Advanced Methods for Sequence Analysis

Gunnar Rätsch

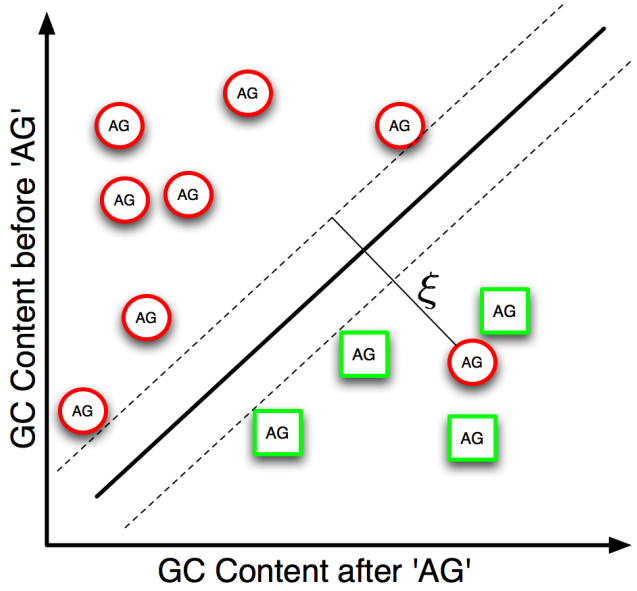
Friedrich Miescher Laboratory, Tübingen

Vorlesung WS 2006/2007
Eberhard-Karls-Universität Tübingen

8 November, 2006

http://www.fml.mpg.de/raetsch/lectures/amsa

SVMs: Geometric View



Minimize

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

Subject to

$$y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

- The examples on the margin are called support vectors [Vapnik, 1995]
- Called the soft margin SVM or the C-SVM [Cortes and Vapnik, 1995]

SVM: Loss View

$$\underset{\mathbf{w},b}{\text{minimize}} \frac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{N} \ell(f_{\mathbf{w},b}(\mathbf{x}), y_i),$$

where

$$\ell(f_{\mathbf{w},b}(\mathbf{x}), y_i) := C \max\{0, 1 - y_i(f_{\mathbf{w},b}(\mathbf{x}))\}$$
$$f_{\mathbf{w},b}(\mathbf{x}) := \mathbf{w}^\top \mathbf{x}_i + b.$$

The above loss function is known as the hinge loss.

Regularizer = $\frac{1}{2}\|\mathbf{w}\|^2$.

Empirical Risk = $\sum_{i=1}^{N} \ell(\mathbf{w}^\top \mathbf{x}_i + b, y_i)$.

How much does a mistake cost us?

Optimization

We have two versions of the same problem

Primal

$$\text{minimize}_{w,b} \frac{1}{2}\|w\|^2 + \sum_{i=1}^{N} \ell(w^\top x_i + b, y_i).$$

For a convex differentiable loss function, we can solve this directly using gradient methods or Newton's method.

Dual (with ℓ = hinge loss)

$$\begin{aligned} \text{maximize}_\alpha \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \\ \text{subject to} \quad & \sum_{i=1}^{N} \alpha_i y_i = 0 \\ & \alpha_i \geqslant 0 \text{ for } i = 1, 2, \ldots, N. \end{aligned}$$

We can use primal dual interior point methods to solve the convex optimization problem.

Representer Theorem for Kernel Algorithms

Let c be an arbitrary cost function and r be a strictly monotonically increasing function on $[0, \infty)$ (for instance $r(\|\mathbf{w}\|_2) = \lambda\|\mathbf{w}\|_2$, $\lambda > 0$).

$$\min_{\mathbf{w}\in\mathcal{F}} \sum_{n=1}^{N} c(y_n, \langle \mathbf{w}, \mathbf{x}_n \rangle) + r(\|\mathbf{w}\|_2) \qquad (1)$$

Then for any N examples $(\mathbf{x}_1, y_1), \ldots, (x_N, y_N)$, then **any solution $\mathbf{w}^*$** of (1) can be expressed by

$$\mathbf{w}^* = \sum_{n=1}^{N} \alpha_n \mathbf{x}_n$$

for some $\alpha_n \in \mathbb{R}$, $n = 1, \ldots, N$.

Only ℓ_2-norm regularizers have this property!

