Advanced Methods for Sequence Analysis

Gunnar Rätsch

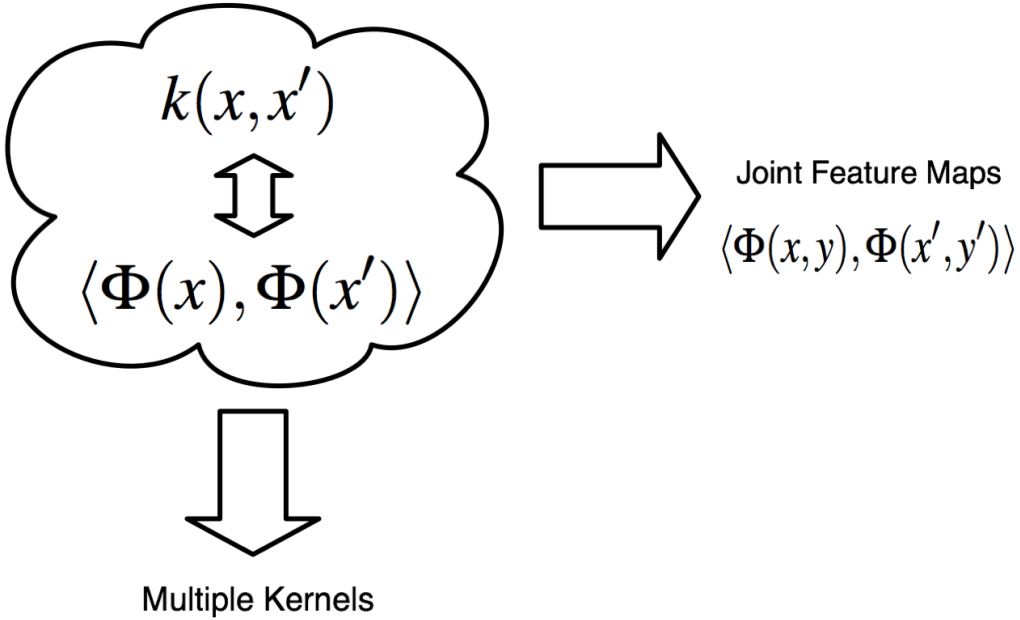
Friedrich Miescher Laboratory, Tübingen

Vorlesung WS 2007/2008
Eberhard-Karls-Universität Tübingen

29 January, 2008

http://www.fml.mpg.de/raetsch/lectures/amsa07

Generalizing kernels



- Learning structured output spaces
- Finding the optimal combination of kernels

Multiple Kernel Learning (MKL)

Possible solution We can add the two kernels, that is

$$k(\mathbf{x}, \mathbf{x}') := k_{sequence}(\mathbf{x}, \mathbf{x}') + k_{structure}(\mathbf{x}, \mathbf{x}').$$

Better solution We can mix the two kernels,

$$k(\mathbf{x}, \mathbf{x}') := (1-t)k_{sequence}(\mathbf{x}, \mathbf{x}') + tk_{structure}(\mathbf{x}, \mathbf{x}'),$$

where t should be estimated from the training data.

In general: use the data to find best convex combination.

$$k(\mathbf{x}, \mathbf{x}') = \sum_{p=1}^{K} \beta_p k_p(\mathbf{x}, \mathbf{x}').$$

Applications

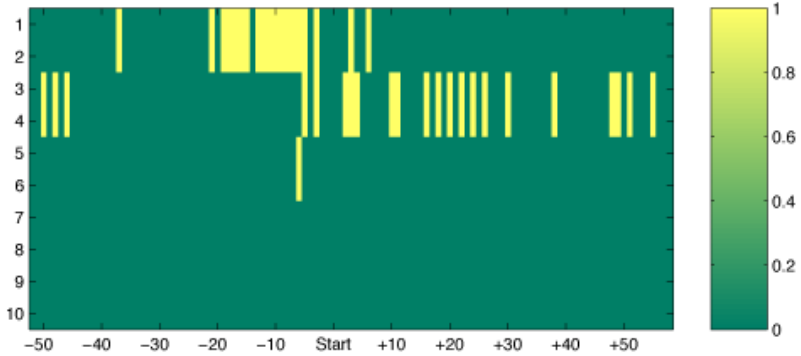
- Heterogeneous data
- Improving interpretability

