Machine learning methods for protein analyses

William Stafford Noble
Department of Genome Sciences
Department of Computer Science and Engineering
University of Washington

Outline

- Remote homology detection from protein sequences
- Identifying proteins from tandem mass spectra
  - Simple probability model
  - Direct optimization approach

Large-scale learning to detect remote evolutionary relationships among proteins

Iain Melvin

Jason Weston

Christina Leslie

Search

Set subsequence From: To:

Choose database nr

Do CD-Search

Now: BLAST! or Reset query Reset all

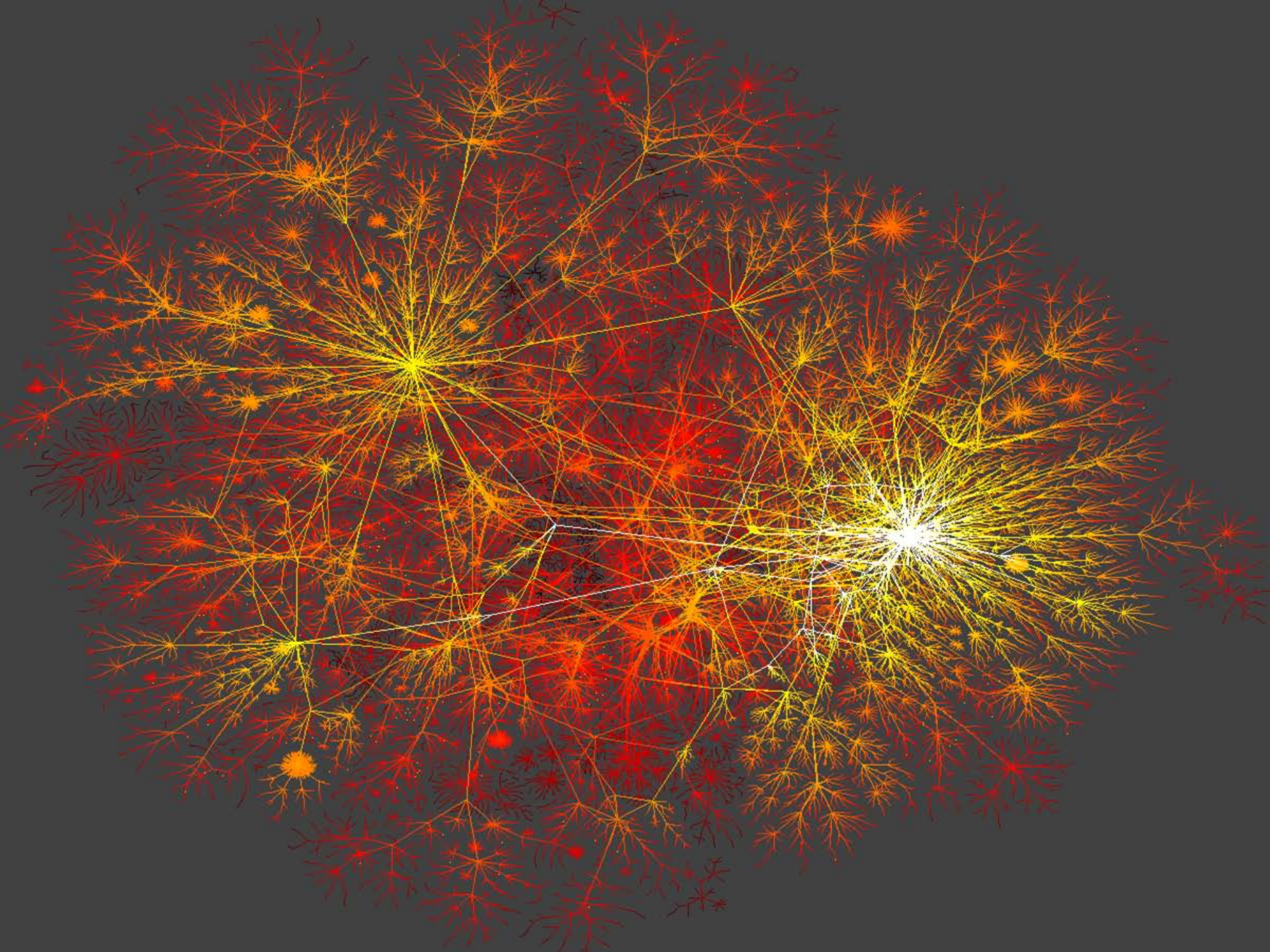
Napredno iskanje

Nastavitve

Jezikovna orodja

Iskanje Google | Klik na srečo

Išči po: celotnem spletu straneh v državi Slovenija



History

- Smith-Waterman (1981)
  - Optimal pairwise local alignment via dynamic programming
- BLAST (1990)
  - Heuristic approximation of Smith-Waterman
- PSI-BLAST (1997)
  - Iterative local search using profiles
- Rankprop (2004)
  - Diffusion over a network of protein similarities
- HHSearch (2005)
  - Pairwise alignment of profile hidden Markov models

Supervised semantic indexing

- Data: 1.8 million Wikipedia documents
- Goal: given a query, rank linked documents above unlinked documents
- Training labels: linked versus unlinked pairs
- Method: ranking SVM (essentially)
  - Margin ranking loss function
  - Low rank embedding
  - Highly scalable optimizer

(Bai et al., ECIR 2009)

Key idea

- Learn an embedding of proteins into a low-dimensional space such that homologous proteins are close to one another.
- Retrieve homologs of a query protein by retrieving nearby proteins in the learned space.

This method requires
- A feature representation
- A training signal
- An algorithm to learn the embedding

Protein similarity network

- Compute all-vs-all PSI-BLAST similarity network.
- Store all E-values (no threshold).
- Convert E-values to weights via transfer function (weight = $e^{-E/\sigma}$).
- Normalize edges leading into a node to sum to 1.

Sparse feature representation

Query protein

Target protein

$$\Phi(p') = (E(p', p_1), \ldots E(p', p_\ell))$$

PSI-BLAST / HHSearch
E-value for query j, target i

Hyperparameter

$$W(p', p_i) = exp(-S_j(i)/\sigma)$$

Probability that a random walk on the protein similarity network moves from protein p' to p_i.

$$E(p', p_i) = W_{p'p_i} / \sum_j W_{p'p_j}$$

Training signal

- Use PSI-BLAST or HHSearch as the teacher.
- Training examples consist of protein pairs.
- A pair *(q,p)* is positive if and only if query *q* retrieves target *p* with E-value < 0.01.
- The online training procedure randomly samples from all possible pairs.

Learning an embedding

- Goal: learn an embedding

$$g(p) = W\Phi(p)$$

  where *W* is an *n*-by-ℓ matrix, resulting in an *n*-dimensional embedding.

- Rank the database with respect to *q* using

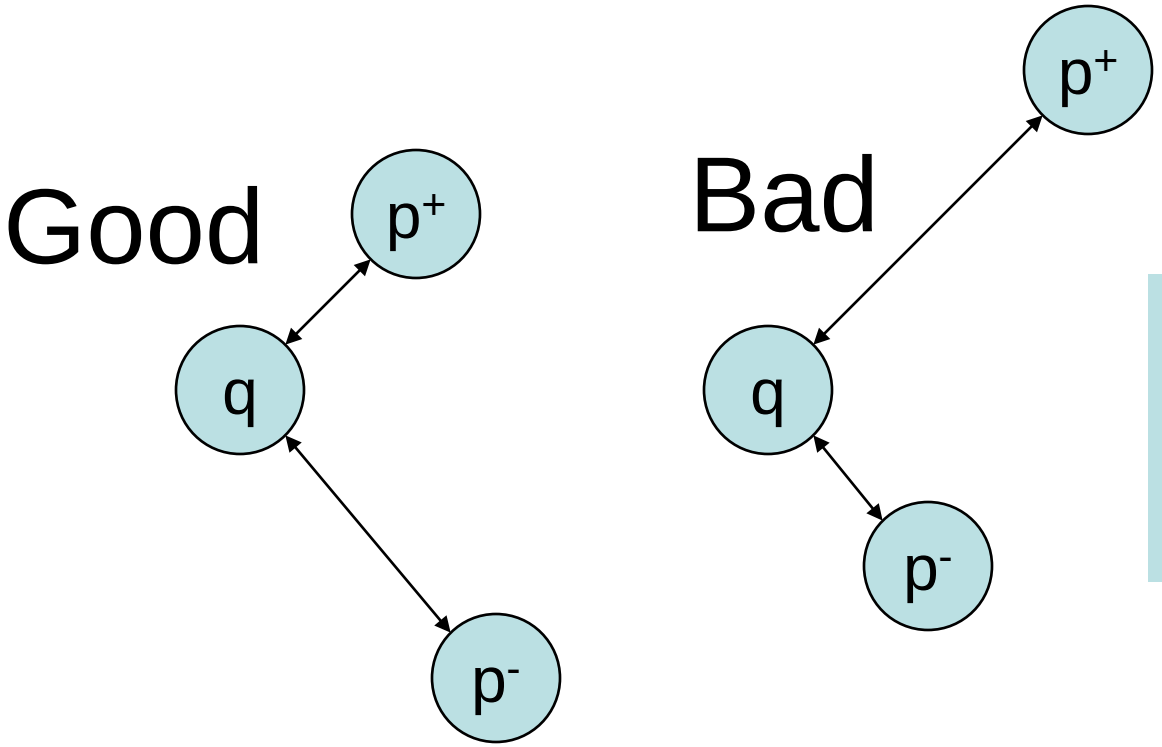
$$f(q, p_i) = ||g(q) - g(p_i)||_1 = ||W\Phi(q) - W\Phi(p_i)||_1$$

  where small values are more highly ranked.

- Choose *W* such that for any tuple

$$f(q, p^+) < f(q, p^-)$$

Learning an embedding



Negative examples should be further from the query than positive examples by a margin of at least 1.