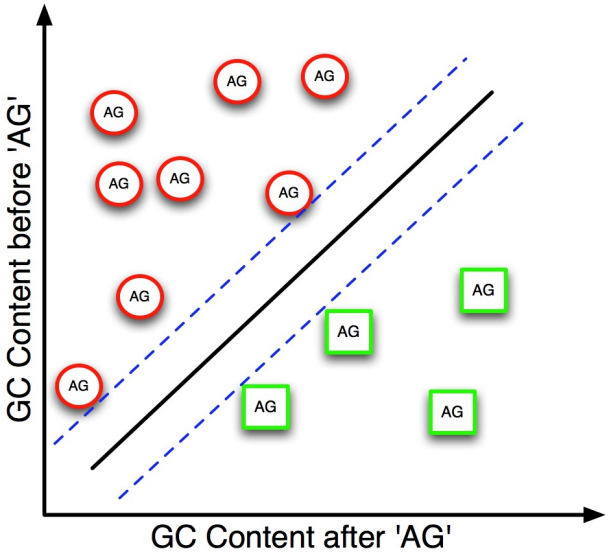
Recognition of Splice Sites

- Given: Potential acceptor splice sites

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
```

intron **exon**

- Goal: Rule that distinguishes true from false ones



Linear Classifiers with large **margin**

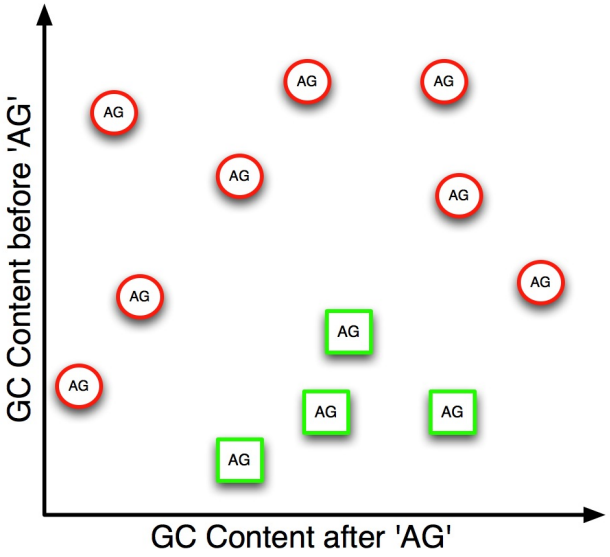
Recognition of Splice Sites

- Given: Potential acceptor splice sites

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
```

intron **exon**

- Goal: Rule that distinguishes true from false ones



More realistic problem!?

- Not linearly separable!
- Need nonlinear separation!?
- Need more features!?

