Kernels for strings and applications in bioinformatics

Jean-Philippe Vert
Jean-Philippe.Vert@ensmp.fr

Centre for Computational Biology
Ecole des Mines de Paris, ParisTech

ISM open forum, Tokyo, Japan, September 28, 2007

Outline

1. Kernels and kernel methods

2. Kernels for biological sequences
   - Explicit vector space embedding
   - Mutual information kernels
   - Alignment kernels
   - Application: remote homology detection

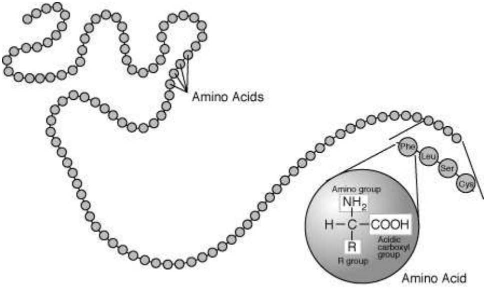
Outline

1. Kernels and kernel methods

2. Kernels for biological sequences
    - Explicit vector space embedding
    - Mutual information kernels
    - Alignment kernels
    - Application: remote homology detection

Kernels and Kernel Methods

Proteins



A : Alanine	V : Valine	L : Leucine
F : Phenylalanine	P : Proline	M : Méthionine
E : Acide glutamique	K : Lysine	R : Arginine
T : Threonine	C : Cysteine	N : Asparagine
H : Histidine	V : Thyrosine	W : Tryptophane
I : Isoleucine	S : Sérine	Q : Glutamine
D : Acide aspartique	G : Glycine	

Challenges with protein sequences

- A protein sequences can be seen as a variable-length sequence over the 20-letter alphabet of amino-acids, e.g., insuline: FVNQHLCGSHLVEALYLVCGERGFFYTPKA
- These sequences are produced at a fast rate (result of the sequencing programs)
- Need for algorithms to compare, classify, analyze these sequences
- Applications: classification into functional or structural classes, prediction of cellular localization and interactions, ...

Example: supervised sequence classification

Data (training)

- Secreted proteins:

  MASKATLLLAFTLLFATCIARHQQRQQQQNQCQLQNIEA...
  MARSSLFTFLCLAVFINGCLSQIEQQSPWEFQGSEVW...
  MALHTVLIMLSLLPMLEAQNPEHANITIGEPITNETLGWL...
  ...

- Non-secreted proteins:

  MAPPSVFAEVPQAQPVLVFKLIADFREDPDPRKVNLGVG...
  MAHTLGLTQPNSTEPHKISFTAKEIDVIEWKGDILVVG...
  MSISESYAKEIKTAFRQFTDFPIEGEQFEDFLPIIGNP..
  ...

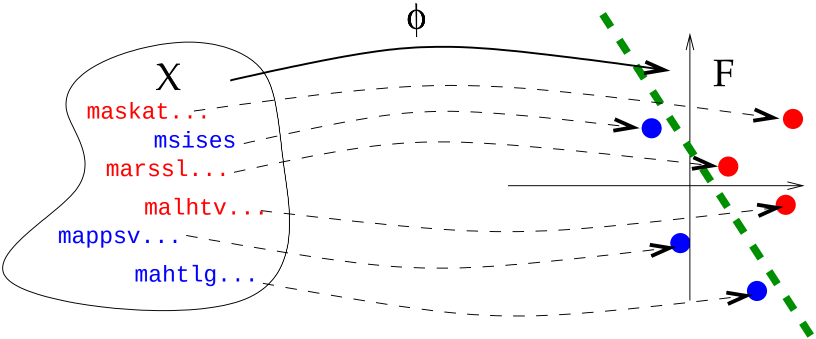
Goal

- Build a classifier to predict whether new proteins are secreted or not.

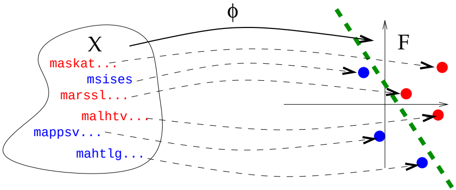
Supervised classification with vector embedding

The idea

- Map each string $x \in \mathcal{X}$ to a vector $\Phi(x) \in \mathbb{R}^p$.
- Train a classifier for vectors on the images $\Phi(x_1), \ldots, \Phi(x_n)$ of the training set (nearest neighbor, linear perceptron, logistic regression, support vector machine...)



