Project: Kernel Principal Component Analysis

Machine Learning, Advanced Course

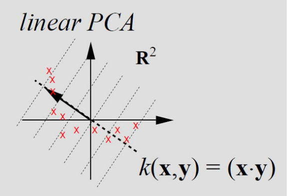
Thayabaran Kathiresan
Rémi Domingues
Agnes Martine Nielsen
Tobias Wiens

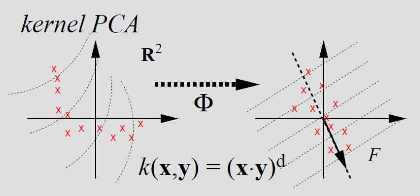
KTH Royal Institute of Technology

December 16, 2014

Kernel Principal Component Analysis: Motivation

Take non-linearities into account





- *PC* are extracted from the high-dimension feature space F
- Kernel functions gives us $\Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})$ for a low computation cost

Kernel PCA

High-dimension feature space F

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

Covariance matrix in F

$$\bar{C} = \frac{1}{l} \sum_{j=1}^{l} \Phi(\mathbf{x}_j) \Phi(\mathbf{x}_j)^T \tag{2}$$

Assumption: data in F is centered in 0

Kernel PCA

Finding eigenvalues and eigenvectors: $\lambda \mathbf{V} = \bar{C}\mathbf{V}$
Write eigenvectors as follows since $\mathbf{V} \in \mathrm{span}\{\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_l)\}$

$$\mathbf{V} = \sum_{i=1}^{l} \alpha_i \Phi(\mathbf{x}_i) \tag{3}$$

This leads to the problem following eigenvalue problem

$$l\lambda\boldsymbol{\alpha} = \mathbf{K}\boldsymbol{\alpha} \tag{4}$$

with

$$K_{ij} := (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)) \tag{5}$$

Kernel PCA

Principal components of a data point $\Phi(\mathbf{x})$

$$(\mathbf{V}^k \cdot \Phi(x)) = \sum_{i=1}^{l} \alpha_i^k (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x})) \tag{6}$$

We replace all inner products in F with a kernel function. We use polynomial kernel in our experiments,

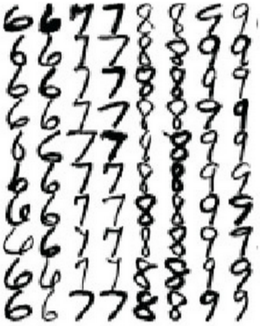
$$k(x, y) = (x \cdot y)^d$$

Experiment

For testing the performance of
the Kernel PCA, we have chosen the digits classification problem. We used the USPS (US Postal Service) Handwritten digits. This data set contains 9300 examples, for better simulation we have used

Training Data : 3000
Test Data : 2000

Types of Classifier used:
Multivariate Gaussian (Linear Classifier)
Linear Support Vector Machine



Original digits from USPS dataset

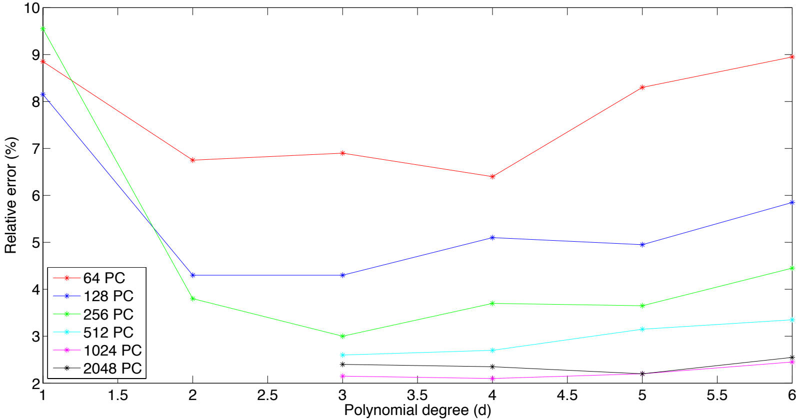
The articles results

Classification using separating hyperplane (SVM)

In case of linear PCA ($d = 1$) best result is 8.6% error for PC$= 128$.
Non-linear PCA ($d = 2, \ldots, 6$ and PC$= 128$) gives around 6% error
With $d > 2$ and 2048 components gives around 4% error

Our results

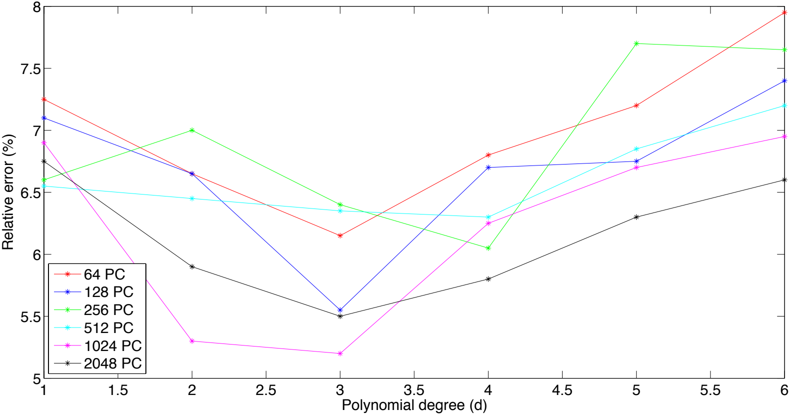
Figure: Classification using Gaussian Multivariate Classifier



Matlab command: `classify`

Our results

Figure: Classification using SVM



Matlab command: `svmtrain` (using LIBSVM)

Discussion

Advantages

- Better results with linear classifiers: Kernel PCA extends PCA by extracting non-linear principal components allowing better performances for linear classifications
- Those components are computed by only solving the eigenvalue problem, which is done in a low complexity due to the Kernel methods

Suggested improvements

- Adapt Kernel PCA to current PCA methods suitable for huge datasets
- Scaling factor applied to $\bar{C}$ is l, should be $l - 1$ to remove the bias (Bessel's correction)

Questions?

Questions?

Equivalent system to $\lambda\mathbf{V} = \bar{C}\mathbf{V}$

$$\lambda(\Phi(\mathbf{x}_k) \cdot \mathbf{V}) = (\Phi(\mathbf{x}_k)) \cdot \bar{C}\mathbf{V}) \text{ for } k = 1, ..., l \textit{ with } \mathbf{V} = \sum_{i=1}^{l} \alpha_i \Phi(\mathbf{x}_i) \quad (7)$$