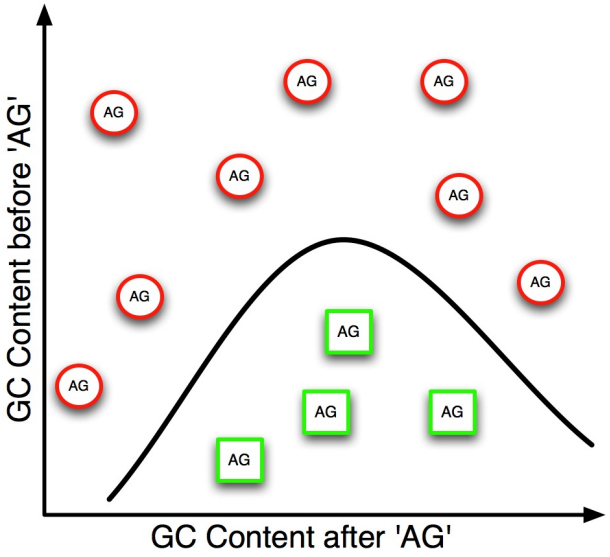
Recognition of Splice Sites

- Given: Potential acceptor splice sites

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
```

intron **exon**

- Goal: Rule that distinguishes true from false ones



More realistic problem!?

- Not linearly separable!
- **Need nonlinear separation!?**
- Need more features!?

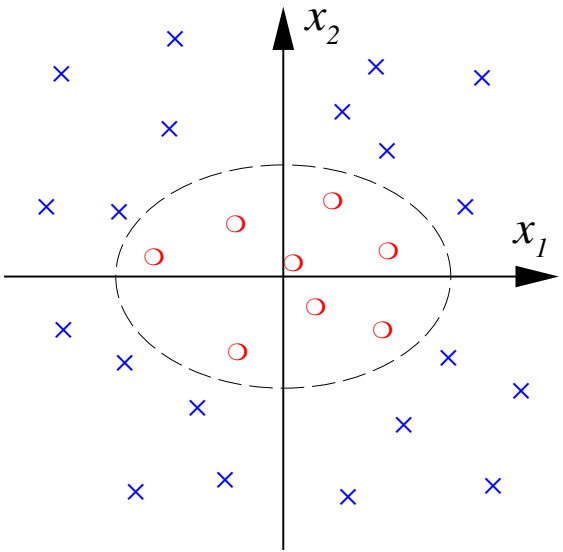
Nonlinear Algorithms in Feature Space

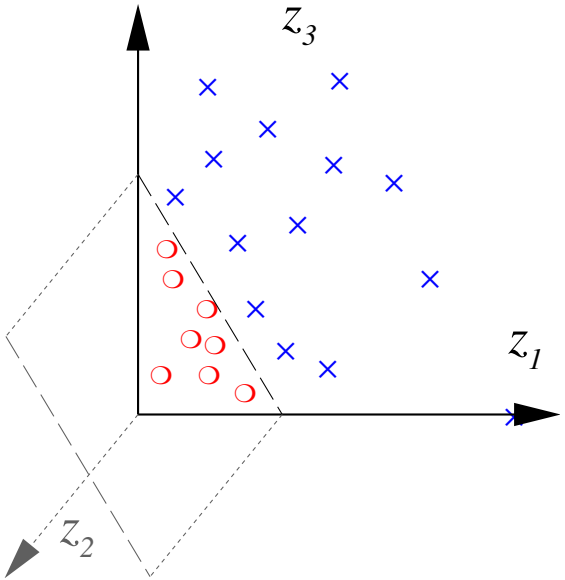
Linear separation might be not sufficient!

$\Rightarrow$ Map into a higher dimensional feature space

Example: all second order monomials

$$\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$





Kernel "Trick"

Example: $\mathbf{x} \in \mathbb{R}^2$ and $\Phi(\mathbf{x}) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$ [Boser et al., 1992]

$$\begin{aligned}
\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle &= \left\langle (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2), (y_1^2, \sqrt{2}\, y_1 y_2, y_2^2) \right\rangle \\
&= \langle (x_1, x_2), (y_1, y_2) \rangle^2 \\
&= \langle \mathbf{x}, \mathbf{y} \rangle^2 \\
&=: \mathrm{k}(\mathbf{x}, \mathbf{y})
\end{aligned}$$

- Scalar product in feature space (here $\mathbb{R}^3$) can be computed in input space (here $\mathbb{R}^2$)!
- Also works for higher orders and dimensions
  $\Rightarrow$ relatively low dimensional input spaces
  $\Rightarrow$ very high dimensional feature spaces
- works only for Mercer Kernels $\mathrm{k}(\mathbf{x}, \mathbf{y})$

Kernel $\approx$ Similarity Measure

Distance:

$$\|\Phi(\mathbf{x}) - \Phi(\mathbf{y})\| = \|\Phi(\mathbf{x})\|^2 - 2\langle\Phi(\mathbf{x}), \Phi(\mathbf{y})\rangle + \|\Phi(\mathbf{y})\|$$

Scalar product:

- If $\|\Phi(\mathbf{x})\|^2 = \|\Phi(\mathbf{y})\|^2 = 1$, then

  scalar product = $2-$distance

- Angle between vectors, i.e.

$$\frac{\langle\Phi(\mathbf{x}), \Phi(\mathbf{y})\rangle}{\|\Phi(\mathbf{x})\| \, \|\Phi(\mathbf{y})\|} = \cos(\Phi(\mathbf{x}), \Phi(\mathbf{y}))$$

Can we use just any similarity measure?

Kernology I

If k is a continuous kernel of a positive integral operator on $L_2(\mathcal{D})$ (where $\mathcal{D}$ is some compact space),

$$\int f(\mathbf{x})\mathrm{k}(\mathbf{x},\mathbf{y})f(\mathbf{y})\,d\mathbf{x}\,d\mathbf{y} \geq 0, \quad \text{for} \quad f \neq 0$$

it can be expanded as

$$\mathrm{k}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{N_F} \lambda_i \psi_i(\mathbf{x})\psi_i(\mathbf{y})$$

with $\lambda_i > 0$, and $N_F \in \mathbb{N}$ or $N_F = \infty$. In that case

$$\Phi(\mathbf{x}) := \begin{pmatrix} \sqrt{\lambda_1}\psi_1(\mathbf{x}) \\ \sqrt{\lambda_2}\psi_2(\mathbf{x}) \\ \vdots \end{pmatrix}$$

satisfies $\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle = \mathrm{k}(\mathbf{x},\mathbf{y})$ [Mercer, 1909].

Kernology I in Matrix Form

If K with $K_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$ is symmetric and positive semi-definite for all choices of $\mathbf{x}$, i.e.

$$K_{i,j} = K_{j,i} \qquad \text{and} \qquad \boldsymbol{\alpha}^\top K \boldsymbol{\alpha} \geq 0,$$

then it can be expanded as

$$\mathrm{k}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{N_F} \lambda_i \psi_i(\mathbf{x}) \psi_i(\mathbf{y})$$

with $\lambda_i > 0$, and $N_F \in \mathbb{N}$ or $N_F = \infty$. In that case

$$\Phi(\mathbf{x}) := \begin{pmatrix} \sqrt{\lambda_1}\psi_1(\mathbf{x}) \\ \sqrt{\lambda_2}\psi_2(\mathbf{x}) \\ \vdots \end{pmatrix}$$

satisfies $\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle = \mathrm{k}(\mathbf{x}, \mathbf{y})$ [Mercer, 1909].

How to construct a kernel

At least two ways to get to a kernel:

- Construct Φ and think about efficient ways to compute $\langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$
- Construct similarity measure, show positiveness and think about what it means