Example: support vector machine



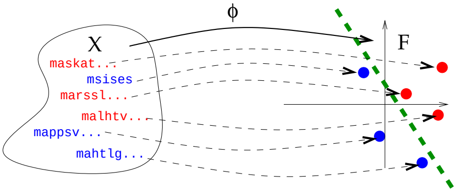
SVM algorithm

$$f(x) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \Phi(x_i)^\top \Phi(x)\right) \; ,$$

where $\alpha_1, \ldots, \alpha_n$ solve, under the constraints $0 \leq \alpha_i \leq C$:

$$\min_{\alpha}\left(\frac{1}{2}\sum_{i=1}^{n}\sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j \Phi(x_i)^\top \Phi(x_j) - \sum_{i=1}^{n} \alpha_i\right) \; .$$

Explicit vector embedding



Difficulties

- How to define the mapping $\Phi : \mathcal{X} \to \mathbb{R}^p$?
- No obvious vector embedding for strings in general.
- How to include prior knowledge about the strings (grammar, probabilistic model...)?

Implicit vector embedding with kernels

The kernel trick

- Many algorithms just require inner products of the embeddings
- We call it a kernel between strings:

$$K(x, x') \triangleq \Phi(x)^\top \Phi(x')$$

Examples

- SVM
- Nearest neighbor:

$$d(x, x')^2 = \| \Phi(x) - \Phi(x') \|^2 = K(x, x) + K(x', x') - 2K(x, x') .$$

- Many other kernel methods (perceptron, regression...)

Implicit vector embedding with kernels

The kernel trick

- Many algorithms just require inner products of the embeddings
- We call it a kernel between strings:

$$K(x, x') \stackrel{\Delta}{=} \Phi(x)^\top \Phi(x')$$

Examples

- SVM
- Nearest neighbor:

$$d(x, x')^2 = \| \Phi(x) - \Phi(x') \|^2 = K(x, x) + K(x', x') - 2K(x, x') .$$

- Many other kernel methods (perceptron, regression...)

Positive Definite Kernels

Definition

A positive definite (p.d.) kernel on the set $\mathcal{X}$ is a function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ symmetric:

$$\forall \left(\mathbf{x}, \mathbf{x}'\right) \in \mathcal{X}^2, \quad K\left(\mathbf{x}, \mathbf{x}'\right) = K\left(\mathbf{x}', \mathbf{x}\right),$$

and which satisfies, for all $N \in \mathbb{N}$, $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) \in \mathcal{X}^N$ et $(a_1, a_2, \ldots, a_N) \in \mathbb{R}^N$:

$$\sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j K\left(\mathbf{x}_i, \mathbf{x}_j\right) \geq 0.$$

Kernels as Inner Products

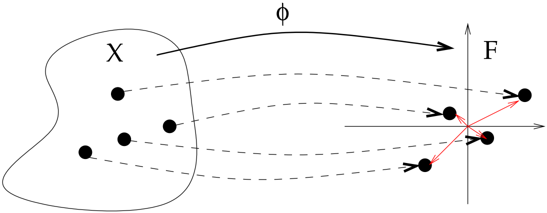
Theorem (Aronszajn, 1950)

K is a p.d. kernel on the set $\mathcal{X}$ if and only if there exists a Hilbert space $\mathcal{H}$ and a mapping

$$\Phi : \mathcal{X} \mapsto \mathcal{H} \, ,$$

such that, for any $\mathbf{x}, \mathbf{x}'$ in $\mathcal{X}$:

$$K\left(\mathbf{x}, \mathbf{x}'\right) = \left\langle \Phi\left(\mathbf{x}\right), \Phi\left(\mathbf{x}'\right) \right\rangle_{\mathcal{H}} \, .$$



Examples

Kernels for vectors

Classical kernels for vectors ($\mathcal{X} = \mathbb{R}^p$) include:

- The linear kernel

$$K_{lin}\left(\mathbf{x}, \mathbf{x}'\right) = \mathbf{x}^\top \mathbf{x}' .$$

- The polynomial kernel

$$K_{poly}\left(\mathbf{x}, \mathbf{x}'\right) = \left(\mathbf{x}^\top \mathbf{x}' + a\right)^d .$$

- The Gaussian RBF kernel:

$$K_{Gaussian}\left(\mathbf{x}, \mathbf{x}'\right) = \exp\left(-\frac{\parallel \mathbf{x} - \mathbf{x}' \parallel^2}{2\sigma^2}\right) .$$

Kernel for strings?

