Patterns in vector spaces

Elisa Ricci
Fondazione Bruno Kessler - FBK (IT)
Overview

- **Pattern analysis**: automatic detection of patterns in data.

- **Pattern**: any relation, regularity and structure present in the data.

- **Learning**: we expect to detect significant patterns in the data and to transfer these informations making predictions about new data coming from the same source.

- **Statistical pattern recognition**: data are represented as vectors (statistical assumption about their distribution).

- Some patterns…
• Can we find a face (pattern) in the image?

• We can learn to discriminate faces from the background.

• Classification task
Overview

• Binary classification:
  • find a function that split up the dataset.

Fisher’s Discriminant Analysis (FDA), Support Vector Machines (SVM)
Face recognition: a template matching problem.

Images are represented as vectors. Problems arise due to the high-dimensional space.

Every face can be reconstructed by taking a weighted summation of some (FEW!!!) basis “faces”.

S. Trivedi. *Face Recognition using Eigenfaces and Distance Classifiers: A Tutorial*
• Principal component analysis (PCA)

• Dimensionality reduction
  • find a low dimensional representation of your data.
Sometimes data are presented in multiple modalities (e.g. images with captions)

President George W. Bush makes a statement in the Rose Garden while Secretary of Defense Donald Rumsfeld looks on. July 23, 2003. Rumsfeld said the United States would release graphic photographs of the dead sons of Saddam Hussein to prove they were killed by American troops. Photo by Larry Downing/Reuters

British director Sam Mendes and his partner actress Kate Winslet arrive at the London premiere of 'The Road to Perdition', September 18, 2002. The film stars Tom Hanks as a Chicago hit man who has a separate family life and co-stars Paul Newman and Jude Law. REUTERS/Dan Chung

World number one Lleyton Hewitt of Australia hits a return to Nicolas Massu of Chile at the Japan Open tennis championships in Tokyo October 3, 2002. REUTERS/Eriko Sugiura

German supermodel Claudia Schiffer gave birth to a baby boy by Caesarean section January 30, 2003, her spokeswoman said. The baby is the first child for both Schiffer, 32, and her husband, British film producer Matthew Vaughn, who was at her side for the birth. Schiffer is seen on the German television show 'Bet It...?!' ('Wetten Dass...?!') in Braunschweig, on January 26, 2002. (Alexandra Winkler/Reuters)

Is there a latent aspect that relates the two modalities?
• Find relations between two paired datasets

Canonical Correlation Analysis (CCA), Partial Least Squares (PLS)
Overview

• Regression
  • find a function that interpolates between the datapoints.

Least Squares Regression (LSR), Ridge regression (RR)
Overview

• A bunch of linear techniques for vectorial data for various tasks

<table>
<thead>
<tr>
<th>Classification</th>
<th>Dimensionality Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>FDA, SVM</td>
<td>PCA</td>
</tr>
<tr>
<td>Regression</td>
<td>Finding relationships between datasets</td>
</tr>
<tr>
<td>LSR, RR</td>
<td>CCA, PLS</td>
</tr>
</tbody>
</table>

• Linear techniques are powerful
  • are intuitive (often solution can be visualized geometrically)
  • often work well (most natural functions are smooth)
  • are fast and easy to solve (typically involve only matrix operations)
• What if we need non linear functions?
  • *Kernels*: data are implicitly mapped in *some* feature space were linear techniques are used.

• What if we have to cope with structured data?
  • Many approaches can be extended naturally to non vectorial data (e.g. sequences, graphs, images)
• PART I: Linear techniques for pattern analysis

  • Least squares problems
    • Linear regression (LSR/RR)
    • Fisher’s discriminant analysis (FDA)

  • Eigenvalue problems
    • Principal component analysis (PCA)
    • Partial least squares (PLS)
    • Canonical correlation analysis (CCA)

  • Convex optimization
    • Support vector machines (SVM)
Outline

• PART II: Kernel methods
  • Support vector machines
  • Kernel ridge regression (KRR)
  • Kernel principal component analysis (KPCA)
  • Kernel canonical correlation analysis (KCCA)
  • Learning in structured output spaces

• Applications in computer vision
Linear techniques for pattern analysis
Least Squares Problems
Linear Regression

Notations:

- Datapoints \( X = \{x_1, x_2, \ldots, x_n\} \) \( x_i \in \mathbb{R}^d \)
- Labels \( y = \{y_1, y_2, \ldots, y_n\} \) \( y \in \mathbb{R} \)
- Linear decision function \( f(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R} \)
  \[
  f(x) = w^T x
  \]
- Parameter vector \( w \)

Least Squares Problems
Linear Regression

- Goal: find a linear function $Xw$ that approximates the labels $y$.

- Assume the model:
  \[
  y = Xw + n
  \]
  with gaussian noise $n$ and $E\{n^Tn\}=I$ and $E\{n\}=0$.

- Maximum likelihood estimator for $w$:
  \[
  \min_w \|y - Xw\|^2
  \]

- For a new test point $x$ the label $y$ can be estimated as $w^Tx$. 
Linear Regression

• Goal: find a linear function $Xw$ that approximates the labels $y$.
• For a new test point $x$ the label $y$ can be estimated as $w^T x$.

$$E = \sum_{i=1}^{n} (y_i - w^T x_i)^2 = \|y - Xw\|^2$$
• Probabilistic interpretation

\[ y_i \sim N(\mathbf{w}^T \mathbf{x}_i, \sigma^2) \quad \text{with} \quad \sigma^2 = 1 \]

Sum Squared Error

\[ E = \sum_{i=1}^{n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 = \|\mathbf{y} - \mathbf{Xw}\|^2 \]

Likelihood

\[ L = \prod_{i=1}^{n} \exp\left(-\frac{1}{2\sigma^2} (y_i - \mathbf{w}^T \mathbf{x}_i)^2\right) \]

\[ = \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2\right) \]
Least Squares Regression

- Solve:

\[
\min_w \|y - Xw\|^2 \quad \Rightarrow
\]

\[
\min_w w^T X^T Xw + y^T y - 2w^T X^T y
\]

(gradient equal to 0):

\[
X^T Xw - 2X^T y = 0 \quad \Rightarrow \quad w = 2(X^T X)^{-1} X^T y
\]

- This is Least Squares Regression.
- In alternative we can solve the linear system:

\[
Aw = b \quad \Rightarrow \quad A = X^T X, \quad b = 2X^T y
\]
• LSR tend to *overfit* for noisy and high dimensional data.

• *Overfitting*: we are doing well on training data but poorly on test data.

• *Generalization*: ability to achieve small error (loss) on test data.

• *Learning*: search in an hypothesis space (e.g. linear functions)

• *Learning theory*: If our hypothesis space is not too complex/flexible (has a *low capacity*) and if our training set is large enough then we probably do well on similar test data.
LSR tend to overfit for noisy and high dimensional data.
Solution: minimize the loss and at the same time restrict the capacity!

\[
\min_w \| y - Xw \|^2 \\
\text{s.t.} \quad C(w) \leq C_{MAX}
\]

Solve:

\[
\min_w \| y - Xw \|^2 \\
\text{s.t.} \quad \| w \|^2 \leq 1
\]

This is known as Regularized LSR or Ridge Regression.
Ridge Regression

- Solve:

\[
\min_w \|y - Xw\|^2
\]

s.t. \( \|w\|^2 \leq 1 \)

- Lagrangian:

\[
L = w^T X^T Xw + y^T y - 2w^T X^T y + \lambda(w^T w - 1)
\]

- Solve (gradient equal to 0):

\[
w = 2(X^T X + \lambda I)^{-1} X^T y
\]
Ridge Regression

- Solve:

\[
\min_w \|y - Xw\|^2 + \lambda \|w\|^2
\]

is equivalent to maximize:

\[
P(w \mid X) = \exp\left( -\frac{\lambda}{2\sigma^2} \|w\|^2 \right) \prod_{i=1}^n \exp\left( -\frac{1}{2\sigma^2} \left( y_i - w^T x_i \right)^2 \right)
\]

\[
= \exp\left( -\frac{1}{2\sigma^2} \left( \lambda \|w\|^2 + \sum_{i=1}^n \left( y_i - w^T x_i \right)^2 \right) \right)
\]

since we assume \( w \sim N\left( 0, \frac{\sigma^2}{\lambda} \right) \) and:

\[
P(w \mid X) = \frac{P(w, X)}{P(X)} \propto P(w, X)
\]
Notations:

• Datapoints \( X = \{x_1, x_2, \ldots, x_n\} \quad x_i \in R^d \)

• Labels \( y = \{y_1, y_2, \ldots, y_n\} \quad y_i \in \{-1,1\} \)

• Linear decision function \( f(\cdot): R^d \rightarrow R \)

\[
f(x) = w^T x + b = \hat{w}^T \hat{x}
\]

• Parameters \( \hat{w} = [w \ b] \) and data \( \hat{x} = [x \ 1] \).
Fisher’s Discriminant Analysis