- Minimize the margin ranking loss with respect to tuples (q, p^+, p^-):

$$\sum_{(q,p^+,p^-)\in\mathcal{R}} \max(0, 1 - f(q,p^-) + f(q,p^+))$$

Training procedure

- Minimize the margin ranking loss with respect to tuples (q, p^+, p^-):

$$\sum_{(q,p^+,p^-)\in\mathcal{R}} \max(0, 1 - f(q,p^-) + f(q,p^+))$$

- Update rules:

$$\text{if } \; 1 - f(q,p^-) + f(q,p^+) > 0$$

$$W \leftarrow W - \lambda \, \text{sign}(W\Phi(q) - W\Phi(p^-))\Phi(q)^\top,$$
$$W \leftarrow W + \lambda \, \text{sign}(W\Phi(q) - W\Phi(p^-))\Phi(p^-)^\top,$$
$$W \leftarrow W + \lambda \, \text{sign}(W\Phi(q) - W\Phi(p^+))\Phi(q)^\top,$$
$$W \leftarrow W - \lambda \, \text{sign}(W\Phi(q) - W\Phi(p^+))\Phi(p^+)^\top,$$

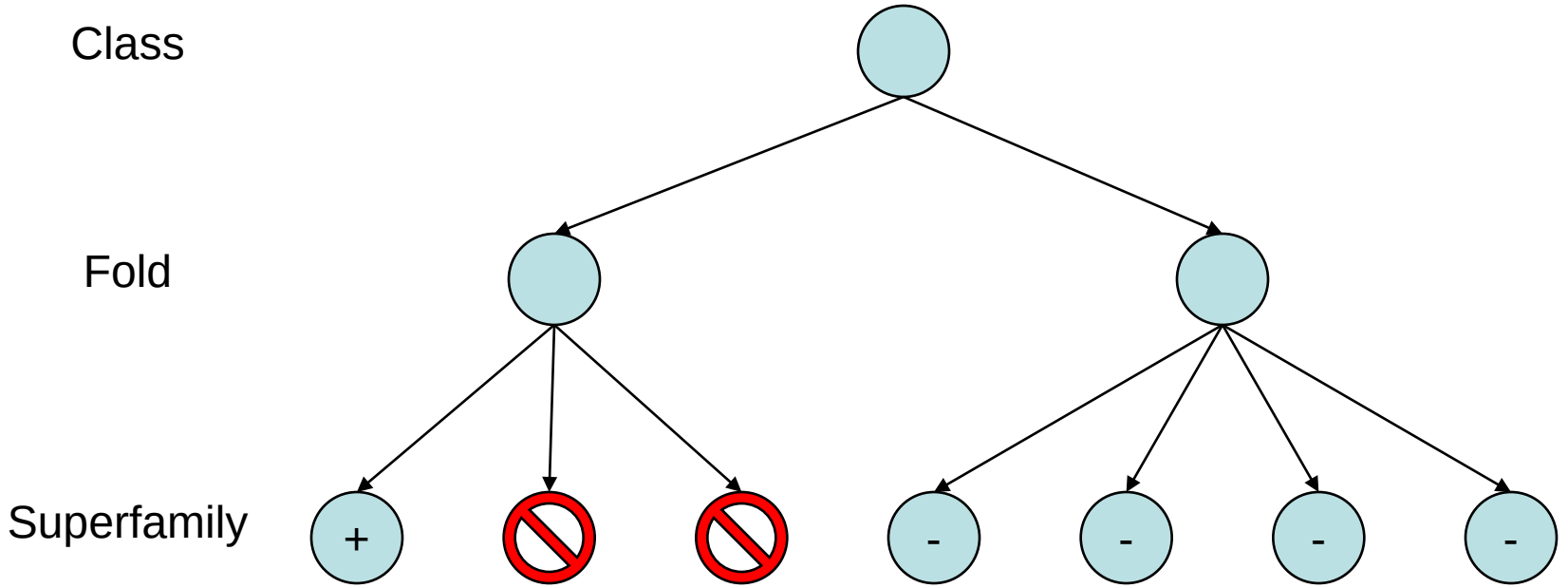
Push q away from p^-

Push p^- away from q

Push q toward p^+

Push p^+ toward q

Remote homology detection



- Semi-supervised setting: initial feature vectors are derived from a large set of unlabeled proteins.
- Performance metric: area under the ROC curve up to the 1st or 50th false positive, averaged over queries.

Results

Method	ROC_1	ROC_{50}
PSI-BLAST	0.624	0.632
Rankprop	0.647	0.707
Protembed PSI-BLAST	0.689	0.739
HHPred	0.771	0.836
Protembed HHPred	0.777	0.853

Results are averaged over 100 queries.

4-helical cytokines
Thioredoxin-like
EF-hand
Snake toxin-like
(Trans)glycosidases
Acid proteases
Scorpion toxin-like
Globin-like
Lysozyme-like
2HC zinc fingers
-binding domain

Key idea #2

- Protein structure is more informative for homology detection than sequence, but is only available for a subset of the data.

- Use *multi-task learning* to include structural information when it is available.

Structure-based labels

- Use the Structural Classification of Proteins to derive labels

$$y_i \in \{1, \ldots, C\}$$

- Introduce a centroid c_i for each SCOP category (fold, superfamily).
- Keep proteins in category *i* close to c_i:

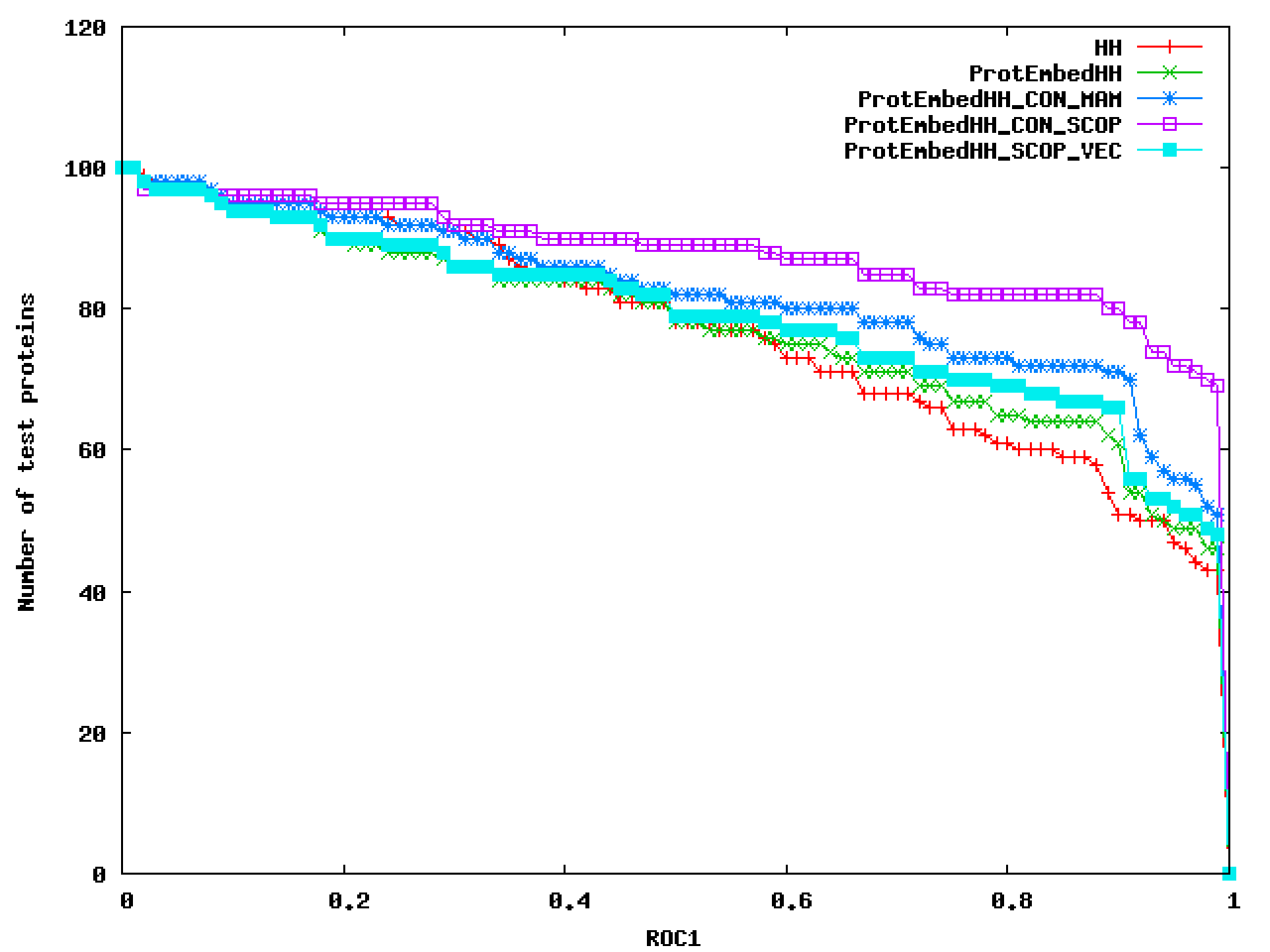
$$f(p_i, c_{y_i}) < f(p_j, c_{y_i}), \quad \forall j : y_j \neq y_i$$