Method for Interpreting SVMs

- Weighted Degree kernel: linear comb. of LD kernels

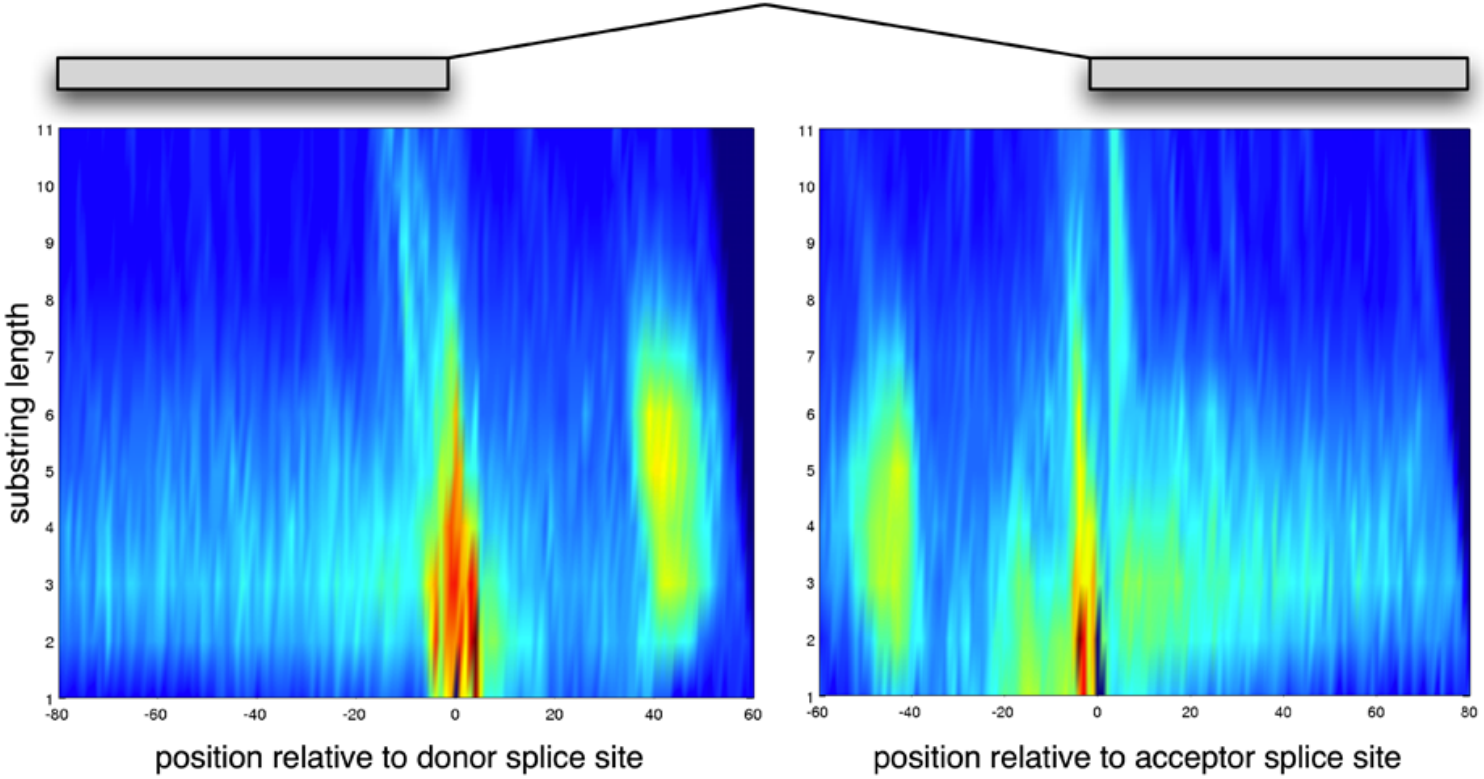
$$k(\mathbf{x}, \mathbf{x}') = \sum_{d=1}^{D} \sum_{l=1}^{L-d+1} \gamma_{l,d} \mathbf{I}(\mathbf{u}_{l,d}(\mathbf{x}) = \mathbf{u}_{l,d}(\mathbf{x}'))$$

- Example: Classifying splice sites



See ? for more details.

POIMs for Splicing



Color-coded importance scores of substrings near splice sites. Long substrings are important upstream of the donor and downstream of the acceptor site [?]

Structured Output Spaces

Learning Task

For a set of labeled data, we predict the label.

Difference from multiclass

The set of possible labels $\mathcal{Y}$ may be very large or hierarchical.

Joint kernel on $\mathcal{X}$ and $\mathcal{Y}$

We define a joint feature map on $\mathcal{X} \times \mathcal{Y}$, denoted by $\Phi(\mathbf{x}, y)$. Then the corresponding kernel function is

$$k((\mathbf{x}, y), (\mathbf{x}', y')) := \langle \Phi(\mathbf{x}, y), \Phi(\mathbf{x}', y') \rangle.$$

For multiclass

For normal multiclass classification, the joint feature map decomposes and the kernels on $\mathcal{Y}$ is the identity, that is

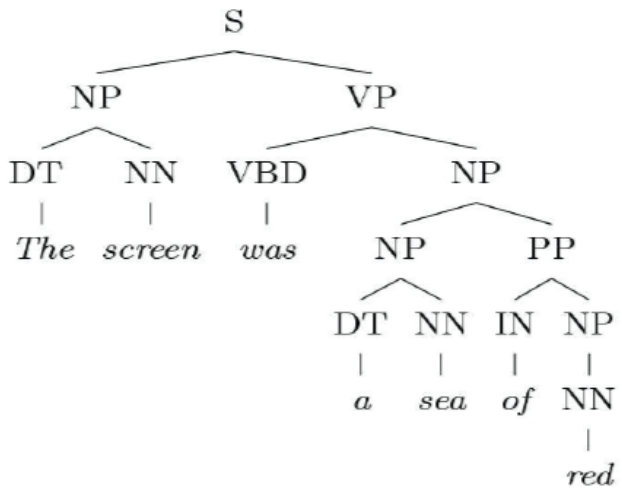
$$k((\mathbf{x}, y), (\mathbf{x}', y')) := [[y = y']] k(\mathbf{x}, \mathbf{x}').$$