What can you do if kernel is not positive definite?

- Optimization problem is not convex!
- Add constant to diagonal (cheap)
- Exponentiate kernel matrix (all eigenvalues become positive)
- SVM-pairwise use similarity as features

Kernology II

Common kernels: [Vapnik, 1995, Müller et al., 2001, Schölkopf and Smola, 2002]

$$
\begin{aligned}
\text{Polynomial} \quad & \mathrm{k}(\mathbf{x},\mathbf{y}) = (\langle \mathbf{x},\mathbf{y}\rangle + c)^d \\
\text{Sigmoid} \quad & \mathrm{k}(\mathbf{x},\mathbf{y}) = \tanh(\kappa\langle \mathbf{x},\mathbf{y}\rangle) + \theta) \\
\text{RBF} \quad & \mathrm{k}(\mathbf{x},\mathbf{y}) = \exp\left(-\|\mathbf{x}-\mathbf{y}\|^2/(2\,\sigma^2)\right) \\
\text{Convex combinations} \quad & \mathrm{k}(\mathbf{x},\mathbf{y}) = \beta_1 \mathrm{k}_1(\mathbf{x},\mathbf{y}) + \beta_2 \mathrm{k}_2(\mathbf{x},\mathbf{y}) \\
\text{Normalization} \quad & \mathrm{k}(\mathbf{x},\mathbf{y}) = \frac{\mathrm{k}'(\mathbf{x},\mathbf{y})}{\sqrt{\mathrm{k}'(\mathbf{x},\mathbf{x})\mathrm{k}'(\mathbf{y},\mathbf{y})}}
\end{aligned}
$$

Notes:

- A kernel implies a mapping Φ to a feature space. In this potentially infinite dimensional space one finds a linear separation hyperplane.
- Every kernel corresponds to a regularization operator implying different smoothness properties in input space.

SVMs with kernels (Primal)

Minimize

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

Subject to

$$y_i\left(\langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle + b\right) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

Plugin: $\mathbf{w} = \sum_i \alpha_i \Phi(\mathbf{x}_i)$

Minimize

$$\frac{1}{2}\sum_{i,j}^{N}\alpha_i\alpha_j\langle\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i\left(\sum_{j=1}^{N}\alpha_j\langle\Phi(\mathbf{x}_j), \Phi(\mathbf{x}_i)\rangle + b\right) \geqslant 1 - \xi_i$$
$$\xi_i \geqslant 0$$
for all $i = 1, \ldots, N$.

SVMs with kernels (Primal)

Minimize

$$\frac{1}{2}\sum_{i,j}^{N}\alpha_i\alpha_j\langle\Phi(\mathbf{x}_i),\Phi(\mathbf{x}_j)\rangle + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i\left(\sum_{j=1}^{N}\alpha_j\langle\Phi(\mathbf{x}_j),\Phi(\mathbf{x}_i)\rangle + b\right) \geqslant 1-\xi_i$$

$$\xi_i \geqslant 0$$

for all $i = 1,\ldots,N$.

Minimize

$$\frac{1}{2}\sum_{i,j}^{N}\alpha_i\alpha_j\mathrm{k}(\mathbf{x}_i,\mathbf{x}_j) + C\sum_{i=1}^{N}\xi_i$$

Subject to

$$y_i\left(\sum_{j=1}^{N}\alpha_j\mathrm{k}(\mathbf{x}_j,\mathbf{x}_i) + b\right) \geqslant 1-\xi_i$$

$$\xi_i \geqslant 0$$

for all $i = 1,\ldots,N$.

Summary "Kernel Trick"

- Representer Theorem: $\mathbf{w} = \sum_{i=1}^{N} \alpha_i \Phi(\mathbf{x}_i)$.
- Hyperplane in $\mathcal{F}$: $y = \text{sgn}\left(\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b\right)$
- Putting things together

$$\begin{aligned} f(\mathbf{x}) &= \text{sgn}\left(\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b\right) \\ &= \text{sgn}\left(\sum_{i=1}^{N} \alpha_i \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}) \rangle + b\right) \\ &= \text{sgn}\left(\sum_{i:\alpha_i \neq 0} \alpha_i \mathrm{k}(\mathbf{x}_i, \mathbf{x}) + b\right) \end{aligned}$$

sparse!

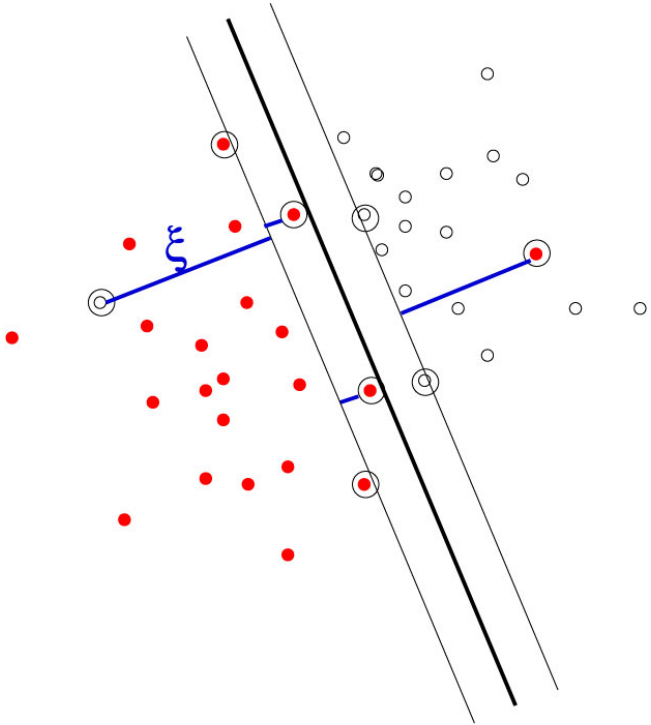
- Trick: $\mathrm{k}(\mathbf{x}, \mathbf{y}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle$, i.e. **do not use Φ, but k!**

See e.g. Vapnik [1995], Müller et al. [2001], Schölkopf and Smola [2002] for details.

Toy Examples

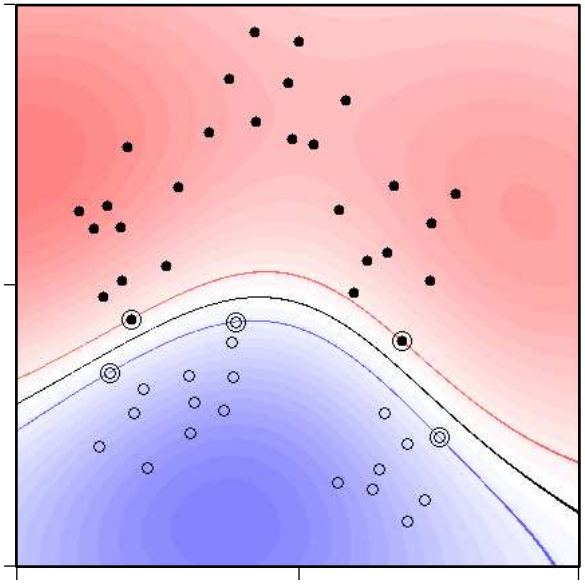
Linear kernel

$k(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle$



RBF kernel

$k(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2 / 2\sigma)$



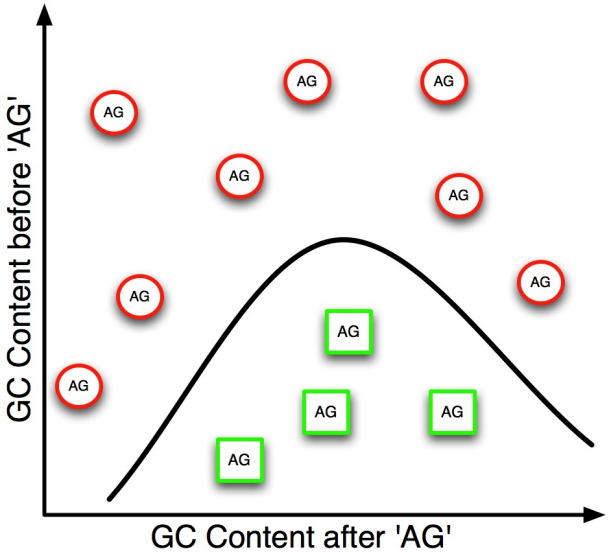
Recognition of Splice Sites

- Given: Potential acceptor splice sites

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
```

