Introduction to Statistical Learning and Kernel Machines

Hichem SAHBI

CNRS UPMC

June 2018

Outline

Introduction to Statistical Learning

- Definitions
- Probability Tools
- Generalization Bounds
- Machine Learning Algorithms

Kernel Machines : Supervised and Unsupervised Learning

- The Representer Theorem
- Supervised Learning (Support vector machines and regression)
- Kernel Design (kernel combination, cdk kernels,...)
- Unsupervised Learning (kernel PCA and CCA)

Outline

Introduction to Statistical Learning

- Definitions
- Probability Tools
- Generalization Bounds
- Machine Learning Algorithms

Kernel Machines : Supervised and Unsupervised Learning

- The Representer Theorem
- Supervised Learning (Support vector machines and regression)
- Kernel Design (kernel combination, cdk kernels,...)
- Unsupervised Learning (kernel PCA and CCA)

Pattern Recognition Problems

- Given a pattern (observation) $X \in \mathcal{X}$, the goal is to predict the unknown label Y of X.
- Character recognition (OCR) : X is an image, Y is a letter.
- Face detection (resp. recognition) : X is an image, Y indicates the presence of a face in the picture (resp. identity).
- Text classification : X is a text, Y is a category (topic, spam/non spam,...).
- Medical diagnosis : X is a set of features (age, genome, ...), Y is the risk.

Section 1

The Representer Theorem

Regularization, Kernel Methods and Representer Theorem

$$\min_{g \in \mathcal{G}} R_n(g) \; + \; \lambda \, \Omega(g), \qquad \lambda \geq 0 \quad \text{(Tikhonov)}$$

- For a particular regularizer $\Omega(g)$ and class $\mathcal{G}$, the solution to the above problem (Kimeldorf and Wahba, 1971)

$$g_\alpha(.) = \sum_{i=1}^{n} \alpha_i k(., X_i), \qquad \{(X_i, Y_i)\}_{i=1}^n \subseteq \mathcal{X} \times \mathcal{Y} \text{ is fixed}$$

- k is a kernel : symmetric, continuous on $\mathcal{X} \times \mathcal{X}$ and positive definite (Mercer, 1909), $k(X, X') = \langle \Phi(X), \Phi(X') \rangle$.
- $R_n(g) = \frac{1}{n} \sum_{i=1}^{n} (g(X_i) - Y_i)^2$ (kernel regression),

  $R_n(g) = \frac{1}{2n} \sum_{i=1}^{n} (sign[g(X_i)] - Y_i)^2$ (e.g., max margin classifier, SVMs).

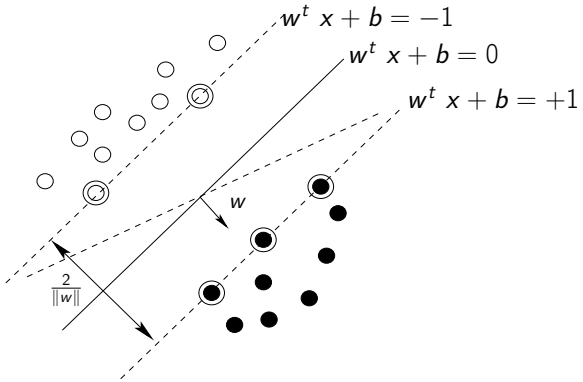
Section 2

Supervised Learning (SVMs and SVRs)

Support Vector Machines

Support Vector Machines (Large Margin Classifiers)

- Let $\{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, be a training set generated i.i.d



$$\min_{w,b} \quad \frac{1}{2} w'w \qquad \text{s.c.} \qquad Y_i \left(w'\ X_i + b\right) - 1 \geq 0, \ \forall\ i$$

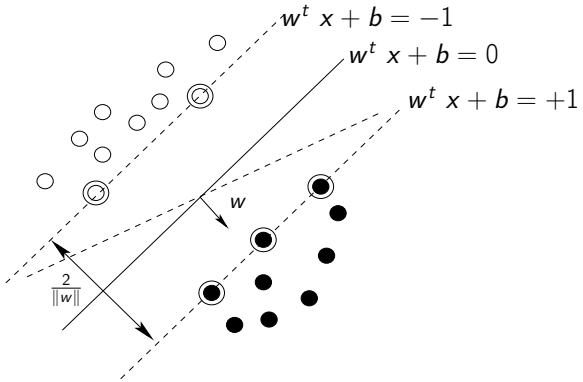
- Optimality conditions lead to $w = \sum_i \alpha_i Y_i X_i$ and dual form

$$\max_{\{\alpha_i\}} \quad \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i\ \alpha_j\ Y_i\ Y_j\ \langle X_i, X_j \rangle$$

$$\text{s.t.} \quad \alpha_i \geq 0, \ \forall\ i \qquad \text{and} \qquad \sum_i \alpha_i\ Y_i\ = 0$$

Support Vector Machines (Large Margin Classifiers)

- Let $\{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, be a training set generated i.i.d



$$\min_{w,b} \quad \frac{1}{2} w'w \qquad \text{s.c.} \qquad Y_i \ \left(w'\ X_i \ + \ b\right) \ - \ 1 \geq 0, \ \forall \ i$$

- Optimality conditions lead to $w = \sum_i \alpha_i Y_i X_i$ and dual form

$$\max_{\{\alpha_i\}} \quad \sum_i \ \alpha_i \ - \ \frac{1}{2} \sum_i \sum_j \ \alpha_i \ \alpha_j \ Y_i \ Y_j \ \langle X_i, X_j \rangle$$

$$\text{s.t.} \qquad \alpha_i \geq 0, \quad \forall \ i \qquad \text{and} \qquad \sum_i \ \alpha_i \ Y_i \ = 0$$

VC Dimension of Large Margin Classifiers

- The set of hyperplane classifiers with a margin at least M has a VC dimension upper bounded by :

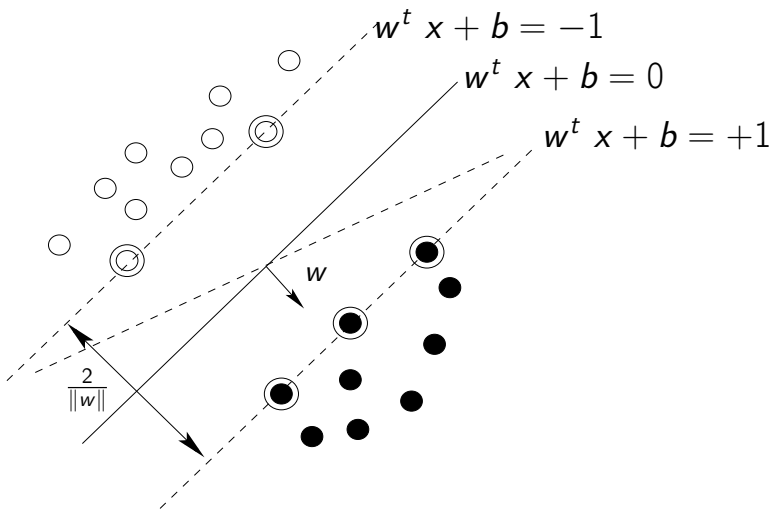
$$h \leq r^2/M^2,$$

- here r is the radius of the smallest sphere containing all the patterns X.

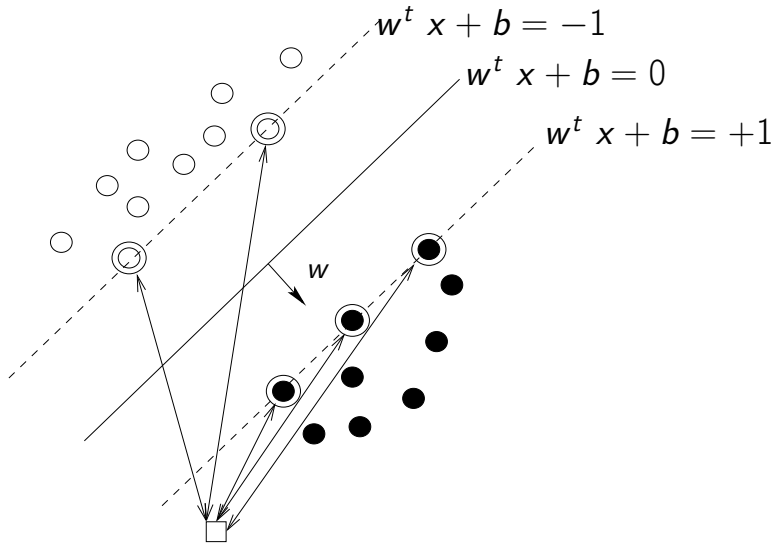
Interpretation of Lagrange Multipliers

- $\alpha_i > 0$ implies $Y_i(w'X_i + b) = 1 : X_i$ is a support vector.

- $\alpha_i = 0$ implies $Y_i(w'X_i + b) > 1 : X_i$ is useless vector.



Classification Function



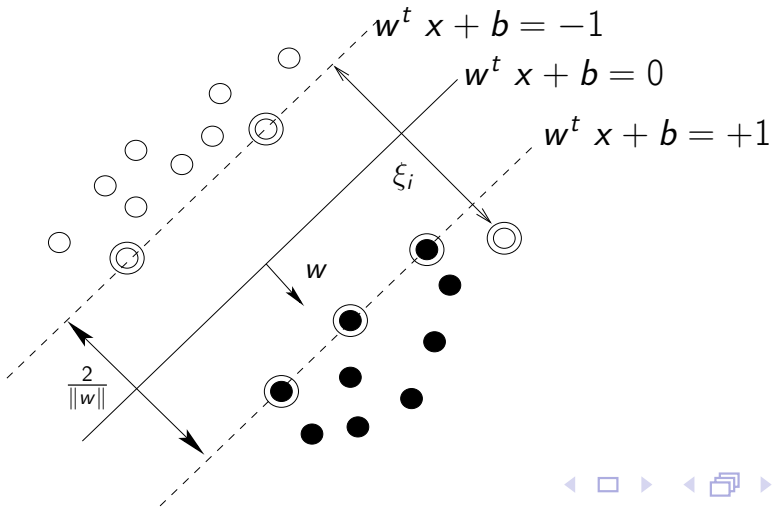
- Classification function

$$\begin{aligned} g_\alpha(X) - b &= \langle w, X \rangle \\ &= \sum_{Y_i=+1} \alpha_i \langle X_i, X \rangle - \sum_{Y_i=-1} \alpha_i \langle X_i, X \rangle \end{aligned}$$

Linear Soft-SVMs

- Introduce slack variables $\{\xi_1, \ldots, \xi_n\}$ to allow misclassification.
- Trade-off large margin and misclassification.

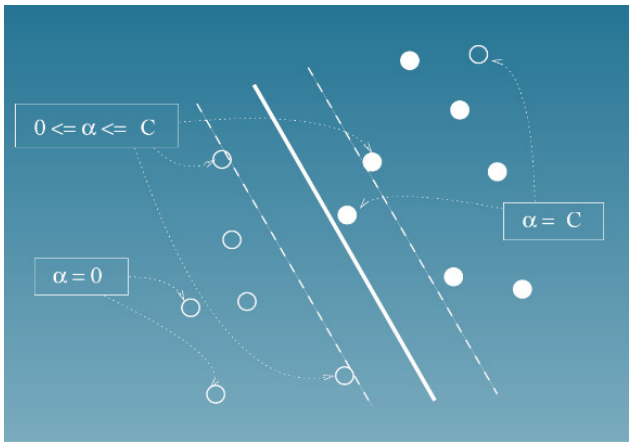
$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} w'w + C \sum_{i=1}^{n} \xi_i \\ \text{s.c.} \quad & Y_i \ (w' \ X_i \ + \ b) + \xi_i \geq 1, \ \forall \ i \\ & \xi_i \ \geq \ 0 \end{aligned}$$



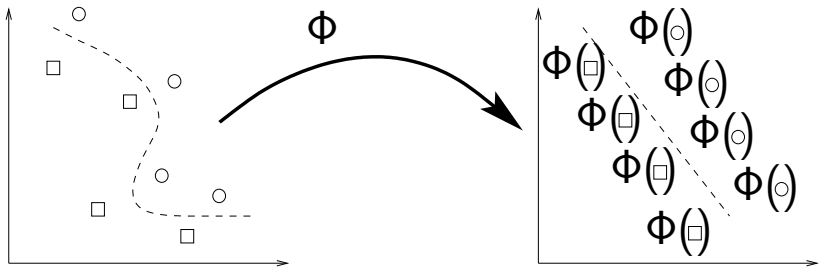
Dual Formulation

$$\max_{\{\alpha_i\}} \quad \sum_i \alpha_i - \frac{1}{2}\sum_i\sum_j \alpha_i \alpha_j Y_i Y_j \langle X_i, X_j \rangle$$

$$\text{s.c.} \quad 0 \leq \alpha_i \leq C, \ \forall i \quad \text{and} \quad \sum_i \alpha_i Y_i = 0$$



Non-Linear SVMs



$$\max_{\{\alpha_i\}} \quad \sum_i \alpha_i \; - \; \frac{1}{2} \sum_i \sum_j \alpha_i \; \alpha_j \; Y_i \; Y_j \; \langle \Phi(X_i), \Phi(X_j) \rangle$$

$$\text{s.t.} \qquad \alpha_i \geq 0, \quad \forall \; i \qquad \text{and} \qquad \sum_i \alpha_i \; Y_i \; = 0$$

$$g_\alpha(X) - b \; = \; \sum_{Y_i=+1} \alpha_i \; \langle \Phi(X_i), \Phi(X) \rangle \; - \; \sum_{Y_i=-1} \alpha_i \; \langle \Phi(X_i), \Phi(X) \rangle$$

- The product $\langle \Phi(X), \Phi(X') \rangle$ defines a kernel $k(X, X')$.