Context Free Grammar Parsing

x

The screen was
a sea of red

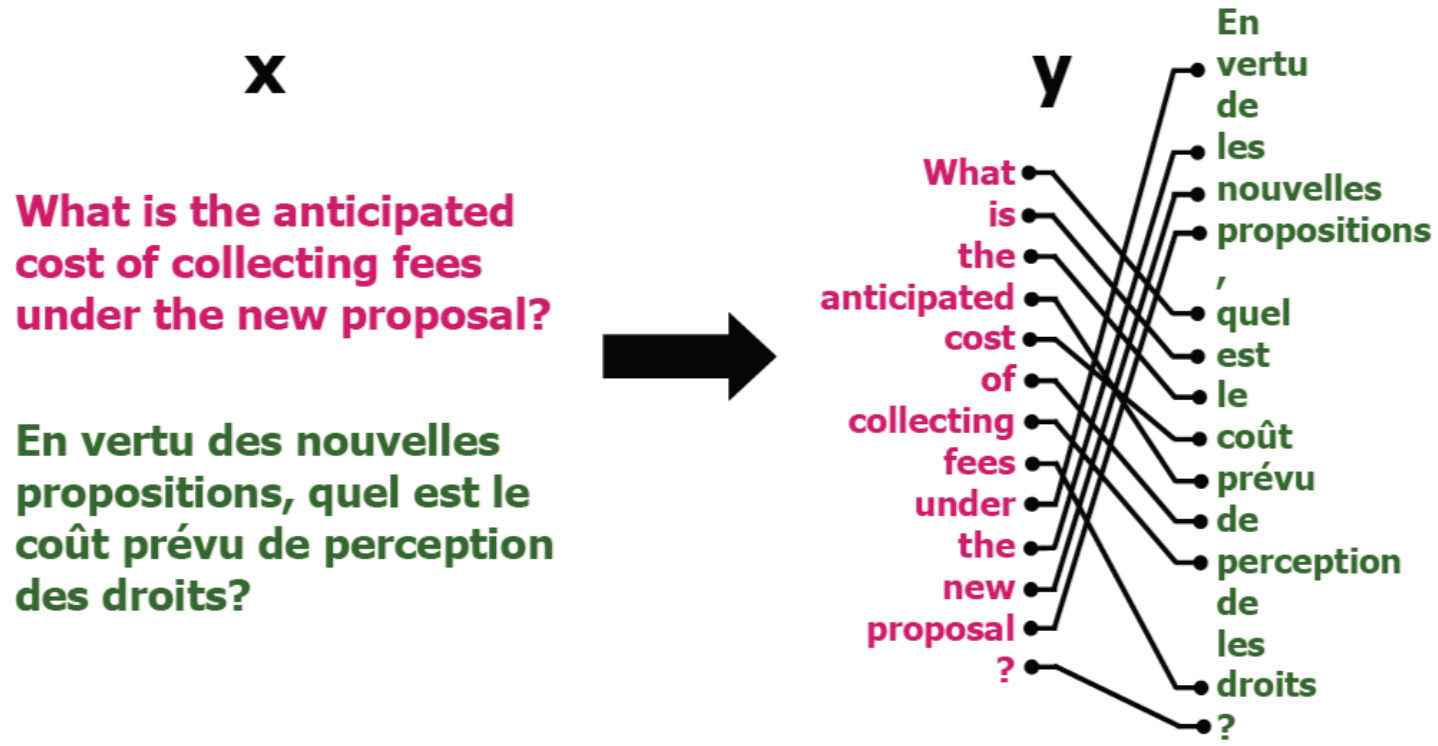
y



Recursive Structure

From Klein & Taskar, ACL'05 Tutorial

Bilingual Word Alignment



Combinatorial Structure

From Klein & Taskar, ACL'05 Tutorial

Handwritten Letter Sequences

x y



Sequential Structure

From Klein & Taskar, ACL'05 Tutorial

Label Sequence Learning

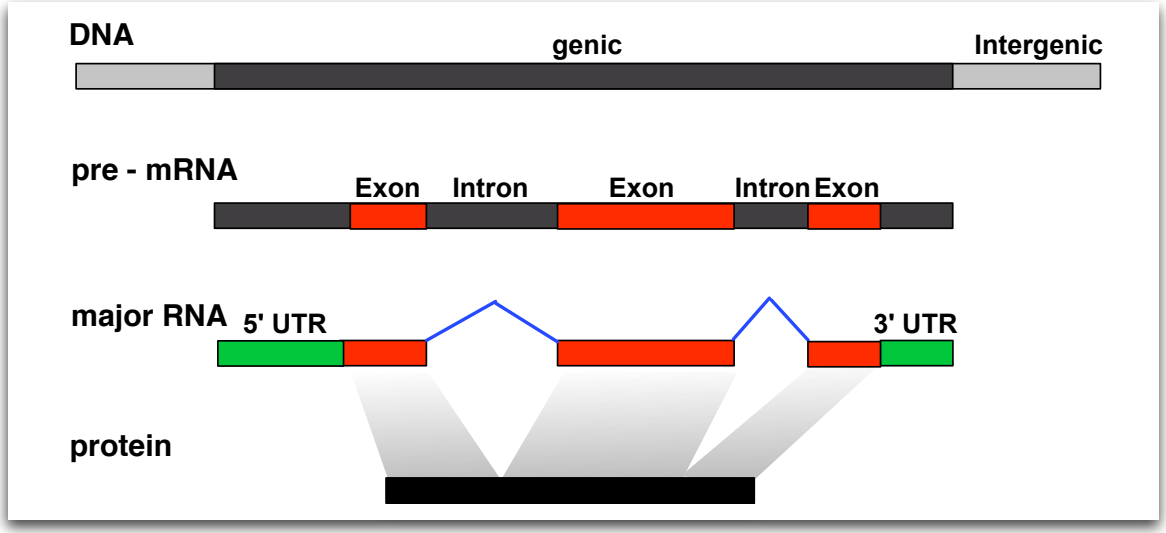
- Given: observation sequence
- Problem: predict corresponding state sequence
- Often: several subsequent positions have the same state
  $\Rightarrow$ state sequence defines a “segmentation”
- Example 1: Protein Secondary Structure Prediction



Residue Sequence:	NWVLSTAADMQGVVTDGMASFLDKD
Secondary Structure:	LLEEEELLLLHHHHHHHHHHLHHHL

Label Sequence Learning

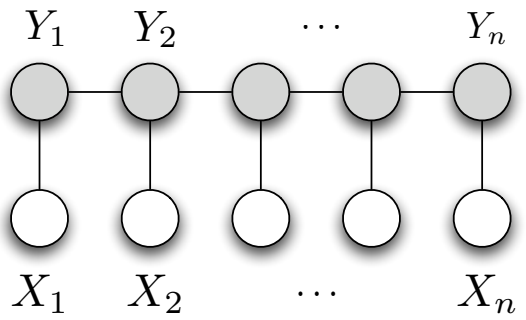
- Given: observation sequence
- Problem: predict corresponding state sequence
- Often: several subsequent positions have the same state
  $\Rightarrow$ state sequence defines a “segmentation”
- Example 2: Gene Finding



Generative Models

- Hidden Markov Models [?]
  - State sequence treated as Markov chain
  - No direct dependencies between observations
  - Example: first-order HMM (simplified)

$$p(\mathbf{x}, \mathbf{y}) = \prod_i p(x_i|y_i)p(y_i|y_{i-1})$$



  - Efficient dynamic programming (DP) algorithms

Decoding *via* Dynamic Programming

$$\begin{aligned}\log p(\mathbf{x}, \mathbf{y}) &= \sum_i (\log p(x_i|y_i) + \log p(y_i|y_{i-1})) \\ &= \sum_i g(y_{i-1}, y_i, x_i)\end{aligned}$$

with $g(y_{i-1}, y_i, x_i) = \log p(x_i|y_i) + \log p(y_i|y_{i-1})$.

