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

\[ k(x, x') \]
\[ \langle \Phi(x), \Phi(x') \rangle \]

Multiple Kernels
\[ \beta_1 k_1(x, x') + \beta_2 k_2(x, x') + \ldots + \beta_p k_p(x, x') \]

- Learning structured output spaces
- Finding the optimal combination of kernels
Multiple Kernel Learning (MKL)

Possible solution We can add the two kernels, that is

\[ k(x, x') := k_{\text{sequence}}(x, x') + k_{\text{structure}}(x, x') \]

Better solution We can mix the two kernels,

\[ k(x, x') := (1 - t)k_{\text{sequence}}(x, x') + tk_{\text{structure}}(x, x') \]

where \( t \) should be estimated from the training data.

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

\[ k(x, x') = \sum_{p=1}^{K} \beta_p k_p(x, x') \]

Applications

- Heterogeneous data
- Improving interpretability
Method for Interpreting SVMs

- **Weighted Degree kernel**: linear comb. of $LD$ kernels

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

- **Example**: Classifying splice sites

See ? for more details.
Color-coded importance scores of substrings near splice sites. Long substrings are important upstream of the donor and downstream of the acceptor site [?]
Learning Task
For a set of labeled data, we predict the label.

Difference from multiclass
The set of possible labels $Y$ may be very large or hierarchical.

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

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

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

$$k((x, y), (x', y')) := [[y = y']] k(x, x').$$
The screen was a sea of red

Recursive Structure

From Klein & Taskar, ACL’05 Tutorial
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?

Combinatorial Structure

From Klein & Taskar, ACL’05 Tutorial
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
  ⇒ state sequence defines a “segmentation”

Example 1: Protein Secondary Structure Prediction
Given: observation sequence

Problem: predict corresponding state sequence

Often: several subsequent positions have the same state
⇒ 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)

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

- Efficient dynamic programming (DP) algorithms
Decoding via Dynamic Programming

\[ \log p(x, y) = \sum_i (\log p(x_i | y_i) + \log p(y_i | y_{i-1})) \]

\[ = \sum_i g(y_{i-1}, y_i, x_i) \]

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 \( x \), find sequence \( y \) such that \( \log p(x, y) \) is maximized, i.e. \( y^* = \arg\max_{y \in y^n} \log p(x, y) \)

**Dynamic Programming Approach:**

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

\[
p(x, y) = \prod_j p\left((x_{i(j-1)} + 1, \ldots, x_{i(j)}) \mid y_j\right) p(y_j \mid y_{j-1})
\]

(use with care)

- Use generalization of DP algorithms of HMMs
Decoding via Dynamic Programming

\[
\log p(x, y) = \prod_j p((x_i(j), \ldots, x_i(j+1)-1) | y_j) p(y_j | y_{j-1}) \\
= \sum_j g(y_{i-1}, y_i, (x_{i(j-1)+1}, \ldots, x_{i(j)}))
\]

with \( g(y_{j-1}, y_j, x_j) = \log p(x_j | y_j) + \log p(y_j | y_{j-1}) \).

**Problem:** Given sequence \( x \), find sequence \( y \) such that \( \log p(x, y) \) is maximized, i.e. \( y^* = \arg\max_{y \in y^*} \log p(x, y) \)

**Dynamic Programming Approach:** \( V(i, y) := \)

\[
\begin{cases} 
\max_{y' \in y, d=1, \ldots, i-1} (V(i - d, y') + g(y', y, x_{i-d+1, \ldots, i})) & i > 1 \\
0 & \text{otherwise}
\end{cases}
\]
Conditional Random Fields

- conditional prob. $p(y|x)$ instead of joint prob. $p(x, y)$

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

- 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 \( f(y|x) \) scoring segmentations \( y \) for \( x \)
- Maximize \( f(y|x) \) w.r.t. \( y \) for prediction:
  \[
  \arg\max_{y \in Y^*} f(y|x)
  \]
- Given \( N \) sequence pairs \((x_1, y_1), \ldots, (x_N, y_N)\) for training
- Determine \( f \) such that there is a large margin between true and wrong segmentations
  \[
  \min_{f} C \sum_{n=1}^{N} \xi_n + P[f] \qquad \text{w.r.t. } f(y_n|x_n) - f(y|x_n) \geq 1 - \xi_n
  \]
  for all \( y_n \neq y \in Y^*, n = 1, \ldots, N \)
- Exponentially many constraints!
Recall the kernel trick