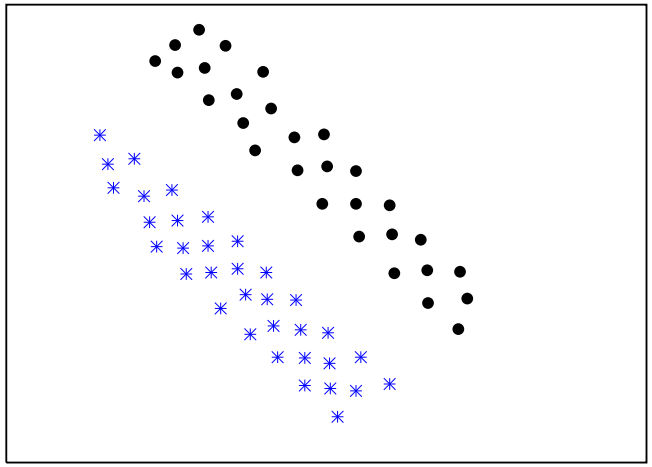
Kernels

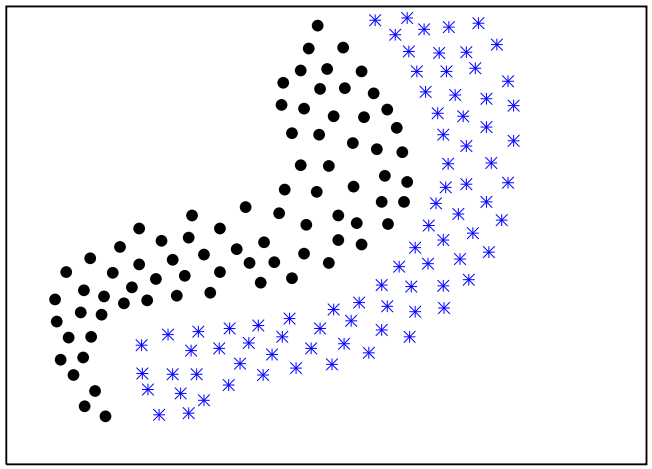
- Kernels are symmetric and positive (semi) definite functions that measure similarity between data.
- Positive semi definite means $k(X, X') = \langle \Phi(X), \Phi(X') \rangle$.

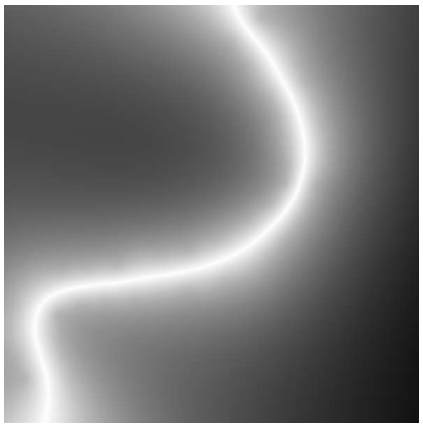
$$\forall X_1, \ldots, X_n \in \mathcal{X}, \quad \forall c_1, \ldots, c_n \in \mathbb{R}, \quad \sum_{i,j} c_i c_j k(X_i, X_j) \geq 0$$

- Equivalently, the Gram (kernel) matrix $\mathbf{K}$ with $\mathbf{K}_{ij} = k(X_i, X_j)$ has positive eigenvalues.
- Kernels on vectorial data : linear $\langle X, X' \rangle$, polynomial $(1 + \langle X, X' \rangle)^p$, Gaussian $\exp\left(-\frac{1}{\sigma^2} \quad \|X - X'\|^2\right)$, etc.
- Kernels can be designed using closure operations (additions, products, exponentiation, etc.)

Examples (Linear vs Gaussian)

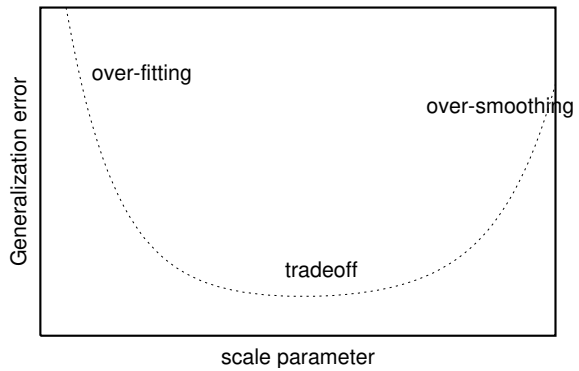






Gaussian Kernel

- $k(X, X') = \langle \Phi(X), \Phi(X') \rangle = \exp\left(-\frac{1}{\sigma^2} \quad \|X - X'\|^2\right)$.
- The dimension of the output space $\mathbb{R}^H$ is ∞.
- The Gaussian kernel has good generalization properties but requires a good selection of the scale parameter σ using the tedious cross validation process.



Gaussian Kernel

$$\langle \Phi(X), w \rangle = \sum_j \alpha_j k(X, X_j) = \sum_j \alpha_j \exp(-\|X - X_j\|^2/\sigma^2)$$

- Large Scale ($\sigma \to \infty$)

$$\begin{aligned} \langle \Phi(X), w \rangle &\simeq \sum_j \alpha_j (1 - \|X - X_j\|^2/\sigma^2) \\ &= -(1/\sigma^2) \sum_j \alpha_j (\langle X, X \rangle + \langle X_j, X_j \rangle - 2\langle X, X_j \rangle) + \dots \\ &= \sum_j \frac{2\alpha_j}{\sigma^2} \langle X, X_j \rangle + Cst = \langle X, \sum_j \frac{2\alpha_j}{\sigma^2} X_j \rangle + Cst \end{aligned}$$

- Small Scale ($\sigma \to 0$)

$$\langle \Phi(X), w \rangle = \sum_j \alpha_j \exp(-\|X - X_j\|^2/\sigma^2) \simeq \sum_j \alpha_j \mathbb{1}_{\{X = X_j\}}$$

Gaussian Kernel

$$\langle \Phi(X), w \rangle = \sum_j \alpha_j k(X, X_j) = \sum_j \alpha_j \exp(-\|X - X_j\|^2/\sigma^2)$$

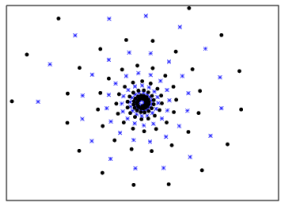
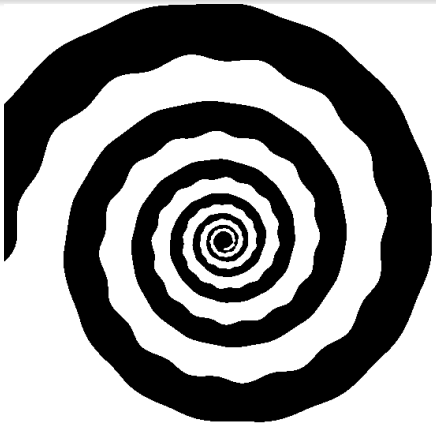
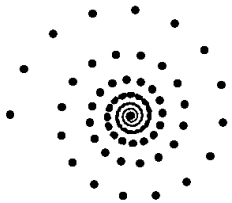
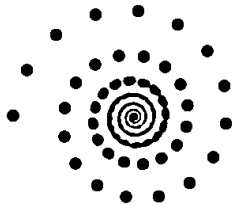
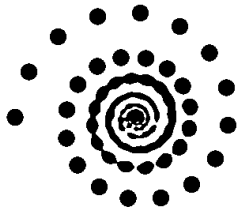
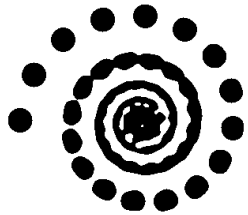
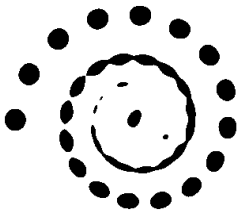
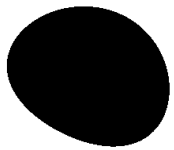
- Large Scale ($\sigma \to \infty$)

$$\begin{aligned} \langle \Phi(X), w \rangle &\simeq \sum_j \alpha_j (1 - \|X - X_j\|^2/\sigma^2) \\ &= -(1/\sigma^2) \sum_j \alpha_j (\langle X, X \rangle + \langle X_j, X_j \rangle - 2\langle X, X_j \rangle) + \dots \\ &= \sum_j \frac{2\alpha_j}{\sigma^2} \langle X, X_j \rangle + Cst = \langle X, \sum_j \frac{2\alpha_j}{\sigma^2} X_j \rangle + Cst \end{aligned}$$

- Small Scale ($\sigma \to 0$)

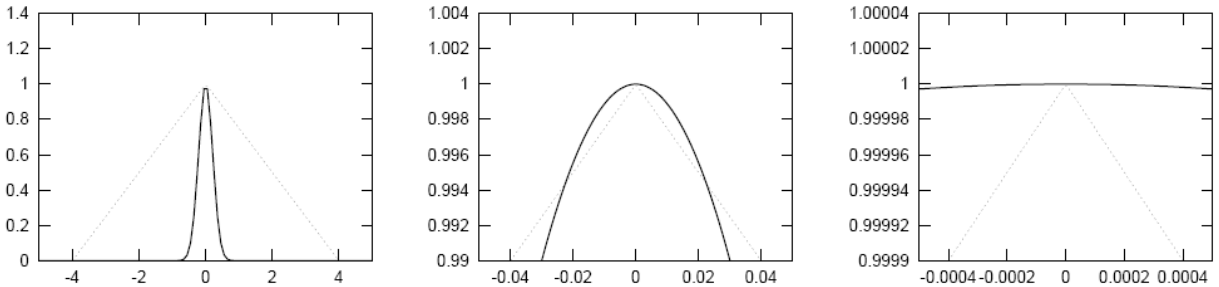
$$\langle \Phi(X), w \rangle = \sum_j \alpha_j \exp(-\|X - X_j\|^2/\sigma^2) \simeq \sum_j \alpha_j \mathbb{1}_{\{X = X_j\}}$$

The Spiral Example


Triangular Kernel

- $k(X, X') = -\|X - X'\|^p, \ p \in]0, 2]$.



- While the Gaussian kernel behaves either as a Dirac-like, a bump or a uniform weighting, the triangular kernel remains similar at all scales.

Scale Invariance

$$\mathcal{T} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$$

$$\gamma\mathcal{T} = \{(\gamma X_1, Y_1), \ldots, (\gamma X_n, Y_n)\}$$

$$F(\mathcal{T}, \alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \, \alpha_j \, Y_i \, Y_j \, k(X_i, X_j)$$

$$F(\gamma\mathcal{T}, \alpha^\gamma) = \sum_i \alpha_i^\gamma - \frac{1}{2} \sum_i \sum_j \alpha_i^\gamma \, \alpha_j^\gamma \, Y_i \, Y_j \, k(\gamma X_i, \gamma X_j)$$

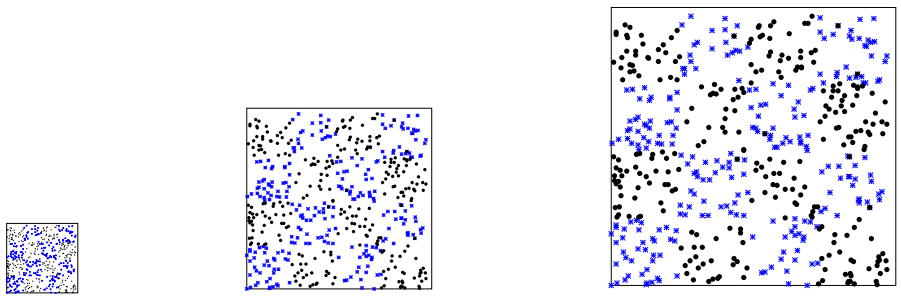
We have [Sahbi & Fleuret, 2002] :

$$\arg\max[F^\gamma(\alpha^\gamma)] = \frac{1}{\gamma^p} \arg\max[F(\alpha)] \quad \text{equivariant}$$

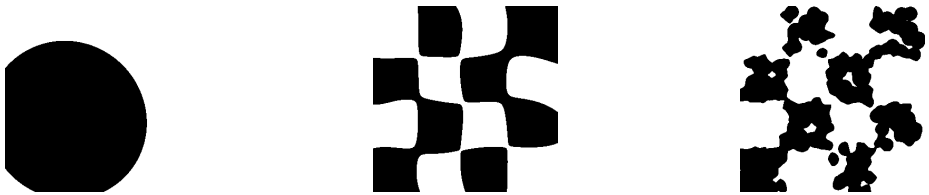
$$g_{\alpha^\gamma}(\gamma X) = g(X) \quad \text{invariant}$$

Scale Invariance (Examples)

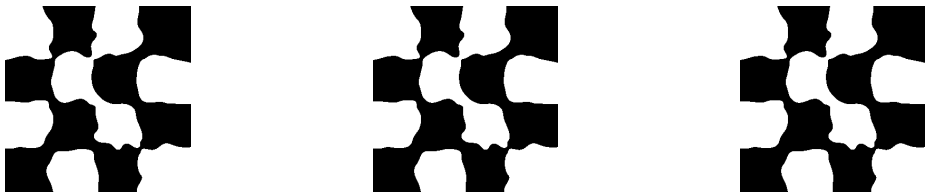
$\times 10^{-1}$ $\times 1$ $\times 10$



Gaussian Kernel ($\sigma = 0.2$)



Triangular Kernel



VC Dimension

- For a given training set $\mathcal{T} = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, we have

$$\begin{bmatrix} g(X_1) \\ g(X_2) \\ \vdots \\ g(X_n) \end{bmatrix} = \begin{bmatrix} k(X_1, X_1) & k(X_1, X_2) & \ldots & k(X_1, X_n) \\ k(X_2, X_1) & k(X_2, X_2) & \ldots & k(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(X_n, X_1) & k(X_n, X_2) & \ldots & k(X_n, X_n) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} + b \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

- Since any Gram matrix built using this kernel is invertible for $0 < p < 2$ [Micchelli86] :
  1. We can learn any function with a null empirical error (i.e., $\forall i$, $g(X_i) = Y_i$).
  2. The VC-dimension of SVMs trained on this kernel is ∞.
- This does not mean that it is bad ! as the actual dimension of data does not exceed n.