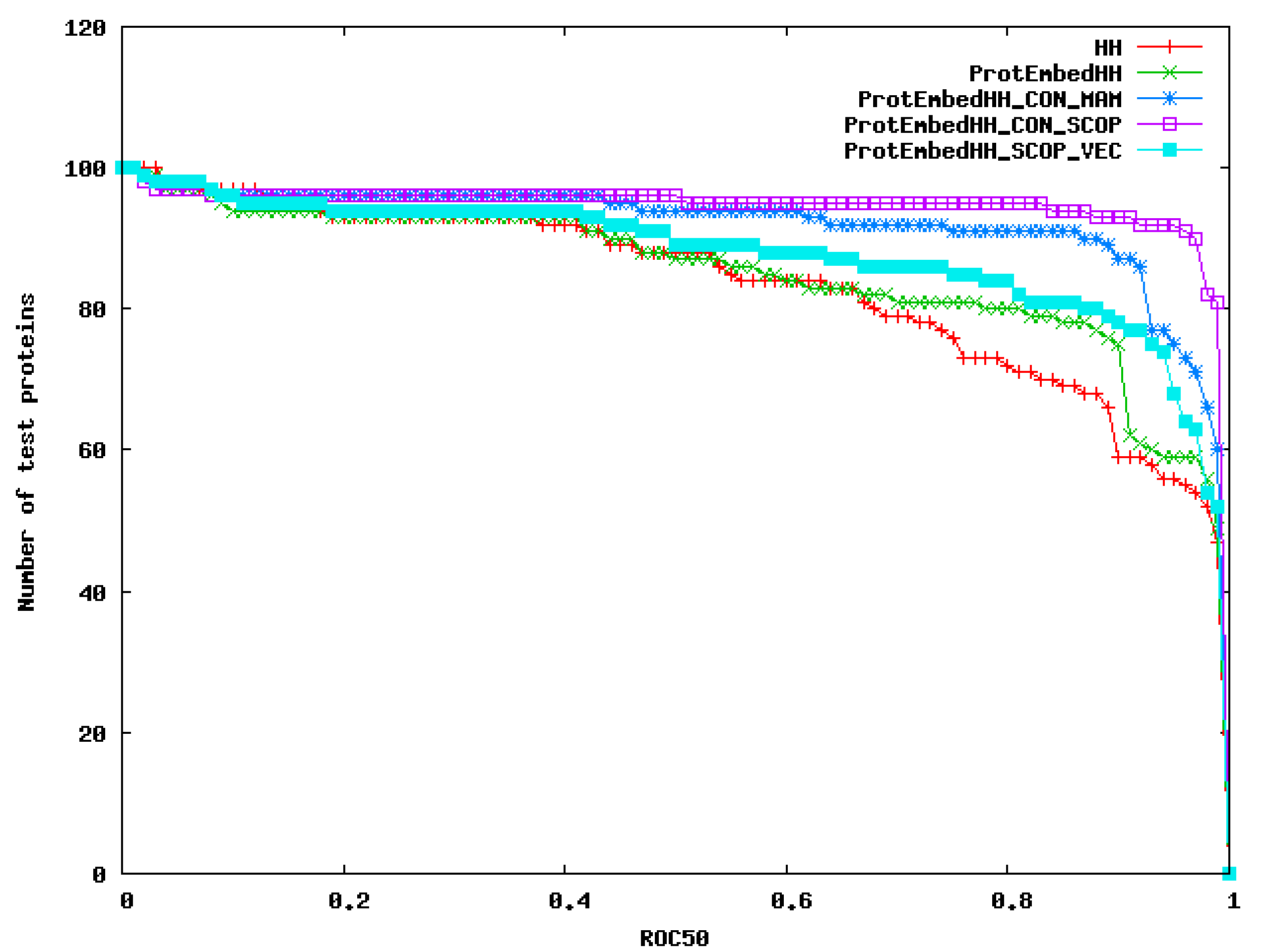
Structure-based ranks

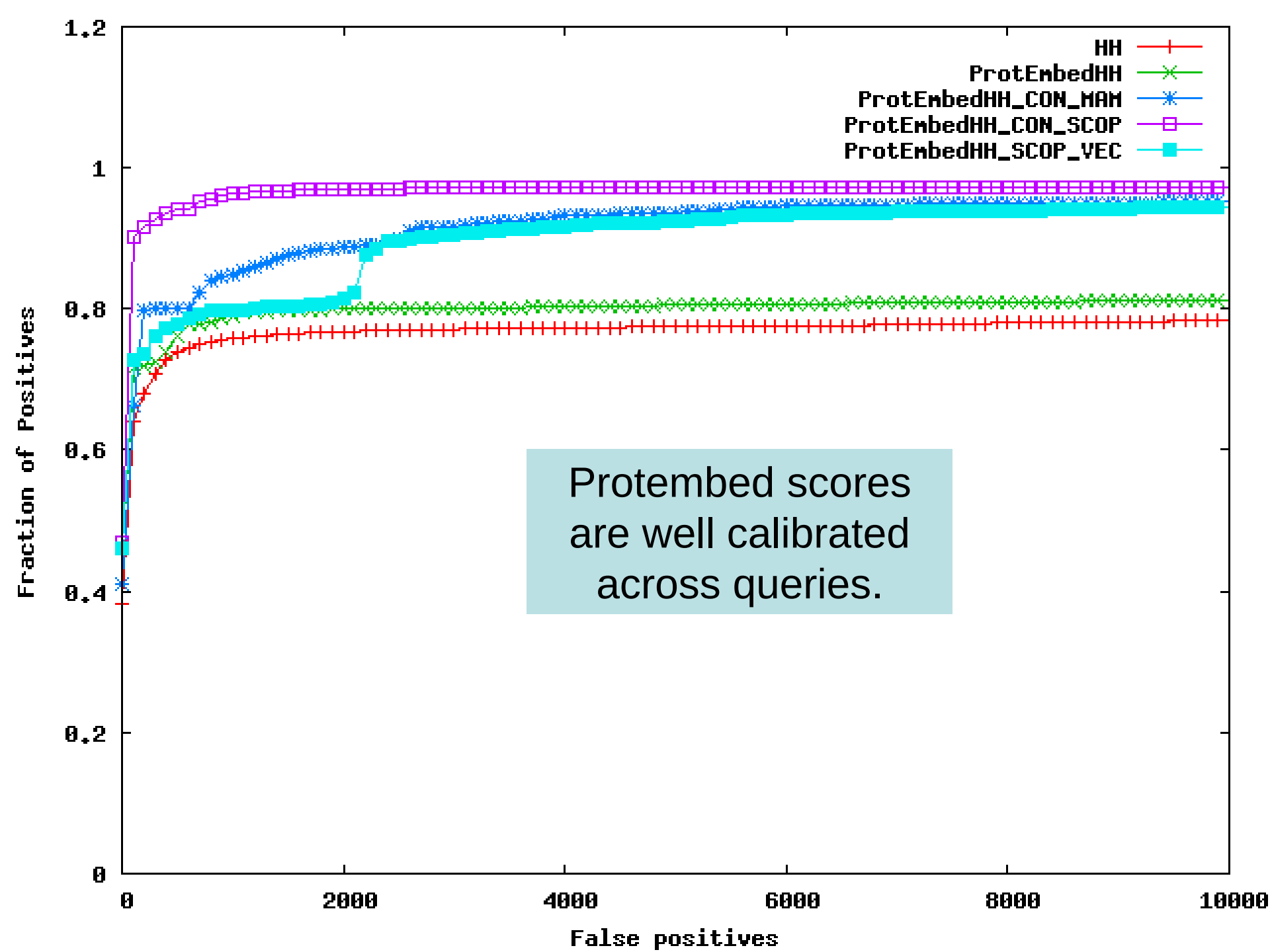
- Use a structure-based similarity algorithm (MAMMOTH) to introduce additional rank constraints.
- Divide proteins into positive and negative with respect to a query by thresholding on the MAMMOTH E-value.

$$f(q, p^+) < f(q, p^-)$$

Method	ROC_1	ROC_{50}
PSI-BLAST	0.624	0.632
Rankprop	0.647	0.707
Protembed PSI-BLAST	0.689	0.739
Protembed PSI-BLAST+SCOP	0.852	0.918
Protembed PSI-BLAST+MAMMOTH	0.744	0.844
HHPred	0.771	0.836
Protembed HHPred	0.777	0.853
Protembed HHPred+MAMMOTH	0.822	0.923
Protembed HHPred+SCOP	0.881	0.949


HH
ProtEmbedHH
ProtEmbedHH_CON_MAM
ProtEmbedHH_CON_SCOP
ProtEmbedHH_SCOP_VEC
Number of test proteins
ROC1
0
20
40
60
80
100
120
0
0.2
0.4
0.6
0.8
1


HH
ProtEmbedHH
ProtEmbedHH_CON_MAM
ProtEmbedHH_CON_SCOP
ProtEmbedHH_SCOP_VEC
Number of test proteins
ROC50
0
20
40
60
80
100
120
0
0.2
0.4
0.6
0.8
1


HH
ProtEmbedHH
ProtEmbedHH_CON_MAM
ProtEmbedHH_CON_SCOP
ProtEmbedHH_SCOP_VEC
Fraction of Positives
False positives
0
0.2
0.4
0.6
0.8
1
1.2
0
2000
4000
6000
8000
10000
Protembed scores are well calibrated across queries.