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

Joint kernel on \( X \) and \( Y \)

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

\[
k((x, y), (x', y')) := \langle \Phi(x, y), \Phi(x', y') \rangle.
\]

For multiclass

For normal multiclass classification, the joint feature map decomposes and the kernels on \( Y \) is the identity, that is

\[
k((x, y), (x', y')) := [[y = y']] k(x, x').
\]
SO Learning with kernels

- Assume $f(y|x) = \langle w, \Phi(x, y) \rangle$, where $w, \Phi(x, y) \in \mathcal{F}$
- Use $\ell_2$ regularizer: $P[f] = \|w\|^2$

$$\min_{w \in \mathcal{F}, \xi \in \mathbb{R}^N} \sum_{n=1}^{N} C \xi_n + \|w\|^2$$

w.r.t. $\langle w, \Phi(x, y_n) - \Phi(x, y) \rangle \geq 1 - \xi_n$
for all $y_n \neq y \in \mathcal{Y}^*, n = 1, \ldots, N$

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

$$\max_{\alpha} \sum_{n,y} \alpha_{n,y} - \sum_{n,y} \sum_{n',y'} \alpha_{n,y} \alpha_{n',y'} \langle \Phi_{n,y}, \Phi_{n',y'} \rangle$$

w.r.t. $\alpha_{n,y} \geq 0, \sum_{y} \alpha_{n,y} \leq C$ for all $n$ and $y$
Kernels

- Recall: \( \Phi_{n,y} :=\Phi(x_n, y_n) - \Phi(x_n, y) \)

- Then

\[
\langle \Phi_{n,y}, \Phi_{n',y'} \rangle = \langle \Phi(x_n, y_n) - \Phi(x_n, y), \Phi(x_{n'}, y_{n'}) - \Phi(x_{n'}, y') \rangle \\
= k((x_n, y_n), (x_{n'}, y_{n'})) - k((x_n, y_n), (x_{n'}, y')) - \\
-k((x_n, y), (x_{n'}, y_{n'})) + k((x_n, y), (x_{n'}, y)),
\]

where

\[
k((x_n, y), (x_{n'}, y')) := \langle \Phi(x_n, y), \Phi(x_{n'}, y') \rangle
\]

- Kernel learning (almost) as usual
Special Case: only two “structures”

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

\[
\min_{w \in \mathcal{F}, \xi \in \mathbb{R}^N} C \sum_{n=1}^{N} \xi_n + \|w\|^2
\]

w.r.t.
\[
\langle w, \Phi(x, y_n) - \Phi(x, 1 - y_n) \rangle \geq 1 - \xi_n
\]

for all \( n = 1, \ldots, N \)

- Dual: Define \( \Phi_n := \Phi(x_n, y_n) - \Phi(x_n, 1 - y_n) \)

\[
\max_{\alpha} \sum_n \alpha_n - \sum_n \sum_{n'} \alpha_n \alpha_{n'} \langle \Phi_n, \Phi_{n'} \rangle
\]

w.r.t.
\[
\alpha_n \geq 0, \alpha_n \leq C \quad \text{for all } n
\]

- Equivalent to standard 2-class SVM
Optimization problem too big (dual as well)

\[
\min_{w \in \mathcal{F}, \xi} \sum_{n=1}^{N} \xi_n + \|w\|^2
\]

w.r.t. \( \langle w, \Phi(x, y_n) - \Phi(x, y) \rangle \geq 1 - \xi_n \)

for all \( y_n \neq y \in \mathcal{Y}^*, n = 1, \ldots, N \)

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
  \[ \langle w, \Phi(x, y_n) - \Phi(x, y) \rangle \geq 1 - \xi_n \]

- Find labeling \( y \) that maximizes
  \[ \langle w, \Phi(x, y) \rangle \]

- Use Dynamic Programming Decoding
  \[ y = \arg \max_{y \in Y^*} \langle w, \Phi(x, y) \rangle \]
  (DP only works if \( \Phi \) has certain decomposition structure)