Leave One Out Error (LOOE) bound

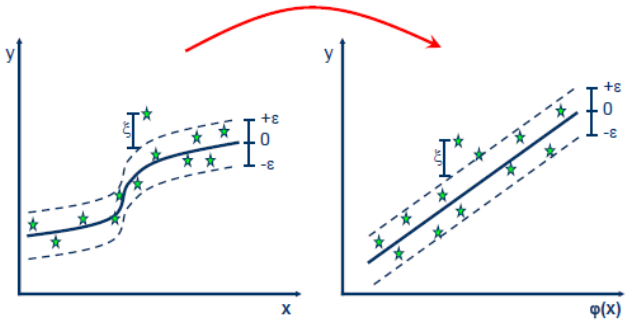
- In SVM, only support vectors are critical for training and classification.
- Training : $\max_{\alpha} \sum_{i}^{n} \alpha_i - \frac{1}{2} \sum_{i,j}^{n} \alpha_i \ \alpha_j \ Y_i \ Y_j \ k(X_i, X_j)$
- Classification : $g(X) = \sum_{i=1}^{n} \alpha_i \ y_i \ k(X, X_i) \ + \ b.$
- When removing the non-support vectors, the optimal solution and hence the decision function remain unchanged.
- The leave-one-out error bound :

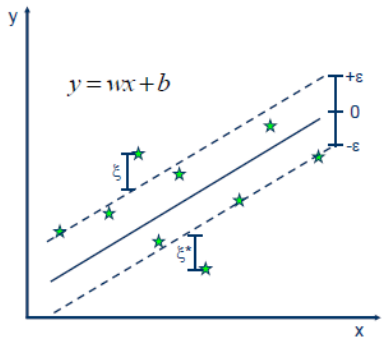
$$R(g) \ \leq \ R_n(g) \ + \ \frac{\#NSV(g)}{n}$$

Support Vector Regression

Support Vector Regression

- Let $\{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, be a training set generated i.i.d according to a probability distribution, now the labels are reals (for instance age prediction, weight prediction, etc.).





- Minimize:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*)$$

- Constraints:

$$y_i - wx_i - b \le \varepsilon + \xi_i$$
$$wx_i + b - y_i \le \varepsilon + \xi_i^*$$
$$\xi_i, \xi_i^* \ge 0$$

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ s.t. \quad & Y_i - (\langle w, \Phi(X_i)\rangle + b) \le \epsilon + \xi_i \\ & (\langle w, \Phi(X_i)\rangle + b) - Y_i \le \epsilon + \xi_i^* \\ & \xi_i, \xi_i^* \ge 0 \end{aligned}$$

Support Vector Regression (Dual Form and Solution)

$$\max_{\alpha} \quad -\frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle\Phi(X_i),\Phi(X_j)\rangle$$

$$-\epsilon\sum_{j=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} Y_i(\alpha_i - \alpha_i^*)$$

$$s.t. \qquad \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0, \quad \alpha_i, \alpha_i^* \in [0, C]$$

$$w = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\Phi(X_i),$$

$$g(X) = \sum_{i}^{n}(\alpha_i - \alpha_i^*)\langle\Phi(X_i),\Phi(X)\rangle + b$$

Section 3

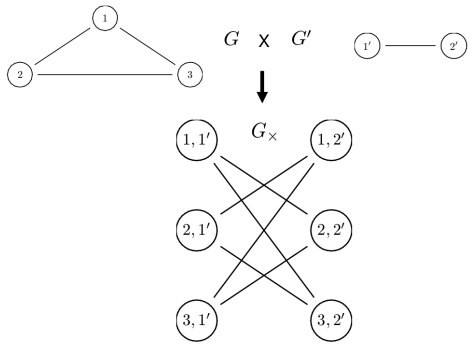
Kernel Design

Standard Kernels (vectorial data)

- Laplacian : $k(X, X') = \exp(-\frac{\|X-X'\|}{\sigma})$
- Hyperbolic Tangent (Sigmoid) : $k(X, X') = tanh(a\langle X, X'\rangle + c)$
- Rational Quadratic : $k(X, X') = 1 - \frac{\|X-X'\|^2}{\|X-X'\|^2+c}$
- Multiquadric : $k(X, X') = \sqrt{\|X - X'\|^2 + c}$
- Inverse Multiquadric : $k(X, X') = \frac{1}{\sqrt{\|X-X'\|^2+c}}$
- Log : $k(X, X') = -\log(\|X - X'\|^d + 1)$
- Cauchy : $k(X, X') = \frac{1}{1+\frac{\|X-X'\|^2}{\sigma^2}}$
- ANOVA : $k(X, X') = \sum_{k=1}^{d} \exp(-\sigma(X^k - X'^k)^2)^d$
- Histogram Intersection : $k(X, X') = \sum_{k=1}^{d} \min(X^k, X'^k)$
- Bayesian : $k(X, X') = \prod_{k=1}^{d} k_l(X^k, X'^k)$ with $k_l(a, b) = \sum_c P(a|c)P(c|b)$.
- Chi-Square : $k(X, X') = 1 - \sum_{k=1}^{d} \frac{X^k-X'^k}{X^k+X'^k}$
- What about data described with graphs or strings (text, social networks, etc.) ?

Standard Kernels on graph data (random walks)

Given $G = (V, E)$ and $G' = (V', E')$.
A product graph $G_x = (V_x, E_x)$ with $V_x = \{(u, u') : u \in V_x,\ u' \in V'_x\}$ and $E_x = \{((u, u'), (v, v')), (u, v) \in E,\ (u', v') \in E'\}$.

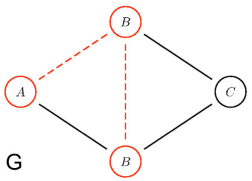


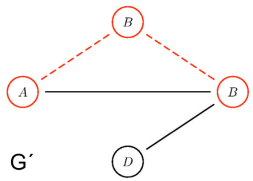
$$A_\times = A_G \otimes A_{G'} = \begin{bmatrix} [A_G]_{11} A_{G'} & \dots & [A_G]_{1n} A_{G'} \\ \vdots & \ddots & \vdots \\ [A_G]_{n1} A_{G'} & \dots & [A_G]_{nn} A_{G'} \end{bmatrix}$$

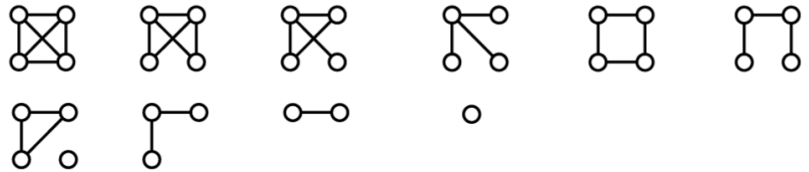
$$k(G, G') = \sum_{(u,u'),(v,v')} \left[\sum_{\ell=0}^{\infty} \lambda^\ell A_\times^\ell \right]_{(u,u'),(v,v')}$$

Standard Kernels on graph data (graphlets)

- Sample and count subgraphs of limited size T in G and G'.
- These subgraphs are referred to as graphlets.
- These subgraphs are not restricted to only chains.







$$k(G, G') = \sum_{g \in \mathcal{D}} \min(H_G(g), H_{G'}(g))$$

Standard Kernels on string data

$K(car, cat)$?

u	c-a	c-t	a-t	b-a	b-t	c-r	a-r	b-r
$\phi_u(cat)$	λ^2	λ^3	λ^2	0	0	0	0	0
$\phi_u(car)$	λ^2	0	0	0	0	λ^3	λ^2	0
$\phi_u(bat)$	0	0	λ^2	λ^2	λ^3	0	0	0
$\phi_u(bar)$	0	0	0	λ^2	0	0	λ^2	λ^3

$$\phi_u(S) = \sum_{\mathbf{i}:u=S[\mathbf{i}]} \lambda^{l(\mathbf{i})}$$

$$k(S, S') = \sum_{u \in \Sigma} \langle \phi_u(S), \phi_u(S') \rangle$$

$K(car, cat) = \lambda^4$.

Explicit Kernel Maps

- Sometimes knowing Φ is very helpful especially when training and testing on very large scale datasets.

$$g(X) \quad = \quad \sum_{i=1}^{n} \alpha_i \; Y_i \; k(X, X_i) \; + \; b$$

- If we know $\Phi()$, then we may pre-compute $w = \sum_i \alpha_i Y_i \Phi(X_i)$, and use

$$g(X) \quad = \quad \langle w, \Phi(X) \rangle + b$$

- The complexity of testing drops from $O(n)$ to $O(1)$, and the complexity of training also drops from $O(n^3)$ to $O(n)$.

Explicit Kernel Maps (Examples)

- Linear : $k(X, X') = \langle X, X' \rangle$ (just identity map).
- Polynomial : $k(X, X') = \langle X, X' \rangle^p = \langle \Phi(X), \Phi(X') \rangle$ with $\Phi(X) = X \otimes ... \otimes X$ (p times).
  Example : Let $k(X, X') = \langle X, X' \rangle^2$, $X = (a_1 \ b_1)$,
  $X' = (a_2 \ b_2)$, $\Phi(X) = X \otimes X = (a_1^2 \ a_1 b_1 \ b_1 a_1 \ b_1^2)$,
  $\Phi(X') = X' \otimes X' = (a_2^2 \ a_2 b_2 \ b_2 a_2 \ b_2^2)$,
  $\langle \Phi(X), \Phi(X') \rangle = a_1^2 a_2^2 + 2 a_1 a_2 b_1 b_2 + b_1^2 b_2^2 = \langle X, X' \rangle^2$.
- Histogram intersection :
  $k(X, X') = \sum_d \min(X^d, X'^d) = \langle \Phi(X), \Phi(X') \rangle$ with
  $\Phi(X) = (\psi(X^1)^t ... \psi(X^d)^t)^t$ and $\psi(.)$ "decimal-to-unary" map.
  Example : $X = \{3, 2, 1\}$, $X' = \{1, 2, 3\}$
  $\langle \Phi(X), \Phi(X') \rangle = \langle 0111 \ 0011 \ 0001, 0001 \ 0011 \ 0111 \rangle = 4$
  $K(X, X') = min(3, 1) + min(2, 2) + min(1, 3) = 4$.

Kernel Design Principles

- So far, we have seen standard kernels. This is not enough !
- How can we design other kernels ? kernel combination ...
- Sums, products and exponentiation of kernels ...

Proposition

$k(X, X') = k_1(X, X') + k_2(X, X')$
$k(X, X') = k_1(X, X').k_2(X, X')$
If k_1, k_2 are p.s.d, then k will also be p.s.d

Proposition

let $k(X, X') = \exp(k_1(X, X'))$
If k_1 is p.s.d, then k will also be p.s.d.

