RNA Secondary Structure Prediction Using Large Margin Methods

F. De Bona[1], C. S. Ong[1,2], A. Zien[1,2], G. Rätsch,[1]
[1] Friedrich Miescher Laboratory, Max Planck Society, Spemannstr. 39, Tübingen, Germany;
[2] Max Planck Institute for Biological Cybernetics, Spemannstr. 38, Tübingen, Germany
fabio@tuebingen.mpg.de

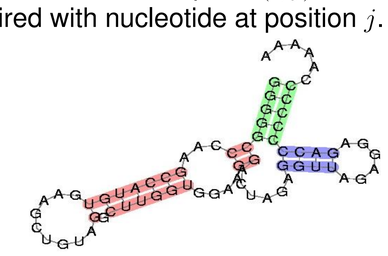
Summary

The majority of traditional computational approaches to RNA secondary structure prediction consist of two components:
- Dynamic programming algorithm for decoding the structure, and
- a parameter inference method.

While the dynamic programming algorithm is quite similar for all these methods parameter inference is usually either than in a probabilistic way or using experimental thermodynamical data. A recently, (1,) proposed probabilistic approach used a conditional maximum likelihood scheme for parameter inference. This model could outperform existing thermodynamic models. We will use a large margin method related to Support Vector Machines. The central idea is to find a parameter vector that assigns highest score to correct and lower score to incorrect structures.

Problem Setting

An RNA Secondary Structure for a nucleotide sequence of length N can be seen as a set of ordered pairs (i, j) denoting that nucleotide at position i is paired with nucleotide at position j.



If (i, j) and (i', j') are two pairs (w.l.o.g $i \leq i'$), then either
- $i = i'$ and $j = j'$,
- $i < j < i' < j'$, or
- $i < i' < j' < j$.

So our input domain is $\mathcal{X} = \Sigma^*$ where $\Sigma = \{A, C, G, U\}$ and our output domain is $\mathcal{Y} = \{\{(i, j) : i, j, \in \mathbb{N}, i < j\}\}$. Based on these pairs one can identify substructures such as stems, hairpins etc..

The Dynamic Programming Component

- In thermodynamic models: Total free energy of y is the sum the energies of the substructures
- In a probabilistic setting: Log probability of a structure is the sum of the log probabilities.

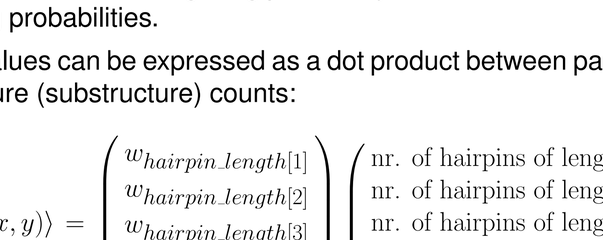
These values can be expressed as a dot product between parameters and feature (substructure) counts:

$$\langle w, \psi(x, y)\rangle = \begin{pmatrix} w_{hairpin\_length}[1] \\ w_{hairpin\_length}[2] \\ w_{hairpin\_length}[3] \\ . \\ . \end{pmatrix} \begin{pmatrix} \text{nr. of hairpins of length 1} \\ \text{nr. of hairpins of length 2} \\ \text{nr. of hairpins of length 3} \\ . \\ . \end{pmatrix}$$

($\psi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^k$ is a feature mapping function). As we are interested in the highest probable structure we calculate:

$$y^* = \underset{\hat{y}}{\text{argmax}}\, P(\hat{y}|x) = \underset{\hat{y}}{\text{argmax}} \langle w, \psi(x_i, \hat{y})\rangle$$

which calculates the structure $\hat{y} \in \mathcal{Y}$ whose probability of being exactly equal to the correct one y is optimal. The result of the above maximization is usually calculated using Dynamic Programming and is called *Viterbi decoding*.
Informally the dynamic programming algorithm has to check for each position i against all remaining positions $j \in \{1, .., n\} \setminus \{i\}$ whether a pair occurs between these positions. When looking at two particular positions i and j of the nucleotide sequence we can identify several possibilities:



SVM Learning for Structured Output Spaces

The following method was proposed by (2,) as a framework for inferring structured output variables such as graphs.
Given the space of input sequences $\mathcal{X}$ and the space of secondary structures $\mathcal{Y}$ we define the prediction function $f : \mathcal{X} \rightarrow \mathcal{Y}$ to be:

$$f(x) = argmax_{y \in \mathcal{Y}} F(x, y; w) = \langle w, \psi(x, y)\rangle,$$

which is exactly the output of the Viterbi decoding.
This means we have to find a w which maximizes the score (Note that our parameters w_i can not be interpreted as probabilities anymore!) of the correct structure.
In optimization lingo this means that we want constraints to hold such that:

$$F(x_i, y_i; w) \geq F(x_i, y; w) + 1 \quad \forall y \in \mathcal{Y} \setminus y_i,$$

which means informally: For each example $(x_i, y_i)_{i \in I}$ F should assign highest score to the correct structure y_i and a lower score to all incorrect structures y. The $+1$ term is a so-called margin we enforce.

