The Pre-Image Problem in Kernel Methods

James Kwok   Ivor Tsang

Department of Computer Science
Hong Kong University of Science and Technology
Hong Kong

Outline

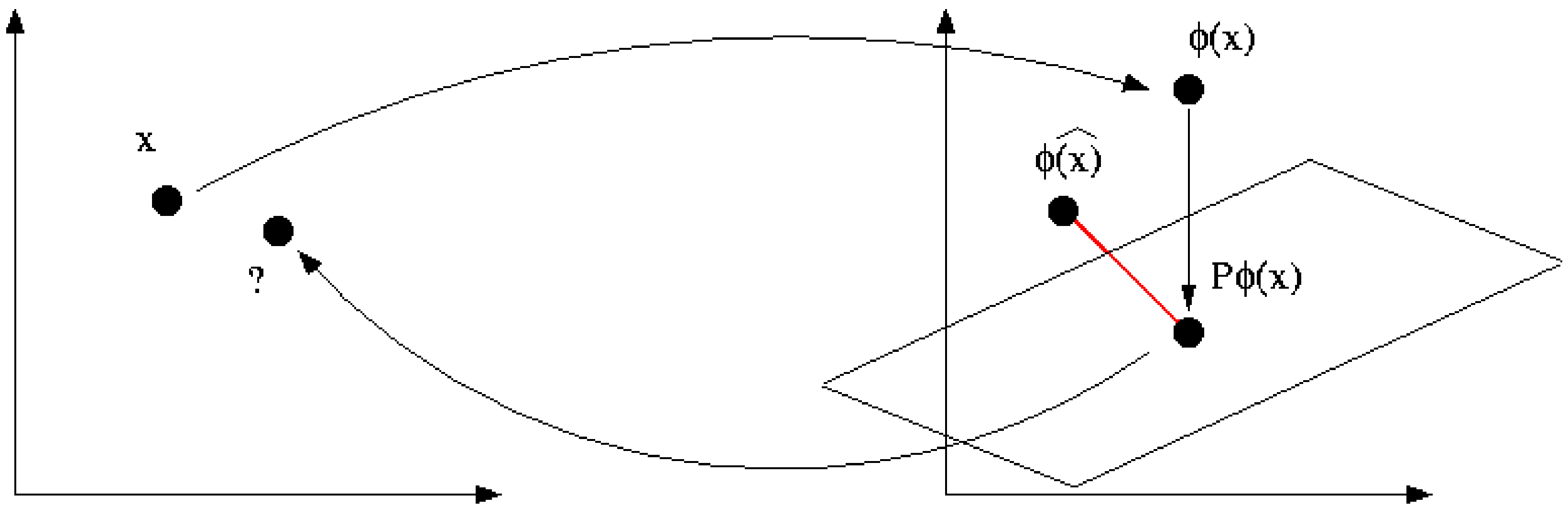
- Introduction
- Proposed Method
- Experimental Results
- Conclusion

Kernel Principal Component Analysis

- Kernel PCA: linear PCA in the feature space
- Given $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, construct $\mathbf{K} = [K_{ij}]$
- $\mathbf{K} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}'$ (assume that $\{\varphi(\mathbf{x}_1), \ldots, \varphi(\mathbf{x}_N)\}$ has been centered)
  - $\mathbf{U} = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_N]$ with $\boldsymbol{\alpha}_i = [\alpha_{i1}, \ldots, \alpha_{iN}]'$
  - $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_N)$
- kth orthonormal eigenvector: $\mathbf{V}_k = \sum_{i=1}^{N} (\alpha_{ki}/\sqrt{\lambda_k})\varphi(\mathbf{x}_i)$
- Projection of $\varphi(\mathbf{x})$ onto $\mathbf{V}_k$: $\beta_k = \varphi(\mathbf{x})'\mathbf{V}_k$
- Projection $P\varphi(\mathbf{x})$ of $\varphi(\mathbf{x})$: $P\varphi(\mathbf{x}) = \sum_{k=1}^{K} \beta_k \mathbf{V}_k$

The Pre-Image Problem

- Given noisy $\mathbf{x}$, can we recover an $\hat{\mathbf{x}}$ such that $\varphi(\hat{\mathbf{x}}) = P\varphi(\mathbf{x})$?