Problem: Given sequence $\mathbf{x}$, find sequence $\mathbf{y}$ such that $\log p(\mathbf{x}, \mathbf{y})$ is maximized, i.e. $\mathbf{y}^* = \operatorname{argmax}_{\mathbf{y} \in \mathcal{Y}^n} \log p(\mathbf{x}, \mathbf{y})$

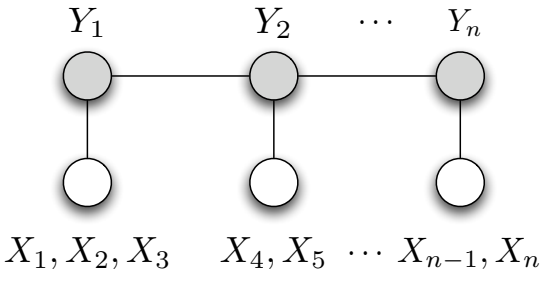
Dynamic Programming Approach:

$$V(i, y) := \begin{cases} \max_{y' \in \mathcal{Y}} (V(i-1, y') + g(y', y, x_i)) & i > 1 \\ 0 & \text{otherwise} \end{cases}$$

Generative Models

- Generalized Hidden Markov Models
  = Hidden Semi-Markov Models
  - Only one state variable per segment
  - Allow non-independence of positions within segment
  - Example: first-order Hidden Semi-Markov Model

$$p(x,y) = \prod_j p(\underbrace{(x_{i(j-1)+1}, \ldots, x_{i(j)})}_{\mathbf{x}_j} | y_j) p(y_j | y_{j-1})$$



(use with care)

- Use generalization of DP algorithms of HMMs

Decoding *via* Dynamic Programming

$$\begin{aligned}\log p(\mathbf{x},\mathbf{y}) &= \prod_j p((x_{i(j)},\ldots,x_{i(j+1)-1})|y_j)p(y_j|y_{j-1}) \\ &= \sum_j g(y_{i-1},y_i,\underbrace{(x_{i(j-1)+1},\ldots,x_{i(j)})}_{\mathbf{x}_j})\end{aligned}$$

with $g(y_{j-1},y_j,\mathbf{x}_j) = \log p(\mathbf{x}_j|y_j) + \log p(y_j|y_{j-1})$.

Problem: Given sequence $\mathbf{x}$, find sequence $\mathbf{y}$ such that $\log p(\mathbf{x},\mathbf{y})$ is maximized, i.e. $\mathbf{y}^* = \operatorname{argmax}_{\mathbf{y}\in\mathcal{Y}^*} \log p(\mathbf{x},\mathbf{y})$

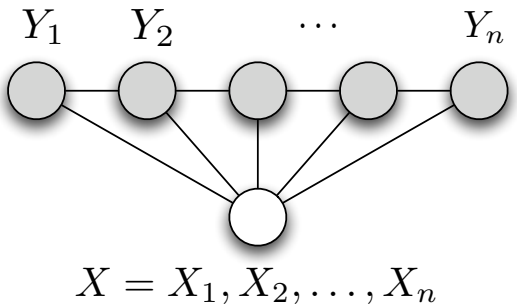
Dynamic Programming Approach: $V(i,y) :=$

$$\begin{cases} \max_{y'\in\mathcal{Y},d=1,\ldots,i-1} (V(i-d,y') + g(y',y,\mathbf{x}_{i-d+1,\ldots,i})) & i>1 \\ 0 & \text{otherwise} \end{cases}$$

Discriminative Models

- Conditional Random Fields [?]
  - conditional prob. $p(y|x)$ instead of joint prob. $p(x, y)$

$$p(y|x, \mathbf{w}) = \frac{1}{Z(x, \mathbf{w})} \exp(\langle \mathbf{w}, \Phi(x, y) \rangle)$$



  - can handle non-independent input features
- Semi-Markov Conditional Random Fields
  - introduce segment feature functions
  - dynamic programming algorithms exist

Max-Margin Structured Output Learning

- Learn function $f(\mathbf{y}|\mathbf{x})$ scoring segmentations $\mathbf{y}$ for $\mathbf{x}$
- Maximize $f(\mathbf{y}|\mathbf{x})$ w.r.t. $\mathbf{y}$ for prediction:

$$\operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}^*} f(\mathbf{y}|\mathbf{x})$$

- Given N sequence pairs $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)$ for training
- Determine f such that there is a large margin between true and wrong segmentations

$$\begin{aligned} \min_f \quad & C\sum_{n=1}^{N}\xi_n + \mathbf{P}[f] \\ \text{w.r.t.} \quad & f(\mathbf{y}_n|\mathbf{x}_n) - f(\mathbf{y}|\mathbf{x}_n) \geq 1 - \xi_n \\ & \text{for all } \mathbf{y}_n \neq \mathbf{y} \in \mathcal{Y}^*, n = 1, \ldots, N \end{aligned}$$

- Exponentially many constraints!

Joint Feature Map

Recall the kernel trick

For each kernel, there exists a corresponding feature mapping $\Phi(\mathbf{x})$ on the inputs such that $k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$.