- Generating a constraint for each possible incorrect structure leads to a number of constraints exponential in the size of the structure.
- We use a technique called *column generation* to approximate the solution

This is done by using a modified viterbi which incorporates some loss terms:

$$\max_{y \in \mathcal{Y} \setminus y_i} \{\Delta(y, y_i) + \langle w, \psi(x_i, y)\rangle\}$$

- Above maximization can be efficiently computed via dynamic programming
- Resembles "standard" secondary structure calculation except the additional loss term $\Delta(y, y_i)$
- Modification of textbook Viterbi needed and possible

New Viterbi for structure calculation with loss:

$$\text{OUTER}(i, j) = \max \begin{cases} \ell_{pair} + \text{PAIR}(i, j) \\ \ell_{res}(i) + \text{OUTER}(i + 1, j) \\ \ell_{res}(j) + \text{OUTER}(i, j - 1) \\ \max_{i \leq k < j} \text{OUTER}(i, k) + \text{OUTER}(k + 1, j) \end{cases}$$

Why bother ?

- Large-margin methods (SVMs et al) outperform probabilistic models in lots of applications.
- What about RNA secondary structure prediction ?
- Is RNA secondary structure prediction a solved problem ?
- Are probabilistic methods "superior" ?
- Is there a single label in RNA secondary structure prediction ?

The ℓ term are contributions of the Hamming loss to the Viterbi score. Removing these terms would lead to a "standard" Viterbi algorithm as it is implemented in $mfold$ and other packages. Using these ideas and formulations we can state a convex mathematical programm which summarizes our requirements:

$$\begin{aligned} \min_{w,\xi} \quad & \tfrac{1}{2}\Omega(w) + \tfrac{C}{n}\textstyle\sum_i \xi_i \\ \text{s.t.} \quad & \xi_i \geq max_{y \in Y \setminus y_i}\{1 - (F(x_i, y_i) - F(x_i, y))\} \quad \forall i \in I \\ & \xi_i \geq 0 \qquad\qquad \forall i \in I. \end{aligned}$$

where Ω is a regularization function taking care of the complexity of our solution w. ξ_i.

- Problem: All structures are penalized equally.
- Better: Penalize incorrect structures according to their distance to the real solution.
- Solution: Structural (Hamming) loss $\Rightarrow$ count number of incorrect positions.

To take the structural loss into account we can multiply the margin with the Hamming loss Δ:

$$\begin{aligned} \min_{w,\xi} \quad & \tfrac{1}{2}||w||^2 + \tfrac{C}{n}\textstyle\sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \max_{y \in \mathcal{Y} \setminus y_i} \langle w, \psi(x_i, y_i) - \psi(x_i, y)\rangle \geq \Delta(y, y_i) - \xi_i \quad \forall i \in I \\ & \xi_i \geq 0 \qquad\qquad \forall i \in I. \end{aligned}$$

This approach where the margin is multiplied with the Hamming loss is known as *margin rescaling*.
It can be argued whether this is the right thing to do. Alternative: *Slack-rescaling*. Formal difference between these methods: $\forall i\ \forall y \in \mathcal{Y} \setminus y_i$:

$$\langle \psi(x_i, y_i) - \psi(x_i, y)\rangle \geq \Delta(y_i, y) - \xi_i \quad \text{(margin-rescaling)}$$

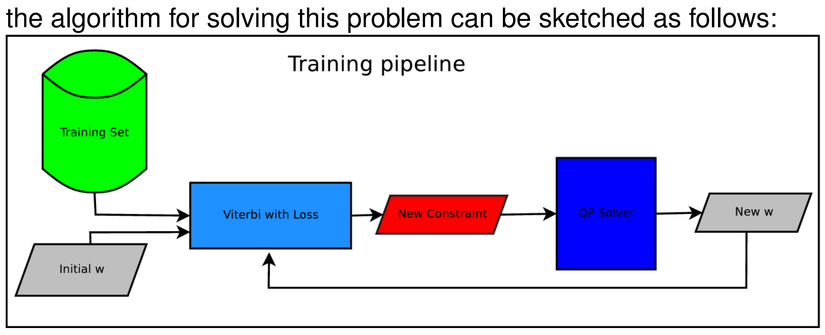
$$\langle \psi(x_i, y_i) - \psi(x_i, y)\rangle \geq 1 - \frac{\xi_i}{\Delta(y_i, y)} \quad \text{(slack-rescaling)}$$