- Non-trivial as φ is not invertible in general
- Moreover, the exact pre-image may not exist
- Only an approximate solution can be obtained

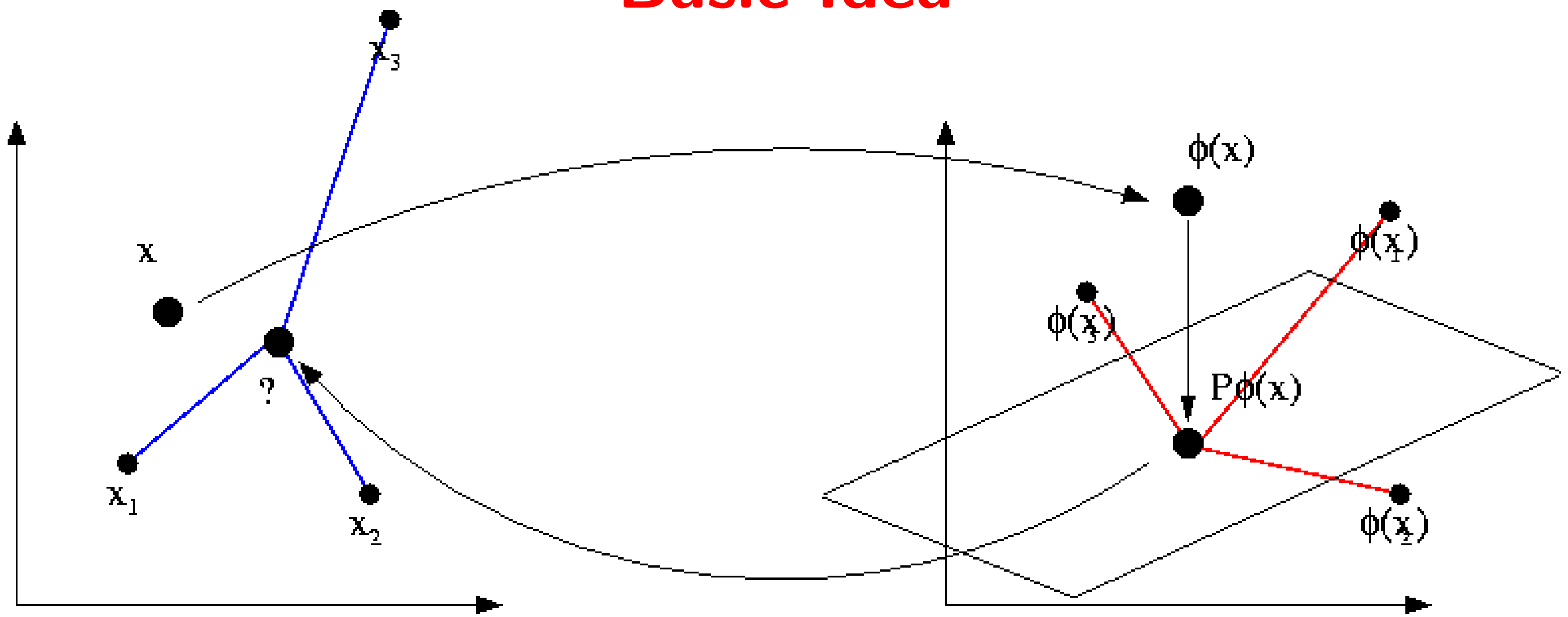
Solution of Mika *et al.* (NIPS11)

- Minimize the feature-space distance $\|\varphi(\hat{\mathbf{x}}) - P\varphi(\mathbf{x})\|$
  - nonlinear optimization
  - iteration scheme for Gaussian kernels $k(\mathbf{x},\mathbf{y}) = \exp(\|\mathbf{x}-\mathbf{y}\|^2/c)$:

$$\hat{\mathbf{x}}_{t+1} = \frac{\sum_{i=1}^{N} \gamma_i \exp(-\|\hat{\mathbf{x}}_t - \mathbf{x}_i\|^2/c)\mathbf{x}_i}{\sum_{i=1}^{N} \gamma_i \exp(-\|\hat{\mathbf{x}}_t - \mathbf{x}_i\|^2/c)}, \qquad \gamma_i = \sum_{k=1}^{K} \beta_k \alpha_{ki}$$

- Problems:
  - numerical instability
  - local minimum

Basic Idea



- Neighbors are most important in determining the location
  - find the distances between $P\varphi(\mathbf{x})$ and its neighboring $\varphi(\mathbf{x}_i)$'s
- Obtain the corresponding input-space distances
- Use these distances to constrain the location of the pre-image

