mGene - a sophisticated Bioinformatics Application

(Transcriptionsstart, Splice Sites, PolyA Site Prediction)
Structure Learning

A. Zien[*,‡], G. Schweikert[*,‡], G. Zeller[*,‡], C. S. Ong[*,‡], F. De Bona[*,‡], P. Philips[*,‡], K. -R. Müller[†,+], S. Sonnenburg[†], G. Rätsch[*,‡]

[‡] Friedrich Miescher Laboratory of the Max Planck Society, Tübingen
[*] Max Planck Institute for Biological Cybernetics, Tübingen
[+] Technical University Berlin, [†] Fraunhofer FIRST.IDA, Berlin

FIRST
Fraunhofer Institut Rechnerarchitektur und Softwaretechnik
MAX-PLANCK-GESELLSCHAFT

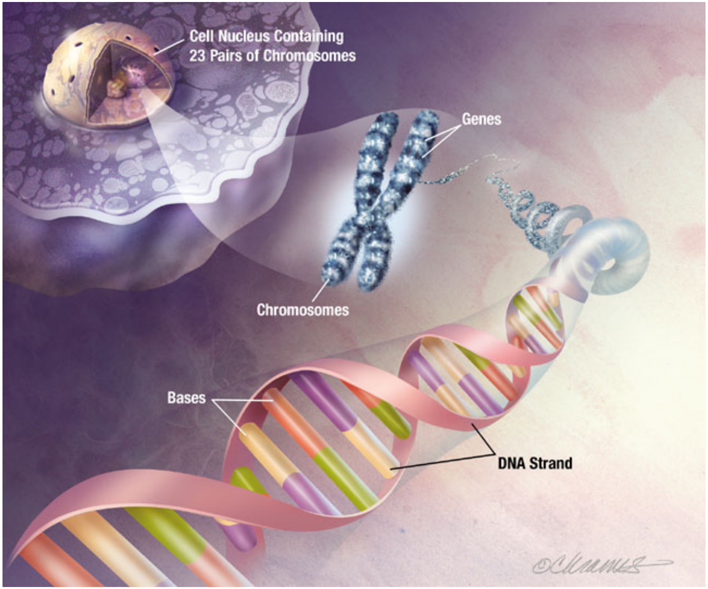
Bioinformatics - Benefits

- Molecular medicine
    - More drug targets
    - Personalised medicine
    - Preventative medicine
    - Gene therapy
- Microbial genome applications
    - Waste cleanup
    - Climate change
    - Alternative energy sources
    - Biotechnology
    - Antibiotic resistance
    - Forensic analysis of microbes
    - Evolutionary studies
- Agriculture
    - Insect resistance
    - Improve nutritional quality
    - Grow crops in poorer soils and that are drought resistant

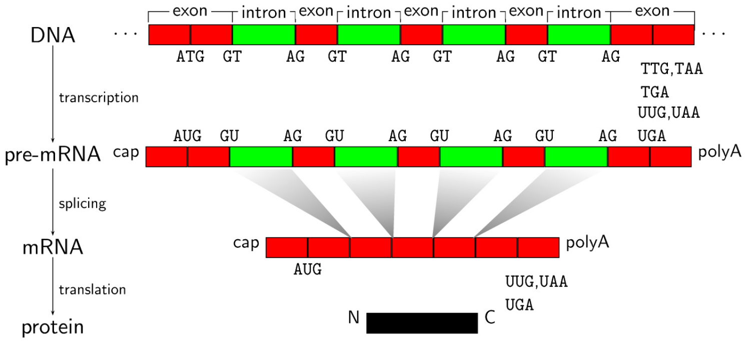
Bioinformatics - Applications

- In Cell
  - which genes are on / off ?
  - in which tissue ?
  - under which conditions ?
- **Sequence Analysis on DNA/RNA $\Leftarrow$ in this Lecture**
  - **locate sequences (genes, start, stop, splice sites,. . .)**
  - detect properties
  - how do individuals of same species differ (SNP's)
  - conservation
  - functional elements
- on Proteins
  - determine structure
  - determine function
  - find protein of similar functions
  - find binding sites (protein-protein, protein-dna)