intron **exon**

- Goal: Rule that distinguishes true from false ones



More realistic problem!?

- Not linearly separable!
- Need nonlinear separation!?
- **Need more features!?**

More Features?

Some ideas:

- statistics for all four letters (or even dimer/codon usage),
- appearance of certain motifs,
- information content,
- secondary structure, . . .

Approaches:

- Manually generate a few *strong* features
  - Requires background knowledge
  - Nonlinear decisions often beneficial
- Include many potentially useful *weak* features
  - Requires more training examples

Best in practice: Combination of both

Spectrum Kernel

General idea [Leslie et al., 2002]

- For each ℓ-mer $s \in \Sigma^\ell$, the coordinate indexed by s will be the number of times s occurs in sequence x.
- Then the ℓ-spectrum feature map is

$$\Phi_\ell^{\text{Spectrum}}(\mathbf{x}) = \left(\phi_s(\mathbf{x})\right)_{s \in \Sigma^\ell}$$

- Here $\phi_s(\mathbf{x})$ is the # occurrences of s in x.
- The spectrum kernel is now the inner product in the feature space defined by this map:

$$k^{\text{Spectrum}}(\mathbf{x}, \mathbf{x}') = \langle \Phi_\ell^{\text{Spectrum}}(\mathbf{x}), \Phi_\ell^{\text{Spectrum}}(\mathbf{x}') \rangle$$

- Dimensionality: exponential in ℓ: $|\Sigma|^\ell$

Spectrum Kernel

Principle

- Spectrum kernel: Count exactly common ℓ-mers

Protein A: ILVFMC

Protein B: WLVFQC

Common 1-mers
L, V, F, C

Common 2-mers
LV
VF

Common 3-mers
LVF

- $\Phi(\mathbf{x})$ has only very few non-zero dimensions
  $\Rightarrow$ efficient kernel computations possible ($\mathcal{O}(|\mathbf{x}| + |\mathbf{x}'|)$)

Substring Kernels

General idea

- Count common substrings in two strings
- Sequences are deemed the more similar, the more common substrings they contain

Variations

- Allow for gaps
- (Include wildcards)
- Allow for mismatches
- (Include substitutions)
- Motif Kernels
- (Assign weights to substrings)

Gappy Kernel

General idea [Lodhi et al., 2002, Leslie and Kuang, 2004]

- Allow for gaps in common substrings
  $\rightarrow$ "subsequences"
- A g-mer then contributes to all its ℓ-mer subsequences

$$\phi^{\text{Gap}}_{(g,\ell)}(s) = \left(\phi_\beta(s)\right)_{\beta\in\Sigma^\ell}$$

- For a sequence x of any length, the map is then extended as

$$\phi^{\text{Gap}}_{(g,\ell)}(\mathbf{x}) = \sum_{g\text{-mers } s \text{ in } \mathbf{x}} \left(\phi^{\text{Gap}}_{(g,\ell)}(s)\right)$$

- The gappy kernel is now the inner product in feature space defined by:

$$k^{\text{Gap}}_{(g,\ell)}(\mathbf{x},\mathbf{x}') = \left\langle \Phi^{\text{Gap}}_{(g,\ell)}(\mathbf{x}), \Phi^{\text{Gap}}_{(g,\ell)}(\mathbf{x}') \right\rangle$$