Input- and Feature-Space Distances

- Feature-space distance between $P\varphi(\mathbf{x})$ and its neighbor $\varphi(\mathbf{x}_i)$

$$\begin{aligned}\tilde{d}^2(P\varphi(\mathbf{x}), \varphi(\mathbf{x}_i)) &= \|P\varphi(\mathbf{x}) - \varphi(\mathbf{x}_i)\|^2 \\ &= \|P\varphi(\mathbf{x})\|^2 + \|\varphi(\mathbf{x}_i)\|^2 - 2P\varphi(\mathbf{x})'\varphi(\mathbf{x}_i)\end{aligned}$$

  - recall that $P\varphi(\mathbf{x}) = \sum_{k=1}^{K} \beta_k \mathbf{V}_k$
  - $\tilde{d}^2(P\varphi(\mathbf{x}), \varphi(\mathbf{x}_i))$ can be computed based on K

- Obtain the corresponding input-space distance d_{ij}

From $\tilde{d}_{ij}$ to d_{ij}

- Isotropic kernels: $k(\mathbf{x}_i, \mathbf{x}_j) = \kappa(\|\mathbf{x}_i - \mathbf{x}_j\|^2)$

$$\begin{aligned} \tilde{d}_{ij}^2 &= k(\mathbf{x}_i, \mathbf{x}_i) + k(\mathbf{x}_j, \mathbf{x}_j) - 2k(\mathbf{x}_i, \mathbf{x}_j) \\ &= K_{ii} + K_{jj} - 2\kappa(\|\mathbf{x}_i - \mathbf{x}_j\|^2) \\ &= K_{ii} + K_{jj} - 2\kappa(d_{ij}^2) \end{aligned}$$

$$\kappa(d_{ij}^2) = \frac{1}{2}(K_{ii} + K_{jj} - \tilde{d}_{ij}^2)$$

  - typically, κ is invertible
  - example: Gaussian kernel $\kappa(z) = \exp(-\beta z)$

$$d_{ij}^2 = -\frac{1}{\beta}\log(\frac{1}{2}(K_{ii} + K_{jj} - \tilde{d}_{ij}^2))$$

- Dot product kernels: $k(\mathbf{x}_i, \mathbf{x}_j) = \kappa(\mathbf{x}_i' \mathbf{x}_j)$

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = \kappa(\mathbf{x}_i' \mathbf{x}_j) = \kappa(s_{ij})$$

  - κ is often invertible
  - example: polynomial kernel $\kappa(z) = z^p$

$$s_{ij} = K_{ij}^{\frac{1}{p}} \quad \text{when } p \text{ is odd}$$

  - corresponding input-space distance: $d_{ij}^2 = s_{ii}^2 + s_{jj}^2 - 2s_{ij}$

Back to Input Space

- Assume that the pre-image $\hat{\mathbf{x}}$ is in the span of the $\mathbf{x}_i$'s
  - Mika *et al.* also obtains the same assumption
- Maintain the distances between $\hat{\mathbf{x}}$ and its neighbors (in the least-square sense)
  - similar to the ideas in classical scaling and metric multidimensional scaling (MDS)
  - reduces to finding the least-square solution for a system of linear equations

Details

- After kernel PCA, we obtain $P\varphi(\mathbf{x})$
- Identify the n neighbors $(\varphi(\mathbf{x}_1), \ldots, \varphi(\mathbf{x}_n))$ of $P\varphi(\mathbf{x})$
  - $\mathbf{X}_{d\times n} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ (input space)
  - assume that they span a q-dimensional space (rank$(\mathbf{X}) = q$)
- Obtain the desired input-space distances between the unknown pre-image and these neighbors
  - $\mathbf{d}^2 = [d^2(\mathbf{x}, \mathbf{x}_1), d^2(\mathbf{x}, \mathbf{x}_2), \ldots, d^2(\mathbf{x}, \mathbf{x}_n)]'$
- Obtain a basis for these centered neighbors
  - center these neighbors: $\mathbf{HX}'$
    - $\mathbf{H}$ is the centering matrix $\mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}'$