Joint kernel on $\mathcal{X}$ and $\mathcal{Y}$

We define a joint feature map on $\mathcal{X} \times \mathcal{Y}$, denoted by $\Phi(\mathbf{x}, y)$. Then the corresponding kernel function is

$$k((\mathbf{x}, y), (\mathbf{x}', y')) := \langle \Phi(\mathbf{x}, y), \Phi(\mathbf{x}', y') \rangle.$$

For multiclass

For normal multiclass classification, the joint feature map decomposes and the kernels on $\mathcal{Y}$ is the identity, that is

$$k((\mathbf{x}, y), (\mathbf{x}', y')) := [[y = y']] k(\mathbf{x}, \mathbf{x}').$$

SO Learning with kernels

- Assume $f(\mathbf{y}|\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$, where $\mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \in \mathcal{F}$
- Use ℓ_2 regularizer: $\mathbf{P}[f] = \|w\|^2$

$$\begin{aligned} \min_{\mathbf{w} \in \mathcal{F}, \boldsymbol{\xi} \in \mathbb{R}^N} \quad & C \sum_{n=1}^{N} \xi_n + \|w\|^2 \\ \text{w.r.t.} \quad & \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}_n) - \Phi(\mathbf{x}, \mathbf{y}) \rangle \geq 1 - \xi_n \\ & \text{for all } \mathbf{y}_n \neq \mathbf{y} \in \mathcal{Y}^*, n = 1, \ldots, N \end{aligned}$$

- Linear classifier that separates true from wrong labelling
- Dual: Define $\Phi_{n,\mathbf{y}} := \Phi(\mathbf{x}_n, \mathbf{y}_n) - \Phi(\mathbf{x}_n, \mathbf{y})$

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{n,\mathbf{y}} \alpha_{n,\mathbf{y}} - \sum_{n,\mathbf{y}} \sum_{n',\mathbf{y}'} \alpha_{n,\mathbf{y}} \alpha_{n',\mathbf{y}'} \langle \Phi_{n,\mathbf{y}}, \Phi_{n',\mathbf{y}'} \rangle \\ \text{w.r.t.} \quad & \alpha_{n,\mathbf{y}} \geq 0, \sum_{\mathbf{y}} \alpha_{n,\mathbf{y}} \leq C \text{ for all } n \text{ and } \mathbf{y} \end{aligned}$$

Kernels

- Recall: $\Phi_{n,\mathbf{y}} := \Phi(\mathbf{x}_n, \mathbf{y}_n) - \Phi(\mathbf{x}_n, \mathbf{y})$
- Then

$$\begin{aligned}\langle \Phi_{n,\mathbf{y}}, \Phi_{n',\mathbf{y}'} \rangle &= \langle \Phi(\mathbf{x}_n, \mathbf{y}_n) - \Phi(\mathbf{x}_n, \mathbf{y}), \Phi(\mathbf{x}_{n'}, \mathbf{y}_{n'}) - \Phi(\mathbf{x}_{n'}, \mathbf{y}') \\ &= k((\mathbf{x}_n, \mathbf{y}_n), (\mathbf{x}_{n'}, \mathbf{y}_{n'})) - k((\mathbf{x}_n, \mathbf{y}_n), (\mathbf{x}_{n'}, \mathbf{y}')) - \\ &\quad -k((\mathbf{x}_n, \mathbf{y}), (\mathbf{x}_{n'}, \mathbf{y}_{n'})) + k((\mathbf{x}_n, \mathbf{y}), (\mathbf{x}_{n'}, \mathbf{y})),\end{aligned}$$

where

$$k((\mathbf{x}_n, \mathbf{y}), (\mathbf{x}_{n'}, \mathbf{y}')) := \langle \Phi(\mathbf{x}_n, \mathbf{y}), \Phi(\mathbf{x}_{n'}, \mathbf{y}') \rangle$$

- Kernel learning (almost) as usual

Special Case: only two "structures"

- Assume $f(\mathbf{y}|\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$, where $\mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \in \mathcal{F}$

$$\begin{aligned} \min_{\mathbf{w} \in \mathcal{F}, \boldsymbol{\xi} \in \mathbb{R}^N} \quad & C \sum_{n=1}^{N} \xi_n + \|w\|^2 \\ \text{w.r.t.} \quad & \langle \mathbf{w}, \Phi(\mathbf{x}, y_n) - \Phi(\mathbf{x}, 1 - y_n) \rangle \geq 1 - \xi_n \\ & \text{for all } n = 1, \ldots, N \end{aligned}$$

- Dual: Define $\Phi_n := \Phi(\mathbf{x}_n, y_n) - \Phi(\mathbf{x}_n, 1 - y_n)$

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_n \alpha_n - \sum_n \sum_{n'} \alpha_n \alpha_{n'} \langle \Phi_n, \Phi_{n'} \rangle \\ \text{w.r.t.} \quad & \alpha_n \geq 0, \alpha_n \leq C \text{ for all } n \end{aligned}$$

- Equivalent to standard 2-class SVM

Optimization

- Optimization problem too big (dual as well)

$$\begin{aligned} \min_{\mathbf{w}\in\mathcal{F},\boldsymbol{\xi}} \quad & C\sum_{n=1}^{N}\xi_n + \|w\|^2 \\ \text{w.r.t.} \quad & \langle \mathbf{w}, \Phi(\mathbf{x},\mathbf{y}_n) - \Phi(\mathbf{x},\mathbf{y})\rangle \geq 1-\xi_n \\ & \text{for all } \mathbf{y}_n \neq \mathbf{y} \in \mathcal{Y}^*, n = 1,\ldots,N \end{aligned}$$

- One constraint per example and wrong labeling
- Iterative solution
  - Begin with small set of wrong labellings
  - Solve reduced optimization problem
  - Find labellings that violate constraints
  - Add constraints, resolve
- Guaranteed Convergence

How to find violated constraints?

- Constraint

$$\langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}_n) - \Phi(\mathbf{x}, \mathbf{y}) \rangle \geq 1 - \xi_n$$

- Find labeling $\mathbf{y}$ that maximizes

$$\langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

- Use Dynamic Programming Decoding

$$\mathbf{y} = \underset{\mathbf{y} \in \mathcal{Y}^*}{\operatorname{argmax}} \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

  (DP only works if Φ has certain decomposition structure)
- If $\mathbf{y} = \mathbf{y}_n$, then compute second best labeling as well
- If constraint is violated, then add to optimization problem

Algorithm

1. $\mathcal{Y}_n^1 = \emptyset$, for $n = 1, \ldots, N$
2. Solve

$$(\mathbf{w}^t, \boldsymbol{\xi}^t) = \underset{\mathbf{w} \in \mathcal{F}, \boldsymbol{\xi}}{\operatorname{argmin}} \quad C \sum_{n=1}^{N} \xi_n + \|w\|^2$$

$$\text{w.r.t.} \quad \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}_n) - \Phi(\mathbf{x}, \mathbf{y}) \rangle \geq 1 - \xi_n$$

$$\text{for all } \mathbf{y}_n \neq \mathbf{y} \in \mathcal{Y}_n^t, n = 1, \ldots, N$$