• FDA or Linear Discriminant Analysis (LDA) solve the same problem of RR but with \( y_i \in \{-1, 1\} \):

\[
\min_w \|y - Xw\|^2 \\
\text{s.t.} \quad \|w\|^2 \leq 1
\]

• Solution:

\[
w = 2\left(X^TX + \lambda I\right)^{-1}X^Ty
\]

• The optimization problem above correspond to maximize a function that represents the difference between the means of the two classes, normalized by a measure of the within-class scatter.
Fisher’s Discriminant Analysis

- Solve:

\[ \max J = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \quad \Rightarrow \quad \max_w \frac{w^T (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T w}{w^T (C_1 + C_2)w} = \frac{w^T S_b w}{w^T S_w w} \]

where

\[ \mu_k = \frac{1}{n} \sum_{i=1}^{n} x_i \quad C_k = \frac{1}{n} \sum_{i=1}^{n} (x_i^k - \mu_k)(x_i^k - \mu_k)^T \]

- Regularized problem:

\[ \min_w \frac{w^T S_b w}{w^T (S_w + \lambda I)w} \]

- Solution:

\[ w = (S_w + \lambda I)^{-1} S_B \alpha 2(X^T X + \lambda I)^{-1} X^T y \]

- Later: another interpretation of FDA.
Eigenvalue Problems
**Eigenvalue problems**

- Eigenvalue problems
  - Principal component analysis (PCA)

- Relations between two paired data sets.
  - Principal component analysis on paired datasets (PCA)
  - Partial least squares (PLS)
  - Canonical correlation analysis (CCA)

- Regularization

- Multi-way: relations between more than two data sources.
Eigenvalue problems

• General form of an eigenvalue problem

\[ Av = \lambda Bv \]

• Properties
  • For symmetric \( A \) and \( B \) the eigenvalues \( \lambda_i \) and eigenvectors \( v_i \) are real.
  • Different eigenvectors are ortonormal in the \( B \)-metric:

\[
\begin{align*}
v_i^T B v_j &= 0 \quad \text{for} \quad \lambda_i \neq \lambda_j \\
v_j^T (A v_i) &= v_i^T (A v_j) \\
\lambda_i (v_j^T B v_i) &= \lambda_j (v_i^T B v_j)
\end{align*}
\]

• Fast to compute: iterative methods
• Goal: find directions of large variance in the data $\mathbf{X}$.

• Solve:

$$\max_w \quad w^T \mathbf{X}^T \mathbf{X} w$$

where the variance of the projection of $\mathbf{X}$ on the vector $w$ is maximized.

• We can again impose a capacity control:

$$\max_w \quad w^T \mathbf{X}^T \mathbf{X} w$$

s.t. $\|w\|^2 = 1$
Principal Component Analysis

• Solve:

\[
\max_w \quad w^T X^T X w
\]

s.t. \[\|w\|^2 = 1\]

• Lagrangian:

\[L = w^T X^T X w - \lambda(w^T w - 1)\]

• Solve by the eigenvalue problem:

\[X^T X w = \lambda w\]
Two equivalent problems:

\[
\begin{align*}
\max_w \quad & w^T X^T X w \\
\text{s.t.} \quad & \|w\|^2 \leq 1
\end{align*}
\]

\[\Longleftrightarrow\]

\[
\begin{align*}
\max_w \quad & \frac{w^T X^T X w}{w^T w}
\end{align*}
\]

Empirical covariance matrix \( C = X^T X \)

- \( C \) positive (semi-)definite symmetric.

- Solutions: Eigenvectors of \( C \). Since \( C \) is real and symmetric, the eigenvectors are real and orthogonal.
• Dimensionality reduction: PCA approximates the data by projecting them in a low dimensional sub-space \((k<<d)\).

• Why dimensionality reduction?
  • Visualization: projection of high-dimensional data into 2D or 3D.
  • Data compression: efficient storage and retrieval.
  • Denoising

• Dimensionality reduction implies information loss (not always) but PCA preserves as much information as possible.

• The new low-dimensional space is centered at the sample mean and has directions determined by the eigenvectors of the covariance matrix \(\mathbf{C}\) corresponding to the largest eigenvalues (principal components).

\[
x_{R} = \mathbf{V}^{T}\mathbf{x}
\]

\(\mathbf{V}\) containing the first \(k\) eigenvectors of \(\mathbf{C}\)
Principal Component Analysis

- How many principal components to keep?
- Study the eigenvalue spectrum (explained variance)
- To choose $k$:

$$\sum_{i=1}^{k} \lambda_i / \sum_{i=1}^{d} \lambda_i > T \quad (\text{e.g. } T = 0.9)$$
• PCA is not always an optimal dimensionality reduction procedure for classification purposes.

• FDA (or LDA): find directions along which the classes are best separated.
  • LDA can be used for dimensionality reduction ($x_R = V^Tx$. $V$ containing the first $k$ eigenvectors of $S_W^{-1}S_B$)
• PCA versus LDA

• Main results:
  • When the training set is small, PCA can outperform LDA.
  • When the number of samples is large and representative for each class, LDA outperforms PCA.
What if we have two datasets?

Goal: find related directions on two paired datasets $X$ and $Y$.

Applications:
- Cross language retrieval of documents – English and French text
- Image retrieval – Image features with text captions
- Speaker Recognition – Audio and Lip movement
What if we have two datasets?

Goal: find related directions on two paired datasets $\mathbf{X}$ and $\mathbf{Y}$.

Series of eigenvalue problems:

- PCA: maximize combined variance on two datasets.
- PLS: maximize covariance.
- CCA: maximize correlation.
PCA on coupled data

- Solve the max variance problem on the data \((X \ Y)\):

\[
\begin{align*}
\max_{w_X, w_Y} & \quad \left( \begin{array}{c} w_X \\ w_Y \end{array} \right)^T \left( \begin{array}{cc} X^T X & X^T Y \\ Y^T X & Y^T Y \end{array} \right) \left( \begin{array}{c} w_X \\ w_Y \end{array} \right) \\
\text{s.t.} & \quad \left( \begin{array}{c} w_X \\ w_Y \end{array} \right)^T \left( \begin{array}{c} w_X \\ w_Y \end{array} \right) = 1
\end{align*}
\]

where:

- \(X^T X\) covariance matrix
- \(X^T Y\) cross-covariance matrix
- \(w_X^T X^T X w_X\) variance along \(w_X\)
- \(w_Y^T Y^T X w_X\) covariance along \(w_X\) and \(w_Y\)
PCA on coupled data

- Solve the max variance problem on the data \((X \ Y)\):

\[
\max_{w_X, w_Y} \left( w_X \right)^T \begin{pmatrix}
X^T X & X^T Y \\
Y^T X & Y^T Y
\end{pmatrix} \left( w_X \right) \\
\text{s.t.} \quad \begin{pmatrix}
w_X \\
w_Y
\end{pmatrix}^T \begin{pmatrix}
w_X \\
w_Y
\end{pmatrix} = 1
\]

- Solve the eigenvalue problem:

\[
\begin{pmatrix}
X^T X & X^T Y \\
Y^T X & Y^T Y
\end{pmatrix} \begin{pmatrix}
w_X \\
w_Y
\end{pmatrix} = \lambda \begin{pmatrix}
w_X \\
w_Y
\end{pmatrix}
\]

- Result is essentially independent on how the data have been split up.
Partial least squares

- Maximize the covariance between projections

\[
\max_{w_X, w_Y} w_X^T X^T Y w_Y
\]

subject to

\[
w_X^T w_X = 1
\]

\[
w_Y^T w_Y = 1
\]

- It is equivalent to the following (we fixed the scale):

\[
\max_{w_X, w_Y} \frac{w_X^T X^T Y w_Y}{\sqrt{w_X^T w_X w_Y^T w_Y}}
\]
Partial least squares

• It is equivalent to the following:

\[
\max_{w_x, w_y} w_x^T X^T Y w_y
\]

s.t. \[ w_x^T w_x + w_y^T w_y = 1 \]

• Lagrangian:

\[
L = w_x^T X^T Y w_y - \frac{1}{2} \lambda (w_x^T w_x + w_y^T w_y - 1)
\]

• Solve the eigenvalue problem (gradient equal to zero):

\[
\begin{pmatrix}
0 & X^T Y \\
Y^T X & 0
\end{pmatrix}
\begin{pmatrix}
w_x \\
w_y
\end{pmatrix}
= \lambda
\begin{pmatrix}
w_x \\
w_y
\end{pmatrix}
\]
Canonical Correlation Analysis

• Maximize the correlation (assuming the variance irrelevant)

\[
\max_{w_X, w_Y} w_X^T X^T Y w_Y
\]

\[\text{s.t. } w_X^T X X w_X + w_Y^T Y Y w_Y = 1\]

• It is equivalent to the following (again we fixed the scale):

\[
\max_{w_X, w_Y} \frac{w_X X^T Y w_Y}{\sqrt{w_X^T X X w_X w_Y^T Y Y w_Y}}
\]
Canonical Correlation Analysis

- Maximize the correlation (assuming the variance irrelevant)

\[
\max_{w_x,w_y} \quad w_x^T X^T Y w_y
\]

s.t. \( w_x^T X^T X w_x + w_y^T Y^T Y w_y = 1 \)

- Lagragian:

\[
L = w_x^T X^T Y w_y - \frac{1}{2} \lambda (w_x^T X^T X w_x + w_y^T Y^T Y w_y - 1)
\]

- Solve the eigenvalue problem (gradient to 0):

\[
\begin{pmatrix}
0 & X^T Y \\
Y^T X & 0
\end{pmatrix}
\begin{pmatrix}
w_x \\
w_y
\end{pmatrix} = \lambda
\begin{pmatrix}
X^T X & 0 \\
0 & Y^T Y
\end{pmatrix}
\begin{pmatrix}
w_x \\
w_Y
\end{pmatrix}
\]
Summarizing...