- A kernel defines an implicit geometry on the space of data, although data do not need to have any prior geometric/algebric structure
- Kernel engineering is the problem of designing specific kernel for specific data and specific tasks. Good place to put prior knowledge!
- We will now see on a practical examples different technical tricks to design kernels.

Kernels for Biological Sequences

Outline

Vector embedding for strings

The idea

Represent each sequence $\mathbf{x}$ by a fixed-length numerical vector $\Phi(\mathbf{x}) \in \mathbb{R}^p$. How to perform this embedding?

Physico-chemical kernel

Extract relevant features, such as:

- length of the sequence
- time series analysis of numerical physico-chemical properties of amino-acids along the sequence (e.g., polarity, hydrophobicity), using for example:
  - Fourier transforms (Wang et al., 2004)
  - Autocorrelation functions (Zhang et al., 2003)

$$r_j = \frac{1}{n-j} \sum_{i=1}^{n-j} h_i h_{i+j}$$

Vector embedding for strings

The idea

Represent each sequence $\mathbf{x}$ by a fixed-length numerical vector $\Phi(\mathbf{x}) \in \mathbb{R}^p$. How to perform this embedding?

Physico-chemical kernel

Extract relevant features, such as:

- length of the sequence
- time series analysis of numerical physico-chemical properties of amino-acids along the sequence (e.g., polarity, hydrophobicity), using for example:
  - Fourier transforms (Wang et al., 2004)
  - Autocorrelation functions (Zhang et al., 2003)

$$r_j = \frac{1}{n-j} \sum_{i=1}^{n-j} h_i h_{i+j}$$

Substring indexation

The approach

Alternatively, index the feature space by fixed-length strings, i.e.,

$$\Phi(\mathbf{x}) = (\Phi_u(\mathbf{x}))_{u \in \mathcal{A}^k}$$

where $\Phi_u(\mathbf{x})$ can be:

- the number of occurrences of u in $\mathbf{x}$ (without gaps) : spectrum kernel (Leslie et al., 2002)
- the number of occurrences of u in $\mathbf{x}$ up to m mismatches (without gaps) : mismatch kernel (Leslie et al., 2004)
- the number of occurrences of u in $\mathbf{x}$ allowing gaps, with a weight decaying exponentially with the number of gaps : substring kernel (Lohdi et al., 2002)

Example: spectrum kernel

- The 3-spectrum of

$$\mathbf{x} = \texttt{CGGSLIAMMWFGV}$$

  is:

$$(\texttt{CGG,GGS,GSL,SLI,LIA,IAM,AMM,MMW,MWF,WFG,FGV})\ .$$

- Let $\Phi_u(\mathbf{x})$ denote the number of occurrences of u in $\mathbf{x}$. The k-spectrum kernel is:

$$K(\mathbf{x}, \mathbf{x}') := \sum_{u \in \mathcal{A}^k} \Phi_u(\mathbf{x}) \Phi_u(\mathbf{x}')\ .$$

- This is formally a sum over $|\mathcal{A}|^k$ terms, but at most $|\mathbf{x}| - k + 1$ terms are non-zero in $\Phi(\mathbf{x})$

Substring indexation in practice

- Implementation in $O(|\mathbf{x}| + |\mathbf{x}'|)$ in memory and time for the spectrum and mismatch kernels (with suffix trees)
- Implementation in $O(|\mathbf{x}| \times |\mathbf{x}'|)$ in memory and time for the substring kernels
- The feature space has high dimension ($|\mathcal{A}|^k$), so learning requires regularized methods (such as SVM)

Dictionary-based indexation

The approach

- Chose a dictionary of sequences $\mathcal{D} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$
- Chose a measure of similarity $s(\mathbf{x}, \mathbf{x}')$
- Define the mapping $\Phi_{\mathcal{D}}(\mathbf{x}) = (s(\mathbf{x}, \mathbf{x}_i))_{\mathbf{x}_i \in \mathcal{D}}$

Examples

This includes:

- Motif kernels (Logan et al., 2001): the dictionary is a library of motifs, the similarity function is a matching function
- Pairwise kernel (Liao & Noble, 2003): the dictionary is the training set, the similarity is a classical measure of similarity between sequences.