Kernel Design Principles (proofs)

- Sum : $\forall X_1, \ldots, X_n \in \mathcal{X}, \quad \forall c_1, \ldots, c_n \in \mathbb{R}$

$$\begin{aligned}\sum_{i,j} c_i c_j (k_1(X_i, X_j) + k_2(X_i, X_j)) &= \sum_{i,j} c_i c_j k_1(X_i, X_j) \\ &+ \sum_{i,j} c_i c_j k_2(X_i, X_j) \geq 0\end{aligned}$$

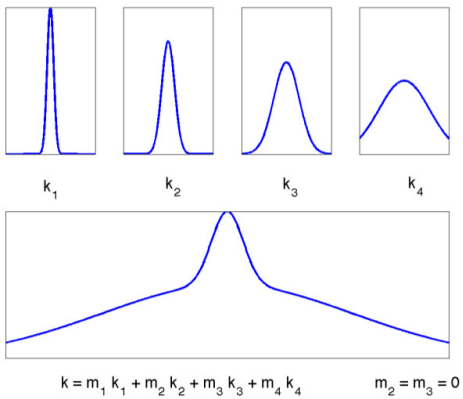
- Product :

$$\begin{aligned}k_1(X_i, X_j).k_2(X_i, X_j) &= \langle \Phi_1(X_i), \Phi_1(X_j) \rangle . \langle \Phi_2(X_i), \Phi_2(X_j) \rangle \\ &= \langle \Phi_1(X_i) \otimes \Phi_2(X_i), \Phi_1(X_j) \otimes \Phi_2(X_j) \rangle \\ &= \langle \Phi(X_i), \Phi(X_j) \rangle\end{aligned}$$

- Exponential (Taylor's expansion and closure of p.s.d w.r.t sums/products) : $exp(k_1(X_i, X_j)) = \sum_{\ell=0}^{\infty} \frac{1}{\ell!} (k_1(X_i, X_j))^{\ell}$

Multiple Kernel Learning (MKL)

- Use a weighted linear combination of kernels $k(X, X') = \sum_{m=1}^{M} \beta_m k_m(X, X')$, with $\beta_m \geq 0$ (to guarantee p.s.d) and $\sum_m \beta_m = 1$ (convex or sparse combination).



$$g(X) = \sum_{m=1}^{M} \beta_m \Big(\sum_{i=1}^{n} \alpha_i k_m(X, X_i) + b_m \Big) = \sum_{m=1}^{M} \beta_m \; g_m(X)$$

Multiple Kernel Learning (MKL), Bi-level Optimization

- $g_m(X) = \langle w_m, \Phi(X) \rangle + b_m$

$$\min_{\{w_m, b_m\}, \xi, \{\beta_m\}} \quad \frac{1}{2} \sum_m \beta_m \|w_m\|^2 + C \sum_i \xi_i$$

$$\text{s.t.} \quad Y_i \sum_m \beta_m \, g_m(X) + \xi_i \geq 1, \ \forall \ i$$

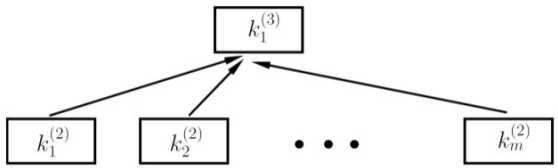
$$\xi_i \ \geq \ 0, \ \beta_m \geq 0, \ \textstyle\sum_m \beta_m = 1$$

$$\max_{\{\alpha_i\}} \quad \sum_i \alpha_i \ - \ \frac{1}{2} \sum_i \sum_j \alpha_i \ \alpha_j \ Y_i \ Y_j \ \sum_m \beta_m k_m(X_i, X_j)$$

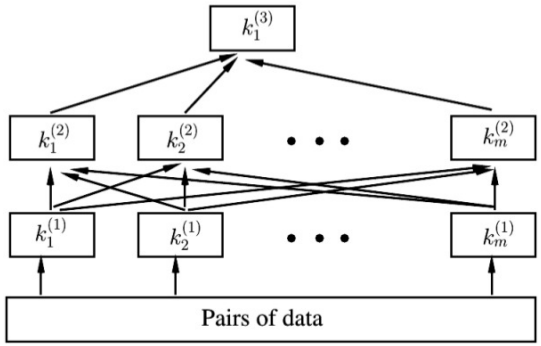
$$\text{s.t.} \quad \alpha_i \geq 0, \ \ \forall \ i \qquad \text{and} \qquad \sum_i \alpha_i \ Y_i \ = 0$$

- Step 1 : fix $\{\beta_m\}$ and solve the above QP
- Step 2 : fix $\{\alpha_i\}$ (so $\{(w_m, b_m)\}_m$) and solve the above LP

Deep Multiple Kernel Learning



shallow MKL



deep MKL

- Deep MKL is a multi-layer perceptron (MLP), where k_p^ℓ at layer ℓ is a nonlinear combination of kernel values at layer $(\ell - 1)$

$$k_p^\ell(X, X') = \sigma\left(\sum_q w_{q,p}^{(\ell-1)} k_q^{(\ell-1)}(X, X')\right)$$

- When combined with SVMs, the whole frame becomes semi-parametric (parametric in kernel design and non-parametric in SVM learning)

Deep Multiple Kernel Learning : Optimization

- Step 1 : fix $\{w_{q,p}\}$ (so the deep kernel) and solve the QP for SVM parameters (similarly to MKL).
- Step 2 : fix the SVM parameters and solve the QP (denoted J) now using backpropagation+gradient descent

Algorithm (Deep MKL)

Inputs : Initial $\mathbf{w}^{(\ell)}(\ell = 1, \ldots, L-1)$, and $\frac{\partial J}{\partial \kappa_{\mathbf{1}}^{(L)}(.,.)}$

Outputs : Optimal $\mathbf{w}^{(\ell)}(\ell = 1, \ldots, L-1)$

Repeat till convergence

for $\ell = L : 2$ **do**

Compute gradient : $\frac{\partial J}{\mathbf{w}_{q,p}^{(\ell-\mathbf{1})}} = \sum_{i,j} \frac{\partial J}{\partial k_p^{(\ell)}(\mathbf{x}_i,\mathbf{x}_j)} \frac{\partial k_p^{(\ell)}(\mathbf{x}_i,\mathbf{x}_j)}{\mathbf{w}_{q,p}^{(\ell)}}$

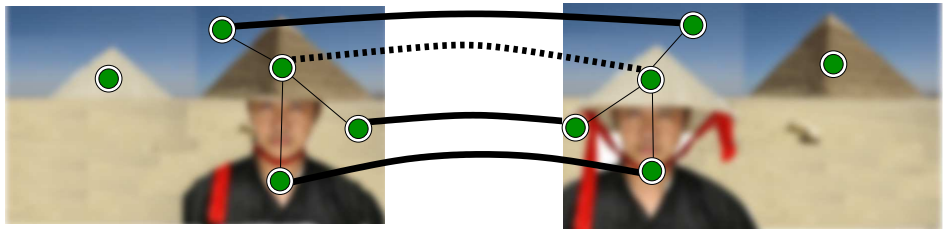
Compute sensitivity : $\frac{\partial J}{\partial k_q^{(\ell-\mathbf{1})}(\mathbf{x}_i,\mathbf{x}_j)} = \sum_q \frac{\partial J}{\partial k_p^{(\ell)}(\mathbf{x}_i,\mathbf{x}_j)} \frac{\partial k_p^{(\ell)}(\mathbf{x}_i,\mathbf{x}_j)}{\partial k_q^{(\ell-\mathbf{1})}(\mathbf{x}_i,\mathbf{x}_j)}$

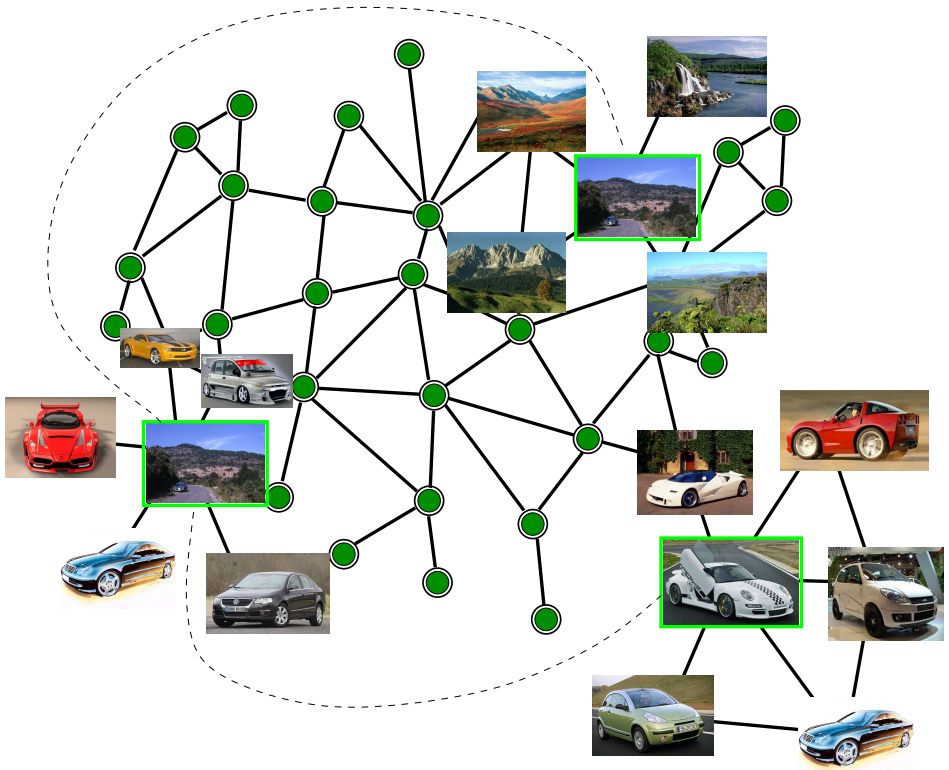
end

Update the weights $\mathbf{w}_{q,p}^{(\ell)}$: $\mathbf{w}_{q,p}^{(\ell)} = \mathbf{w}_{q,p}^{(\ell)} - \beta \frac{\partial J}{\mathbf{w}_{q,p}^{(\ell)}} \quad \ell = 1, \ldots, L-1.$

EndRepeat

Context Dependent Kernels (Image Comparison)





Context Dependent Kernels (Problem Formulation)

$$\min_{k} \quad \sum_{i,j} k(X_i, X_j)\; d(X_i, X_j) \;+\; \beta \sum_{i,j} k(X_i, X_j)\; \log(k(X_i, X_j))$$

$$+\alpha \sum_{i,j} k(X_i, X_j) \left(- \sum_{\substack{X_k \in \mathcal{N}(X_i), \\ X_\ell \in \mathcal{N}(X_j)}} k(X_k, X_\ell) \right)$$

$$\text{s.t.} \quad k(X_i, X_j) \in [0, 1], \quad \sum_{i,j} k(X_i, X_j) = 1$$

$$k_t(X_i, X_j) \quad \propto \quad \exp\left(-\frac{d(X_i, X_j)}{\beta} - 1 \right)$$

$$\exp\left(\frac{2\alpha}{\beta} \sum_{k,\ell} 1_{\{X_k \in \mathcal{N}(X_i)\}} 1_{\{X_\ell \in \mathcal{N}(X_j)\}}\; k_{t-1}(X_k, X_\ell) \right)$$

Context Dependent Kernels (P.S.D)

- By induction : $k_0(X_i, X_j) = \exp\left(-\frac{d(X_i,X_j)}{\beta} - 1\right)$ is p.s.d
- Assuming k_{t-1} p.s.d, $k_t(X_i, X_j) \propto$

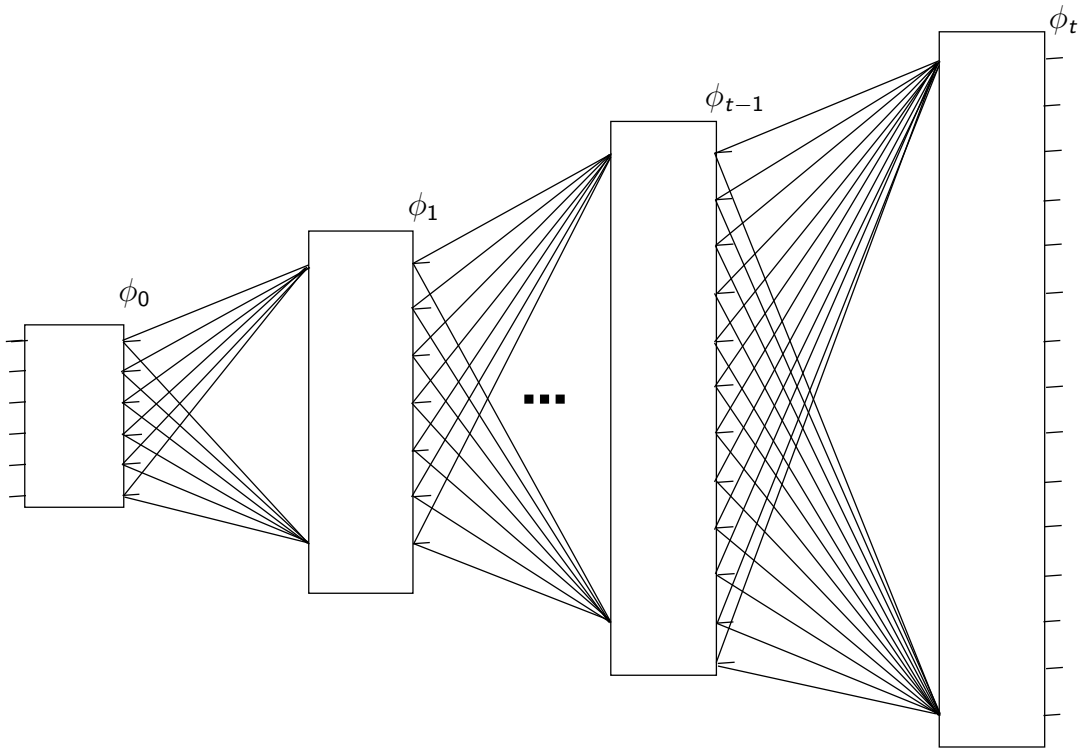
$$\exp\left(-\frac{d(X_i,X_j)}{\beta} - 1\right).$$

$$\exp\left(\frac{2\alpha}{\beta}\left\langle \sum_k 1_{\{X_k \in \mathcal{N}(X_i)\}} \Phi_{t-1}(X_k), \sum_\ell 1_{\{X_\ell \in \mathcal{N}(X_j)\}}\ \Phi_{t-1}(X_l)\right\rangle\right)$$

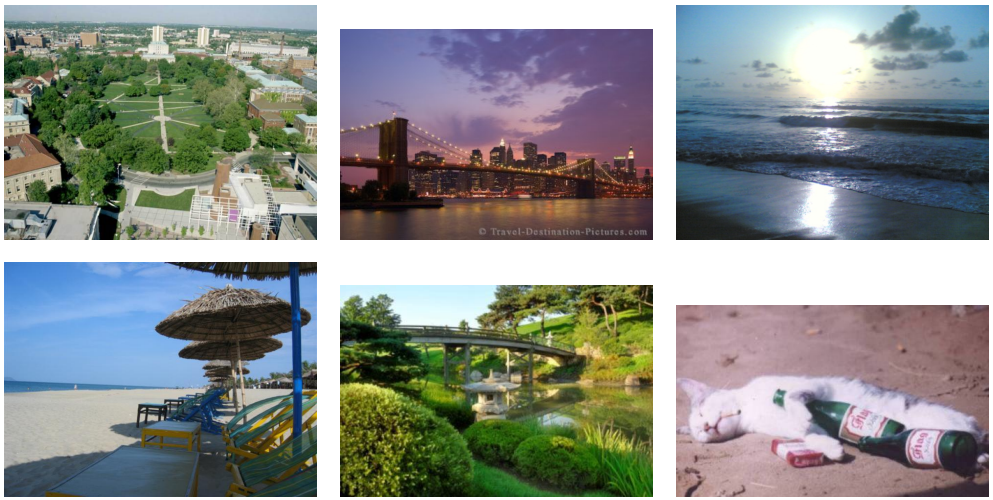
- Resulting from the closure of p.s.d w.r.t exponentiation and product, it follows that k_t is also p.s.d.