Parameters and Algorithm Details

We want to solve:

$$\min_{w,\xi} \frac{1}{2}||w||^2 + \frac{C}{n}\sum_{i=1}^{n} max_{\forall y \in \mathcal{Y} \setminus y_i}(\ell(\langle w, \psi(x_i, y_i) - \psi(x_i, y)\rangle))$$

the algorithm for solving this problem can be sketched as follows:



Tuning of our method involves:

- Selection of adequate features,
- choice of regularization term, and
- choice of an adequate loss function.

Choice of Features and Regularization

- The total number of features: 1216.
- Features include hairpin, stem sizes,
- ocurrence of certain motifs, etc.
- Feature dependencies/relations modeled via regularization term

Our quadratic programm has a regularization term $w^T P w$.
- Naive regularization could be P = I (identity matrix)
- We would like to include biological prior knowledge in P, such as:
  - Loop size parameters should be "smooth",
  - no difference between AU and UA.
- Coupling of the parameters: $(w_i - w_j)^2$

$\Rightarrow$ Proper regularization affects sensitivity / specificity by 2% (in total) each.

Structural Loss

What if there is no best structure ?
- Maybe the highest scoring structure is not the correct one but the second highest scoring.
  $\Rightarrow$ Achieving structural loss of zero is not possible at all.
- It might make sense to allow for certain number of positions to be incorrect during training.
  $\Rightarrow$ Δ could be insensitive to a certain loss range.

We investigated several forms of this ϵ-insensitive loss, namely
- Loss insensitive for 3 positions,
- Loss insensitive for $0.1 \cdot length(x)$ positions, and
- Loss insensitive for $0.02 \cdot length(x)$ at least 2 positions.

The 10% insensitive loss performed best for these 3 cases. However only marginal better than a standard Hamming loss.

Results

- Data set proposed in (1,).
- Data set is a subset of the Rfam database (3,)
- Consists of 151 secondary structures

Our method is denoted with RFP.

Method	Sensitivity	Specificity
Contrafold	0.73	0.66
CG	0.73	0.65
mfold	0.69	0.60
RFP	0.66	0.67

Results were obtained by five-fold cross validation.

Discussion

As we have seen there is still a question to be adressed: Are probabilistic models superior to large-margin methods in structure prediction ?
The conditional likelihood approach and our large-margin method have a lot in common:
- Feature set seems pretty fixed and very similar
- Regularization offers a lot of tuning possibilities but is also very similar and reported to contribute only 2% in total.
- Objective is convex.
- Results reported with using plain Viterbi are very close.

$\Rightarrow$ However there is still a gap in terms of sensitivity and specificity.

Possible explanantions:
- It was reported that maximum expected accuracy outperforms Viterbi (1,).
- Rfam consists of consensus structures we instead assume that there is one correct label

Among the things we tried are:
- Different regularization terms max $\pm 2\%$ sensitivity / specificity.
- Slack / Margin rescaling workarounds showed no benefit.
- ϵ-insensitive structural loss function

References

Do C.B. and Woods D.A. and Batzoglou S., CONTRAfold: RNA Secondary Structure Prediction without Energy-Based Models. *Bioinformatics* 22(14), 2006.

Ioannis Tsochantaridis and Thomas Hofmann and Thorsten Joachims and Yasemin Altun Support Vector Machine Learning for Interdependent and Sturcutured Output Spaces *Proceedings of the 16th International Conference on Machine Learning*, 2004

Sam Griffiths-Jones and Simon Moxon and Mhairi Marshall and Ajay Khanna and Sean R. Eddy and Ale Bateman. Rfam: annotating non-coding RNAs in complete genomes. *Nucleic Acids Res.*, 2005.

R. Durbin and S. Eddy and A. Krogh and G. Mitchison Biological Sequence Analysis Cambridge University Press 1998

Michael Zuker Lectures on RNA Secondary Structure Prediction, 2003 http://www.bioinfo.rpi.edu/ zukerm/lectures/RNAfold-html/

Joachim Dahl and Lieven Vandenberghe CVXOPT: A Python Package for Convex Optimization http://www.ee.ucla.edu/ vandenbe/cvxopt/ Current release: 0.81. 2006