- General eigenvalue problem
  \[ \mathbf{A} \mathbf{w} = \lambda \mathbf{B} \mathbf{w} \]

- PCA
  \[
  \begin{pmatrix}
  \mathbf{X}^T \mathbf{X} & \mathbf{X}^T \mathbf{Y} \\
  \mathbf{Y}^T \mathbf{X} & \mathbf{Y}^T \mathbf{Y}
  \end{pmatrix}
  \begin{pmatrix}
  \mathbf{w}_X \\
  \mathbf{w}_Y
  \end{pmatrix}
  = \lambda
  \begin{pmatrix}
  \mathbf{w}_X \\
  \mathbf{w}_Y
  \end{pmatrix}
  \]

- PLS
  \[
  \begin{pmatrix}
  0 & \mathbf{X}^T \mathbf{Y} \\
  \mathbf{Y}^T \mathbf{X} & 0
  \end{pmatrix}
  \begin{pmatrix}
  \mathbf{w}_X \\
  \mathbf{w}_Y
  \end{pmatrix}
  = \lambda
  \begin{pmatrix}
  \mathbf{w}_X \\
  \mathbf{w}_Y
  \end{pmatrix}
  \]

- CCA
  \[
  \begin{pmatrix}
  0 & \mathbf{X}^T \mathbf{Y} \\
  \mathbf{Y}^T \mathbf{X} & 0
  \end{pmatrix}
  \begin{pmatrix}
  \mathbf{w}_X \\
  \mathbf{w}_Y
  \end{pmatrix}
  = \lambda
  \begin{pmatrix}
  \mathbf{X}^T \mathbf{X} & 0 \\
  0 & \mathbf{Y}^T \mathbf{Y}
  \end{pmatrix}
  \begin{pmatrix}
  \mathbf{w}_X \\
  \mathbf{w}_Y
  \end{pmatrix}
  \]
PCA and PLS tend to be robust against small variance noise since they look at \textit{large} variance directions.

CCA only looks at correlation. Variance can be an irrelevant feature of the data.

Regularization helps to make CCA robust against noise.

Try to obtain solutions $w_X$ and $w_Y$ of small norm.
Regularized CCA

• Original problem

$$\max_{w_x, w_y} \quad w_x^T X^T Y w_y$$

s.t. $$w_x^T X^T X w_x + w_y^T Y^T Y w_y = 1$$

• Regularized problem

$$\max_{w_x, w_y} \quad w_x^T X^T Y w_y - \frac{1}{2} C (w_x^T w_x + w_y^T w_y)$$

s.t. $$w_x^T X^T X w_x + w_y^T Y^T Y w_y = 1$$
Regularized CCA

- Regularized problem

\[
\max_{w_x, w_y} \quad w_x^T X^T Y w_y - \frac{1}{2} \lambda (w_x^T w_x + w_y^T w_y) \\
\text{s.t.} \quad w_x^T X^T X w_x + w_y^T Y^T Y w_y = 1
\]

(again computing the Lagrangian and equating the gradient to 0)

- Solve the following eigenvalue problem:

\[
\begin{pmatrix}
-CI & X^T Y \\
Y^T X & -CI
\end{pmatrix}
\begin{pmatrix}
w_x \\
w_y
\end{pmatrix} = \lambda
\begin{pmatrix}
X^T X & 0 \\
0 & Y^T Y
\end{pmatrix}
\begin{pmatrix}
w_x \\
w_y
\end{pmatrix}
\]
R-CCA and FDA

- Regularized problem

\[
\begin{pmatrix}
-CI & X^T Y \\
Y^T X & -CI
\end{pmatrix}
\begin{pmatrix}
w_x \\
w_y
\end{pmatrix} = \lambda
\begin{pmatrix}
X^T X & 0 \\
0 & Y^T Y
\end{pmatrix}
\begin{pmatrix}
w_x \\
w_y
\end{pmatrix}
\]

- If \( Y = y \) (\( y \) is of one dimension):

\[
\begin{pmatrix}
-CI & X^T y \\
y^T X & -C
\end{pmatrix}
\begin{pmatrix}
w_x \\
w_y
\end{pmatrix} = \lambda
\begin{pmatrix}
X^T X & 0 \\
0 & y^T y
\end{pmatrix}
\begin{pmatrix}
w_x \\
w_y
\end{pmatrix}
\]

- Solution ("same" that FDA solution):

\[
w_x = (\lambda X^T X + CI)^{-1} X^T y
\]

- FDA is a special case of R-CCA (also RR).
- CCA is a form of multivariate regression.
Relations between multiple sources

- Goal: Identifying common factors between more than two data sources.

- Multiway extension of PLS, PCA, CCA.

- We have $k$ datasets $X_i$ and we want to compute the associated $w_i$.

![Diagram showing relations between datasets](image)
Multiway CCA

• Solve:

$$\max_{w_i} \left\| \sum_i X_i w_i \right\|^2$$

s.t. $$\sum_i \left\| X_i w_i \right\|^2 = 1$$

• It is equivalent to:

$$\max_{w_i} \sum_{i \neq j} w_i^T X_i^T X_j w_j$$

s.t. $$\sum_i w_i^T X_i^T X_i w_i = 1$$

• Since:

$$\left\| \sum_i X_i w_i \right\|^2 = \sum_{i \neq j} w_i^T X_i^T X_j w_j + \sum_i w_i^T X_i^T X_i w_i$$
Multiway CCA

- Solve the following eigenvalue problem:
  \[
  \begin{pmatrix}
  0 & X_1^T X_2 & \cdots & X_1^T X_N \\
  X_2^T X_1 & 0 & \cdots & X_2^T X_N \\
  \vdots & \vdots & \ddots & \vdots \\
  X_N^T X_1 & X_N^T X_2 & \cdots & 0
  \end{pmatrix}
  \begin{pmatrix}
  w_1 \\
  w_2 \\
  \vdots \\
  w_N
  \end{pmatrix}
  = \lambda
  \begin{pmatrix}
  X_1^T X_1 & 0 & \cdots & 0 \\
  0 & X_2^T X_2 & \cdots & 0 \\
  \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & \cdots & X_N^T X_N
  \end{pmatrix}
  \begin{pmatrix}
  w_1 \\
  w_2 \\
  \vdots \\
  w_N
  \end{pmatrix}
  \]

- Admit a regularized form too.
  \[
  \begin{pmatrix}
  -C I & X_1^T X_2 & \cdots & X_1^T X_N \\
  X_2^T X_1 & -C I & \cdots & X_2^T X_N \\
  \vdots & \vdots & \ddots & \vdots \\
  X_N^T X_1 & X_N^T X_2 & \cdots & -C I
  \end{pmatrix}
  \begin{pmatrix}
  w_1 \\
  w_2 \\
  \vdots \\
  w_N
  \end{pmatrix}
  = \lambda
  \begin{pmatrix}
  X_1^T X_1 & 0 & \cdots & 0 \\
  0 & X_2^T X_2 & \cdots & 0 \\
  \vdots & \vdots & \ddots & \vdots \\
  0 & 0 & \cdots & X_N^T X_N
  \end{pmatrix}
  \begin{pmatrix}
  w_1 \\
  w_2 \\
  \vdots \\
  w_N
  \end{pmatrix}
  \]
Many important problems in pattern analysis
  - dimensionality reduction
  - finding relations between two representations of the data
  - classification (FDA)

They can be reduced to eigenvalue problems
  - They can be studied with simple linear algebra
  - They can be solved or approximated efficiently with standard techniques

All these problems require computing only the inner product between datapoints.
Convex Optimization
**Classification**

- **Binary classification:**
  
  \[
  \mathbf{w}^T \mathbf{x} + b > 0
  \]
  
  \[
  \mathbf{w}^T \mathbf{x} + b = 0
  \]
  
  \[
  \mathbf{w}^T \mathbf{x} + b < 0
  \]

- **Which is the optimal separating hyperplane?**

  \[
  f(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} + b)
  \]
We discussed Fisher’s Discriminant Analysis
The best hyperplane is the one obtained maximizing the difference between the means of the two classes normalized by a measure of the within-class scatter.