Relation to Deep-Learning

$$k_t(X, X') = \underbrace{\phi_t(\phi_{t-1}(\ldots\phi_1(\phi_0(X))))}_{t \text{ times}} \cdot \underbrace{\phi_t(\phi_{t-1}(\ldots\phi_1(\phi_0(X'))))}_{t \text{ times}}$$



Context Dependent Kernels : ImageCLEF Benchmark



- 250k training images, 1k images for dev and 2k images for test (labels not available for participants). Nbr of concepts : 116.
- Visual features provided by the organizers including GIST, Color Histograms, SIFT, C-SIFT, RGB-SIFT and OPPONENT-SIFT. For all the SIFT-based descriptors, a bag-of-words representation is provided.

Context Dependent Kernels : ImageCLEF Benchmark

- In order to build the graph of images, the latter are connected if the number of their shared keywords (in meta-data) is sufficiently large.
- We use the resulting context dependent kernel and "one-versus-all" SVMs for each concept/class. The score of these SVMs decide about the presence of these concepts in images.
- Histogram intersection kernel is used for initialization.

	CF	CD				
$\frac{2\alpha}{\beta}$	0	10^{-3}	10^{-2}	10^{-1}	1	10
F-scores (samples)	41.40	40.17	41.31	46.70	**51.30**	48.14
F-scores (concepts)	30.90	29.77	31.03	37.55	**45.00**	42.21

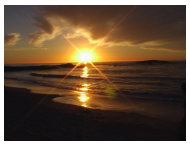
Context Dependent Kernels : Comparison with MKL

- For comparison, we build 8 kernel matrices corresponding to the 7 visual features and 1 tag-based (TF-IDF) features ; we learn an "optimal" linear combination of these kernels via MKL.

	F-scores (samples)	F-scores (concepts)
Visual without Context		
Standard SVM+Linear kernel	37.80	28.45
Standard SVM+Polynomial kernel	33.06	26.04
Standard SVM+HI kernel	41.40	30.90
Visual + Context		
MKL SVM based on Linear	46.58	43.01
MKL SVM based on Polynomial	43.81	42.18
MKL SVM based on HI	49.49	45.10
Visual + Context		
SVM+ CD kernel	51.30	45.00

- Improvement of CDK is not only due to the exploitation of the tags but also due to the way we use context in kernel design.

Context Dependent Kernels : Annotation Examples

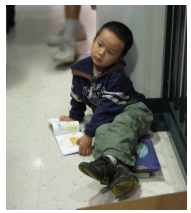


(a)
CF : beach cloud coast outdoor reflection sea silhouette sky sun sunset water
CA : beach cloud coast reflection **sand** sea silhouette sky sun sunset water

(b)
CF : countryside daytime grass outdoor plant sky soil
CA : countryside daytime grass **motorcycle** outdoor plant sky

(c)
CF : **building** closeup daytime flower furniture garden grass outdoor park plant tree
CA : daytime **dog** flower furniture grass outdoor plant tree

(d)
CF : **boat** building castle cityscape cloud cloudless daytime grass **harbor lake** outdoor plant shadow sky tree vehicle
CA : building castle cloud cloudless daytime grass **mountain** outdoor plant shadow sky tree



(e)
CF : **bird building** child **dog elder** footwear furniture indoor male person **sport teenager**
CA : child daytime footwear indoor person

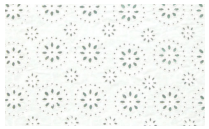
(f)
CF : book **building church cityscape** furniture indoor **painting rain sign train**
CA : book furniture indoor

(g)
CF : closeup **fish** flower outdoor plant
CA : closeup flower plant

(h)
CF : **bicycle book car** cartoon diagram drum **guitar** instrument **motorcycle poster**
CA : cartoon diagram drum instrument logo

(i)
CF : airplane cloudless daytime **helicopter logo** outdoor **sea** sky sport vehicle water
CA : airplane cloudless daytime outdoor sky vehicle water

(j)
CF : **aerial** beach **boat bridge cityscape** cloud coast countryside daytime **harbor** outdoor **plant river** sand sea sky water
CA : beach cloud coast daytime outdoor sand sea sky water



(k)
CF : **cat** diagram embroidery **plant**
CA : **cartoon** diagram embroidery unpaved

(l)
CF : **beach bridge cat** cloudless **coast** countryside daytime desert outdoor **plant road** sand sky soil **vehicle**
CA : cloudless daytime desert **mountain** outdoor sand sky soil

(m)
CF : cloud **coast** countryside daytime grass mountain outdoor plant **sea** sky soil **water**
CA : cloud countryside daytime forest grass mountain outdoor plant sky **tree**

(n)
CF : daytime **dog** forest outdoor park plant **reflection river** soil tree unpaved **water**
CA : bridge daytime forest **grass** outdoor plant road **sky** soil tree unpaved

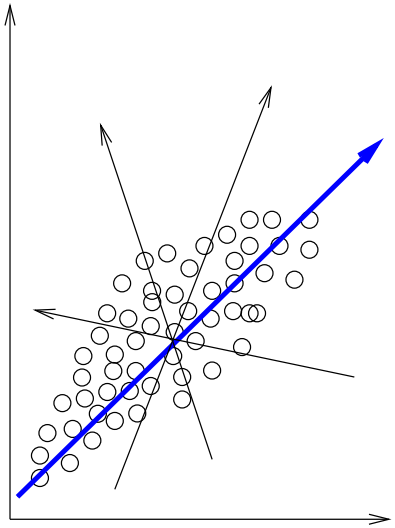
(o)
CF : **cat** daytime elder male outdoor person **plant** sand sky soil teenager unpaved
CA : **beach building castle** daytime outdoor person sand sculpture unpaved

Section 4

Unsupervised Learning (Kernel PCA and CCA)

Principal Component Analysis

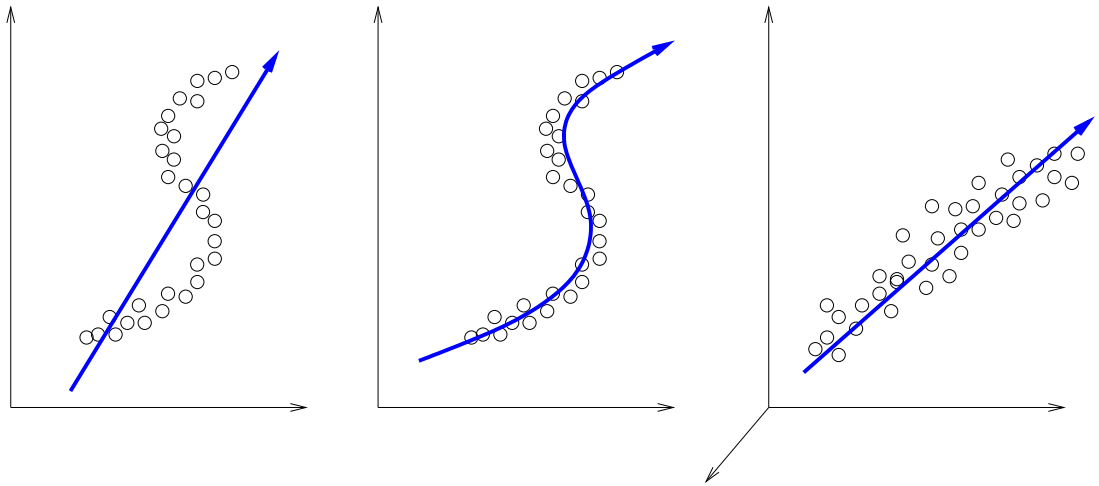
Principal Component Analysis



- Given a training set $\{X_1, \ldots, X_n\}$ (no labels are available).
- PCA finds the principal axes as the eigenvectors of

$$M = \frac{1}{n} \sum_{j=1}^{n} X_j \, {X_j}^t$$

Kernel Principal Component Analysis



- A function Φ maps the data from the original (input) space into a high dimensional (feature) space.
- A space of dimension n, guarantees the existence of a linear manifold for at most $n+1$ training data.

Kernel Principal Component Analysis

$$M = \frac{1}{n} \sum_{j=1}^{n} \Phi(X_j)\ \Phi(X_j)^t$$

$$\forall k = 1, ..., n, \quad \exists \alpha_{k1}, ..., \alpha_{kn} \in \mathbb{R} \quad \text{s.t.} \quad V_k = \sum_{j=1}^{n} \alpha_{kj}\ \Phi(X_j)$$

where $\alpha_k = (\alpha_{k1}, ..., \alpha_{kn})$ are found by solving the following eigenproblem :

$$K\ \alpha_k = \lambda_k\ \alpha_k$$

- K is the Gram matrix. $K_{ji} = \langle \Phi(X_j), \Phi(X_i) \rangle = k(X_j, X_i)$.
- Using the kernel trick, the projection of a training sample in the mapping space :

$$\langle \Phi(X), V_k \rangle = \sum_{j=1}^{n} \alpha_{kj}\ \langle \Phi(X), \Phi(X_j) \rangle = \sum_{j=1}^{n} \alpha_{kj}\ k(X, X_j)$$

Similarity Invariance

- KPCA is translation and rotation invariant for some kernels (e.g., triangular). It's encoded in kernels.

$$\begin{aligned} k(\mathbf{R}\, X_i + \mathbf{t}, \mathbf{R}\, X_j + \mathbf{t}) &= -\|\mathbf{R}\, X_i + \mathbf{t} - \mathbf{R}\, X_j - \mathbf{t}\|^p \\ &= -\|X_i - X_j\|^p = k(X_i, X_j) \end{aligned}$$

- Scale ? the kernel is not invariant (just equivariant), but KPCA projections can be made invariant.

$$\mathcal{S} = \{X_1, ..., X_n\}$$

$$\gamma\mathcal{S} = \{\gamma\, X_1, ..., \gamma\, X_n\}$$

- We need to show :

$$\forall\, X \in \mathbb{R}^n \quad \forall\, k = 1, ..., n \quad \langle V_k^{(\gamma)}, \gamma\, X \rangle = \langle V_k^{(1)}, X \rangle$$