3. Find violated constraints ($n = 1, \ldots, N$)

$$\mathbf{y}_n^t = \underset{\mathbf{y}_n \neq \mathbf{y} \in \mathcal{Y}^*}{\operatorname{argmax}} \langle \mathbf{w}^t, \Phi(\mathbf{x}, \mathbf{y}) \rangle$$

   If $\langle \mathbf{w}^t, \Phi(\mathbf{x}, \mathbf{y}_n) - \Phi(\mathbf{x}, \mathbf{y}_n^t) \rangle < 1 - \xi_n^t$, set $\mathcal{Y}_n^{t+1} = \mathcal{Y}_n^t \cup \{\mathbf{y}_n^t\}$
4. If violated constraint exists then go to 2
5. Otherwise terminate $\Rightarrow$ Optimal solution

Loss functions

- So far 0-1-loss with slacks: If $\mathbf{y} \neq \mathbf{y}$, then prediction is wrong, but it does not matter how wrong
- Introduce loss function on labellings $\ell(\mathbf{y}, \mathbf{y}')$, e.g.
  - How many segments are wrong or missing
  - How different are the segments, etc

Loss functions

- So far 0-1-loss with slacks: If $\mathbf{y} \neq \mathbf{y}$, then prediction is wrong, but it does not matter how wrong
- Introduce loss function on labellings $\ell(\mathbf{y}, \mathbf{y}')$, e.g.
  - How many segments are wrong or missing
  - How different are the segments, etc
- Extend optimization problem (Margin rescaling):

$$\begin{aligned} \min_{\mathbf{w}\in\mathcal{F},\boldsymbol{\xi}} \quad & C\sum_{n=1}^{N}\xi_n + \|w\|^2 \\ \text{w.r.t.} \quad & \langle \mathbf{w}, \Phi(\mathbf{x},\mathbf{y}_n) - \Phi(\mathbf{x},\mathbf{y})\rangle \geq \ell(\mathbf{y},\mathbf{y}') - \xi_n \\ & \text{for all } \mathbf{y}_n \neq \mathbf{y} \in \mathcal{Y}^*, n = 1,\ldots,N \end{aligned}$$

- Finding violated constraints ($n = 1,\ldots,N$)

$$\mathbf{y}_n^t = \operatorname*{argmax}_{\mathbf{y}_n\neq\mathbf{y}\in\mathcal{Y}^*} \langle \mathbf{w}^t, \Phi(\mathbf{x},\mathbf{y})\rangle + \ell(\mathbf{y},\mathbf{y}_n)$$

Loss functions

- So far 0-1-loss with slacks: If $\mathbf{y} \neq \mathbf{y}$, then prediction is wrong, but it does not matter how wrong
- Introduce loss function on labellings $\ell(\mathbf{y}, \mathbf{y}')$, e.g.
  - How many segments are wrong or missing
  - How different are the segments, etc
- Extend optimization problem (Slack rescaling):

$$\begin{aligned} \min_{\mathbf{w}\in\mathcal{F},\boldsymbol{\xi}} \quad & C\sum_{n=1}^{N}\xi_n + \|w\|^2 \\ \text{w.r.t.} \quad & \langle \mathbf{w}, \Phi(\mathbf{x},\mathbf{y}_n) - \Phi(\mathbf{x},\mathbf{y})\rangle \geq 1 - \xi_n/\ell(\mathbf{y},\mathbf{y}') \\ & \text{for all } \mathbf{y}_n \neq \mathbf{y} \in \mathcal{Y}^*, n = 1,\ldots,N \end{aligned}$$

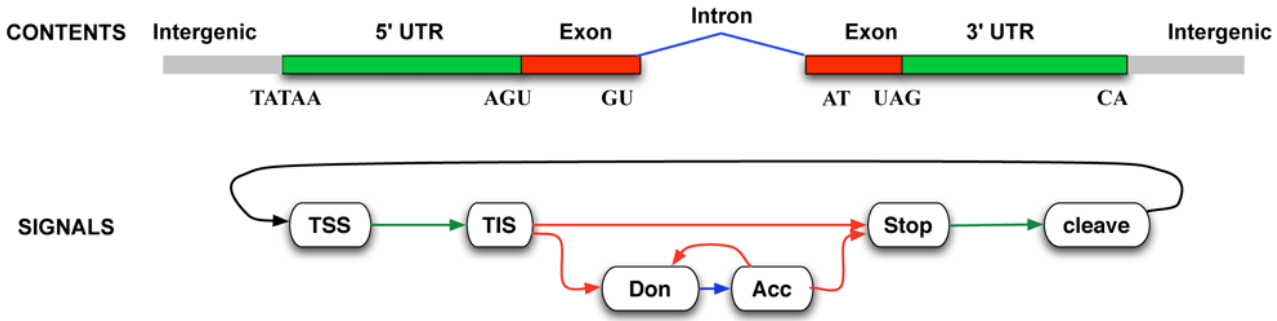
- Finding violated constraints more difficult

Problems

- Optimization may require many iterations
- Number of variables increases linearly
- When using kernels, solving optimization problems can become infeasible
- Evaluation of $\langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$ in Dynamic programming can be very expensive
  - Optimization and decoding become too expensive
- Approximation algorithms useful
- Decompose problem
  - First part uses kernels, can be precomputed
  - Second part without kernels and only combines ingredients

Gene Finding as Segmentation Task

- Nodes correspond to sequence signals
  - Depend on recognition of signals on the DNA
- Transitions correspond to segments
  - Depend on length or sequence properties of segment
- Markovian on segment level, non-Markovian within segments
  - Allows efficient decoding and modeling of segment lengths