Conclusions

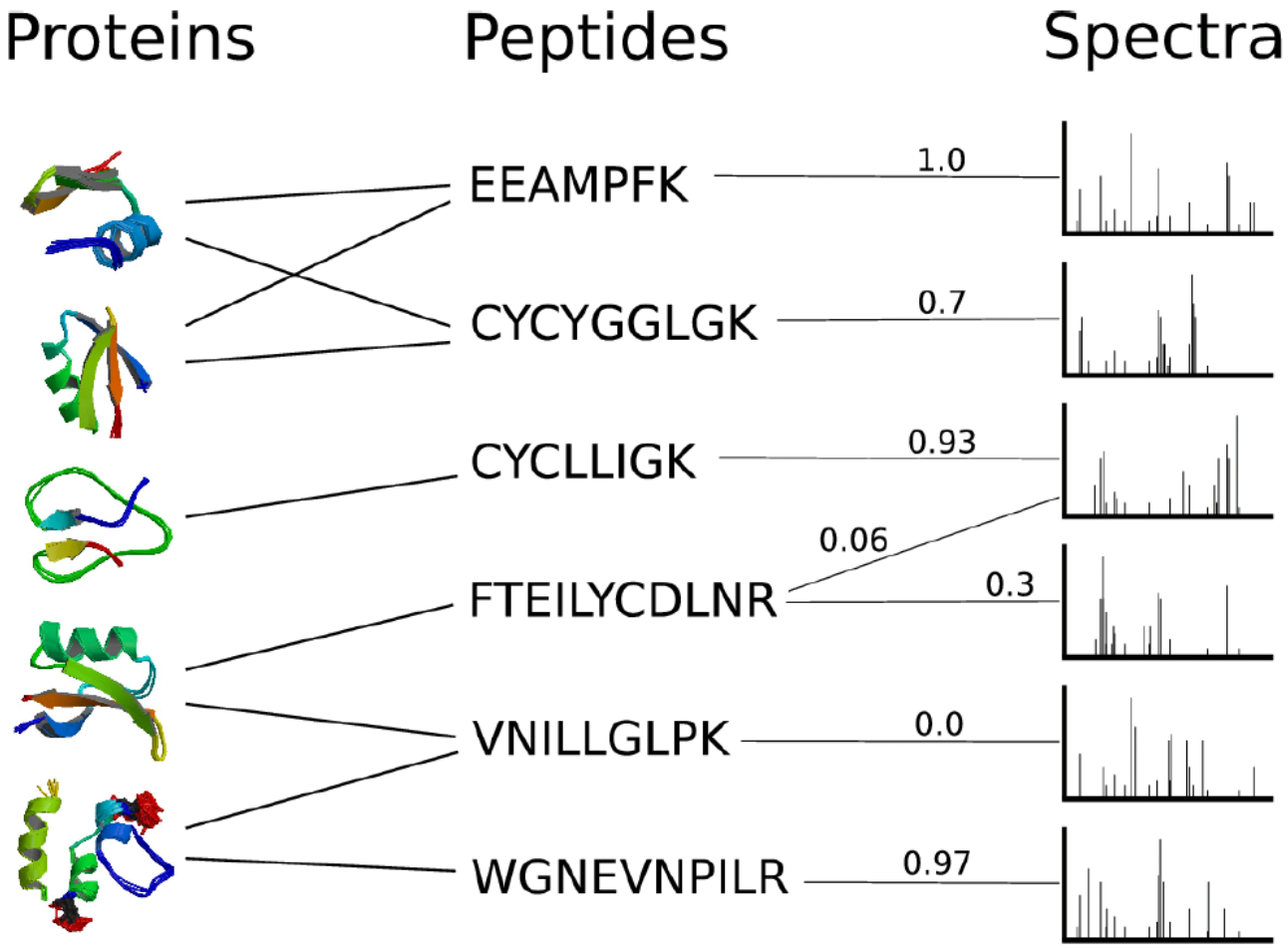
- Supervised semantic indexing projects proteins into a low-dimensional space where nearby proteins are homologs.
- The method bootstraps from unlabeled data and a training signal.
- The method can easily incorporate structural information as additional constraints, via multi-task learning.

Calculation of exact protein posterior probabilities for identifying proteins from shotgun mass spectrometry data

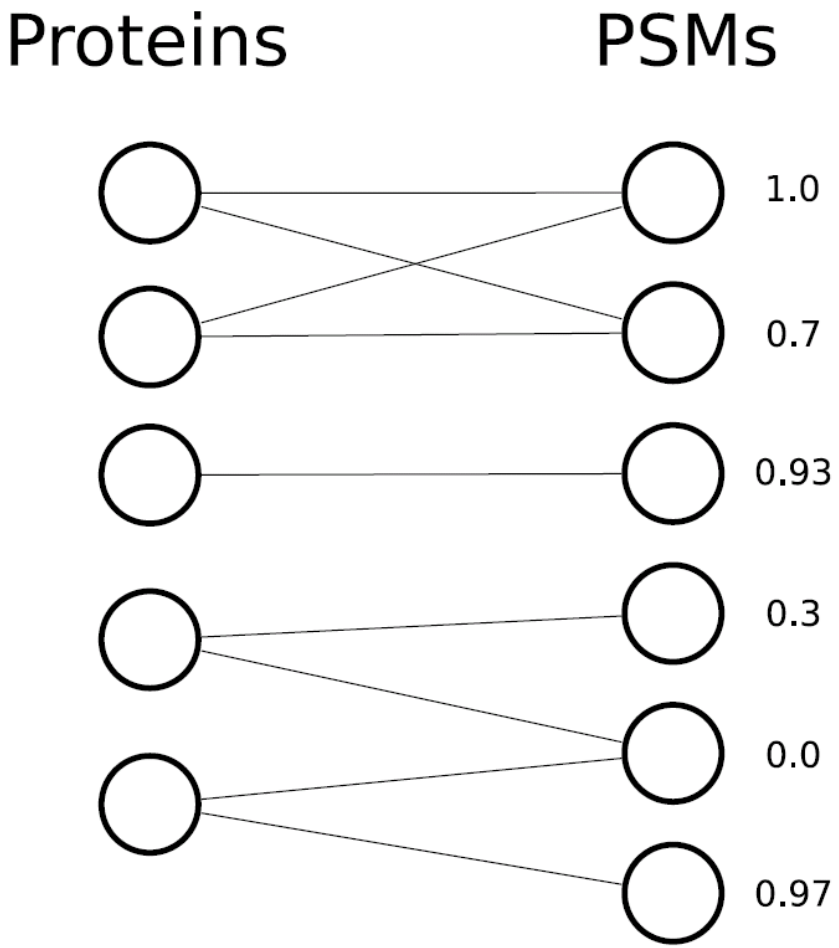
Oliver Serang

Michael MacCoss

The protein ID problem



The protein ID problem



Input:

- Bipartite, many-to-many graph linking proteins to peptide-spectrum matches (PSMs)
- Posterior probability associated with each PSM.

Output:

- List of proteins, ranked by probability.

Existing methods

- ProteinProphet (2003)
  - Heuristic, EM-like algorithm
  - Most widely used tool for this task
- MSBayes (2008):
  - Probability model
  - Hundreds of parameters
  - Sampling procedure to estimate posteriors

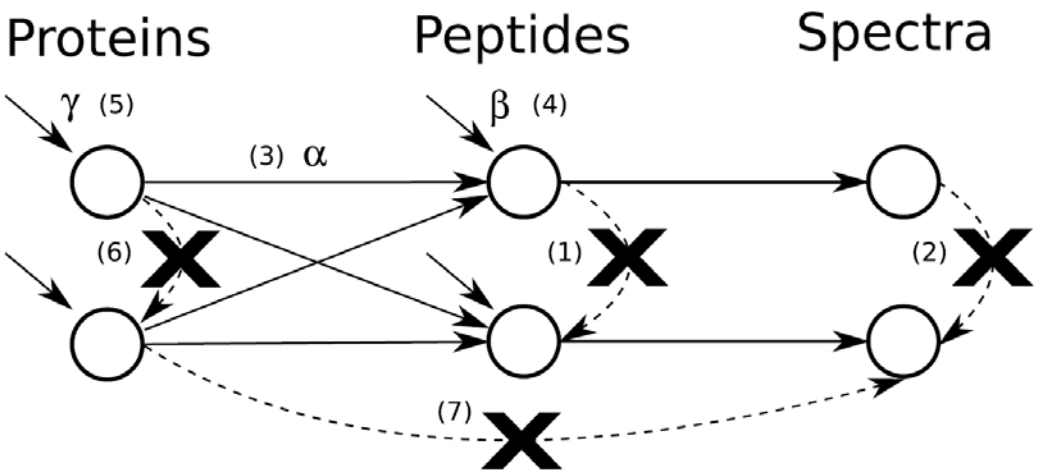
Key idea

- Use a simple probability model with few parameters.
- Employ graph manipulations to make the computation tractable.

Three parameters

- The probability α that a peptide will be emitted by the protein.
- The probability β that the peptide will be emitted by the noise model.
- The prior probability γ that a protein is present in the sample.

Assumptions



1. Conditional independence of peptides given proteins.
2. Conditional independence of spectra given peptides.
3. Emission of a peptide associated with a present protein.
4. Creation of a peptide from the noise model.
5. Prior belief that a protein is present in the sample.
6. Independence of prior belief between proteins.
7. Dependence of a spectrum only on the best-matching peptide.

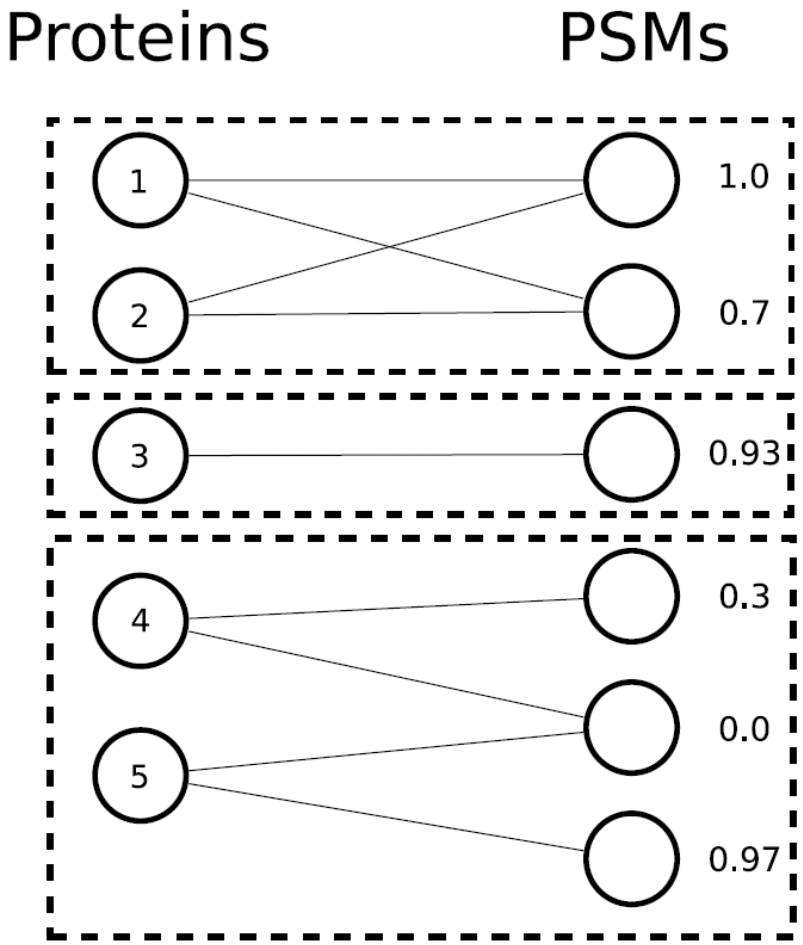
The probability model

$$L(R^{(i)} = r^{(i)}|D) = \sum_{\forall e^{(i)}} \prod_{\epsilon} \frac{\Pr(E_\epsilon^{(i)} = e_\epsilon^{(i)}|D_{\delta(\epsilon)}^{(i)}, Q)}{\Pr(E_\epsilon^{(i)} = e^{(i)}, Q)} \Pr(E_\epsilon^{(i)} = e_\epsilon^{(i)}|R^{(i)} = r^{(i)})$$