Scale Invariance

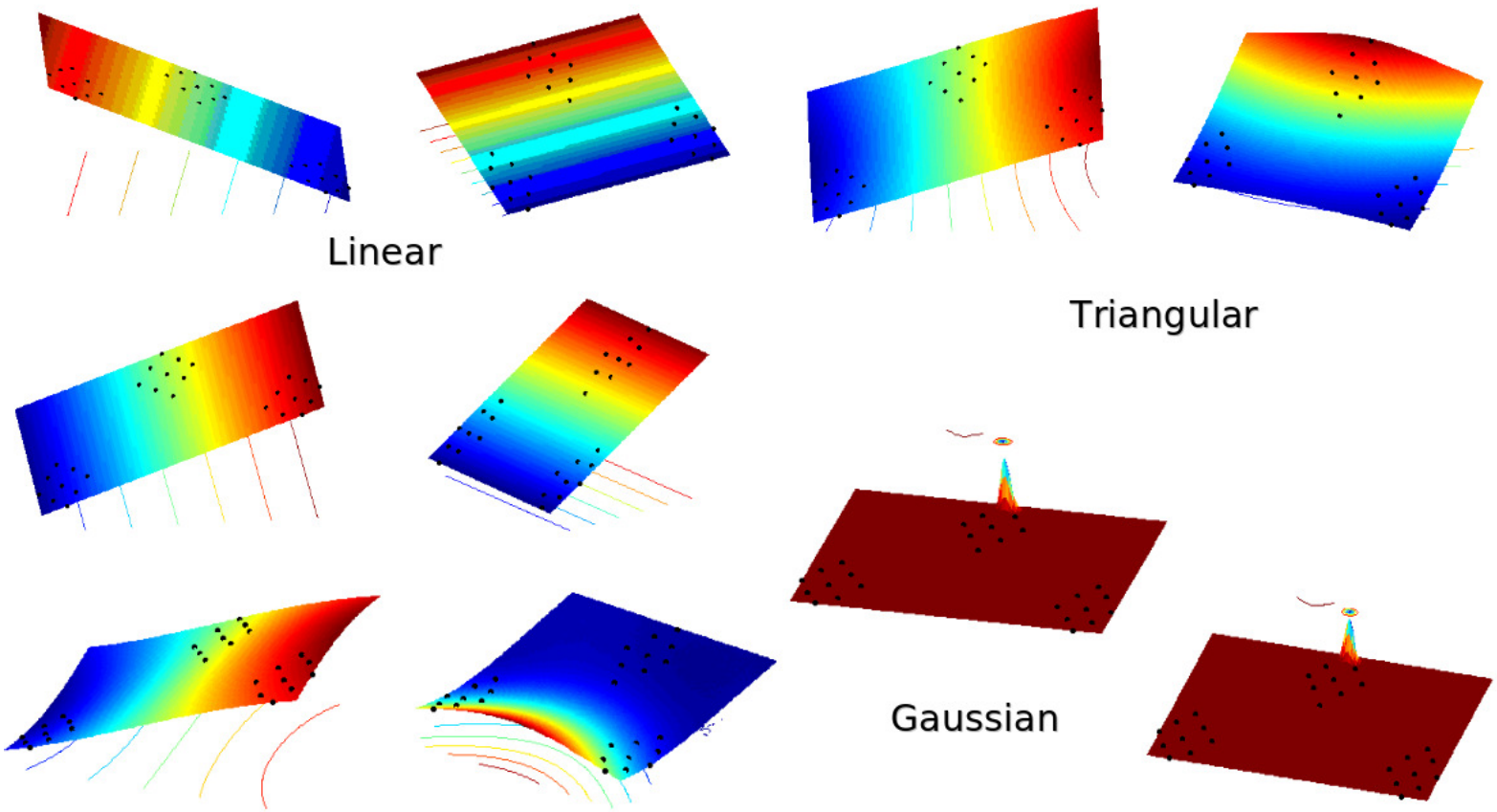
- Let's consider a new expansion of each eigenvector :

$$V_k^{(\gamma)} = \frac{1}{\lambda_1^\gamma} \sum_j^n \alpha_{kj}^\gamma \, \Phi(\gamma \, X_j)$$

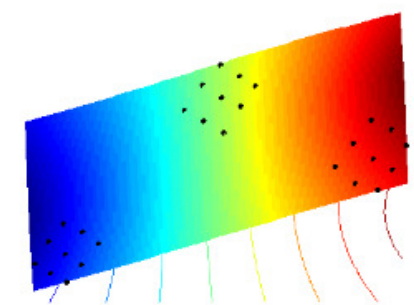
here α_k^γ comes from the eigenproblem $K^\gamma \, \alpha_k^\gamma = \lambda_k^\gamma \, \alpha_k^\gamma$.

- The proof comes from the fact that the Gram matrix K^γ of the scaled set can be written as : $K^\gamma = \gamma^p \, K^1$
- which implies : $\forall \; k = 1, ..., n, \; \lambda_k^\gamma = \gamma^p \, \lambda_k^1 \quad and \quad \alpha_k^\gamma = \alpha_k^1$.
- So $\forall \; X \qquad \langle V_k^{(\gamma)}, \gamma \, X \rangle = \langle V_k^{(1)}, X \rangle \quad \square$

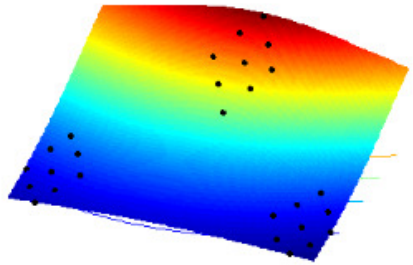
Examples



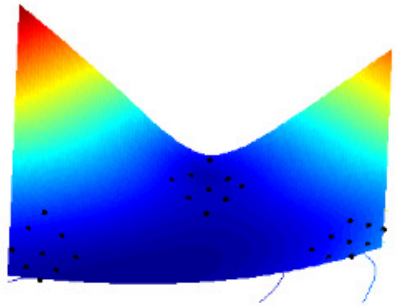
Examples : Dimensions



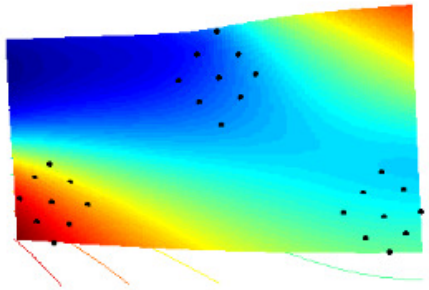
d=1



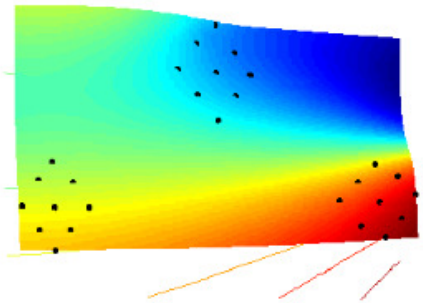
d=2



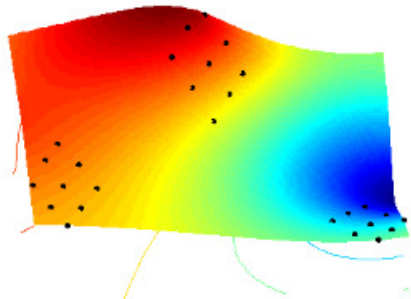
d=3



d=4

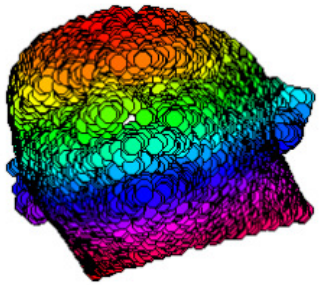


d=5

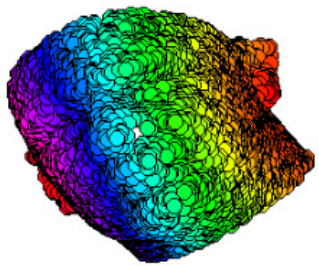


d=6

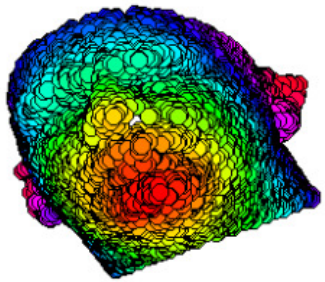
Examples : Dimensions



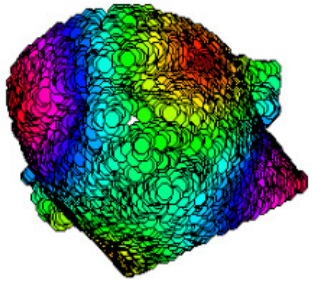
d=1



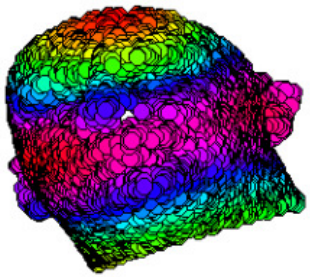
d=2



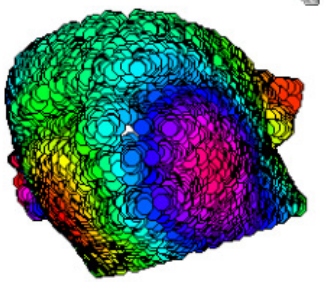
d=3



d=4

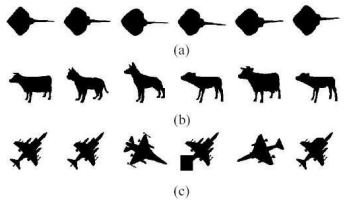


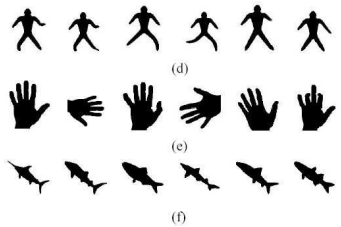
d=5

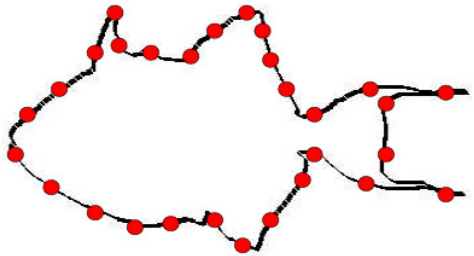


d=6

Kernel PCA and Shape Description & Recognition

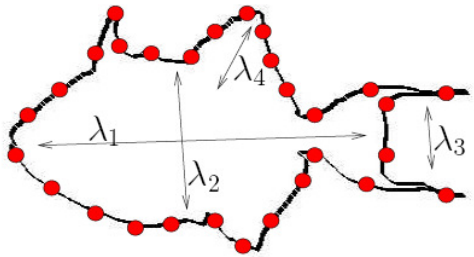






$$X=\begin{bmatrix} \bullet & \bullet & \cdots & \bullet \end{bmatrix} \qquad K_{ji} = k(X_j, X_i)$$

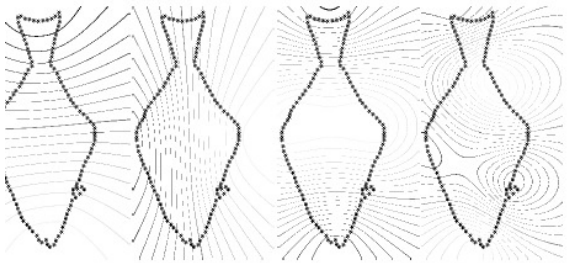
$$\begin{aligned} K\,\alpha_1 &= \lambda_1\,\alpha_1 \\ &\vdots \\ K\,\alpha_N &= \lambda_N\,\alpha_N \end{aligned}$$

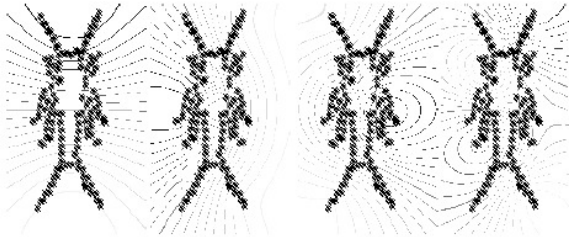


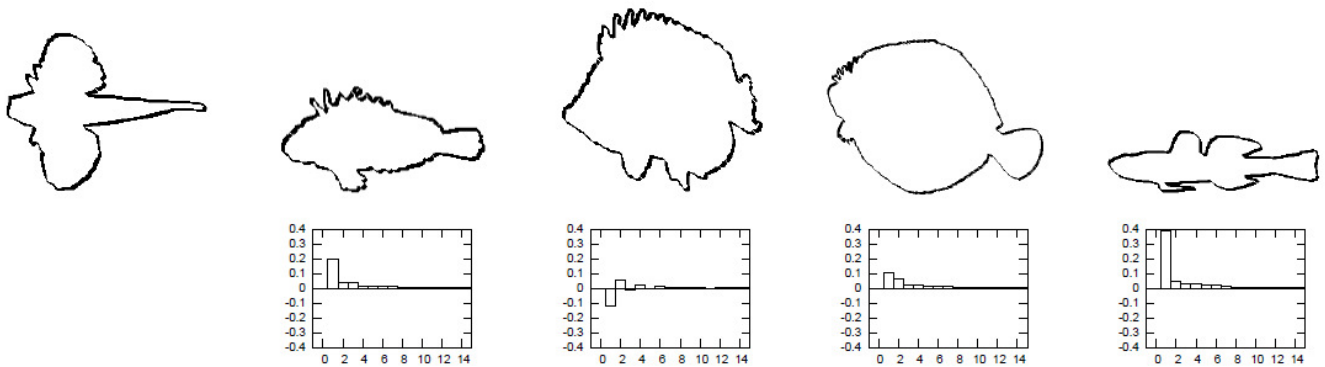
$$\begin{bmatrix} \lambda_1/\lambda_1 \\ \lambda_2/\lambda_1 \\ \lambda_3/\lambda_1 \\ \vdots \\ \lambda_p/\lambda_1 \end{bmatrix}$$

p<< N, this will be translation, rotation and scale invariant.