- use SVD on $(\mathbf{H}\mathbf{X}')'$: $\mathbf{X}\mathbf{H} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}' = \mathbf{U}\mathbf{Z}$
  - $\mathbf{U}_{d\times q} = [\mathbf{e}_1, \ldots, \mathbf{e}_q]$ ($\mathbf{e}_i$'s orthonormal)
  - $\mathbf{Z}_{q\times n} = [\mathbf{z}_1, \ldots, \mathbf{z}_n]$ ($\mathbf{z}_i$ is the projections of $\mathbf{x}_i$ onto $\mathbf{e}_j$'s)
- distances of $\mathbf{x}_i$'s to the centroid: $\mathbf{d}_0^2 = [\|\mathbf{z}_1\|^2, \ldots, \|\mathbf{z}_n\|^2]'$

- embed the desired pre-image $\hat{\mathbf{x}}$ in the span of these neighbors

$$d^2(\hat{\mathbf{x}}, \mathbf{x}_i) \simeq d_i^2, \quad i = 1, \ldots, n.$$

  - if exact pre-image exists, distances exactly preserved
  - least-square solution

$$\hat{\mathbf{z}} = -\frac{1}{2}\boldsymbol{\Lambda}^{-1}\mathbf{V}'(\mathbf{d}^2 - \mathbf{d}_0^2)$$

    - expressed in terms of the coordinate system defined by the $\mathbf{e}_j$'s

- transform back to the original coordinate system in the input space

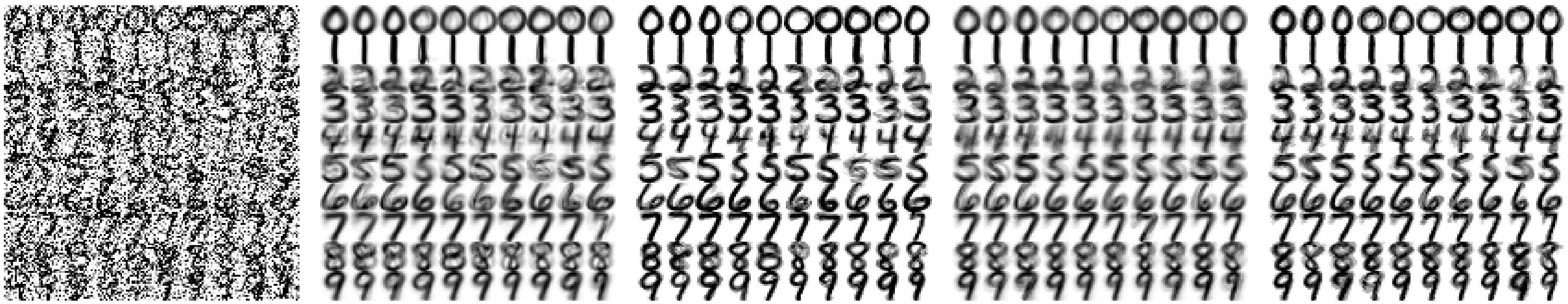
$$\hat{\mathbf{x}} = \mathbf{U}\hat{\mathbf{z}} + \bar{\mathbf{x}}$$

Experiment: Pre-Images in Kernel PCA

- images from USPS dataset, 100 test images
- ten neighbors are used in locating the pre-image
- Gaussian noise ($\sigma^2 = 0.25, 0.3, 0.4, 0.5$)
- similar results on salt-and-pepper noise

Results (Gaussian kernel)

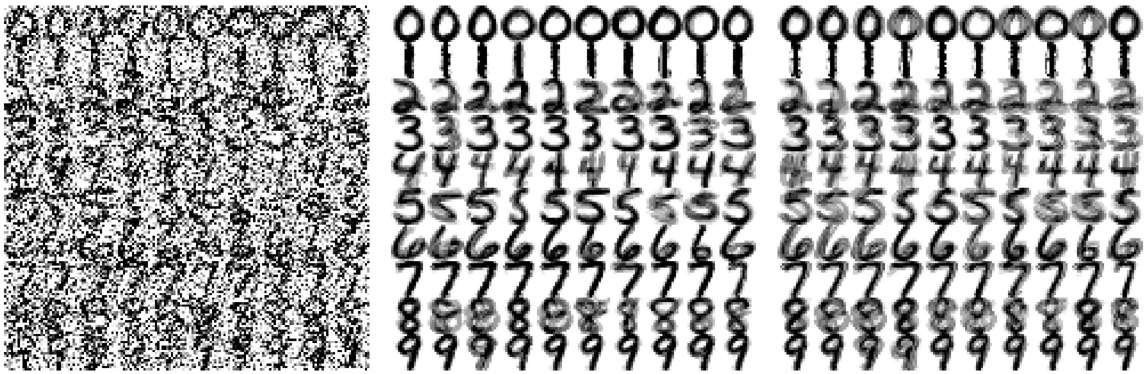
- noisy image; (300 training images) Mika *et al.*; proposed method; (60 training images) Mika *et al.*; proposed method