Dictionary-based indexation

The approach

- Chose a dictionary of sequences $\mathcal{D} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$
- Chose a measure of similarity $s(\mathbf{x}, \mathbf{x}')$
- Define the mapping $\Phi_{\mathcal{D}}(\mathbf{x}) = (s(\mathbf{x}, \mathbf{x}_i))_{\mathbf{x}_i \in \mathcal{D}}$

Examples

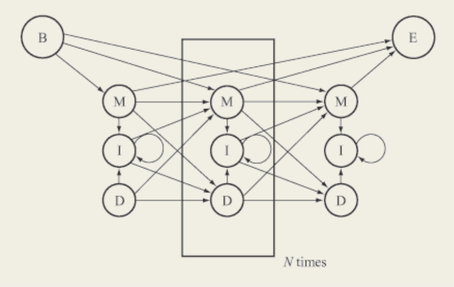
This includes:

- Motif kernels (Logan et al., 2001): the dictionary is a library of motifs, the similarity function is a matching function
- Pairwise kernel (Liao & Noble, 2003): the dictionary is the training set, the similarity is a classical measure of similarity between sequences.

Outline

Probabilistic models for sequences

Probabilistic modeling of biological sequences is older than kernel designs. Important models include HMM for protein sequences, SCFG for RNA sequences.



Parametric model

A model is a family of distribution

$$\{P_\theta, \theta \in \Theta \subset \mathbb{R}^m\} \subset \mathcal{M}_1^+(\mathcal{X})$$

Mutual information kernels

Definition

- Chose a prior $w(d\theta)$ on the measurable set Θ
- Form the kernel (Seeger, 2002):

$$K(\mathbf{x}, \mathbf{x}') = \int_{\theta \in \Theta} P_\theta(\mathbf{x}) P_\theta(\mathbf{x}') w(d\theta) \, .$$

- No explicit computation of a finite-dimensional feature vector
- $K(\mathbf{x}, \mathbf{x}') = < \phi(\mathbf{x}), \phi(\mathbf{x}') >_{L_2(w)}$ with

$$\phi(\mathbf{x}) = (P_\theta(\mathbf{x}))_{\theta \in \Theta} \, .$$

Example: coin toss

- Let $P_\theta(X = 1) = \theta$ and $P_\theta(X = 0) = 1 - \theta$ a model for random coin toss, with $\theta \in [0, 1]$.
- Let $d\theta$ be the Lebesgue measure on $[0, 1]$
- The mutual information kernel between $\mathbf{x} = 001$ and $\mathbf{x}' = 1010$ is:

$$\begin{cases} P_\theta(\mathbf{x}) & = \theta (1 - \theta)^2 \ , \\ P_\theta(\mathbf{x}') & = \theta^2 (1 - \theta)^2 \ , \end{cases}$$

$$K(\mathbf{x}, \mathbf{x}') = \int_0^1 \theta^3 (1 - \theta)^4 \, d\theta = \frac{3!4!}{8!} = \frac{1}{280} \ .$$

Context-tree model

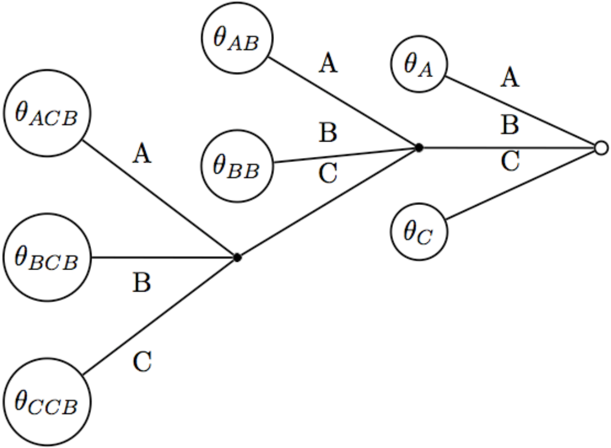
Definition

A context-tree model is a variable-memory Markov chain:

$$P_{\mathcal{D},\theta}(\mathbf{x}) = P_{\mathcal{D},\theta}\left(x_1 \dots x_D\right) \prod_{i=D+1}^{n} P_{\mathcal{D},\theta}\left(x_i \mid x_{i-D} \dots x_{i-1}\right)$$