Solve:

\[
\min_w \| y - Xw \|^2 \\
\text{s.t. } \|w\|^2 \leq 1
\]

Solution:

\[
w = 2(X^TX + \lambda I)^{-1}X^Ty
\]
Support Vector Machines

- SVM: the best hyperplane is the one with maximum margin $\rho$.
- Intuitive idea: maximize the “stability”
  - use $w$ such that we can maximally perturb the input samples without introducing misclassifications.
- Examples closest to the hyperplane are support vectors.
- We can rescale $w$, $b$ such that for support vectors

$$d = \frac{y_i (w^T x_i + b)}{||w||} = \frac{1}{||w||}$$

- Then the margin is.

$$\rho = 2d = 2 \frac{y_i (w^T x_i + b)}{||w||} = \frac{2}{||w||}$$
Support Vector Machines

- Given a training set:
  \[ T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \quad x_i \in \mathbb{R}^d, \quad y_i \in \{-1,1\} \]

- Goal:
  - Correctly classify all training data
    \[ y_i (w^T x_i + b) \geq 1 \quad \forall i \]
  - Maximize the Margin
    \[ \rho = \frac{2}{\|w\|} \]

- This is equivalent to the following
  \[
  \min_{w,b} \quad \frac{1}{2} \|w\|^2 \\
  \text{s.t.} \quad y_i (w^T x_i + b) \geq 1 \quad \forall i
  \]

- This is a Quadratic Programming (QP) problem.
Support Vector Machines

From an optimization perspective:

\[
\begin{align*}
\min_{w,b} & \quad \frac{1}{2}\|w\|^2 \\
\text{s.t.} & \quad y_i (w^T x_i + b) \geq 1 \quad \forall i
\end{align*}
\]

is a rather easy problem.

- The objective function is differentiable and convex.
- The constraints are all linear.
- We can find the global optimal \( w \).
- In principle the solution can be computed by standard QP solvers.
Support Vector Machines

- The solution is typically computed constructing the *dual problem.*

\[
\max_{\alpha} \quad \sum_{i} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j \\
\text{s.t.} \quad \sum_{i} \alpha_i y_i = 0 \quad \alpha_i \geq 0, \quad \forall i
\]

- This is also a QP problem.
- Each \(\alpha_i\) (Lagrange multiplier) is associated with a constraint in the primal problem.
- Each data sample corresponding to a non zero \(\alpha_i\) is a support vector.
- *Sparsity:* usually we have only few support vectors!
- *Note:* solving the dual problem involves computing only the inner products \(x_i^T x_j\) between all training points.
Given the solution $\alpha$ of the dual problem, the solution of the primal is:

\[ w = \sum_{i=1}^{N_s} \alpha_i y_i x_i \]

\[ b = \frac{1}{N_s} \sum_{k=1}^{N_s} \left( y_k - \sum_{i=1}^{N_s} \alpha_i y_i x_i^T x_k \right) \]

The solution is computed only with non-zero $\alpha_i$ i.e. only with data points $x_i$ corresponding to a support vector.

The classifying function is:

\[ f(x) = \sum_i \alpha_i y_i x_i^T x + b \]

**Note:** It relies only on the *inner product* between the test point $x$ and the support vectors $x_i$. 

**Support Vector Machines**

Convex optimization
• What if the training set is not linearly separable?

• *Slack variables* $\xi_i$ can be added to allow misclassification of difficult or noisy examples.

• This is usually referred as *soft-margin SVM*. 
Support Vector Machines

• Solve:

\[
\min_{w, b, \xi_i} \quad \frac{1}{2} \|w\|^2 + C \sum_i \xi_i \\
\text{s.t.} \quad y_i (w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \forall i
\]

• The problem is still convex and efficiently solvable.

• \(C\) is a \textit{regularization} parameter. It controls overfitting.
• It allows a trade-off between maximizing the margin and fitting the training data.
• A large \(C\) corresponds to assign a higher penalty to the errors.
SVM in practice

• Solve (dual):

\[
\max_\alpha \sum \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j
\]

\[
s.t. \quad \sum \alpha_i y_i = 0 \quad 0 \leq \alpha_i \leq C, \quad \forall i
\]

• We could solve the dual QP with standard QP solvers
  • QP packages (MINOS, LOQO, MATLAB toolbox etc.)
• In practice faster *ad hoc* solvers have been developed
  • Decomposition methods: decompose the large scale QP problem into a series of smaller QP problems [Joachims '99] (e.g. SVM\textsuperscript{light} http://svmlight.joachims.org/)
  • Sequential Minimal Optimization (SMO) [Platt '99] algorithm converts the QP problem to an analytical one by analyzing two points at a time.
SVM in practice

- A widely used package:
  - LibSVM (http://www.csie.ntu.edu.tw/~cjlin/libsvm/) with a nice tool for model selection

- Recently very fast solvers have been proposed
  - SVM-Perf (training in linear time) [Joachims, 2006] (http://svmlight.joachims.org/)
  - Pegasos (simple stochastic subgradient optimizer) (http://ttic.uchicago.edu/~shai) [Shalev-Schwartz, 2007]

Convex optimization
Multiclass SVM

- Given a training set:
  \[ T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \quad x_i \in \mathbb{R}^d, \quad y_i \in \{1, \ldots, k\} \]

- Several possible approaches:
  - One versus all:
    - We have \( k \) classes, we train \( k \) binary SVMs:
      - \( 1^{st} \) class vs. \( 2^{nd}, 3^{rd} \ldots k^{th} \) class
      - \( 2^{nd} \) class vs. \( 1^{st}, 3^{rd} \ldots k^{th} \) class
    - We get \( k \) decision functions:
      \[ f_m(x) = w_m^T x + b_m \]
    - Prediction can be made by:
      \[ \hat{m} = \arg\max_{m=1,\ldots,k} f_m(x) \]
Multiclass SVM

- **One versus one:**
  - We have $k$ classes, we train $k(k-1)/2$ binary SVMs, e.g. for $k=3$:
    1. $1^{st}$ class vs. $2^{nd}$
    2. $1^{st}$ class vs. $3^{rd}$
    3. $2^{nd}$ class vs. $3^{rd}$
  - Predictions is made by majority of voting

- **Other approaches:**
  - Many methods imply solving an optimization problem (e.g. [Weston and Watkins, 1999], [Crammer and Singer, 2001])
  - A comparison in [Hsu and Lin, 2002]
  - Accuracy similar for many problems (may be one vs. one is fastest for training)
Multiclass SVM

- [Crammer and Singer, 2001] Solve multiclass classification as a single optimization problem.

\[
\begin{align*}
\min_{\mathbf{w}_m, \xi_i} & \quad \frac{1}{2} \sum_m \|\mathbf{w}_m\|^2 + C \sum_i \xi_i \\
\text{s.t.} & \quad \mathbf{w}_{y_i}^T \mathbf{x}_i \geq \mathbf{w}_m^T \mathbf{x}_i + 1 - \xi_i, \quad \forall i, \quad \forall m \neq y_i, m \in \{1, \ldots, k\} \\
& \quad \xi_i \geq 0
\end{align*}
\]

- A slack variable for each training point.
- Prediction:

\[
\hat{m} = \arg \max_{m=1, \ldots, k} \mathbf{w}_m^T \mathbf{x}
\]
Support Vector Machines

- Solve:
  \[
  \min_{w,b,\xi} \quad \frac{1}{2} \|w\|^2 + C \sum_i \xi_i \\
  \text{s.t.} \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \forall i
  \]

- The problem is still convex and efficiently solvable.