Gappy Kernel

Principle

- Gappy kernel: Count common ℓ-subsequences of g-mers

g=3, l=2

Protein A: ILVFMC

ILV: IL, IV, LV
LVF: LV, VF, LF
VFM: VF, FM, VM
FMC: FM, FC, MC

Protein B: WLVFQC

WLV: WL, LV, WV
LVF: LV, VF, LF
VFQ: VF, FQ, VQ
FQC: FQ, QC, FC

Mismatch Kernel

General idea [Leslie et al., 2003]

- Do not enforce strictly exact matches
- Define mismatch neighborhood of ℓ-mer s with up to m mismatches:

$$\phi^{\text{Mismatch}}_{(l,m)}(s) = (\phi_\beta(s))_{\beta \in \Sigma^\ell}$$

- For a sequence x of any length, the map is then extended as

$$\phi^{\text{Mismatch}}_{(l,m)}(\mathbf{x}) = \sum_{\ell\text{-mers } s \text{ in } \mathbf{x}} (\phi^{\text{Mismatch}}_{(l,m)}(s))$$

- The mismatch kernel is now the inner product in feature space defined by:

$$k^{\text{Mismatch}}_{(l,m)}(\mathbf{x}, \mathbf{x}') = \left\langle \Phi^{\text{Mismatch}}_{(l,m)}(\mathbf{x}), \Phi^{\text{Mismatch}}_{(l,m)}(\mathbf{x}') \right\rangle$$

Mismatch Kernel

Principle

- Mismatch kernel: Count common ℓ-mers with max. m mismatches

Protein A: ILVFMC

Protein B: WLVFQC

Common 3-mers
with max. 1 mismatch:

ILV - WLV
LVF - LVF
VFM - VFQ
FMC - FQC

Motif kernels

General idea

- Conserved motif in sequences indicate structural and functional characteristics
- Model sequence as feature vector representing motifs
- i-th vector component is 1 $\Leftrightarrow$ $\mathbf{x}$ contains i-th motif

Motif databases

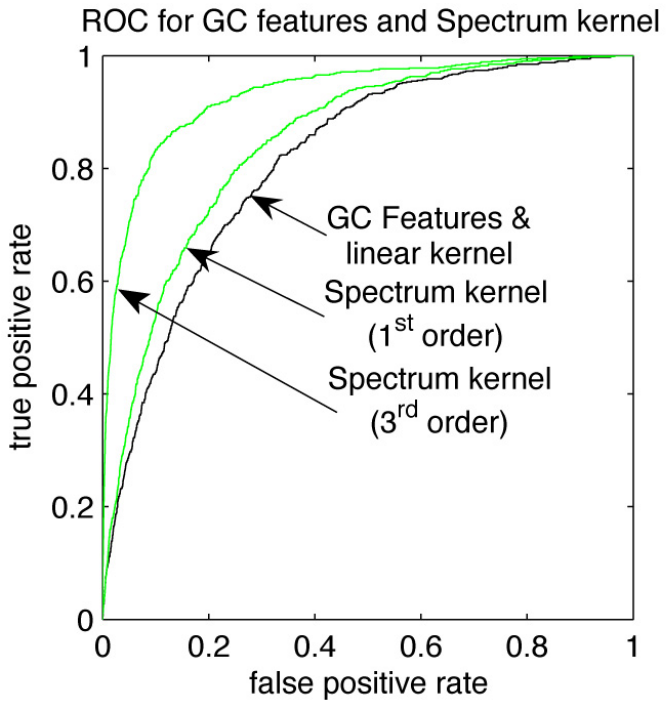
- Protein: Pfam, PROSITE, ...
- DNA: Transfac, Jaspar, ...
- RNA: Rfam, Structures, Regulatory sequences, ...

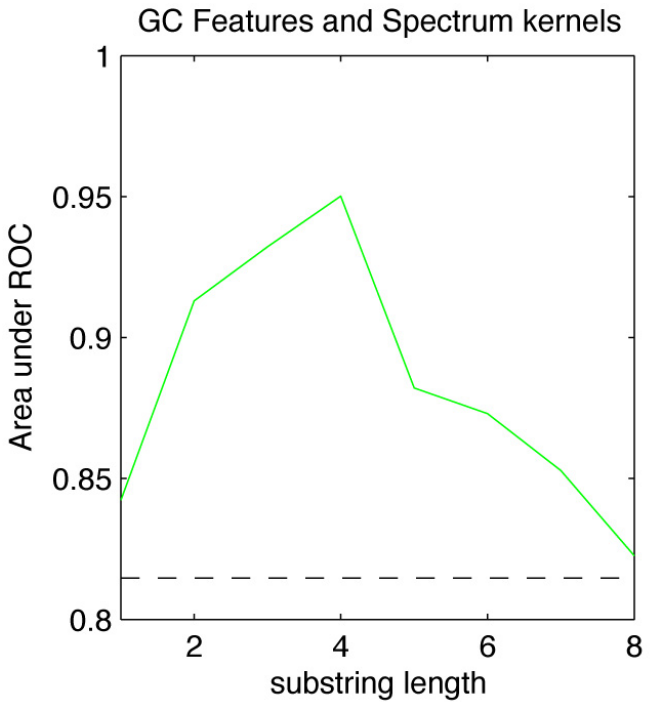
Generated by

- manual construction/prior knowledge
- multiple sequence alignment (do not use test set!)