Learning to Predict Segmentations

- Learn function $f(\mathbf{y}|\mathbf{x})$ scoring segmentations $\mathbf{y}$ for $\mathbf{x}$
- f considers signal, content and length information
- Maximize $f(\mathbf{y}|\mathbf{x})$ w.r.t. $\mathbf{y}$ for prediction: $\underset{\mathbf{y}}{\operatorname{argmax}}\, f(\mathbf{y}|\mathbf{x})$
- Determine f such that there is a large margin between true and wrong segmentations

$$\begin{aligned} \min_{f} \quad & \sum_{n=1}^{N} \xi_n + \mathbf{P}[f] \\ \text{w.r.t.} \quad & f(\mathbf{y}_n|\mathbf{x}_n) - f(\mathbf{y}|\mathbf{x}_n) \geq 1 - \xi_n \\ & \text{for all } \mathbf{y} \neq \mathbf{y}_n, n = 1, \ldots, N \end{aligned}$$

- Use approximation (Rätsch & Sonnenburg, NIPS'06)
  - Train signal and content detectors separately
  - Combine in large margin fashion

Signal and Content Sensors

SVMs to recognize signals:

- Transcription start and cleavage site, polyA site
- Translation initiation site and stop codon
- Donor and acceptor splice sites

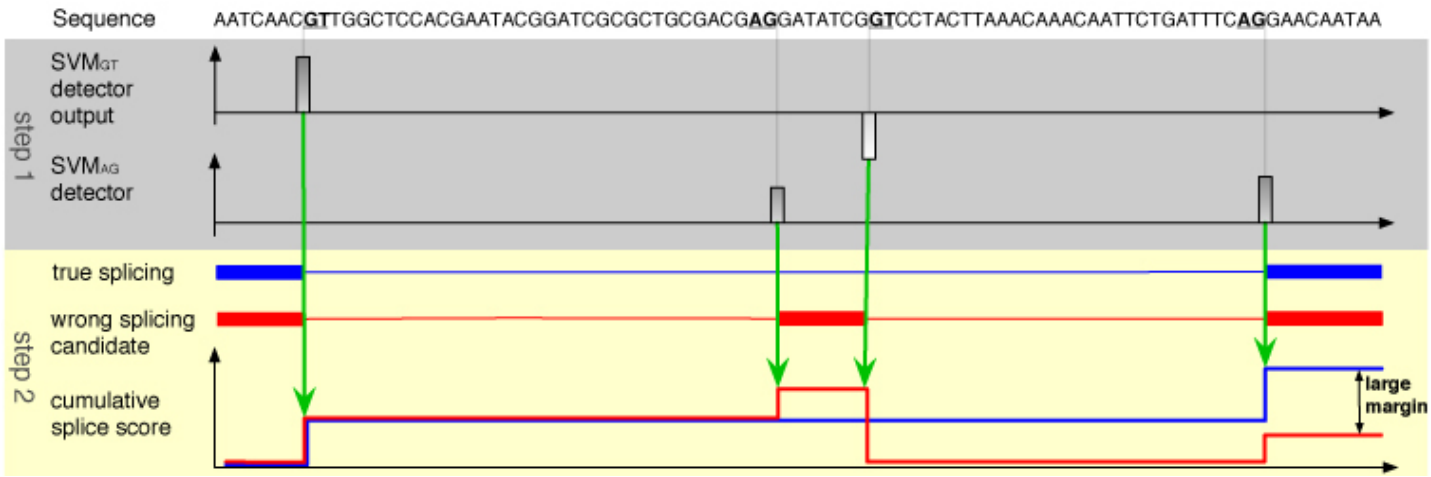
Every non-signal position is a negative $\Rightarrow$ unbalanced problem
Use Weighted Degree Kernel & Spectrum kernel

SVMs to recognize contents:

- exons & utr
- introns
- intergenic

Train one type against all others. Use Spectrum kernel.

Large Margin Combination (simplified)

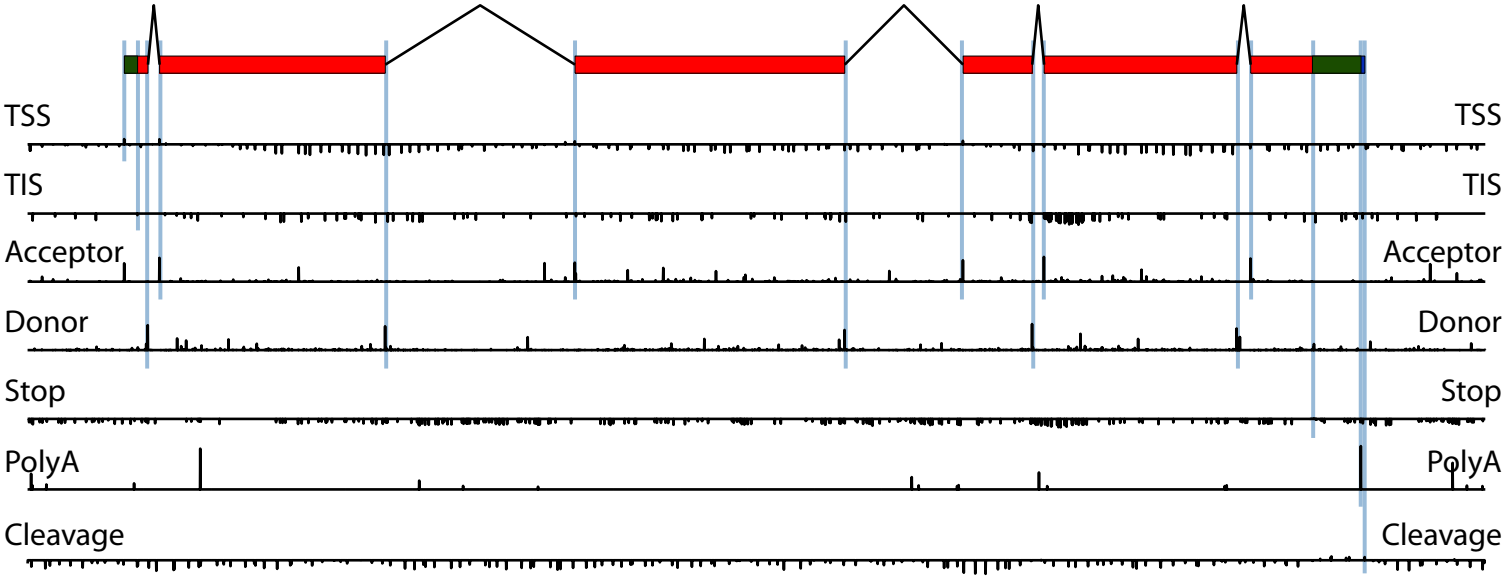


- Simplified Model: Score for splice form $\mathbf{y} = \{(p_j, q_j)\}_{j=1}^{J}$:

$$f(\mathbf{y}) := \underbrace{\sum_{j=1}^{J-1} S_{GT}(f_j^{GT}) + \sum_{j=2}^{J} S_{AG}(f_j^{AG})}_{\text{Splice signals}} + \underbrace{\sum_{j=1}^{J-1} S_{L_I}(p_{j+1} - q_j) + \sum_{j=1}^{J} S_{L_E}(q_j - p_j)}_{\text{Segment lengths}}$$