- \( C \) is a *regularization* parameter. It controls overfitting.
- It allows a trade-off between maximizing the margin and fitting the training data.
- A large \( C \) corresponds to assign a higher penalty to the errors.
Loss functions

• General rule to design classifiers:
  • Choose an appropriate loss function (error on training data) and a regularization term:

\[
\min_w \sum_{i=1}^n L_w(x_i, y_i) + \frac{1}{C} R(w)
\]

• Efficiently solvable as long as \( L_w \) is convex in \( w \).
  • square loss (FDA)

\[
L_w(x, y) = (y - x^T w)^2 = (1 - yx^T w)^2
\]

• hinge loss (soft- margin SVM)

\[
L_w(x, y) = \max(0, 1 - yx^T w)
\]
Loss functions

- Many possible loss functions:

\[ L(z) \]

- **square loss** \((1 - z)^2\)
- **Adaboost loss** \(e^{-z}\)
- **hinge loss**, \(\max(0, 1-z)\)
- **logistic loss**, \(\log(1+e^{-z})\)

\[ 0/1 \text{ loss} = \text{ideal loss} !!! \]
Support Vector Machines

- Soft margin SVMs can successfully handle noisy data.
- What about very hard problems with non separable data?
- Idea: mapping data to some higher-dimensional feature space where the training set is separable!!!
Support Vector Machines

• Solve:

\[
\min_{w,b,\xi} \frac{1}{2} \| w \|^2 + C \sum_i \xi_i \\
\text{s.t. } y_i (w^T \phi(x_i) + b) \geq 1 - \xi_i \quad \xi_i \geq 0 \quad \forall i
\]

• Explicit mapping is expensive since \( \phi(x) \) is usually high dimensional
• Solving the problem without explicitly mapping the data is desirable
• Solution: exploit dual problem with

\[
\max_{\alpha} \quad \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\
\text{s.t. } \sum_i \alpha_i y_i = 0 \quad 0 \leq \alpha_i \leq C, \forall i
\]

\[k(x_i, x_j) = \phi(x_i)^T \phi(x_j)\]

**Kernel !!!**
• Kernels are not the only possible way to approach non-linearity
• Other possibility: Stacking (use classifier output as input to other (linear) classifiers)
• *Multilayer Perceptron* (ANN)
• *Boosting*: Combination of weak learners:
  • A weak learner is a "rough and moderately inaccurate" predictor.
• The decision function $f$ turns out to be non-linear.
Kernel methods
Support Vector Machines
Support Vector Machines

- Non linear SVM:

\[
\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j k(x_i, x_j)
\]

s.t. \( \sum_i \alpha_i y_i = 0 \quad 0 \leq \alpha_i \leq C, \forall i \)

- The kernel function implicitly maps the data into a high-dimensional space feature space

\[ k(x_i, x_j) = \phi(x_i)^T \phi(x_j) \]

- The decision function is

\[ f(x) = \sum_i \alpha_i y_i k(x_i, x) \]
Support Vector Machines

• Why the kernels?
  • It is often faster to evaluate the kernel than the feature function
    • Example (e.g. polynomial kernel)

\[
\begin{align*}
\text{feature function} & \quad \phi \left( \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right) = \left( 1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2 \right) \\
\text{kernel} & \quad k(x, y) = \phi(x)^T \phi(y) = (1 + x_1 y_1 + x_2 y_2)^2 
\end{align*}
\]

• There are kernel functions for which we know that a feature transformation exists, but we don’t know what it is.

• Which functions are admissible as kernels?
Kernels

- Mercer’s condition:
  *Every semi-positive definite symmetric function is a kernel.*

**Theorem:** If a kernel is positive semi-definite i.e.:

$$\sum_{i,j} k(x_i,x_j)c_ic_j \geq 0$$

where $c_i$ are real numbers, then there exists function $\phi(x)$ defining an inner product of possibly higher dimensions i.e.

$$k(x_i,x_j) = \phi(x_i)^T \phi(x_j)$$

- Semi-positive definite symmetric functions correspond to a semi-positive definite symmetric matrix.
- Every symmetric semi-positive definite matrix is a kernel.
• Simple kernels:

  - **Linear kernel** \( k(x_i, x_j) = x_i^T x_j \)

  - **Polynomial kernel** \( k(x_i, x_j) = (1 + x_i^T x_j)^m, \quad \forall x_i, x_j \in \mathbb{R}^d, m > 0 \)

  - **Gaussian or RBF kernel** \( k(x_i, x_j) = \exp \left( \frac{\|x_i - x_j\|^2}{2\sigma^2} \right), \quad \sigma > 0 \)

  - **Sigmoid with parameter \( \kappa \) and \( \theta \)**
    
    \( k(x_i, x_j) = \tanh (\kappa x_i^T x_j + \theta) \),

    *(It does not satisfy the Mercer condition on all \( \kappa \) and \( \theta \))*
• Kernels for other type of data
  • For $N$-bin histograms
    \[ k(h_i, h_j) = \sum_{k=1}^{N} \min(h_{i,k}, h_{j,k}) \]
    • Intersection kernel
    \[ k(h_i, h_j) = \exp\left(-\gamma \sum_{k=1}^{K} \frac{(h_{i,k} - h_{j,k})^2}{h_{i,k} + h_{j,k}}\right) \]
    • $\chi^2$ kernel
  • For comparing probability distributions
    \[ k(p_i, p_j) = \exp(-KL(p_i, p_j)) \]
  • For strings
    \[ k(s_i, s_j) = \exp(-D_E(s_i, s_j)) \]
Constructing Kernels

- We can construct kernel from arbitrary distance functions
  - \( d : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R} \) is a distance function i.e.
    - (non-negativity) \( d(x_i, x_j) \geq 0, \ \forall x_i, x_j \in \mathbb{R}^d \)
    - (identity of indiscernibles) \( d(x_i, x_j) = 0, \iff x_i = x_j \)
    - (symmetry) \( d(x_i, x_j) = d(x_j, x_i), \ \forall x_i, x_j \in \mathbb{R}^d \)
    - (triangle inequality) \( d(x_i, x_k) \leq d(x_i, x_j) + d(x_j, x_k), \ \forall x_i, x_j, x_k \in \mathbb{R}^d \)

then \( k(x_i, x_j) = \exp(-d(x_j, x_i)) \) is a kernel.
We can construct kernels from other kernels:

1) If $k$ is a kernel and $\alpha > 0$, then $\alpha k$ and $k + \alpha$ are kernels.

2) If $k_1, k_2$ are kernels, then $k_1 + k_2$ and $k_1 \cdot k_2$ are kernels.
Applications

- OCR/handwriting recognition (MNIST)
  - 60,000 training examples, 10000 test examples, images 28x28 pxls
  - Features: raw pixels
  - Images have been resized, normalized w.r.t. brightness/contrast

- Choosing a good mapping $\phi(x)$ improves the results
- Kernel provides an effective way to encode prior knowledge
- State of the art results: [DeCoste & Scholkopf, 2002]

<table>
<thead>
<tr>
<th>Kernel</th>
<th>Test error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear SVM</td>
<td>8.5%</td>
</tr>
<tr>
<td>Polynomial SVM</td>
<td>1%</td>
</tr>
<tr>
<td>Translation invariant SVM</td>
<td>0.56%</td>
</tr>
</tbody>
</table>
Applications

- **Face Detection**
  

- **Algorithm**
  - Consider the same image at several scales
  - 19x19pxls windows pattern
  - Preprocessing (light correction, histogram equalization)
  - Classify using SVM with polynomial kernel and 50,000 datapoints
Applications

- State of the art result in pedestrian detection
  - Features: local intensity gradient directions
  - linear kernel [Dalal, Triggs. CVPR 2005]
  - histogram intersection kernel [Maji, Berg, Malik. CVPR 2008]

\[
k(h_i, h_j) = \sum_{k=1}^{N} \min(h_{i,k}, h_{j,k})
\]

Maji et al. Classification using intersection kernel SVMs is efficient, 2008
Applications

- Object category recognition
  - extract local image descriptors, e.g. SIFT
  - calculate multi-level pyramid histograms $h_{l,k}$
  - pyramid match kernel
    [Grauman, Darrell. ICCV 2005]
  - Fast evaluation of kernel functions

$$k(h_i, h_j) = \sum_{l=1}^{L} 2^l \sum_{k=1}^{2^{l-1}} \min(h_{i,k}, h_{j,k})$$

Grauman, Darrel. The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features, 2005
Which kernel?

• In previous examples knowledge about the application domain give assistance in defining appropriate similarity measures

• If we have no prior knowledge which is the best kernel?
  • Gaussian or polynomial kernel is default
  • Very often more elaborate kernels are needed
  • What about the best kernel parameters?
    • e.g. width in Gaussian kernel
    • In the absence of reliable criteria, applications rely on the use of a validation set or cross-validation to set the parameters.
Which kernel?

- Model selection is important to achieve highest accuracy
  - Action Classification, KTH dataset

- Model selection makes the difference (same features, same RBF kernel, $C$ and $\sigma$ obtained by crossvalidation)

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Novozin et al. ICCV 2007</td>
<td>85.1%</td>
</tr>
<tr>
<td>Dollar et al, PETS 2005</td>
<td>80.6%</td>
</tr>
</tbody>
</table>

Nowozin et al. Discriminative Subsequence Mining for Action Classification, ICCV 2007
Learning the kernel

- Idea: what about learning the kernel matrix?
- [Lanckriet et al., 2002] Specify a convex cost function to learn the kernel matrix
  - Transductive setting: optimize the embedding for training data and learn it for unlabeled data.
  - For labeled data: find the embedding which shows maximal margin keeping the trace of $K$ constant

\[
\min_{K > 0} \max_{\alpha} \frac{1}{2} \alpha^T e - \frac{1}{2} \alpha^T G(K) \alpha
\]

\[
\text{s.t. } \alpha^T y = 0, \quad \alpha \geq 0, \quad \text{Trace}(K) = c
\]

- This optimization problem is convex: SemiDefinite Programming (SDP)

\[
\min_{K > 0, t} \quad t
\]

\[
t \geq \max_{\alpha} \frac{1}{2} \alpha^T e - \frac{1}{2} \alpha^T G(K) \alpha
\]

\[
\text{s.t. } \alpha^T y = 0, \quad \alpha \geq 0, \quad \text{Trace}(K) = c
\]
Learning the kernel

- With some rewriting:

\[
\begin{align*}
\min_{K>0,t,\lambda,v} & \quad t \\
\text{s.t.} & \quad \text{Trace}(K) = c \\
& \quad \left( G(K) \begin{bmatrix} \lambda y + v + e \\ t \end{bmatrix} \right) \succeq 0 \\
& \quad v \geq 0
\end{align*}
\]

and considering also the unlabeled data...