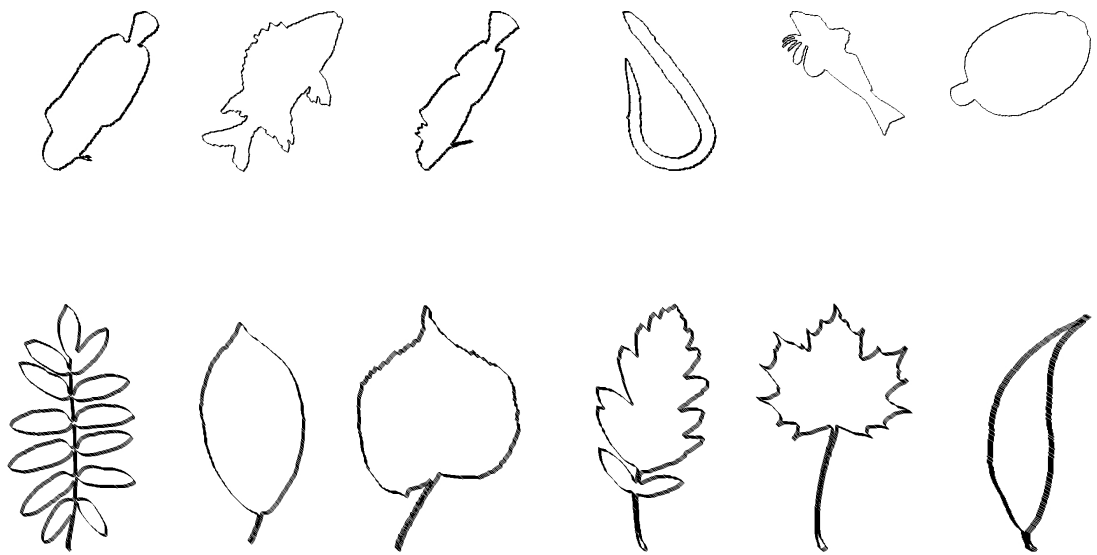
Interpretation





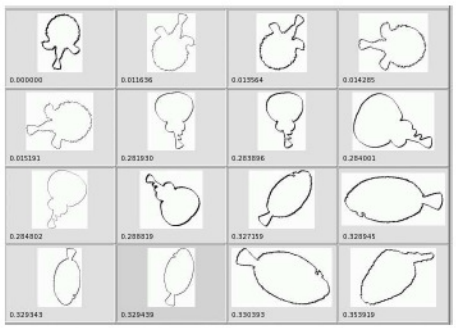


Databases

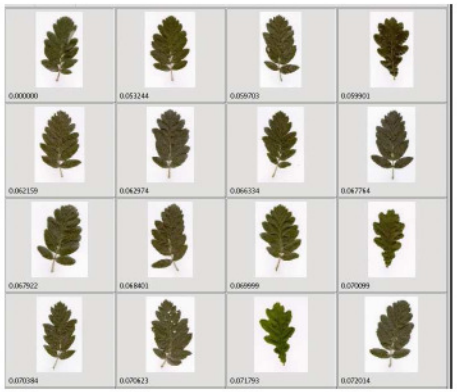


- **SQUID** : unavailable ground truth (5500 contours).
- **Swedish** : (15 categories with 75 images each).
- **SmithSonian** : (135 categories).
- Performances measured using Recall, Precision.

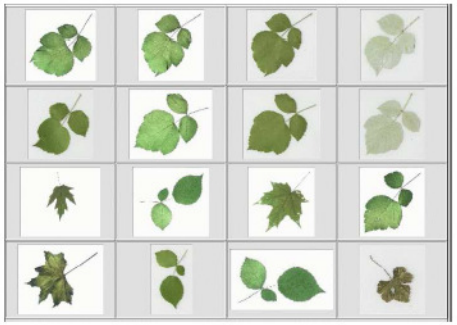
Performance

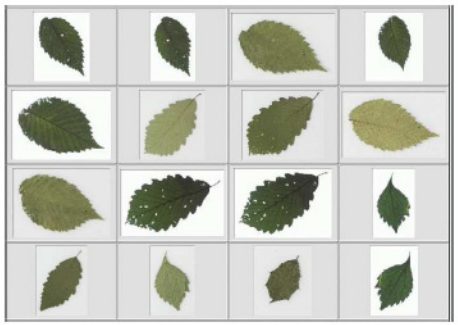


SQUID



Swedish

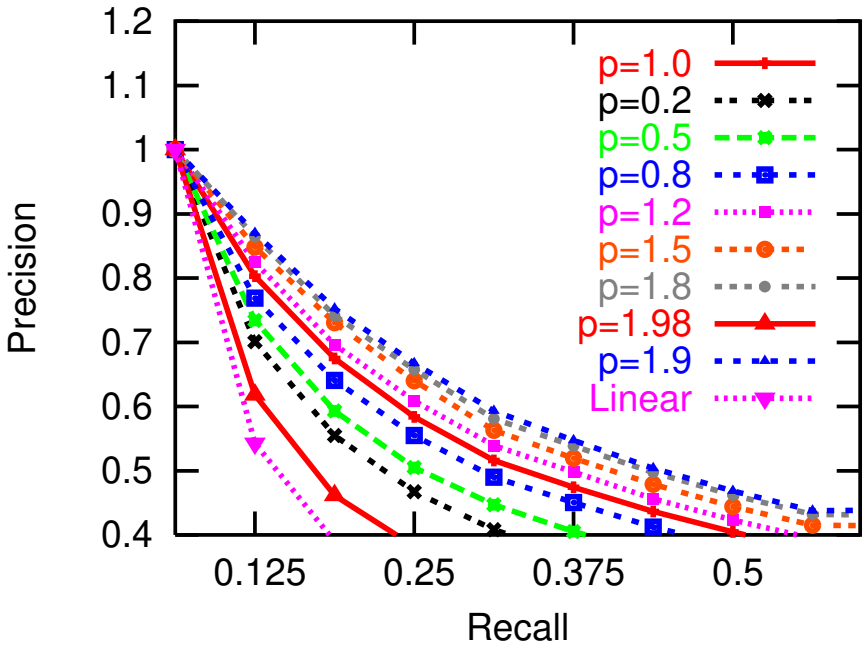


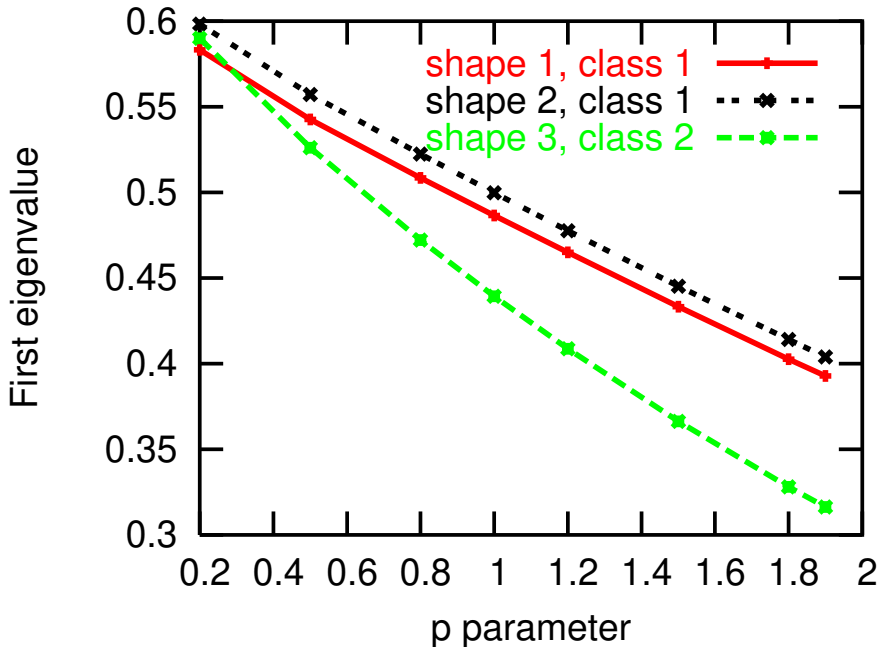


SmithSonian

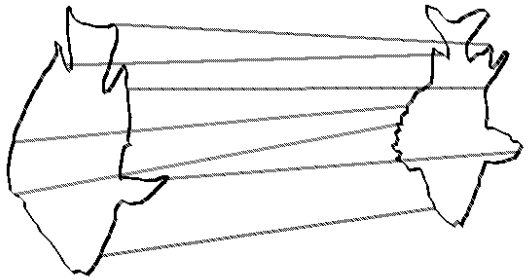
Recall (over 16)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Precision																
Linear PCA	1.0	.59	.46	.40	.35	.32	.31	.29	.28	.27	.27	.26	.25	.25	.24	.24
Kernel PCA	**1.0**	**.967**	**.949**	.935	.925	.914	.903	.892	.882	.871	.860	.851	.841	.832	.825	.816
Hough	1.0	.899	.848	.814	.785	.765	.749	.737	.724	.711	.702	.693	.684	.677	.671	.664
EOH	1.0	.752	.651	.592	.554	.528	.508	.490	.474	.462	.453	.444	.436	0.431	.424	.418
CSS	1.0	.959	.936	.924	.916	.914	.905	.899	.892	.888	.884	.879	.876	.872	.867	.864

Kernel Parameters





Matching

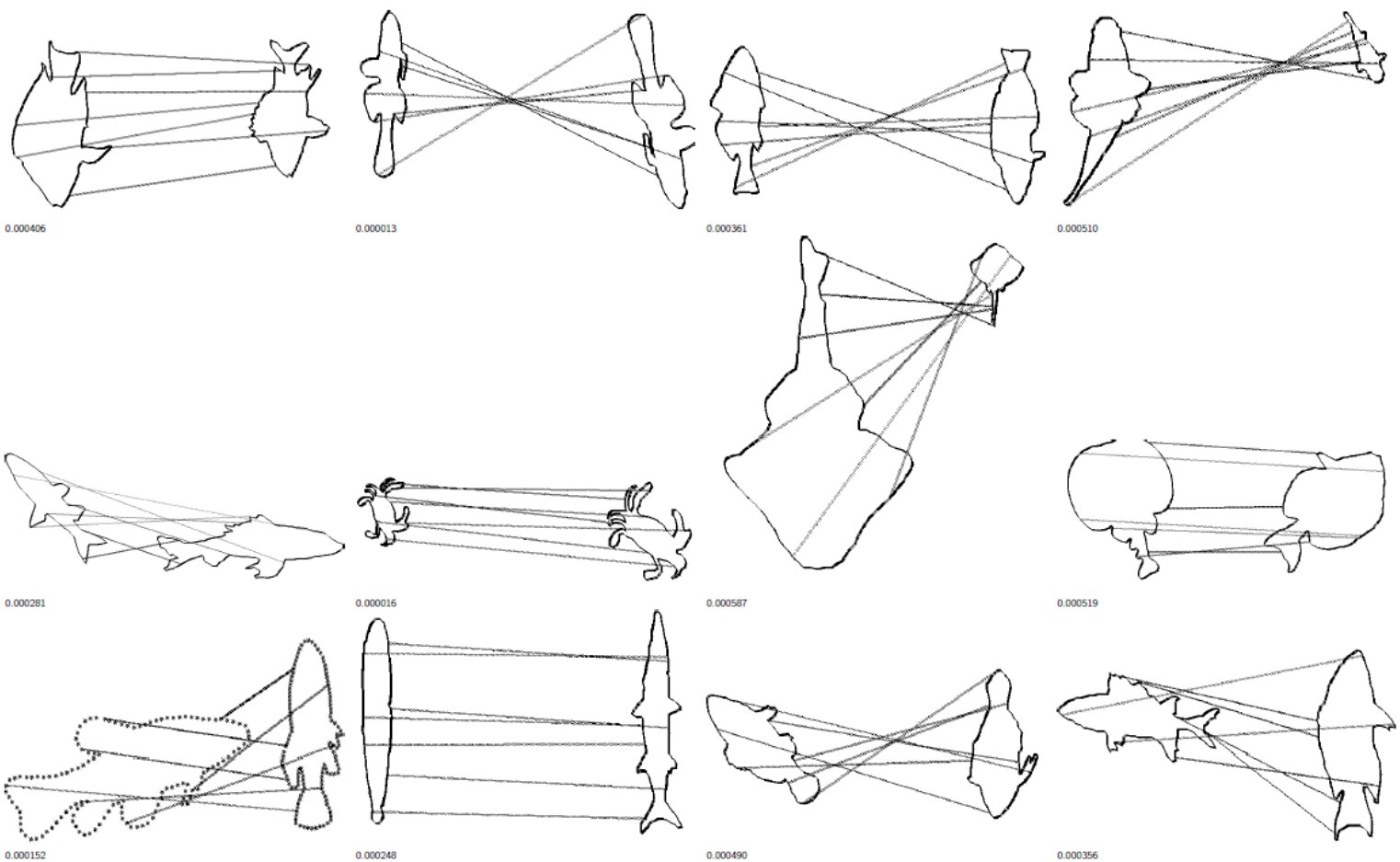


Given two curves $\mathcal{S}_1$, $\mathcal{S}_2$ and $X_i \in \mathcal{S}_1$, we define $X'_j \in \mathcal{S}_2$ as a match of X_i if :

$$X'_j = \arg \min_{X' \in \mathcal{S}_2} \|\pi(X_i) - \pi(X')\|^2$$

$$\pi(X_i) = \begin{bmatrix} \langle \Phi(X_i), V_1 \rangle \\ . \\ . \\ . \\ \langle \Phi(X_i), V_d \rangle \end{bmatrix} \quad \pi(X'_j) = \begin{bmatrix} \langle \Phi(X'_j), V'_1 \rangle \\ . \\ . \\ . \\ \langle \Phi(X'_j), V'_d \rangle \end{bmatrix}$$

Matching

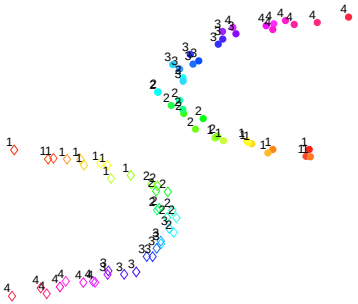
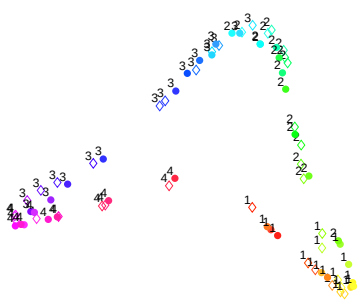

0.000406
0.000013
0.000361
0.000510
0.000281
0.000016
0.000587
0.000519
0.000152
0.000248
0.000490
0.000356

Canonical Correlation Analysis

Canonical Correlation Analysis

- Given aligned training data $\{(X_1, X_1'), \ldots, (X_n, X_n')\}$ (taken from two different views ; for instance camera 1 and camera 2, two languages, document and text, etc.)