- $\mathcal{D}$ is a suffix tree
- $\theta \in \Sigma^{\mathcal{D}}$ is a set of conditional probabilities (multinomials)

Context-tree model: example



$$P(AABACBACC) = P(AAB)\theta_{AB}(A)\theta_A(C)\theta_C(B)\theta_{ACB}(A)\theta_A(C)\theta_C(A) .$$

The context-tree kernel

Theorem (Cuturi et al., 2004)

- *For particular choices of priors, the context-tree kernel:*

$$K\left(\mathbf{x},\mathbf{x}'\right)=\sum_{\mathcal{D}}\int_{\theta\in\Sigma^{\mathcal{D}}}P_{\mathcal{D},\theta}(\mathbf{x})P_{\mathcal{D},\theta}(\mathbf{x}')w(d\theta|\mathcal{D})\pi(\mathcal{D})$$

  can be computed in $O(|\mathbf{x}|+|\mathbf{x}'|)$ with a variant of the Context-Tree Weighting algorithm.
- *This is a valid mutual information kernel.*
- *The similarity is related to information-theoretical measure of mutual information between strings.*

Outline

1. Kernels and kernel methods

2. Kernels for biological sequences
   - Explicit vector space embedding
   - Mutual information kernels
   - Alignment kernels
   - Application: remote homology detection

Sequence alignment

Motivation

How to compare 2 sequences?

$$\mathbf{x}_1 = \texttt{CGGSLIAMMWFGV}$$
$$\mathbf{x}_2 = \texttt{CLIVMMNRLMWFGV}$$

Find a good alignment:

```
CGGSLIAMM----WFGV
|...|||||....||||
C---LIVMMNRLMWFGV
```

Alignment score

In order to quantify the relevance of an alignment π, define:

- a substitution matrix $S \in \mathbb{R}^{\mathcal{A}\times\mathcal{A}}$
- a gap penalty function $g : \mathbb{N} \rightarrow \mathbb{R}$

Any alignment is then scored as follows

```
CGGSLIAMM----WFGV
|...|||||....||||
C---LIVMMNRLMWFGV
```

$$s_{S,g}(\pi) = S(C,C) + S(L,L) + S(I,I) + S(A,V) + 2S(M,M) + S(W,W) + S(F,F) + S(G,G) + S(V,V) - g(3) - g(4)$$

Local alignment kernel

Smith-Waterman score

- The widely-used Smith-Waterman local alignment score is defined by:

$$SW_{S,g}(\mathbf{x},\mathbf{y}) := \max_{\pi\in\Pi(\mathbf{x},\mathbf{y})} s_{S,g}(\pi).$$

- It is symmetric, but not positive definite...

LA kernel

The local alignment kernel:

$$K_{LA}^{(\beta)}(\mathbf{x},\mathbf{y}) = \sum_{\pi\in\Pi(\mathbf{x},\mathbf{y})} \exp\left(\beta s_{S,g}(\mathbf{x},\mathbf{y},\pi)\right),$$

is symmetric positive definite (Vert et al., 2004).

Local alignment kernel

Smith-Waterman score

- The widely-used Smith-Waterman local alignment score is defined by:

$$SW_{S,g}(\mathbf{x},\mathbf{y}) := \max_{\pi \in \Pi(\mathbf{x},\mathbf{y})} s_{S,g}(\pi).$$

- It is symmetric, but not positive definite...

LA kernel

The local alignment kernel:

$$K_{LA}^{(\beta)}(\mathbf{x},\mathbf{y}) = \sum_{\pi \in \Pi(\mathbf{x},\mathbf{y})} \exp\left(\beta s_{S,g}(\mathbf{x},\mathbf{y},\pi)\right),$$

is symmetric positive definite (Vert et al., 2004).