- R = the set of present proteins
- D = the set of observed spectra
- E = the set of present peptides
- Q = peptide prior probability

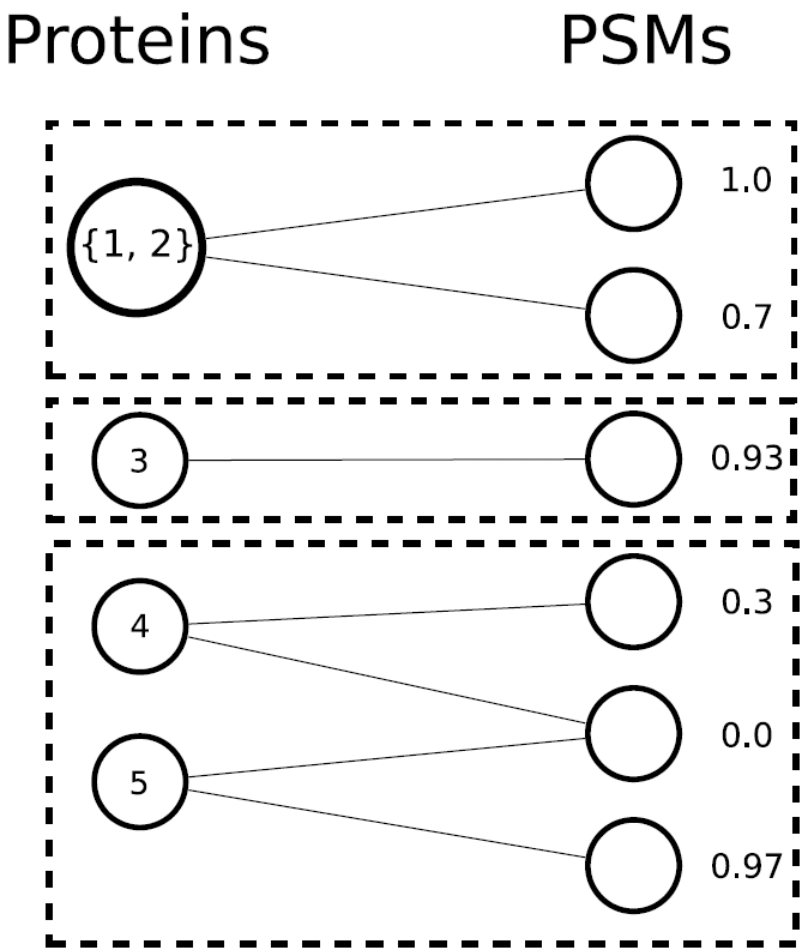
- Computational challenge: Exactly computing posterior probabilities requires enumerating the power set of all possible sets of proteins.

Speedup #1: Partitioning



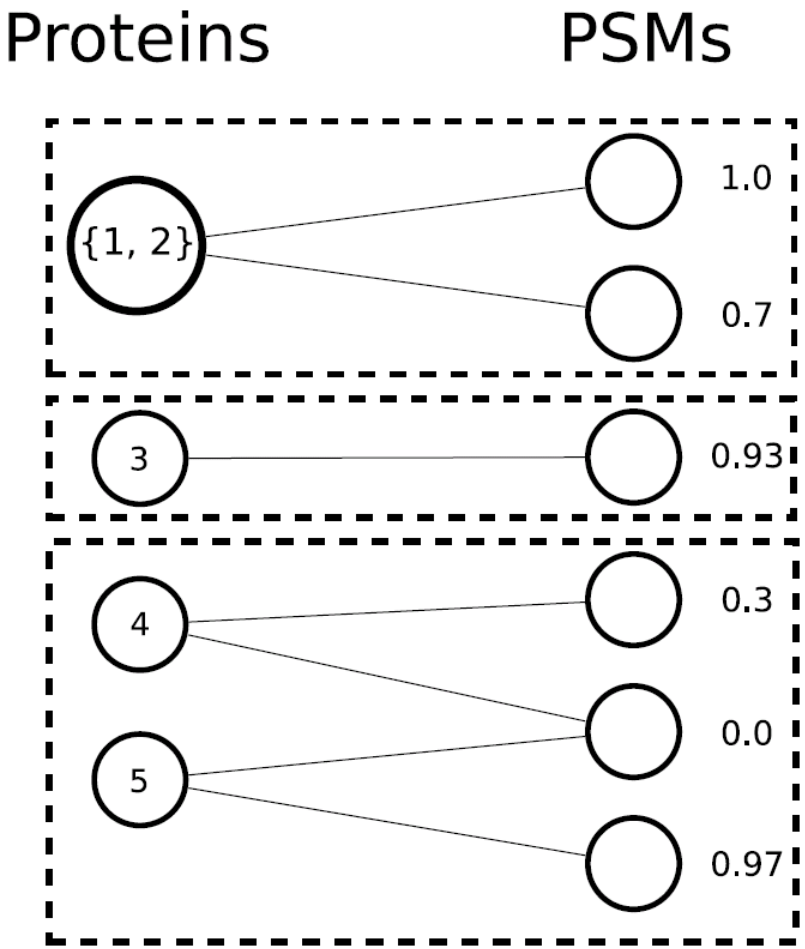
- Identify connected components in the input graph.
- Compute probabilities separately for each component.

Speedup #2: Clustering



- Collapse proteins with the same connectivity into a super-node.
- Do not distinguish between “absent/present” versus “present/absent.”
- Reduce state space from 2^n to n.

Speedup #3: Pruning

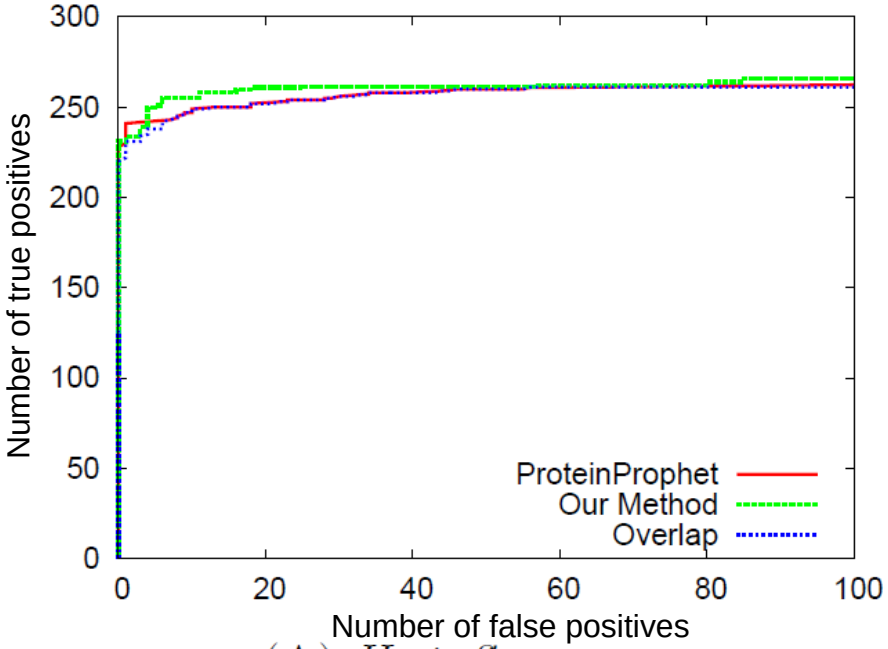


- Split zero-probability proteins in two.
- This allows the creation of two smaller connected components.
- When necessary, prune more aggressively.