Bioinformatics - The Genome

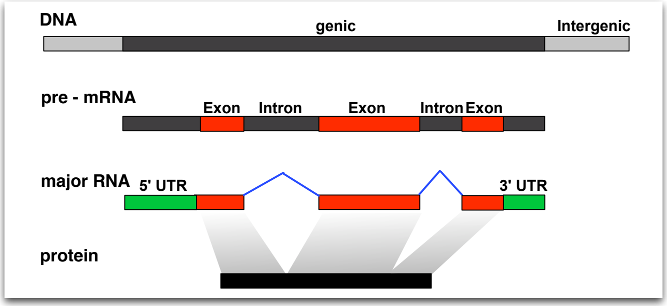


Bioinformatics - From DNA to Protein



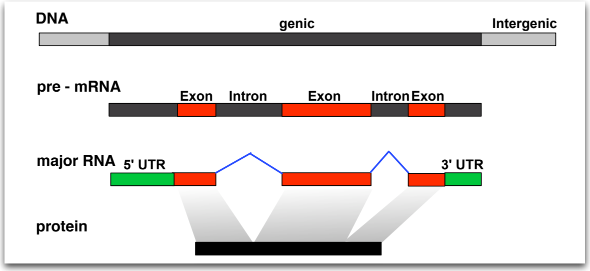
Finding Genes - I

- What is a Gene ?
  - A segment on DNA that codes for a certain property (protein).
  - Proteins control everything, Enzymes (catalyze; involved in metabolism, DNA replication/repair, RNA synthesis)..., Cell signaling (Insulin), ligand binding (Haemoglobin),...



Finding Genes - II

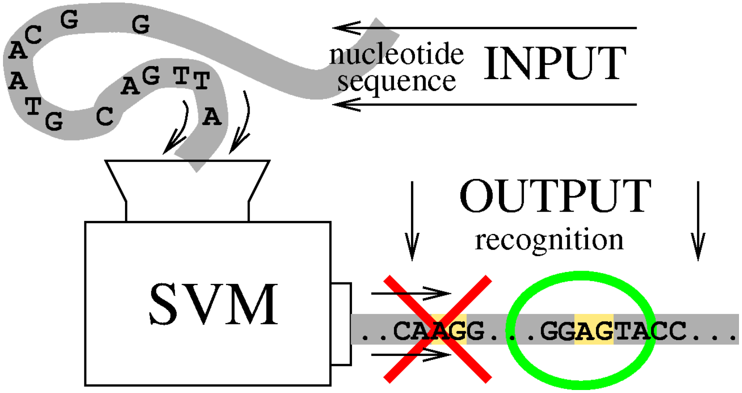
- Sites to detect
  - Gene has a **transcription start**, **transcription end** - only part from ATG. . . TAA,. . . is transcribed ⇒ pre-mRNA
  - Only **exons** code for protein, inserted **introns** are cut out in splicing ⇒ mRNA
  - Gene has a **translation start** and **translation end** - that part is translated to ⇒ Protein



Finding Genes - III

- Requirements
  - human genome (all DNA) has 3 billion base pairs - huge!!
  - method needs to be fast + fit in memory
- 2-step approach:
  1. Detect Signals (focus on splice site and transcription start site prediction) - $\Rightarrow$ SVM on *sliding windows*
     - define kernels on strings
     - (spectrum kernel, weighted degree kernel)
  2. Learn Structure/Gene Segmentation (complex task)

1st pass - Proceedure for splice sites



Preparing data

- Collecting data for training and evaluation is a complex, non-trivial task (half the work)
- two kinds, one for 1st pass (2-class classification positive/negative data); one for 2nd pass (correct segmentations)
- we assume data is given (others have done it for us :-)

2-class problem: solve with SVMs Classifier

$$f(\mathbf{x}) = \mathrm{sign}\left(\sum_{i=1}^{N} y_i \alpha_i \mathsf{k}(\mathbf{x}, \mathbf{x}_i) + b\right)$$

$\Rightarrow$ **How to design the kernel ?**

Data Classes

- Position Independent (e.g. Which Tissue? Promoter Region)