LA kernel is p.d.: proof

- If K_1 and K_2 are p.d. kernels for strings, then their convolution defined by:

$$K_1 \star K_2(\mathbf{x}, \mathbf{y}) := \sum_{\mathbf{x}_1\mathbf{x}_2 = \mathbf{x}, \mathbf{y}_1\mathbf{y}_2 = \mathbf{y}} K_1(\mathbf{x}_1, \mathbf{y}_1) K_2(\mathbf{x}_2, \mathbf{y}_2)$$

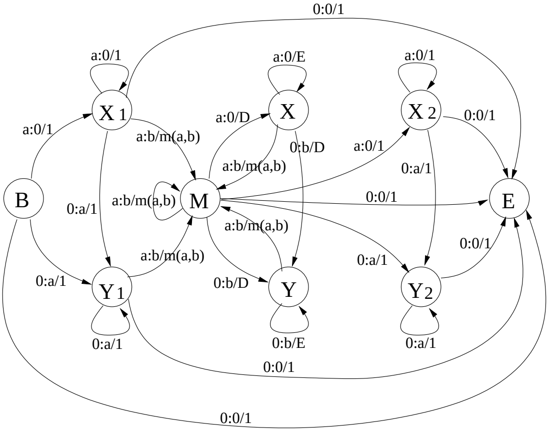
  is also p.d. (Haussler, 1999).

- LA kernel is p.d. because it is a convolution kernel (Haussler, 1999):

$$K_{LA}^{(\beta)} = \sum_{n=0}^{\infty} K_0 \star \left(K_a^{(\beta)} \star K_g^{(\beta)} \right)^{(n-1)} \star K_a^{(\beta)} \star K_0 .$$

  where K_0, K_a and K_g are three basic p.d. kernels (Vert et al., 2004).

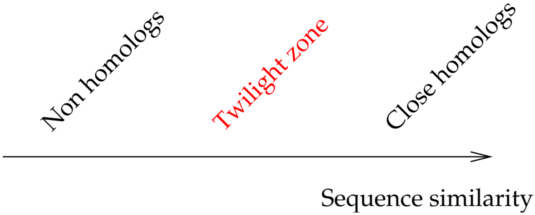
- Implementation by dynamic programming in $O(|\mathbf{x}| \times |\mathbf{x}'|)$



- In practice, values are too large (exponential scale) so taking its logarithm is a safer choice (but not p.d. anymore!)

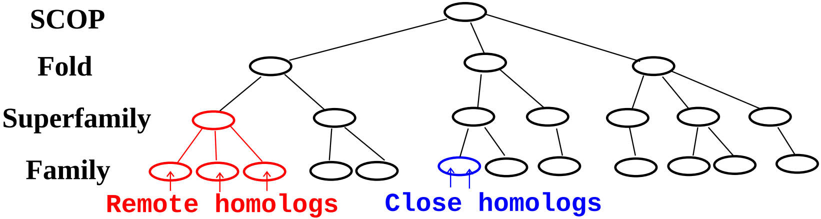
Outline

Remote homology



- Homologs have common ancestors
- Structures and functions are more conserved than sequences
- Remote homologs can not be detected by direct sequence comparison

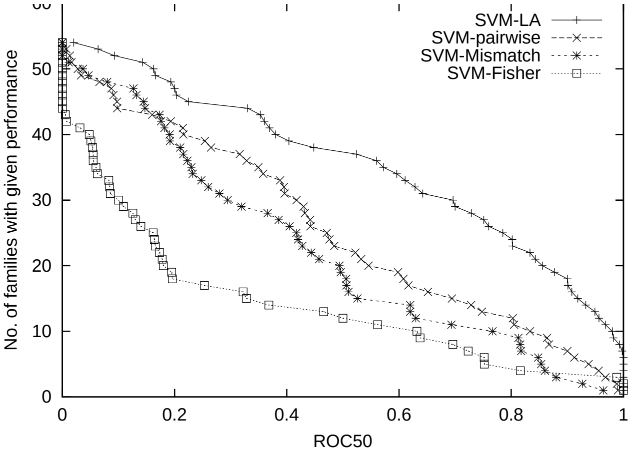
SCOP database



A benchmark experiment

- Goal: recognize directly the superfamily
- Training: for a sequence of interest, positive examples come from the same superfamily, but different families. Negative from other superfamilies.
- Test: predict the superfamily.

Difference in performance



Performance on the SCOP superfamily recognition benchmark (from Vert et al., 2004).

Conclusion

Conclusion

- Many multivariate statistical methods can be used with strings when a string kernel is defined.
- We saw several principles for string kernel design
  - explicit vector embedding
  - mutual information kernels
  - alignment kernels
- We omitted many other examples (marginalized kernels, Fisher kernels, ...)
- The choice of the kernel does matter in the final performance.
- Many open questions: which principles to choose / select good kernels?