Simulation Example

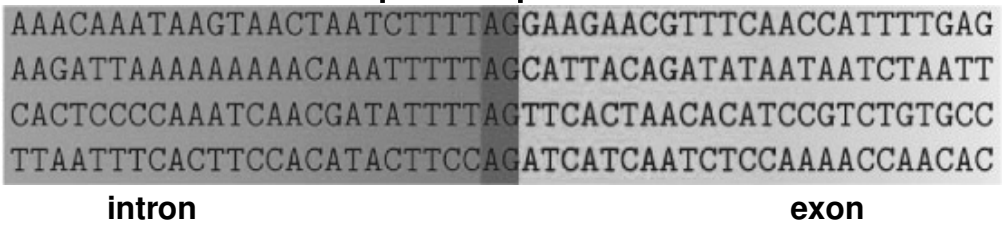
- Linear Kernel on GC-content features
- Spectrum kernel $\mathrm{k}_{\ell}^{\mathrm{Spectrum}}(\mathbf{x}, \mathbf{x}')$





Position Dependence

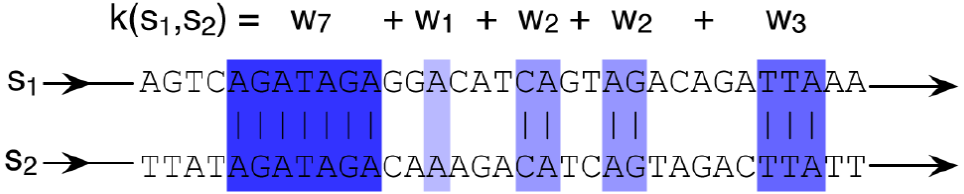
- Given: Potential acceptor splice sites



- Goal: Rule that distinguishes true from false ones
- Position of Motif is important ('T' rich just before 'AG')
- Spectrum Kernel is *blind* w.r.t. positions
- New kernels for sequences with constant length
  - Substring kernel per position (sum over positions)
  - Can detect motifs at specific positions
    - weak if positions vary
    - Extension: allow "shifting"

Weighted Degree Kernel

Weighted degree kernel compares two sequences by identifying the largest matching blocks which contribute depending on their length [Rätsch and Sonnenburg, 2004].

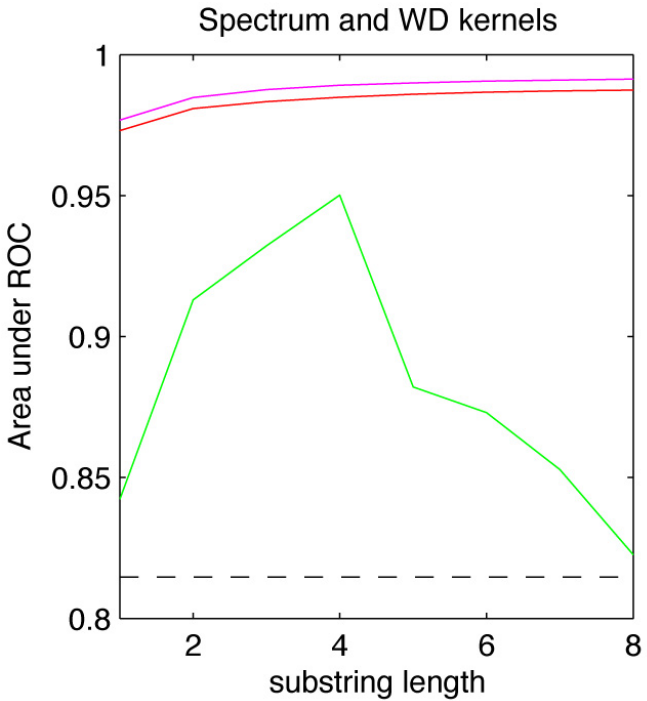


Equivalent to a mixture of spectrum kernels (up to order ℓ) at every position for appropriately chosen weights w (depending on ℓ).

Weighted degree kernel w/ shifts allows matching subsequences to be offset from each other [Rätsch et al., 2005].



Substring Kernel Comparison



- Linear Kernel on GC-content features
- Spectrum kernel
- Weighted Degree Kernel
- Weighted Degree Kernel with shifts

Remark: Higher order substring kernels typically exploit that correlations appear locally and not between arbitrary parts of the sequence (other than e.g. the polynomial kernel).