  AAACAAAAACGTAACTAATCTTTTAGAGAGAACGTTTCAACCATTTTGAG
  AAGATTAACTCATCACAGATTTCATTACATACAGATATAATTCAAAAATT
  CACTCCCCAAATCAACGATATTTAAAAATCACTAACACATCCGTCTGTGC

  - Task: separate DNA strings, '-' class random ACGT, '+' class contains 'AAAAA' motif

- Position Dependent (e.g. Splice Site Classification)

  AAACAAATAAGTAACTAATCTTTTAAGAAGAACGTTTCAACCATTTTGAG
  AAGATTAAAAAAAAACAAATTTTTAACATTACAGATATAATAATCTAATT
  CACTCCCCAAATCAACGATATTTTAATTCACTAACACATCCGTCTGTGCC

  - Task: separate DNA strings, '-' class random ACGT, '+' class 'AA' in the middle

- Mixture Position Dependent/Independent (e.g. Promoter)

  AAACAAATAAGTAACTAATCTTTTAAAGAGAACGTTTCAACCATTTTGAG
  AAGATTAAAAAAAAACAAATTTCATTAAATACAGATATAATAATCTAATT
  CACTCCCCAAATCAACGATATTTAAATTTCACTAACACATCCGTCTGTGC

  - Task: separate DNA strings, '-' class random 'ACGT', '+' class 'AAA' in the middle shifted ± 15

Spectrum Kernel

To make use of position independent motifs:

- Idea: like bag of words kernel (text classification) but for Bioinformatics (words are now strings of length k (k-mers))
  - count k-mers in sequence A and sequence B.
  - Spectrum Kernel is sum of product of counts (for same k-mer)

Example $k = 3$:

x `AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG`

x' `TACCTAATTATGAAATTAAATTTCAGTGTGCTGATGGAAACGGAGAAGTC`

3-mer	AAA	AAC	...	CCA	CCC	...	TTT
# in **x**	2	4	...	1	0	...	3
# in **x**′	3	1	...	0	0	...	1

$$\mathrm{k}(\mathbf{x}, \mathbf{x}') = 2 \cdot 3 + 4 \cdot 1 + \ldots 1 \cdot 0 + 0 \cdot 0 \ldots 3 \cdot 1$$

Weighted Degree Kernel

To make use of position dependent motifs:

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

- L length of the sequence $\mathbf{x}$
- d maximal "match length" taken into account
- $\mathbf{u}_{k,l}(\mathbf{x})$ subsequence of length k at position l of sequence $\mathbf{x}$

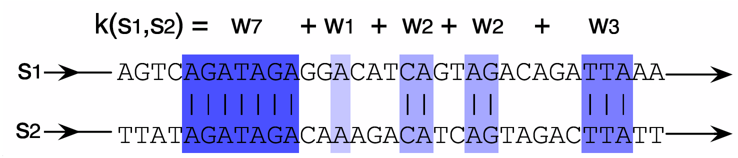
Example degree $d = 3$:

```
       x  AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
#1-mers   .|.|.|||.|..||.|.|..|||.||...|....|...|||......|..
#2-mers   .....||.....|.......||..|.............||..........
#3-mers   .....|..............|...|.............|...........
      x'  TACCTAATTATGAAATTAAATTTCAGTGTGCTGATGGAAACGGAGAAGTC
```

$$\mathrm{k}(\mathbf{x},\mathbf{x}') = \beta_1 \cdot 21 + \beta_2 \cdot 8 + \beta_3 \cdot 4$$

Weighted Degree Kernel

- for weighting we use $\beta_k = 2\frac{d-k+1}{d(d+1)}$.
- effort is $O(L \cdot d)$
- Speedup Idea: Reduce effort to $O(L)$ by finding matching "blocks"


k(S1,S2) = W7 + W1 + W2 + W2 + W3
S1
AGTCAGATAGAGGACATCAGTAGACAGATTAAA
S2
TTATAGATAGACAAAGACATCAGTAGACTTATT

Exercise: Show that WD kernel and its "block" formulation are equivalent

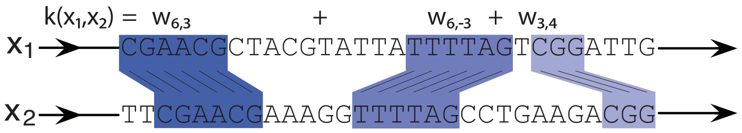
Weighted Degree Kernel with *shifts*

To make use of partially position-dependent motifs:

- If sequence is slightly mutated (Insertion,Deletion) WD kernel fails.
- Extension: Allow for some positional variance (shifts $S(l)$)

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{d} \beta_k \sum_{l=1}^{L-k+1} \gamma_l \sum_{\substack{s=0 \\ s+l \leq L}}^{S(l)} \delta_s \, \mu_{k,l,s,\mathbf{x}_i,\mathbf{x}_j},$$

$$\mu_{k,l,s,\mathbf{x}_i,\mathbf{x}_j} = \mathbf{I}(\mathbf{u}_{k,l+s}(\mathbf{x}_i) = \mathbf{u}_{k,l}(\mathbf{x}_j)) + \mathbf{I}(\mathbf{u}_{k,l}(\mathbf{x}_i) = \mathbf{u}_{k,l+s}(\mathbf{x}_j)),$$


k(x1,x2) = w6,3 + w6,-3 + w3,4
x1 CGAACGCTACGTATTATTTTAGTCGGATTG
x2 TTCGAACGAAAGGTTTTAGCCTGAAGACGG

The Final Signal and Content Sensors

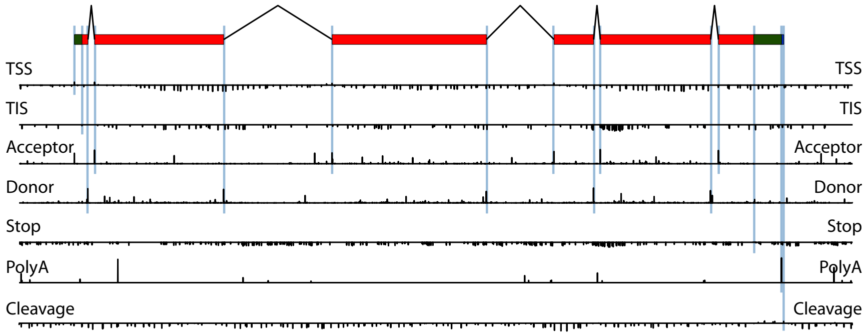
- Exon vs. Intron - **Spectrum Kernel**
- splice sites - **Weighted Degree Kernel**
- transcription start, transcription stop - **Weighted Degree Kernel with shifts**

Perform Model Selection:

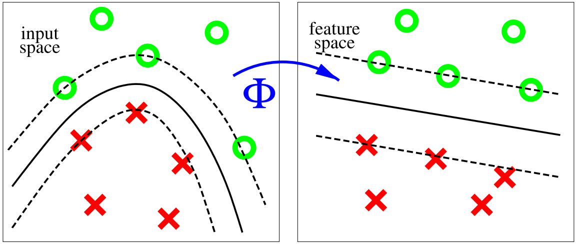
- window length
- k-mer length (spectrum kernel), degree,shift (WD-kernel)
- SVM regularization parameter C
- ...
- takes a long time (cluster)

We now have Signal and content sensors

Example


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

What did we learn ?



- SVM decision function in kernel feature space:

$$f(\mathbf{x}) = \sum_{i=1}^{N} y_i \alpha_i \underbrace{\Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}_i)}_{=k(\mathbf{x},\mathbf{x}_i)} + b \quad (1)$$

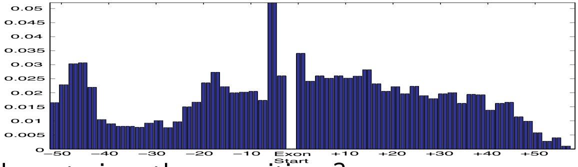
- learned parameters α by solving quadratic optimization problem