- Canonical correlation analysis maps the two views into a common (latent) space where data become highly correlated.

Canonical Correlation Analysis

$$\begin{aligned} \max_{\mathbf{P}_1,\mathbf{P}_2} \quad & \sum_i \langle \mathbf{P}_1^t X_i, \mathbf{P}_2^t X_i' \rangle \\ \text{s.t.} \quad & \sum_i \langle \mathbf{P}_1^t X_i, \mathbf{P}_1^t X_i \rangle = 1 \\ & \sum_i \langle \mathbf{P}_2^t X_i', \mathbf{P}_2^t X_i' \rangle = 1 \end{aligned}$$

- In a matrix form :

$$\begin{aligned} \max_{\mathbf{P}_1,\mathbf{P}_2} \quad & \mathbf{P}_1^t \mathbf{X} \mathbf{X}'^t \mathbf{P}_2 \\ \text{s.t.} \quad & \mathbf{P}_1^t \mathbf{X} \mathbf{X}^t \mathbf{P}_1 = 1 \\ & \mathbf{P}_2^t \mathbf{X}' \mathbf{X}'^t \mathbf{P}_2 = 1 \end{aligned} \qquad \begin{aligned} \max_{\mathbf{P}_1,\mathbf{P}_2} \quad & \mathbf{P}_1^t \mathbf{C}_{12} \mathbf{P}_2 \\ \text{s.t.} \quad & \mathbf{P}_1^t \mathbf{C}_{11} \mathbf{P}_1 = 1 \\ & \mathbf{P}_2^t \mathbf{C}_{22} \mathbf{P}_2 = 1 \end{aligned}$$

Canonical Correlation Analysis

- The corresponding Lagrangian is

$$L(\lambda, \mathbf{P}_1, \mathbf{P}_2) = \mathbf{P}_1^t \mathbf{C}_{12} \mathbf{P}_2 - \frac{\lambda_1}{2}(\mathbf{P}_1^t \mathbf{C}_{11} \mathbf{P}_1 - 1) - \frac{\lambda_2}{2}(\mathbf{P}_2^t \mathbf{C}_{22} \mathbf{P}_2 - 1)$$

- Taking derivatives w.r.t $\mathbf{P}_1$ and $\mathbf{P}_2$, one obtains

$$\frac{\partial L}{\partial \mathbf{P}_1} = \mathbf{C}_{12}\mathbf{P}_2 - \lambda_1 \mathbf{C}_{11}\mathbf{P}_1 = 0$$

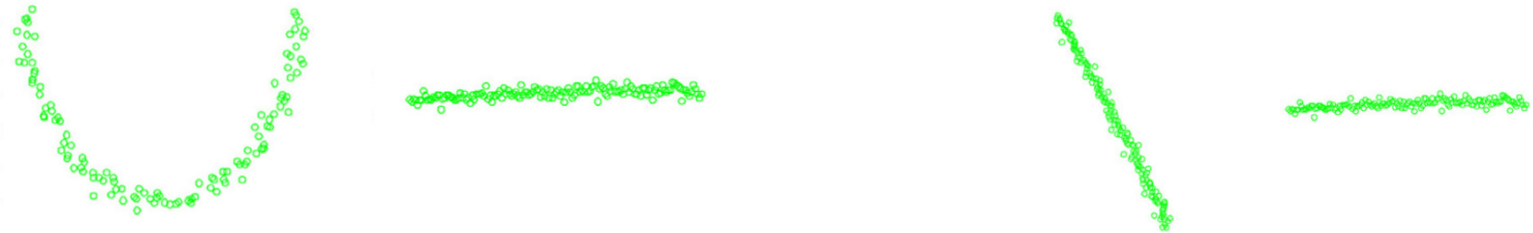
$$\frac{\partial L}{\partial \mathbf{P}_2} = \mathbf{C}_{21}\mathbf{P}_1 - \lambda_2 \mathbf{C}_{22}\mathbf{P}_2 = 0$$

$$\mathbf{C}_{12}\mathbf{C}_{22}^{-1}\mathbf{C}_{21}\mathbf{P}_1 = \lambda^2 \mathbf{C}_{11}\mathbf{P}_1$$

$$\mathbf{P}_2 = \frac{\mathbf{C}_{22}^{-1}\mathbf{C}_{21}\mathbf{P}_1}{\lambda}$$

Kernel Canonical Correlation Analysis

- when $p \ll n$, linear transformations are not appropriate.



- $\mathbf{X} = [X_1, \ldots, X_n] \mapsto \Phi(\mathbf{X}) = [\Phi(X_1), \ldots, \Phi(X_n)]$.
- $\mathbf{X}' = [X'_1, \ldots, X'_n] \mapsto \Phi(\mathbf{X}') = [\Phi(X'_1), \ldots, \Phi(X'_n)]$.

$$\mathbf{P}_1 = \Phi(\mathbf{X})\alpha_1, \quad \mathbf{P}_2 = \Phi(\mathbf{X}')\alpha_2$$

$$\begin{array}{lll} \max\limits_{\alpha_1,\alpha_2} & \alpha_1^t \Phi(\mathbf{X}^t)\Phi(\mathbf{X})\Phi(\mathbf{X}'^t)\Phi(\mathbf{X}')\alpha_2 = \max\limits_{\alpha_1,\alpha_2} \alpha_1^t K_1 K_2 \alpha_2 & \\ \text{s.t.} & \alpha_1^t \Phi(\mathbf{X}^t)\Phi(\mathbf{X})\Phi(\mathbf{X}^t)\Phi(\mathbf{X})\alpha_1 = \alpha_1^t K_1^2 \alpha_1 = 1 \\ & \alpha_2^t \Phi(\mathbf{X}'^t)\Phi(\mathbf{X}')\Phi(\mathbf{X}'^t)\Phi(\mathbf{X}')\alpha_2 = \alpha_2^t K_2^2 \alpha_2 = 1 \end{array}$$

Kernel Canonical Correlation Analysis

- The corresponding Lagrangian

$$L(\lambda, \alpha_1, \alpha_2) = \alpha_1^t K_1 K_2 \alpha_2 - \frac{\lambda_1}{2}(\alpha_1^t K_1^2 \alpha_1 - 1) - \frac{\lambda_2}{2}(\alpha_2^t K_2^2 \alpha_2 - 1)$$

- Taking derivatives w.r.t. α_1 and α_2, one obtains

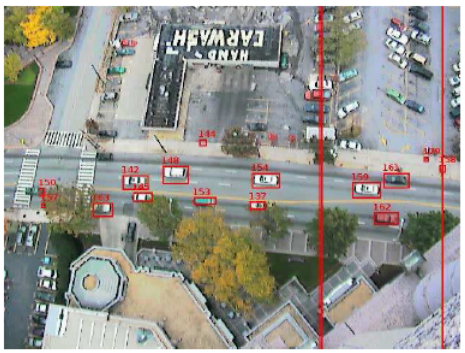
$$\frac{\partial L}{\partial \alpha_1} = K_1 K_2 \alpha_2 - \lambda_1 K_1^2 \alpha_1 = 0$$

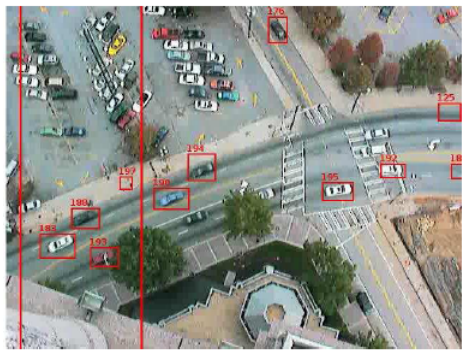
$$\frac{\partial L}{\partial \alpha_2} = K_2 K_1 \alpha_1 - \lambda_2 K_2^2 \alpha_2 = 0$$

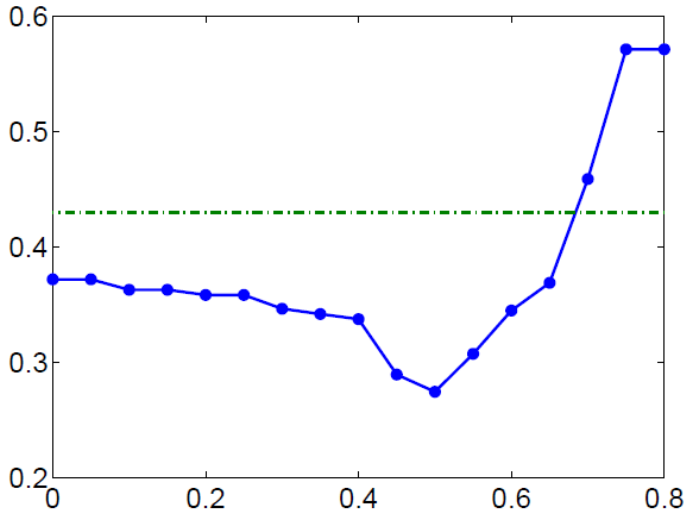
$$\alpha_2 = \frac{K_2^{-1} K_1 \alpha_1}{\lambda}$$

$$I\alpha_1 = \lambda^2 \alpha_1$$

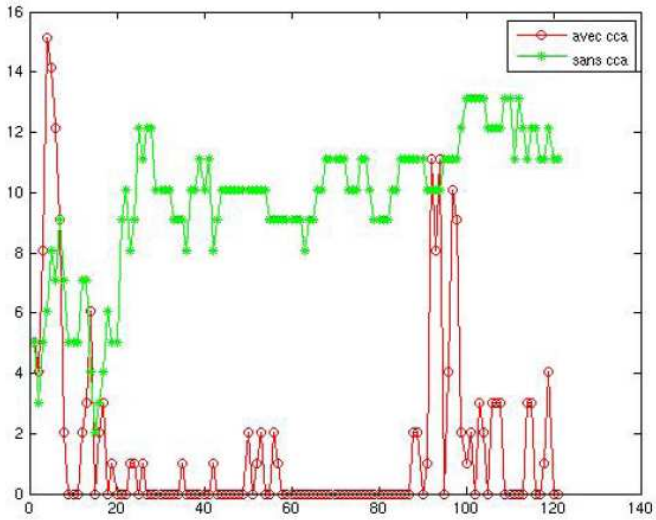
Examples : multi-view recognition



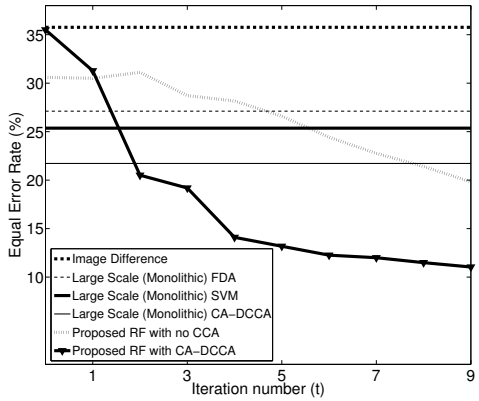




Examples : multi-view recognition



Examples : remote sensing change detection



General Conclusion & Good Practices

- Use shallow models (kernel methods) when you have few data in high dimensional spaces.
- Representation learning converts into a similarity (kernel) learning.
- Convexity is an asset but not a necessity (the game is more open with non-convex problems but finding "good" solution is another story)
- Use deep models when data (actually ground truth) is not an issue.
- Is the future of ML semi-parametric ?

Some References

- V. Vapnik, Statistical Learning Theory, 1998.
- C. Bishop, Pattern Recognition and machine learning, 2006.
- J. Shawe-Taylor and N. Cristianini, Kernel Methods for Pattern Analysis- Cambridge University Press, 2004.
- B. Scholkopf, K. Tsuda, J-P. Vert. Kernel Methods in Computational Biology, 2004.
- S. Boyd and L. Vandenberghe. Convex optimization. Cambridge university press, 2004.
- J.C. Spall. Introduction to stochastic search and optimization : estimation, simulation, and control (Vol. 65). John Wiley & Sons. 2005.
- I. Goodfellow, Y. Bengio and A. Courville. Deep learning (Vol. 1). Cambridge : MIT press, 2016.
- M. Jiu and H. Sahbi. Nonlinear Deep Kernel Learning for Image Annotation. IEEE Trans. Image Processing 26(4) : 1820-1832, 2017.