- We optimize the margin only on training data and learn the other matrix blocks.

- The capacity of the search space of possible kernel matrices is controlled in order to prevent overfitting

\[
K = \sum_i \mu_i K_i
\]
Kernel combination

- Learning kernels combination have been widely studied [Bach et al. 2004]
- Given multiple kernels \((K_1, \ldots, K_K)\) all convex combinations are kernels:
  \[
  K = \sum_i \mu_i K_i, \quad \mu_i \geq 0 \quad \sum_i \mu_i = 1
  \]
- We could compute \(\mu_i\) by crossvalidation but too many combinations!
- Instead of cross validation, maximize margin (again):
  \[
  \min_{\mathbf{w}, \mu, \xi} \frac{1}{2} \sum_{i=1}^{K} \frac{1}{\mu_i} \|\mathbf{w}_i\|^2 + C \sum_j \xi_j
  \]
  \[\text{s.t.} \quad y_i \sum_{i=1}^{K} \mathbf{w}_i^T \phi(x_j) \geq 1 - \xi_j, \quad \xi_j \geq 0 \quad \forall j\]
  \[\mu_i \geq 0, \quad \sum_{i=1}^{K} \mu_i = 1\]
- This optimization problem is \textit{jointly convex} in \(\mathbf{w}\) and \(\mu_i\)
- There is a unique global minimum.
Kernel combination

- Many kernels: each capturing different aspects of the data (e.g. texture, color, edges)
- **Combining** aspects is better than **selecting**!
- Simple combination is enough if all features are valid but insufficient if some features are not discriminative

<table>
<thead>
<tr>
<th>Kernels</th>
<th>Accuracy (time - sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td>60.9% (3)</td>
</tr>
<tr>
<td>Shape</td>
<td>70.2% (4)</td>
</tr>
<tr>
<td>Texture</td>
<td>63.7% (3)</td>
</tr>
<tr>
<td>HOG</td>
<td>58.3 % (4)</td>
</tr>
<tr>
<td>HSV</td>
<td>61.3% (3)</td>
</tr>
<tr>
<td>SIFT</td>
<td>70.6% (4)</td>
</tr>
<tr>
<td><strong>product</strong></td>
<td>85.5% (2)</td>
</tr>
<tr>
<td><strong>averaging</strong></td>
<td>84.5% (10)</td>
</tr>
<tr>
<td><strong>Kernel learning</strong></td>
<td>85.5% (152)</td>
</tr>
</tbody>
</table>

Nowozin et al. On Feature Combination for Multiclass Object Classification, ICCV 2009
SVM – Summarizing…

- SVMs were originally proposed by Boser, Guyon and Vapnik in 1992 and gained increasing popularity in late 1990s.

- (20 years later) SVMs are still widely used.

- SVM are currently among the best performers for a wide number of classification tasks ranging from text, images, genomic data.

- Which are the motivations of the success?
SVM – Summarizing...

- Flexibility in choosing a similarity function
  - SVMs can be applied to complex data types beyond feature vectors (e.g. graphs, images) by designing appropriate kernel functions.

- Sparseness of the solution when dealing with large data sets
  - only support vectors are used to specify the separating hyperplane

- Ability to handle large feature spaces
  - complexity does not depend on the dimensionality of the feature space

- Convex optimization
  - There is only a single global solution
• Strong theoretical guarantees (generalization bounds)
• SVM for regression [Vapnik et al. ’97]
  • I did not cover these aspects in the talk
• Kernels have been used for a wider number of tasks:
  • KRR [C. Saunders et al. ’98]
  • KPCA [Schölkopf et al. ’99]
  • KCCA [D.R Hardoon et al., 2004]
  • KFDA, spectral clustering…
• Linear techniques are more powerful with the kernels
  • implicit preprocessing, non-linear in the original data.
  • still linear in some feature space then still intuitive/interpretable
KRR, KPCA, KCCA
Kernel RR

• Linear RR:

\[
\min_w \| y - Xw \|^2 + \lambda \| w \|^2
\]

• Consider a non linear mapping: \( w = \Phi_x \alpha \)

\[
\min_{\alpha} \| y - K\alpha \|^2 + \lambda \alpha^T K\alpha
\]

• Solve (gradient equal to 0):

\[
\alpha = (K + \lambda I)^{-1} y
\]
Kernel PCA

- PCA
- Maximize the variance \( \max_w w^T X^T X w \)

assuming \( \mu = X e = 0 \),
the solution is
\[
X^T X w = \lambda w
\]

- KPCA [Schölkopf et al. ’99]
- Solve

\[
\max_w w^T \Phi^T_x \Phi_x w
\]

Considering the non linear vector \( w = \Phi_x \alpha \) and
assuming \( \mu = \Phi_x e = 0 \),
the solution is

\[
\Phi^T_x \Phi_x w = \lambda w \quad \Rightarrow \quad K^2 \alpha = \lambda K \alpha \quad \Rightarrow \quad K \alpha = \lambda \alpha
\]
• CCA [H. Hotelling, 1936]
  - Maximize the correlation of projections

\[
\max_{{w_x, w_y}} \frac{w_x X^T Y w_y}{\sqrt{w_x^T X^T X w_x \cdot w_y^T Y^T Y w_y}}
\]

• KCCA [D.R Hardoon et al., 2004]
  - Two non linear canonical vectors

\[
w_x = \Phi_x \alpha \quad \quad \quad \quad w_y = \Phi_y \beta
\]

• Solve

\[
\max_{\alpha, \beta} \frac{\alpha K_x^T K_y \beta}{\sqrt{\alpha^T K_x^T K_x \alpha \cdot \beta^T K_y^T K_y \beta}}
\]
Three categories: sport, aviation and paintball
Data: 400 images for each class and associated text
Features
- Image – Color (HSV) and texture (Gabor)
- Text – Term frequencies
• Tasks
  • Classification of web pages into 3 categories

<table>
<thead>
<tr>
<th></th>
<th>SVM</th>
<th>KCCA + SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>2.13%±0.23%</td>
<td>1.21%±0.27%</td>
</tr>
</tbody>
</table>

• Text query for image retrieval (% success)

<table>
<thead>
<tr>
<th></th>
<th>GVSM</th>
<th>KCCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top 10</td>
<td>1.27 %</td>
<td>17.3%</td>
</tr>
<tr>
<td>Top 30</td>
<td>5 %</td>
<td>30.3%</td>
</tr>
</tbody>
</table>

• The data set and the number of classes used were small
Kernel methods offer a modular framework. Two steps:
• Data are preprocessed in the kernel matrix. Data can be of various nature (sequences, images…)
• A kernel algorithm is used to analyze the data, looking only at the information contained in the kernel matrix

\[
f(x) = \sum_{i} \alpha_i y_i k(x_i, x)
\]
Learning in Structured Output Spaces
Learning in structured output

• Traditional *supervised* learning setting:
  • Data are available in form of examples and their associated correct answers.

\[ T = \{(x^1, y^1), \ldots, (x^\ell, y^\ell)\}, \quad x^i \in \mathcal{X}, y^i \in \mathcal{Y} \]

**Learning:**

Find \( h: \mathcal{X} \rightarrow \mathcal{Y} \) s.t.
\[ h(x^i) \approx y^i \quad \forall i = 1 \ldots \ell \]

**Hypotheses space**

\( h \in \mathcal{H} \)

**Prediction:**

\( h(x) = y \) on a new test sample \( x \).
Learning in structured output

- In the traditional supervised learning setting the decision function is a map from an input space to an output space
  \[ h : X \rightarrow Y \]

- Regression \( y = R \)
- Binary classification \( y = \{-1, 1\} \)
- Multiclass classification \( y = \{1, \ldots, k\} \)

- Structured output learning consider more complex output spaces.
- Why structured output learning?
Learning in structured output

• Nowadays many problems involve highly structured data which can be represented by sequences and graphs.

• This phenomenon is observed in several fields such as computational biology, computer vision or natural language processing.

