is the nonlinear dimensionality reduction method that has been used most in climate applications. Along with its extensions to correlation and spectral analysis, NLPCA has been applied to questions as diverse as tropical Pacific climate variability, regimes in Northern Hemisphere atmospheric dynamics, and the quasi-biennial oscillation. The first climate application where NLPCA was used was the question of tropical variability in the Pacific. Application of NLPCA with one and two-dimensional bottleneck layers to tropical Pacific observational SST data demonstrated that low-dimensional NLPCA approximations can characterize variability in this data better than linear PCA approximations, and that NLPCA approximations are able to represent the observed El Nio/La Nia asymmetry [Monahan, 2000, 2001].

An extension of NLPCA that has proven useful in some applications uses complex-valued networks [Rattan and Hsieh, 2005]. These can be used for the analysis of vector field data such as winds or currents, where it is possible to exploit spatial correlations between different vector components. This approach has been applied in the study of tropical Pacific winds [Rattan and Hsieh, 2004]

4 Implementations of example application

4.1 Iris Dataset Visualization

Principal Component Analysis (PCA) is a dimensionality reduction technique that is used to transform and a high-dimensional dataset into a smaller dimensional subspace to give a directed impression of the dataset prior to running a machine learning algorithm on the data. The Iris dataset is in a 4_{th} dimensions (features) of three different iris ower species. Linear PCA can only extract two visible clusters, however, with Kernel PCA we can visualize better in a two dimensional subspace.

4.1.1 Pseudocode for Iris Dataset Visualization

a. Loading Iris Dataset

b. Loading built-in functions in Python libraries

c. Training PCA/KPCA model on the dataset

d. Reconstruct data

e. Projecting data in the principal component coordinate system

4.2 USPS Handwriting Dataset Recognition

The dataset contains numeric data obtained from the scanning of handwritten digits from envelopes by the U.S. Postal Service. The original scanned digits are binary and of different sizes and orientations; the images here have been deslanted and size normalized, resulting in 16×16 grayscale. We will first extract features via Kernel PCA and apply that to a SVM classifier to train and test on the splitted USPS dataset.

4.2.1 Pseudocode for USPS Handwriting Dataset Recognition

a. Loading USPS Dataset

b. Determining the Kernel function used

c. Creating matrix K and centering K in F feature space

d. Sorting the eigenvalues and eigenvectors

e. Selecting the smallest number of eigenvectors which can contains 99% of variance

f. Projecting data in the principal component coordinate system

g. Training SVM classifier with extracted features (80% of original data)

h. Testing classifier results on the test data (20% of the original)

References

Bernhard Scholkopf, Alexander Smola and Klaus-Robert Muller. *Nonlinear Component Analysis as a Kernel Eigenvalue Problem*

Ian Ross. *Nonlinear Dimensionality Reduction Methods in Climate Data Analysis*

Generalized Principal Component Analysis. Tutorial @ CVPR 2007. JHU vision labs

Quan Wang. *Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models*

Simon Gunter, Nicol N. Schraudolph and S.V:N. Vishwanathan. *Fast Iterative Kernel Principal Component Analysis.*