number of training images	σ^2	SNR		
		noisy images	our method	Mika *et al.*
300	0.25	2.32	**6.36**	5.90
	0.3	1.72	**6.24**	5.60
	0.4	0.91	**5.89**	5.17
	0.5	0.32	**5.58**	4.86
60	0.25	2.32	**4.64**	4.50
	0.3	1.72	**4.56**	4.39
	0.4	0.90	**4.41**	4.19
	0.5	0.35	**4.29**	4.06

Experiment on Polynomial Kernel (Order=3)

- Mika *et al.*
  - we derive a similar iteration scheme as for Gaussian kernels
  - however, the iteration does not converge
- Proposed method
  - Noisy image; 300 training images; 60 training images



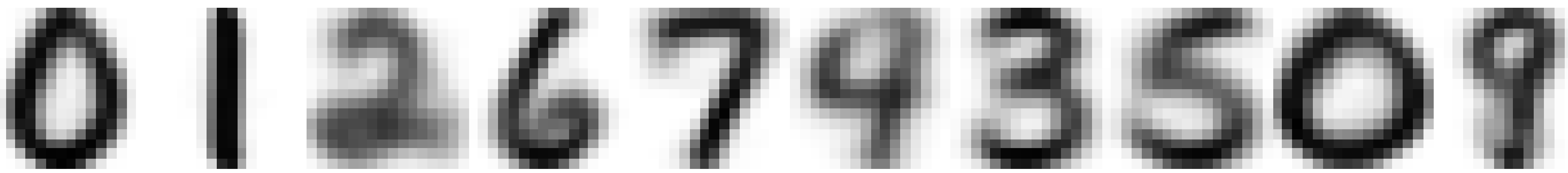
number of training images	σ^2	SNR from our method
300	0.25	5.39
	0.3	5.08
	0.4	4.61
	0.5	4.24
60	0.25	4.33
	0.3	4.09
	0.4	3.74
	0.5	3.50

Pre-Images in Kernel k-Means Clustering

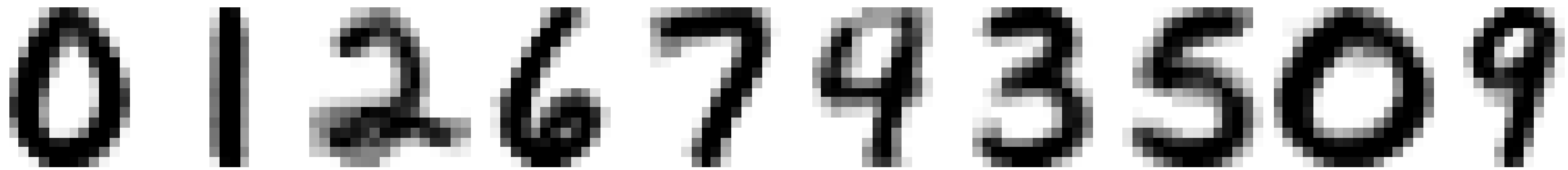
- Kernelized version of k-means clustering
  - 3000 images randomly selected from the USPS data set
  - $k = 10$ (number of digits)
  - Gaussian kernel
- Use the proposed method to find pre-images of cluster centroids
- For comparison, also obtain cluster centroids by averaging in the input space over all patterns belonging to the same cluster

Results

- Input space averaging



- Proposed method



  – yields centroids that are visually more appealing

Conclusion

- Finding the pre-image of a feature vector in the kernel-induced feature space
- Advantages:
  - non-iterative
  - involves only linear algebra
  - does not suffer from numerical instabilities or the local minimum problem
- Can be applied equally well to both isotropic kernels and dot product kernels
- Experimental results show significant improvements over the traditional method

Future Directions

- Can be applied to other kernel applications that also require computation of the pre-images
- Extension to other classes of kernels
  - only addressed isotropic kernels and dot product kernels in this paper
  - kernel dependency estimation: pre-images for structured objects