• Structured output learning allows to cope with problems involving structured data in a principled way without losing information about the structure of the problem.
Learning in structured output

- **Named Entity Recognition**: locate named entities in text. Entities of interest are person names, location names, organization names, miscellaneous (dates, times...)

- Can we consider the interactions between adjacent words and realize a joint labeling for all the words in the sentence?

Multiclass classification

- **Observed variable**: words in a sentence.
- **Label**: entity tag.

PP  ESTUDIA  YA  PROYECTO  LEY  TV  REGIONAL  REMITIDO  POP  LA  JUNTA  Merida.

X    N N N    M m m    N N N O L

y    Label: entity tag.
Learning in structured output

- **Sequence labeling**: given an input sequence $x$, reconstruct the associated label sequence $y$ of equal length.

  \[
  \begin{array}{c}
  y_1 \quad y_2 \quad y_3 \\
  x_1 \quad x_2 \quad x_3 \\
  \end{array}
  \]

- **Named entity recognition**: given a training set of correct pairs of sentences and their associated entity tags can we *learn* how to extract entities from a new sentence?
Learning in structured output

- **Biological sequence alignment**: is used to determine the similarity between biological sequences.

  $$S = \{A, T, G, C\}, \quad S_1, S_2 \in S$$

  $S_1 = \text{ATGCTTTTC}$
  $S_2 = \text{CTGTCGCC}$

  $\text{ACTGATTACGTGAACTGGATCCA}$
  $\text{ACTC--TAGGTGAAGTG--ATCCCA}$

- Given two sequences $S_1, S_2 \in S$ a global alignment is an assignment of gaps, so as to line up each letter in one sequence with either a gap or a letter in the other sequence.
Learning in structured output

• Alignments can be represented as paths from the upper-left to the lower-right corner in the alignment graph.

• Sequence alignment: given a sequences pair $x$, predict the correct sequence $y$ of alignment operations (e.g. matches, mismatches, gaps).

$\begin{align*}
x &= \begin{cases} 
S_1 & \text{ATGCTTTTC} \\
S_2 & \text{CTGTCGCC} 
\end{cases} \\
y &= \begin{cases} 
\text{ATGCTTTTC} \\
\text{CTGTCGCC} 
\end{cases}
\end{align*}$
• **RNA secondary structure prediction**: given an RNA sequence, predict the most likely secondary structure.

```
AUCCCCGUAUCGAUC
AAAAUCCCAUGGGUAC
CCUAGUGAAAGUGUA
UAUACGUGCUCUGAU
UCUUUACUGAGGAGU
CAGUGAACGAACUGA
```
**Learning in structured output**

- *Sequence parsing*: given an input sequence \( x \), determine the associated parse tree \( y \) given an underlying context-free grammar.

  Example:

  Context-free grammar \( \mathcal{G} = \{ \mathcal{V}, \mathcal{A}, \mathcal{R}, S \} \)

  \( \mathcal{V} = \{ S \} \) set of non-terminals symbols

  \( \mathcal{A} = \{ G, A, U, C \} \) set of terminals symbols

  \( \mathcal{R} = \{ S \rightarrow SS | gSc | cSc | aSu | uSa | \varepsilon \} \).

- *RNA secondary structure prediction*: given a training set of correct RNA secondary structures can we *learn* how to determine the secondary structure of a new RNA sequence?

  Example:

  \( x = \text{GAUCGAUCGAUC} \)

  \( y \):

  \[
  \begin{align*}
  & S \\
  & \downarrow \quad \downarrow \\
  & G \\
  & \quad \quad A \\
  & \quad \quad \downarrow \\
  & C \\
  & \quad \quad \downarrow \\
  & U \\
  & \quad \quad \downarrow \\
  & A \\
  & \quad \quad \downarrow \\
  & U \\
  & \quad \quad \downarrow \\
  & C \quad \quad \downarrow \\
  & G \quad \quad \downarrow \\
  & \quad \quad \downarrow \\
  & S \quad \quad \downarrow \\
  & \quad \quad \downarrow \\
  & S \quad \quad \downarrow \\
  & \quad \quad \downarrow \\
  & S \quad \quad \downarrow \\
  & \quad \quad \downarrow \\
  & \varepsilon \\
  \end{align*}
  \]
**Learning in structured output**

**Multilabel** supervised classification (Output: $y = (y_1...y_n)$).

**Training set:**

$\mathcal{T} = \{(x^1, \bar{y}^1),..., (x^\ell, \bar{y}^\ell)\}, \quad x^i \in \mathcal{X}, \bar{y}^i \in \mathcal{Y}$

**Learning:**

Find $h : \mathcal{X} \rightarrow \mathcal{Y}$ s.t.

$h(x^i) \approx \bar{y}^i \quad \forall i = 1...\ell$

$h \in \mathcal{H}$

**Prediction:**

$h(x) = \arg \max_{y \in \mathcal{Y}} s(x,y)$

$s(x,y) = w^T \phi(x,y), \quad w \in \mathbb{R}^d$

$|\mathcal{Y}|$ is typically huge.

**Score**

**Hypotheses space**
Learning in structured output

• Many algorithms for structured output learning
  • CRFs [Lafferty et al. 2001]
  • Structured Output Perceptron [Collins, 2002]
  • M³N [Taskar et al. 2003], SVMISO [Tsochantaridis et al. 04]
  • SVMISO (1-slack) [Joachims et al., 08]
  • SEARN [Daume et al. 2006]
  • SODA [Ricci et al. 2008]
• Many others now!
Learning in structured output

- Many algorithms for structured output learning.
  - What do they have in common?

- Two main phases in designing the algorithm:
  - *Encoding*: define a suitable feature map $\phi(x,y)$. (Challenging!) We need to characterize the output space (huge) in a synthetic and compact way.
  
  - *Optimization*: define a suitable objective function and use it for learning.
• Features must be defined in a way such that prediction can be computed efficiently.

\[ h(x) = \arg \max_{y \in Y} w^T \phi(x, y) \]

| \( |Y| \) is typically huge. |

• The feature vector \( \phi(x, y) \) decomposes as sum of elementary features \( \phi(x, y) \) on “parts”.

• Parts are typically edges or nodes in a graph.
**Sequence labeling:**
- Example: CRF with HMM features

- In general features reflect long range interactions (when labeling $x_i$ past and future observations are taken into account).
- Arbitrary features of the observations are considered (e.g. spelling properties in NER).

\[
\phi_{pq}^{y,y}(x_k, y_k) = I(x_k = q) I(y_k = p) \quad q \in \Sigma_x, p \in \Sigma_y
\]

\[
\phi_{pz}^{y}(y_k, y_{k-1}) = I(y_{k-1} = p) I(y_k = z) \quad p, z \in \Sigma_y
\]
**Encoding**

**Sequence alignment:**

- **3-parameter model:**

\[
\phi(x, y) = \begin{bmatrix}
\text{#matches} \\
\text{#mismatches} \\
\text{#gaps}
\end{bmatrix} = \begin{bmatrix}
4 \\
1 \\
4
\end{bmatrix}
\]

- In practice more complex models are used:
  - **4-parameter model:** affine function for gap penalties, i.e.
different costs if the gap starts (gap opening penalty) in a given position or if it continues (gap extension penalty).
  - **211/212-parameter model:** \(\phi(x, y)\) contains the statistics associated to the gap penalties and all the possible pairs of amino acids.
**Sequence parsing:**

- The feature vector contains the number of occurrences of each rule.

\[
\phi(x, y) = \begin{bmatrix}
1 \\
1 \\
2 \\
2 \\
2
\end{bmatrix}
\begin{align*}
S &\to SS \\
S &\to gSc \\
S &\to cSc \\
S &\to aSu \\
S &\to uSa \\
S &\to \varepsilon.
\end{align*}
\]
**Encoding**

- Having defined these features predictions

\[ h(x) = \arg \max_{y \in \mathcal{Y}} \, w^T \phi(x, y) \]

can be computed efficiently with dynamic programming (DP).

- **Sequence labeling**: Viterbi algorithm
- **Sequence alignment**: Needleman-Wunsch algorithm
- **Sequence parsing**: Cocke-Younger-Kasami (CYK) algorithm
The number $N$ of possible output vector $y_k$ given an observation $x$ is typically huge.

SODA [Ricci et al. 2008]:
- To characterize the distribution of the scores its mean and its variance are considered.

$$\mu(x,y) = \frac{1}{N} \sum_{k=1}^{N} s(x,y_k) = w^T \left( \frac{1}{N} \sum_{k=1}^{N} \phi(x,y_k) \right) = w^T \mu$$

$$\sigma^2(x,y) = w^T \left( \frac{1}{N} \sum_{k=1}^{N} (\phi(x,y_k) - \mu)(\phi(x,y_k) - \mu)^T \right) w = w^T C w$$

- $C$ and $\mu$ can be computed efficiently with DP techniques.
SODA

- **SODA**: Minimize the number of output vectors with score higher than the score of the correct pairs.