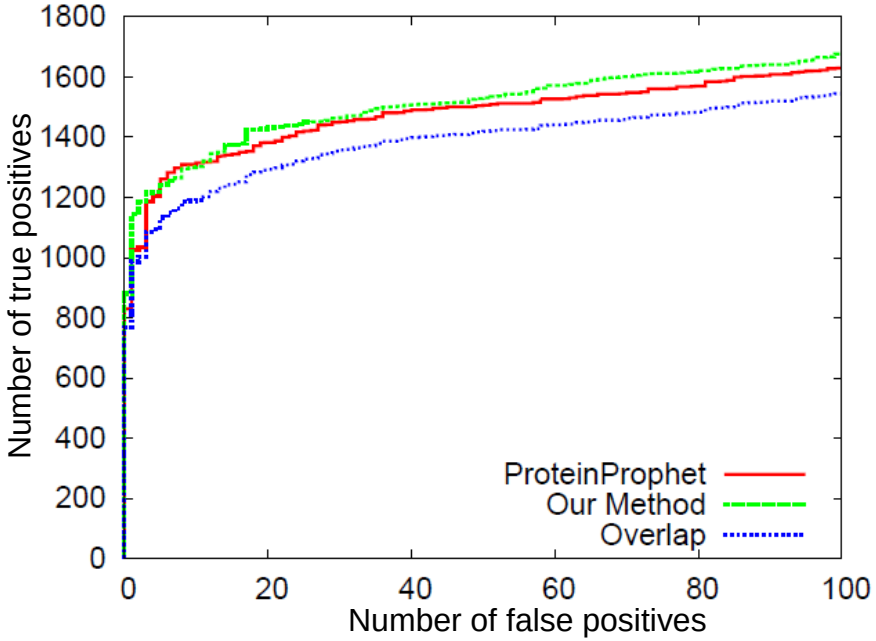
Effects of speedups

	H. Influenzae	Yeast	ISB 18	Sigma 49
PSMs	29,123	10,390	21,166	23,694
Proteins	32,748	3742	1777	392
Edges	60,844	12,202	21,720	24,392
Full problem	33,000	3700	1800	390
After partitioning	930	74	15	11
After clustering	170	47	15	11
After pruning	60	47	15	11

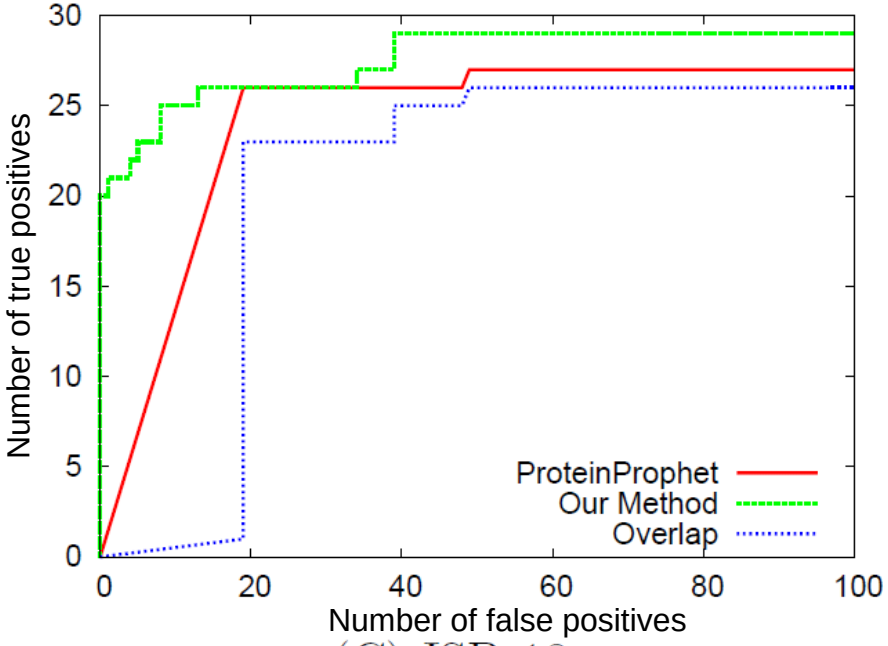
Numbers in the lower half of the table represent the $\log_2$ of the size of problem.



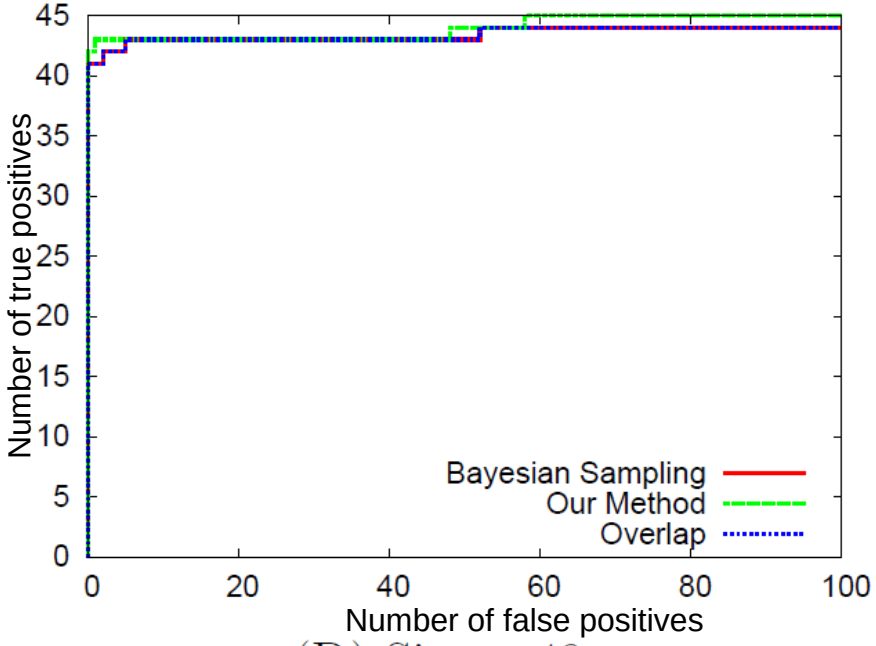
(A) *H. influenzae*



(B) Yeast

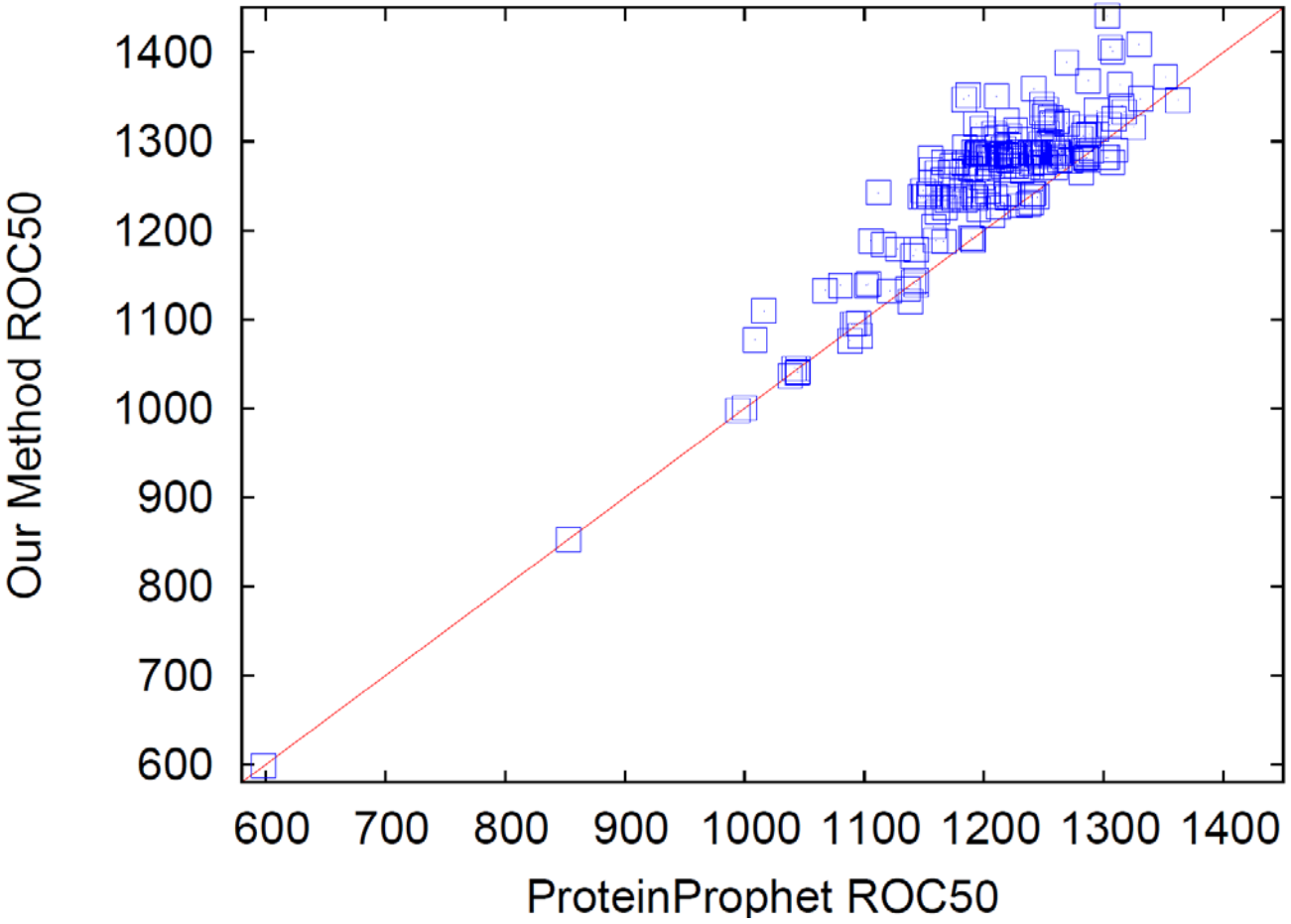


(C) ISB 18



(D) Sigma 49

Robustness to parameter choice



- Results from all ISB 18 data sets.
- Parameters selected using the *H. influenzae* data set.

Conclusions

- We provide a simple probability model and a method to efficiently compute exact protein posteriors.
- The model performs as well or slightly better than the state of the art.