Problem: Decision function (2) **is hard to interpret**

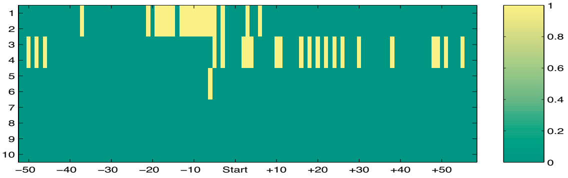
Understanding the SVM Decision

Splice Sites

1. Which positions in the sequence are important for discrimination?



2. What characterizes those positions?



3. Which motifs at which position are important?

Approach: Optimize Combination of Kernels

- Define Kernel as Convex Combination of Subkernels:

$$\mathrm{k}(\mathbf{x},\mathbf{y}) = \sum_{l=1}^{L} \beta_l \, \mathrm{k}_l(\mathbf{x},\mathbf{y})$$

e.g. Weighted Degree Kernel

$$\mathrm{k}(\mathbf{x},\mathbf{x}') = \sum_{l=1}^{L} \beta_l \sum_{k=1}^{d} \mathbf{I}(\mathbf{u}_{k,l}(\mathbf{x}) = \mathbf{u}_{k,l}(\mathbf{x}'))$$

- optimize weights $\boldsymbol{\beta}$ such that margin is maximized

$\Rightarrow$ determine $(\boldsymbol{\beta}, \boldsymbol{\alpha}, b)$ simultaneously

$\Rightarrow$ **Multiple Kernel Learning** (Bach, Lanckriet and Jordan 2004)

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,

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

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

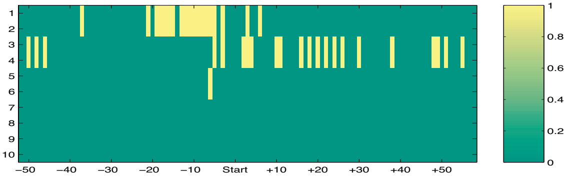
- Heterogeneous data
- Improving interpretability

Method for Interpreting SVMs

- Weighted Degree kernel: linear comb. of $L \cdot D$ 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 Rätsch & Sonnenburg 2006 for more details.

Using SVM $\mathbf{w}$ from feature Space

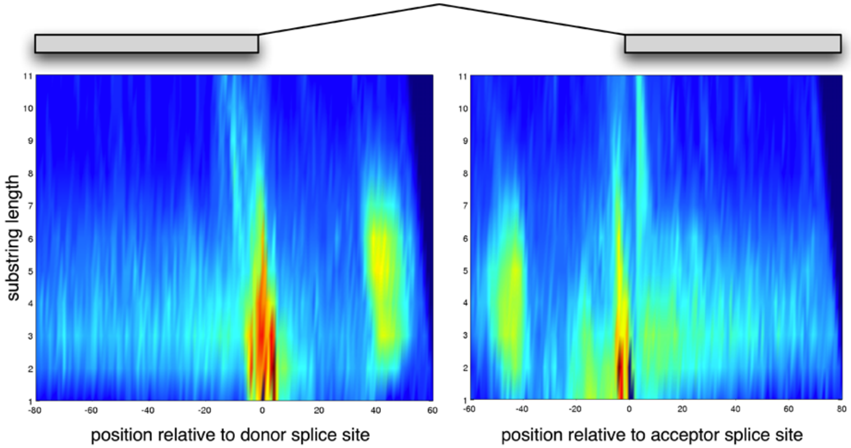
- Recall SVM decision function in kernel feature space:

$$f(\mathbf{x}) = \sum_{i=1}^{N} y_i \alpha_i \underbrace{\Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}_i)}_{=\mathrm{k}(\mathbf{x},\mathbf{x}_i)} + b \qquad (2)$$

- Could explicitly compute $\mathbf{w} = \sum_{i=1}^{N} \alpha_i \Phi(\mathbf{x}_i)$
- **Problem:** Φ and thus $\mathbf{w}$ too big
- **Solution:**
  - Reduce dimensionality by considering a small WD kernel degree, (like $1, \ldots, 8$)
  - Still consider high degree for learning, only project on lower degree for interpretation
  - Idea: long, overlapping k-mers contribute to small ones

We get so called Positional Oligomer Importance Matrices

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 (Rätsch et.al 2007)