- Maximize the Z-score:

\[
Z(x) = \frac{s(x, y) - \mu(x, y)}{\sigma(x, y)}
\]
• The Z-score can be expressed as a function of the parameters $\mathbf{w}$.

$$Z(\mathbf{x}) = \frac{s(\mathbf{x}, \mathbf{y}) - \mu(\mathbf{x}, \mathbf{y})}{\sigma(\mathbf{x}, \mathbf{y})} = \frac{\mathbf{w}^T (\phi(\mathbf{x}, \mathbf{y}) - \mu)}{\sqrt{\mathbf{w}^T \mathbf{Cw}}} = \frac{\mathbf{w}^T \mathbf{b}}{\sqrt{\mathbf{w}^T \mathbf{Cw}}}$$

• Two equivalent optimization problems:

$$\max_{\mathbf{w}} \frac{\mathbf{w}^T \mathbf{b}}{\sqrt{\mathbf{w}^T \mathbf{Cw}}} \quad \leftrightarrow \quad \min_{\mathbf{w}} \frac{1}{N} \sum_{k=1}^{N} \xi_k^2$$

s.t. $\mathbf{w}^T (\phi(\mathbf{x}, \mathbf{y}) - \phi(\mathbf{x}, \mathbf{y}_k)) = 1 + \xi_k$
SODA

- Ranking loss:

\[
\mathcal{L}^{rk}(x, \overline{y}) = \frac{1}{N} \sum_{y_k \neq \overline{y}} I[s(x, \overline{y}) \leq s(x, y_k)]
\]

- An upper bound on the ranking loss is minimized:

\[
\mathcal{L}^{urk}(x, \overline{y}) = \frac{1}{N} \sum_{k=1}^{N} \xi_k^2 = \frac{1}{N} \sum_{y_k \neq \overline{y}} (w^T (\phi(x, \overline{y}) - \phi(x, y_k)) - 1)^2 \geq \mathcal{L}^{rk}(x, \overline{y})
\]

- The number of output vectors with score higher than the score of the correct pairs is minimized.
SODA

- Given a training set the empirical risk associated to the upper-bound on the ranking loss is minimized.

\[
\min \limits_w \sum_{i=1}^{\ell} \frac{1}{N_i} \sum_{k=1}^{N_i} \xi^2_{ik} \\
\text{s.t.} \quad w^T (\phi(x_i, y_i) - \phi(x_i, y_k)) = 1 + \xi_{ik} \quad \forall i
\]

- To solve it an equivalent formulation in terms of matrices \(C\) and \(b\) is considered.

\[
\max \limits_w \frac{w^T \sum_{i=1}^{\ell} b_i}{\sqrt{w^T \sum_{i=1}^{\ell} (C_i + b_i b_i^T) w}} = \frac{w^T b^*}{\sqrt{w^T C^* w}} \quad \text{SODA (Structured Output Discriminant Analysis)}
\]

- Convex optimization problem (simple matrix inversion \(w = C^{-1} b^*\))
- Fast conjugate gradient methods available.
Loss in structured output learning

- Minimize the number of incorrect macrolabels $y$.

$$\mathcal{L}^{0/1}(x, \bar{y}) = I[h(x) \neq \bar{y}]$$

CRFs [Lafferty et al. 2001], HMSVM [Altun et al. 2003], structured output perceptron [Collins 2002].

- Minimize the number of incorrect microlabels $y$.

$$\mathcal{L}^{hm}(x, \bar{y}) = I[h(x)_j \neq y_j]$$

M$^3$Ns [Taskar et al., 03], SVMISO [Tsochantaridis et al., 04].
**SVMISO**

- M³Ns and SVMISO: structured output extension of SVMs

\[
\min_{w, \xi_i} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i \\
\text{s.t. } w^T \phi(x^i, \bar{y}^i) - w^T \phi(x^i, y^i_k) \geq \Delta(y^i_k, \bar{y}^i) - \xi_i \quad \forall y^i_k \neq \bar{y}^i
\]

where \( \Delta(y^i_k, \bar{y}^i) = \sum_j I[y^i_k, j \neq \bar{y}_j] \)

- The idea is to separate with *maximal margin* the score of the correct input-output pairs from the best scoring uncorrect pair
Crammer and Singer approach to multiclass SVM: the first example of SVMISO.

\[
\min_{w, \xi_i} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i \\
\text{s.t. } w^T \phi(x^i, y^i) - w^T \phi(x^i, y_k^i) \geq \Delta(y_k^i, y^i) - \xi_i \quad \forall y_k^i \neq y^i
\]

where the feature encoding is:

\[
\phi(x^i, y = 1)^T = \begin{bmatrix} x^i T & 0 & \cdots & 0 \end{bmatrix} \\
\phi(x^i, y = 2)^T = \begin{bmatrix} 0 & x^i T & \cdots & 0 \end{bmatrix} \\
\vdots
\]

\[
\phi(x^i, y = k)^T = \begin{bmatrix} 0 & \cdots & 0 & x^i T \end{bmatrix}
\]

and the weight vector:

\[
w^T = [w_1^T \ w_2^T \ \cdots \ w_k^T]
\]
**SVMISO**

\[
\min_{w, \xi} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i \\
\text{s.t.} \quad w^T \phi(x^i, \bar{y}^i) - w^T \phi(x^i, y_k^i) \geq \Delta(y_k^i, \bar{y}^i) - \xi_i \quad \forall y_k^i \neq \bar{y}^i
\]

- SVMISO: Solve by cutting plane method
  - Initialize constraint set to be empty
  - Iterate until convergence:
    - Solve optimization using current constraint set
    - Add maximally violated constraint for current solution

- In the applications discussed before the most violated constraints corresponds to the current best scoring pair and can be found with DP.

\[
\hat{y}_k^i = \arg \max_{y \in Y} w^T \phi(x^i, y_k^i)
\]
SVMISO

- SVMISO: 1-slack formulation [Joachims et al., 08]

\[
\min_{w, \xi} \frac{1}{2} \|w\|^2 + C\xi
\]

s.t. \[
\frac{1}{\ell} \sum_{i=1}^{\ell} [w^T \phi(x^i, y^i) - w^T \phi(x^i, y_k^i)] \geq \frac{1}{\ell} \sum_{i=1}^{\ell} [\Delta(y_k^i, y^i)] - \xi \quad \forall y_k^i \neq y^i
\]

- Solve by cutting plane method: quite fast and scalable approach.
- Sparsity is essential!!!
- SVMISO allows to use of kernels.
Applications

- Sequence labeling: artificial data.
  - Chain CRF with HMM features.
  - Average number of incorrect labels varying the level of noise $p$.

$|\Sigma_x| = 4 \quad |\Sigma_y| = 2$

$|\Sigma_x| = 5 \quad |\Sigma_y| = 3$
Applications

- Sequence labeling: NER
  - Spanish news wire article - Special Session of CoNLL02
  - 300 sentences with average length of 30 words.
  - 9 labels: non-name, beginning and continuation of persons, organizations, locations and miscellaneous names.
  - Two sets of binary features: $S_1$ (HMM features) and $S_2$ ($S_1$ and HMM features for the previous and the next word).

Labeling error on test set (five-fold crossvalidation)

<table>
<thead>
<tr>
<th>Method</th>
<th>$S_1$</th>
<th>$S_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SODA</td>
<td>10.13</td>
<td>8.27</td>
</tr>
<tr>
<td>SVMISO</td>
<td>10.97</td>
<td>8.11</td>
</tr>
<tr>
<td>Perceptron</td>
<td>20.99</td>
<td>13.78</td>
</tr>
<tr>
<td>CRFs</td>
<td>12.01</td>
<td>8.29</td>
</tr>
</tbody>
</table>
Structured output learning provides a powerful framework to handle problems with structured data.

It is somehow an extension of generative models (GMs):
- Both encode potential dependencies among and between input and output.
- GMs are typically trained by maximum likelihood estimation. Discriminative models (DMs) specify the probability of a possible output \( y \) given an observation \( x \) (consider conditional probability \( P(y|x) \) rather than joint probability \( P(y,x) \)).
- DMs encode long term dependencies between input.

Many algorithms have nice convergence properties and bounds.

Drawbacks
- Learning does not scale much.
- In many algorithms loss should be decomposable, e.g. sum of local costs on subparts.
Summarizing

Many applications:

- Sequence labeling (e.g. part-of-speech tagging, named-entity recognition) [Lafferty et al. 01, Altun et al. 03]
- Hyperlinked documents classification [Taskar et al. 03]
- Reinforcement learning / planning [Abbeel, Ng 04]
- Classification with non-linear performance measures (e.g. F-measure) [Joachims 05]
- Segmentation of 3D scan data [Anguelov et al., 2005]
- Sequence alignment for protein structure prediction [Yu et al., 2006]
- Object localization [Blaschko & Lampert, 2008]
- Image segmentation [Szummer et al., 2008]

and more….
Thank you