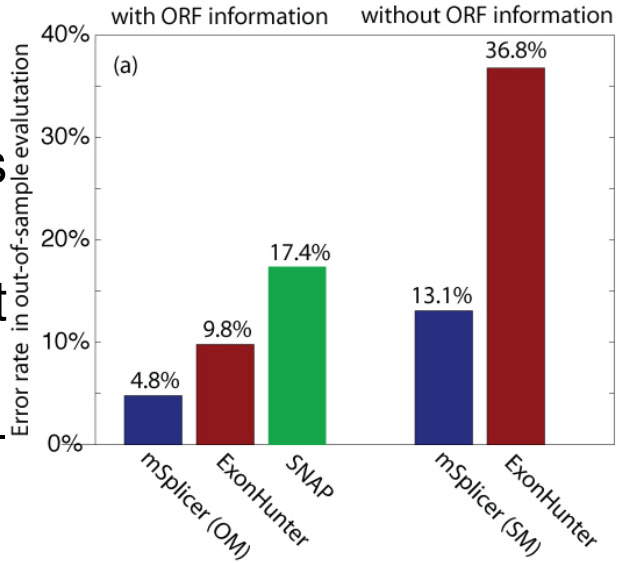
- Tune free parameters (in functions $S_{GT}, S_{AG}, S_{L_E}, S_{L_I}$) by solving **linear program** using training set with known splice forms

Example


TSS
TSS
TIS
TIS
Acceptor
Acceptor
Donor
Donor
Stop
Stop
PolyA
PolyA
Cleavage
Cleavage

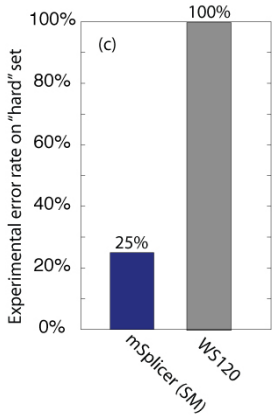
Results I (Splice forms only)

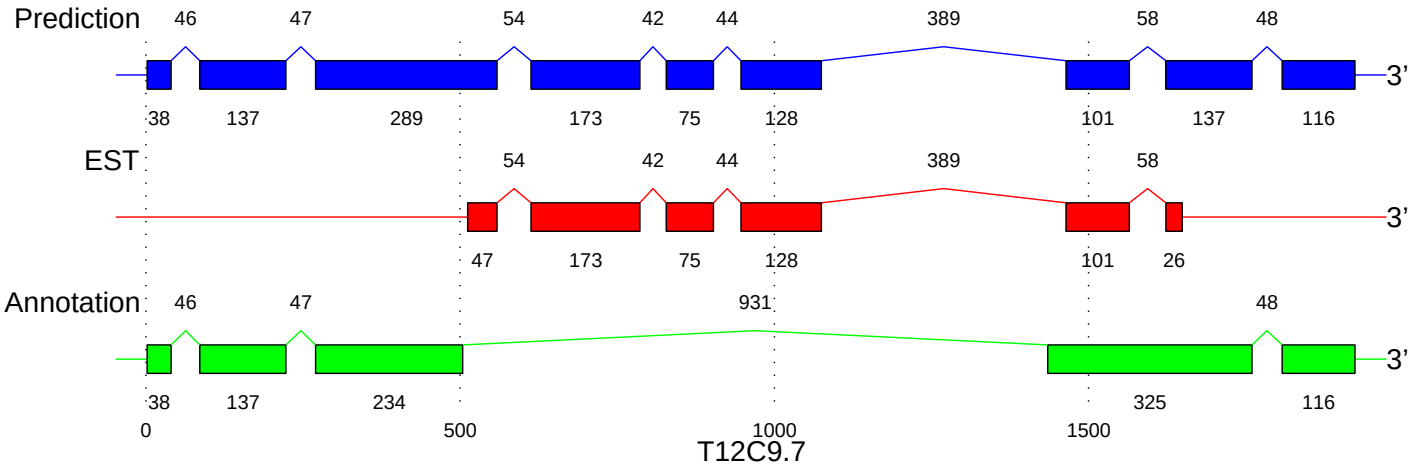
- ≈3,800 gene models derived from cDNAs and ESTs
  - 60% for training and validation
  - 40% for testing (exclude alt. spliced genes)
- Out-of-sample accuracy (≈1100 gene models):
  - Splice form error rate
    - 4.8% (coding)
    - 13.1% (mixed)
  - Much lower error rates than state-of-the-art
    - Exonhunter (Brejova et al., ISMB'05)
    - Snap (Korf, BMC Bioinformatics 2004)



Results II (Splice forms only)

- Validation by RT-PCR & direct sequencing
  - Consider 20 disagreeing cases
  - Annotation was never correct
  - 75% of our predictions were correct





Summary

- Joint feature maps for inputs and outputs
  - Good for multiclass and structure prediction
- Related to (generalized) HMMs
  - Don't estimate $p(\mathbf{x}, \mathbf{y})$ but predict $\mathbf{y}$ given $\mathbf{x}$
- Result in large optimization problems
  - Can be solved iteratively
  - But still too large for medium size problems
- Decomposition of the Problem
  - Use efficient kernel-based two-class detectors
  - Integrate without kernels
- Beats HMM based approaches in Gene finding :-)