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

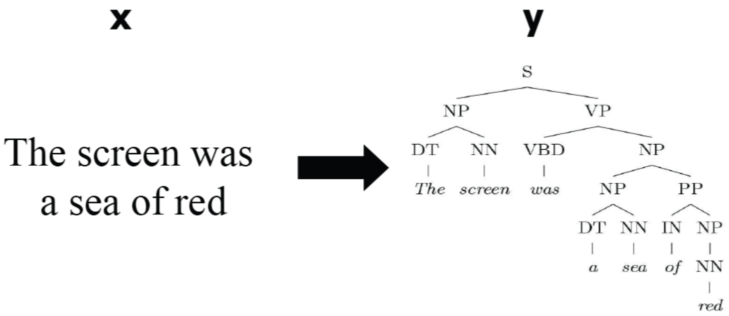
$$k((\mathbf{x}, y), (\mathbf{x}', y')) := [[y = y']] k(\mathbf{x}, \mathbf{x}').$$

Joint Feature Map

Interdependent Outputs
For example a hierarchy of classes like part of speech tagging.

Label Sequence Learning
Given an input sequence predict a label sequence annotating the input

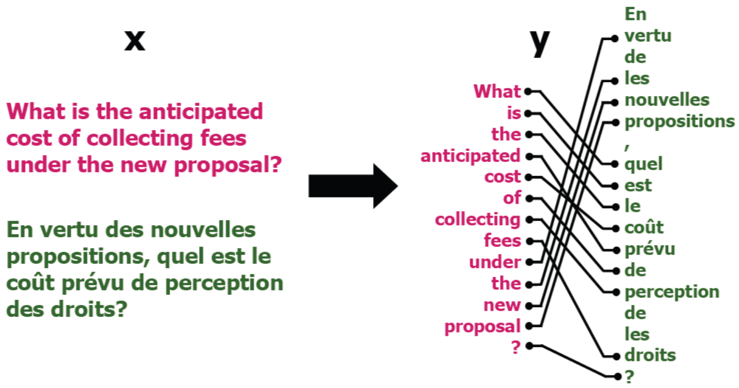
Context Free Grammar Parsing



Recursive Structure

From Klein & Taskar, ACL'05 Tutorial

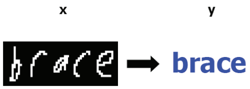
Bilingual Word Alignment


x
What is the anticipated cost of collecting fees under the new proposal?
En vertu des nouvelles propositions, quel est le coût prévu de perception des droits?
y
What
is
the
anticipated
cost
of
collecting
fees
under
the
new
proposal
?
En
vertu
de
les
nouvelles
propositions
,
quel
est
le
coût
prévu
de
perception
de
les
droits
?

Combinatorial Structure

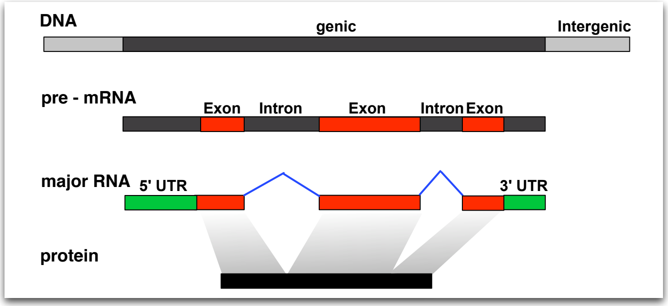
From Klein & Taskar, ACL'05 Tutorial

Handwritten Letter Sequences



Label Sequence Learning

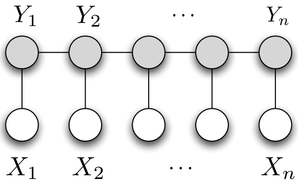
- Given: observation sequence
- Problem: predict corresponding state sequence
- Often: several subsequent positions have the same state
  $\Rightarrow$ state sequence defines a "segmentation"
- Learn Segmentation for **Gene Finding**



Generative Models

- Hidden Markov Models (Rabiner, 1989)
  - 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

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

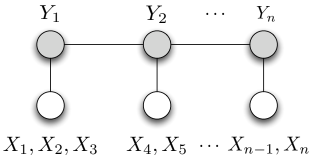
with $g(y_{i-1}, y_i, x_i) = \log p(x_i|y_i) + \log p(y_i|y_{i-1})$.
Problem: Given sequence $\mathbf{x}$, find sequence $\mathbf{y}$ such that $\log p(\mathbf{x},\mathbf{y})$ is maximized, i.e. $\mathbf{y}^* = \mathrm{argmax}_{\mathbf{y}\in\mathcal{Y}^n} \log p(\mathbf{x},\mathbf{y})$
Dynamic Programming 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