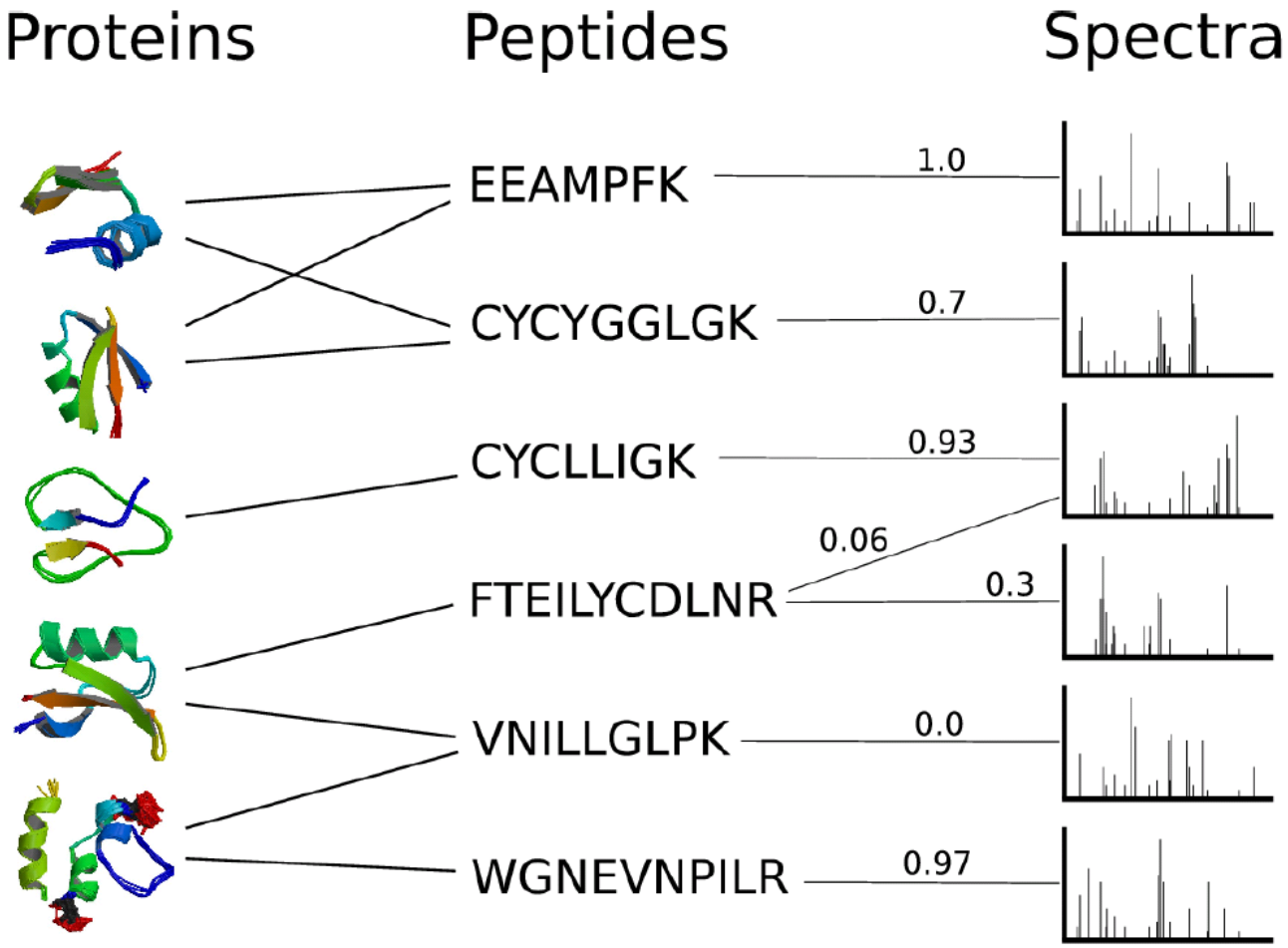
Direct maximization of protein identifications from tandem mass spectra

Marina Spivak

Jason Weston

Michael MacCoss

The protein ID problem



Key ideas

Previous methods:

- First compute a single probability per PSM, then do protein-level inference.
- First control error at peptide level, then at the protein level.

Our approach:

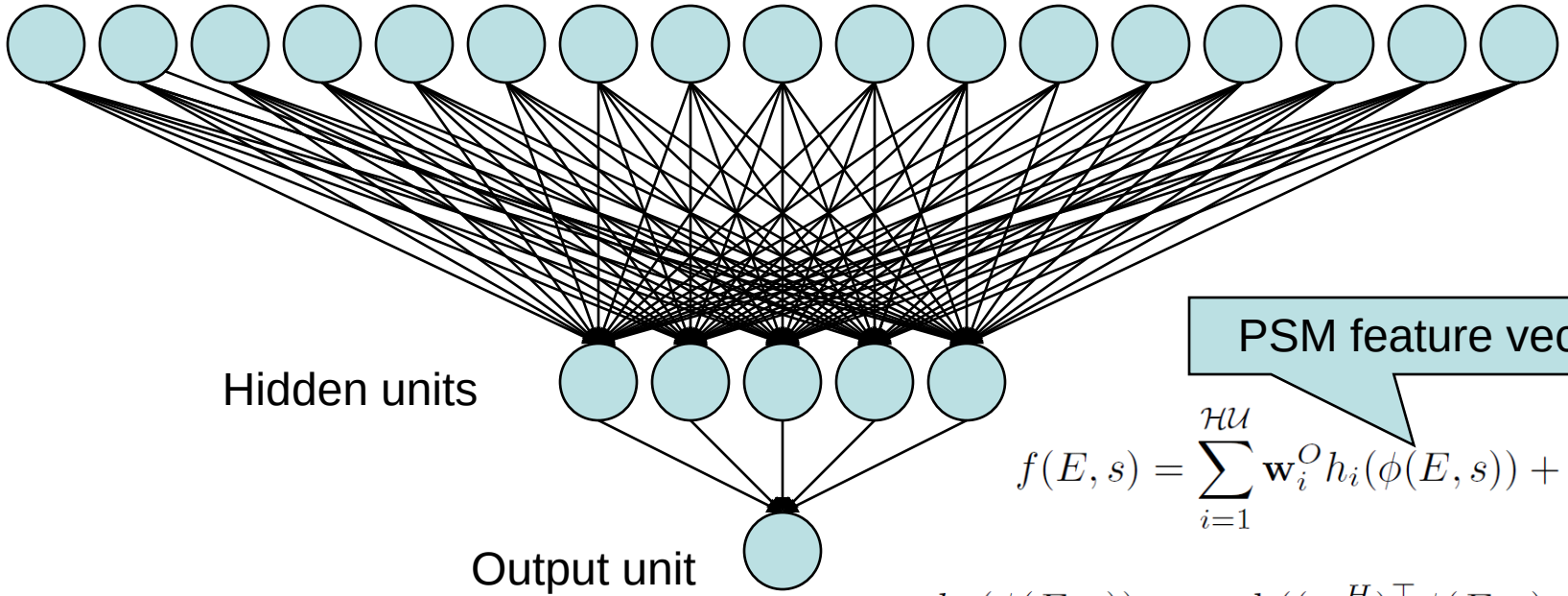
- Perform a single joint inference, using a rich feature representation.
- Directly minimize the protein-level error rate.

Features representing each PSM

- Cross-correlation between observed and theoretical spectra (XCorr)
- Fractional difference betweeen 1st and 2nd XCorr.
- Fractional difference between 1st and 5th XCorr.
- Preliminary score for spectrum versus predicted fragment ion values (Sp)
- Natural log of the rank of the Sp score.
- The observed mass of the peptide.
- The difference between the observed and theoretical mass.
- The absolute value of the previous feature.
- The fraction of matched b- and y-ions.
- The log of the number of database peptides within the specified mass range.
- Boolean: Is the peptide preceded by an enzymatic (tryptic) site?
- Boolean: Does the peptide have an enzymatic (tryptic) C-terminus?
- Number of missed internal enzymatic (tryptic) sites.
- The length of the matched peptide, in residues.
- Three Boolean features representing the charge state.