Fisher Kernel

General idea [Jaakkola et al., 2000, Tsuda et al., 2002a]

- Combine probabilistic models and SVMs
- Best-paper award at ISMB 1999

Sequence representation

- Arbitrary length sequences s
- Probabilistic model $p(s|\boldsymbol{\theta})$ (e.g. HMM, PSSMs)
- Maximum Likelihood estimate $\boldsymbol{\theta}^* \in \mathbb{R}^d$
- Transformation into *Fisher score* features $\Phi(s) \in \mathbb{R}^d$
  - $\Phi(s) = \dfrac{\partial p(s|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$
  - Describes contribution of every parameter to $p(s|\boldsymbol{\theta})$
- $\mathrm{k}(s, s') = \langle \Phi(s), \Phi(s') \rangle$

Example: Fisher Kernel on PSSMs

- Fixed length sequences $s \in \Sigma^N$
- PSSMs: $p(s|\boldsymbol{\theta}) = \prod_{i=1}^{N} \theta_{i,s_i}$
- Fisher scores features: $(\Phi(s))_{i,\sigma} = \frac{\mathrm{d}p(s|\boldsymbol{\theta})}{\mathrm{d}\theta_{i,\sigma}} = \mathrm{Id}(s_i = \sigma)$
- Kernel: $\mathrm{k}(s, s') = \langle \Phi(s), \Phi(s') \rangle = \sum_{i=1}^{N} \mathrm{Id}(s_i = s'_i)$
- Identical to WD kernel with order 1

Note: Marginalized-count kernels [Tsuda et al., 2002b] can be understood as a generalization of Fisher kernels.

Pairwise comparison kernels

General idea [Liao and Noble, 2002]

- Employ empirical kernel map on Smith-Waterman/Blast scores

Advantage

- Utilizes decades of practical experience with Blast

Disadvantage

- High computational cost ($O(N^3)$)

Alleviation

- Employ Blast instead of Smith-Waterman
- Use a smaller subset for empirical map

Local Alignment Kernel

In order to compute the score of an alignment, one needs

- substitution matrix $S \in \mathbb{R}^{\Sigma \times \Sigma}$
- gap penalty $g : \mathbb{N} \to \mathbb{R}$

An alignment π is then scored as follows:

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

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

Smith-Waterman score (not positive definite)

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

Local Alignment Kernel [Vert et al., 2004]

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

Local Alignment Kernel

In order to compute the score of an alignment, one needs

- substitution matrix $S \in \mathbb{R}^{\Sigma \times \Sigma}$
- gap penalty $g : \mathbb{N} \rightarrow \mathbb{R}$

An alignment π is then scored as follows:

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

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

Smith-Waterman score (not positive definite)

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

Local Alignment Kernel [Vert et al., 2004]

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

Kernel Summary

- Kernel extend SVMs to nonlinear decision boundaries, while keeping the simplicity of linear classification
- Good kernel design is important for every single data analysis task
- String kernels perform computations in very high dimensional feature space
- Kernels on strings can be:
  - Substring kernels (e.g. Spectrum & WD kernel)
  - Based on probabilistic methods (e.g. Fisher Kernel)
  - Derived from similarity measures (e.g. Alignment kernels)
- Not mentioned: Kernels on graphs, images, structures
- Application goes far beyond computational biology

Next Lectures

Invited Lectures:

- November 15
  Koji Tsuda (MPI for biol. Cybernetics, Tübingen)
  Marginalized Kernels (generalization of the Fischer kernel), Diffusion kernels for networks
- November 22
  Karsten Borgwardt (University of Munic)
  More kernels on graphs, random walks on graphs
- November 29
  Alexander Zien (FML & MPI for biol. Cybernetics)
  Semi-supervised Learning
- December 6
  Ulrike von Luxburg (MPI for biol. Cybernetics)
  Unsupervised Learning (clustering, (kernel) PCA, etc)

References

B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152, 1992.

C. Cortes and V.N. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

T.S. Jaakkola, M. Diekhans, and D. Haussler. A discriminative framework for detecting remote protein homologies. *J. Comp. Biol.*, 7: 95–114, 2000.

C. Leslie and R. Kuang. Fast string kernels using inexact matching for protein sequences. *Journal of Machine Learning Research*, 5: 1435–1455, 2004.

C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 564–575, 2002.

C. Leslie, E. Eskin, J. Weston, and W.S. Noble. Mismatch string kernels for discriminative protein classification. *Bioinformatics*, 20(4), 2003.

L. Liao and W.S. Noble. Combining pairwise sequence similarity and support vector machines. In *Proc. 6th Int. Conf. Computational Molecular Biology*, pages 225–232, 2002.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Roy. Soc. London*, A 209:415–446, 1909.

K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

G. Rätsch and S. Sonnenburg. Accurate splice site detection for *Caenorhabditis elegans*. In K. Tsuda B. Schoelkopf and J.-P. Vert, editors, *Kernel Methods in Computational Biology*. MIT Press, 2004.

G. Rätsch, S. Sonnenburg, and B. Schölkopf. RASE: recognition of alternatively spliced exons in *C. elegans*. *Bioinformatics*, 21(Suppl. 1):i369–i377, June 2005.

B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

K. Tsuda, M. Kawanabe, G. Rätsch, S. Sonnenburg, and K.R. Müller. A new discriminative kernel from probabilistic models. *Neural Computation*, 14:2397–2414, 2002a.

K. Tsuda, T. Kin, and K. Asai. Marginalized kernels for biological sequences. *Bioinformatics*, 18:268S–275S, 2002b.

V.N. Vapnik. *The nature of statistical learning theory*. Springer Verlag, New York, 1995.

J.-P. Vert, H. Saigo, and T. Akutsu. Local alignment kernels for biological sequences. In K. Tsuda B. Schoelkopf and J.-P. Vert, editors, *Kernel Methods in Computational Biology*. MIT Press, 2004.