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

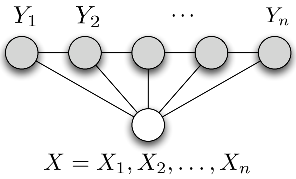
with $g(y_{j-1},y_j,\mathbf{x}_j)=\log p(\mathbf{x}_j|y_j)+\log p(y_j|y_{j-1})$.
Problem: Given sequence $\mathbf{x}$, find sequence $\mathbf{y}$ such that $\log p(\mathbf{x},\mathbf{y})$ is maximized, i.e. $\mathbf{y}^* = \mathrm{argmax}_{\mathbf{y}\in\mathcal{Y}^*}\log p(\mathbf{x},\mathbf{y})$
Dynamic Programming Approach: $V(i,y) :=$

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

Discriminative Models

- Conditional Random Fields (Lafferty et.al 2001)
  - 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

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

- 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

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

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)

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

- 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

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

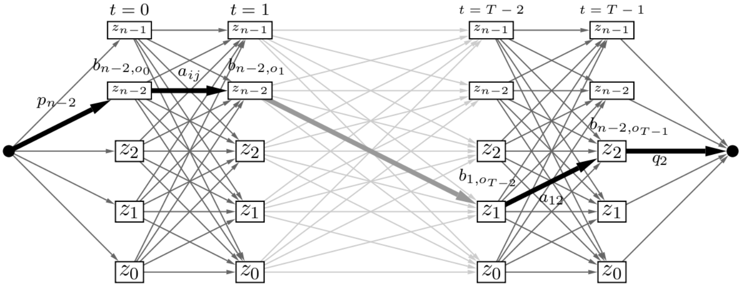
(DP only works if Φ has certain decomposition structure)

- If $\mathbf{y} = \mathbf{y}_n$, then compute second best labeling as well
- If constraint is violated, then add to optimization problem

Dynamic Programming

- number of possible paths of length T for a (fully connected) model with n states is n^T
- infeasible already for small T

Solution: Use dynamic programming (Viterbi decoding)



- runtime complexity before: $\mathcal{O}(n^T) \Rightarrow$ **NOW:** $\mathcal{O}(n^2 \cdot T)$

Algorithm

1. $\mathcal{Y}_n^1 = \emptyset$, for $n = 1, \dots, N$
2. Solve
$$\begin{aligned}(\mathbf{w}^t, \boldsymbol{\xi}^t) = \operatorname*{argmin}_{\mathbf{w}\in\mathcal{F},\boldsymbol{\xi}} \quad & C\sum_{n=1}^{N}\xi_n + \|w\|^2 \\ \text{w.r.t.} \quad & \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}_n) - \Phi(\mathbf{x}, \mathbf{y})\rangle \geq 1 - \xi_n \\ & \text{for all } \mathbf{y}_n \neq \mathbf{y} \in \mathcal{Y}_n^t, n = 1, \dots, N\end{aligned}$$
3. Find violated constraints ($n = 1, \dots, N$)
$$\mathbf{y}_n^t = \operatorname*{argmax}_{\mathbf{y}_n \neq \mathbf{y} \in \mathcal{Y}^*} \langle \mathbf{w}^t, \Phi(\mathbf{x}, \mathbf{y})\rangle$$
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):

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

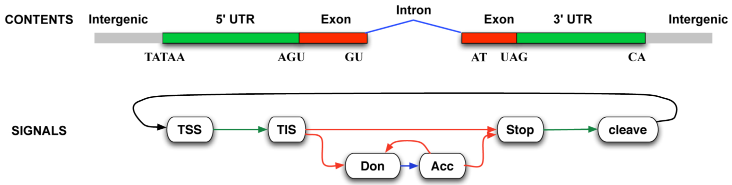
- Finding violated constraints more difficult