- If \( y = y_n \), then compute second best labeling as well

- If constraint is violated, then add to optimization problem
Algorithm

1. $y_n^1 = \emptyset$, for $n = 1, \ldots, N$

2. Solve

$$ (w^t, \xi^t) = \arg\min_{w \in \mathcal{F}, \xi} C \sum_{n=1}^{N} \xi_n + \|w\|^2 $$

w.r.t. $\langle w, \Phi(x, y_n) - \Phi(x, y) \rangle \geq 1 - \xi_n$

for all $y_n \neq y \in Y_n^t, n = 1, \ldots, N$

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

$$ y_n^t = \arg\max_{y_n \neq y \in Y^*} \langle w^t, \Phi(x, y) \rangle $$

If $\langle w^t, \Phi(x, y_n) - \Phi(x, y_n^t) \rangle < 1 - \xi_n$, set $y_n^{t+1} = y_n^t \cup \{y_n^t\}$

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 \( y \neq y' \), then prediction is wrong, but it does not matter how wrong
- Introduce loss function on labellings \( \ell(y, y') \), 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 \( y \neq y \), then prediction is wrong, but it does not matter how wrong.

- Introduce loss function on labellings \( \ell(y, y') \), e.g.
  - How many segments are wrong or missing
  - How different are the segments, etc.

- Extend optimization problem (Margin rescaling):

\[
\min_{w \in \mathcal{F}, \xi} \frac{C}{N} \sum_{n=1}^{N} \xi_n + \|w\|^2
\]

\[
\text{w.r.t. } \langle w, \Phi(x, y_n) - \Phi(x, y) \rangle \geq \ell(y, y') - \xi_n
\]

for all \( y_n \neq y \in \mathcal{Y}^*, n = 1, \ldots, N \)

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

\[
y_n^t = \arg\max_{y \neq y \in \mathcal{Y}^*} \langle w^t, \Phi(x, y) \rangle + \ell(y, y_n)
\]
Loss functions

- So far 0-1-loss with slacks: If \( y \neq y' \), then prediction is wrong, but it does not matter how wrong
- Introduce loss function on labellings \( \ell(y, y') \), e.g.
  - How many segments are wrong or missing
  - How different are the segments, etc
- Extend optimization problem (Slack rescaling):
  
  \[
  \min_{w \in \mathcal{F}, \xi} C \sum_{n=1}^{N} \xi_n + \|w\|^2
  \]
  w.r.t. \( \langle w, \Phi(x, y_n) - \Phi(x, y) \rangle \geq 1 - \xi_n / \ell(y, y') \)
  for all \( y_n \neq y \in \mathcal{Y}^*, n = 1, \ldots, N \)
- 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 $\langle w, \Phi(x, y) \rangle$ 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
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
Learn function $f(y|x)$ scoring segmentations $y$ for $x$

- $f$ considers signal, content and length information

Maximize $f(y|x)$ w.r.t. $y$ for prediction: $\arg\max_y f(y|x)$

Determine $f$ such that there is a large margin between true and wrong segmentations

$$\min_f \sum_{n=1}^{N} \xi_n + P[f]$$

w.r.t. $f(y_n|x_n) - f(y|x_n) \geq 1 - \xi_n$

for all $y \neq y_n, n = 1, \ldots, N$

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

$$f(y) := \sum_{j=1}^{J-1} S_{GT}(f_j^{GT}) + \sum_{j=2}^J S_{AG}(f_j^{AG}) + \sum_{j=1}^{J-1} S_{LI}(p_{j+1} - q_j) + \sum_{j=1}^J S_{LE}(q_j - p_j)$$

- Splice signals
- Segment lengths

Tune free parameters (in functions $S_{GT}, S_{AG}, S_{LE}, S_{LI}$) by solving linear program using training set with known splice forms.
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)
Validation by RT-PCR & direct sequencing
Consider 20 disagreeing cases
Annotation was never correct
75% of our predictions were correct
Joint feature maps for inputs and outputs
- Good for multiclass and structure prediction
- Related to (generalized) HMMs
  - Don’t estimate $p(x, y)$ but predict $y$ given $x$
- 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 :-(