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



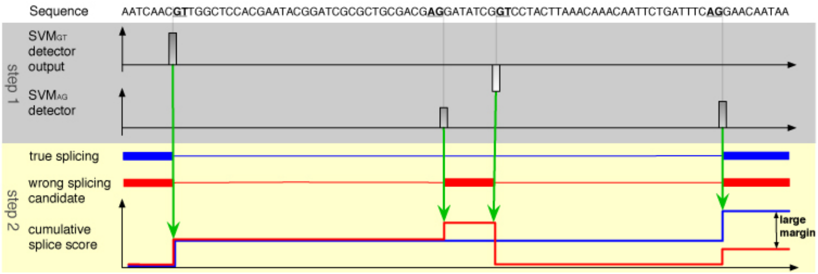
Learning to Predict Segmentations

- 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

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

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

Large Margin Combination (simplified)

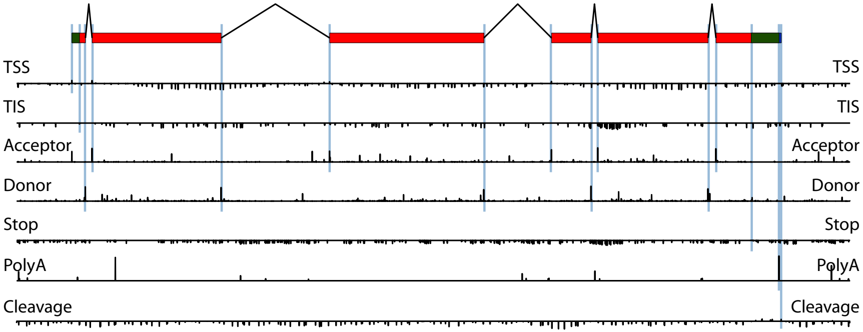


- Simplified Model: Score for splice form $\mathbf{y} = \{(p_j, q_j)\}_{j=1}^{J}$:

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

- 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

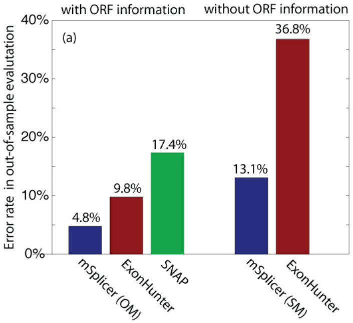

TSS
TIS
Acceptor
Donor
Stop
PolyA
Cleavage
TSS
TIS
Acceptor
Donor
Stop
PolyA
Cleavage

Results Summary

- Splicing only (Rätsch et al., PLoS Comp. Biol., 2007)
  - Comparison with other methods
  - Analysis of a few disagreeing cases
  - Results available on `http://www.wormbase.org`
- Full gene predictions
  - Relevant for the nGASP competition
  - Evaluation by organizers still pending

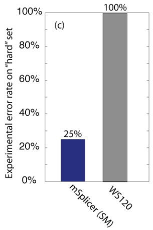
Results I (Splice forms only)

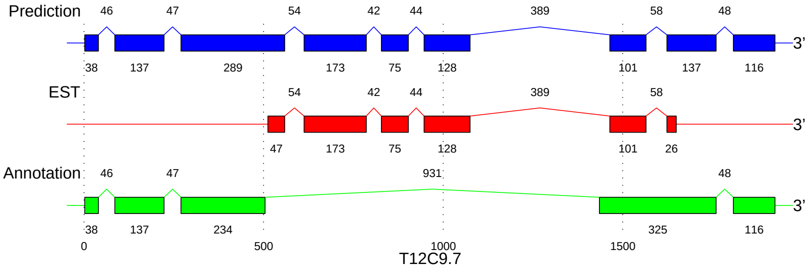
- ≈3,800 gene models derived from cDNAs and ESTs
  - 60% for training and validation
  - 40% for testing (exclude alt. spliced genes)
- Out-of-sample accuracy (≈1100 gene models):
  - Splice form error rate
    - 4.8% (coding)
    - 13.1% (mixed)
  - Much lower error rates than state-of-the-art
    - 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





mGene - Summary

2-step approach

- Content and Signal Sensors(transcription start,. . .)
  - Support Vector Machine with String Kernel (spectrum,weighted degree,. . .)
- Label Sequence (Segmentation) Learning
  - Joint feature maps for inputs and outputs
  - Related to (generalized) HMMs
  - 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 :-)