PSM scoring

Input units: 17 PSM features



PSM feature vector

$$f(E,s) = \sum_{i=1}^{\mathcal{HU}} \mathbf{w}_i^O h_i(\phi(E,s)) + b,$$

$$h_k(\phi(E,s)) = \tanh((\mathbf{w}_k^H)^\top \phi(E,s) + b_k)$$

The Barista model

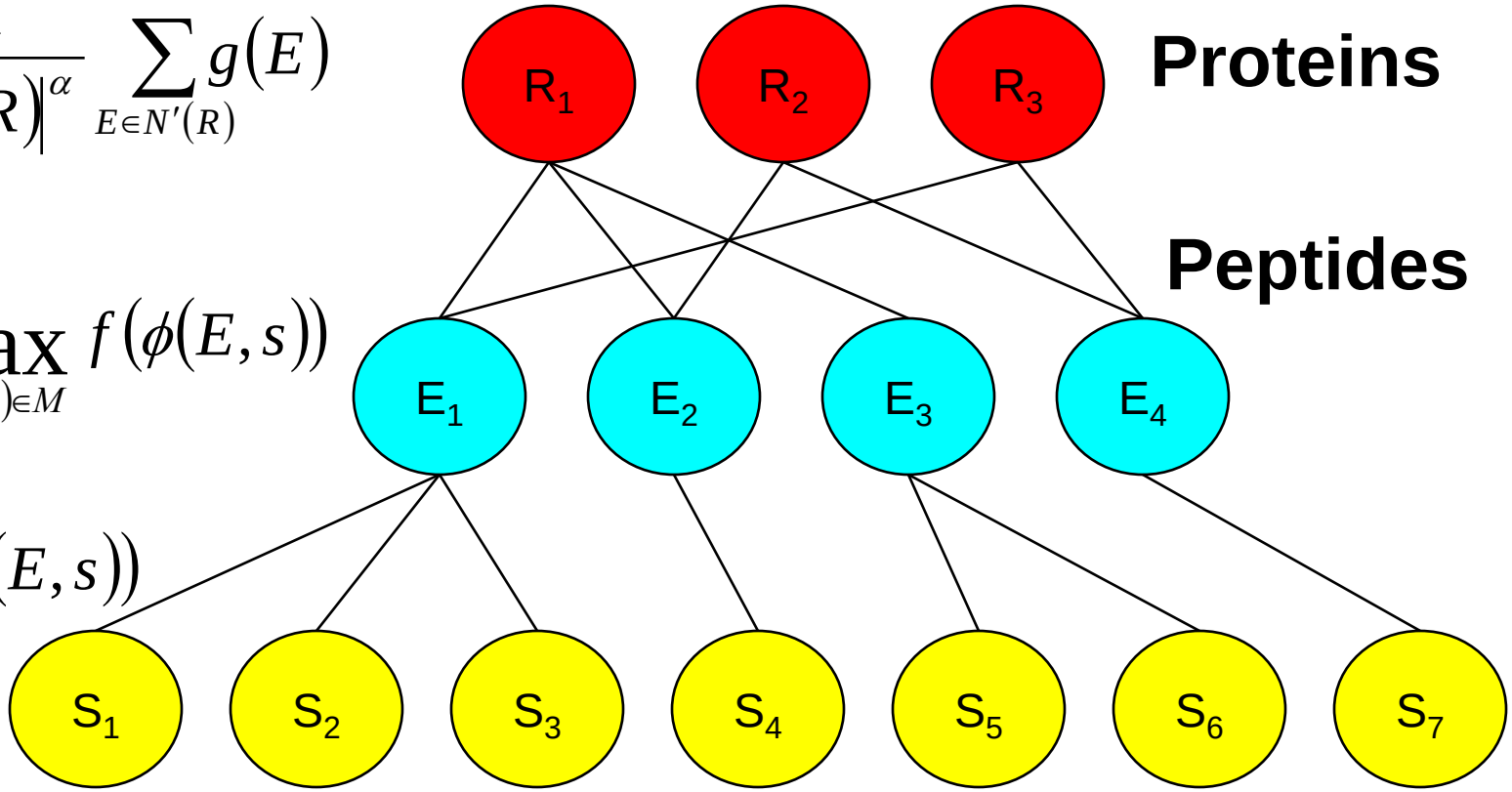
Number of peptides in protein R

$$F(R) = \frac{1}{|N(R)|^{\alpha}} \sum_{E \in N'(R)} g(E)$$

$$g(E) = \max_{s:(E,s) \in M} f(\phi(E,s))$$

$f(\phi(E,s))$

Neural network score function

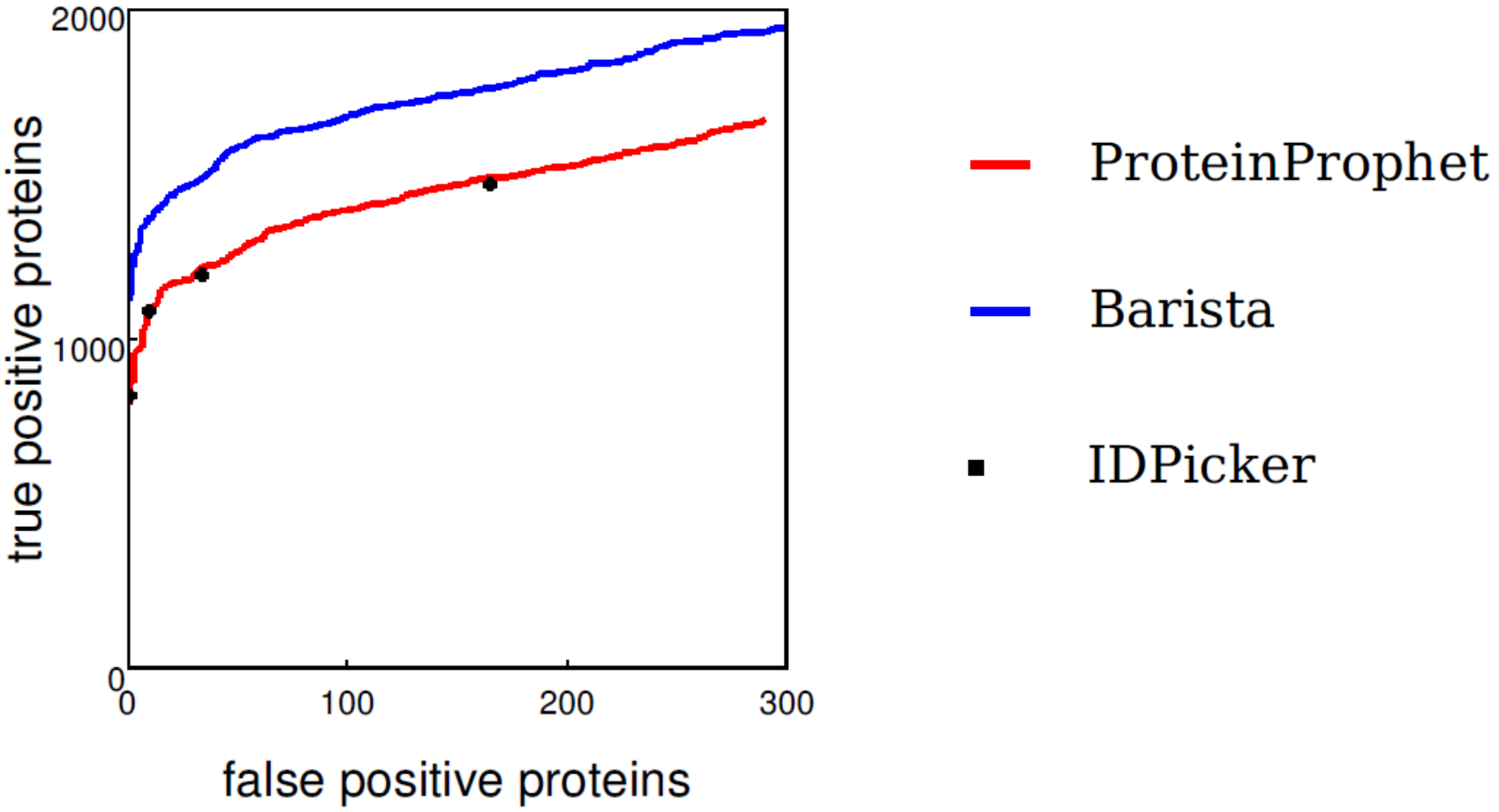


Model Training

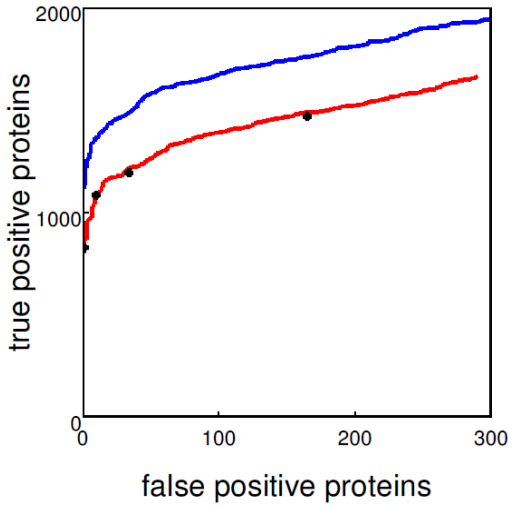
repeat
    Pick a random protein (R_i, y_i)
    Compute $F(R_i)$
    if $(1 - yF(R_i)) > 0$ then
        Make a gradient step to optimize $L(F(R_i),y_i)$
    end if
until convergence

- Search against a database containing real (*target*) and shuffled (*decoy*) proteins.
- For each protein, the label $y \in \{+1, -1\}$ indicates whether it is a target or decoy.
- Hinge loss function: $L(F(R),y) = \max(0, 1-yF(R))$
- Goal: Choose parameters *W* such that $F(R) > 0$ if $y = 1$, $F(R) < 0$ if $y = -1$.

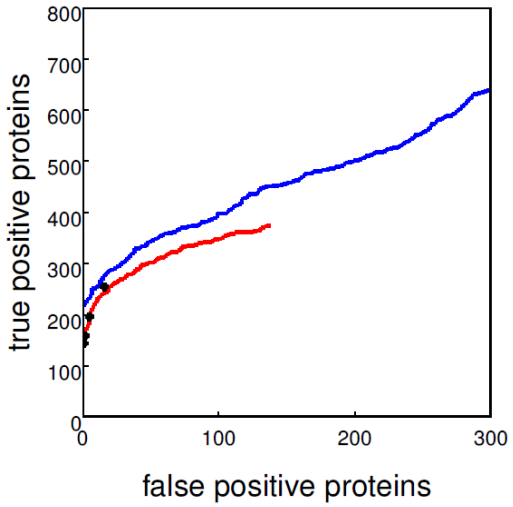
Target/decoy evaluation



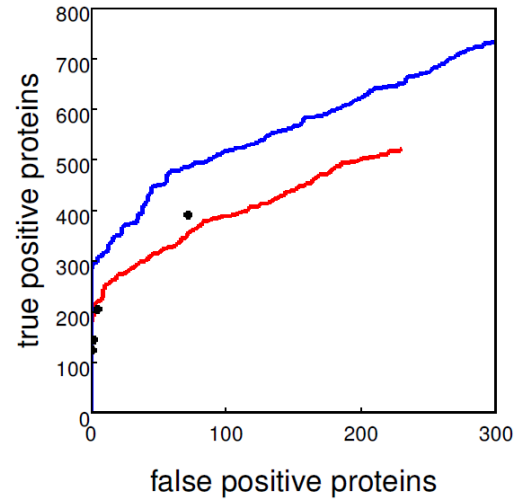
(A) Yeast trypsin



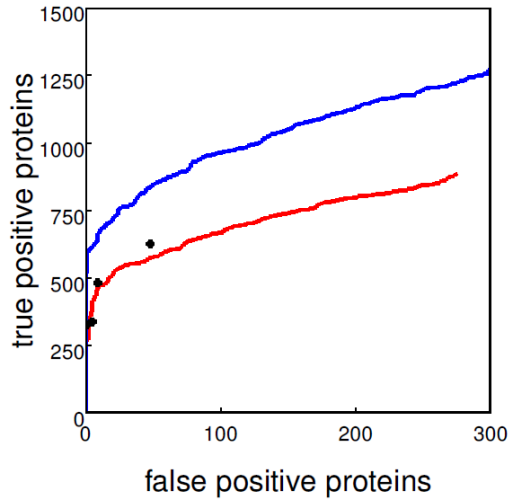
(B) Yeast elastase

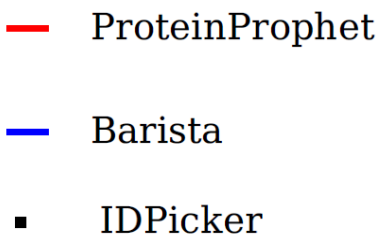


(C) Yeast chymotrypsin

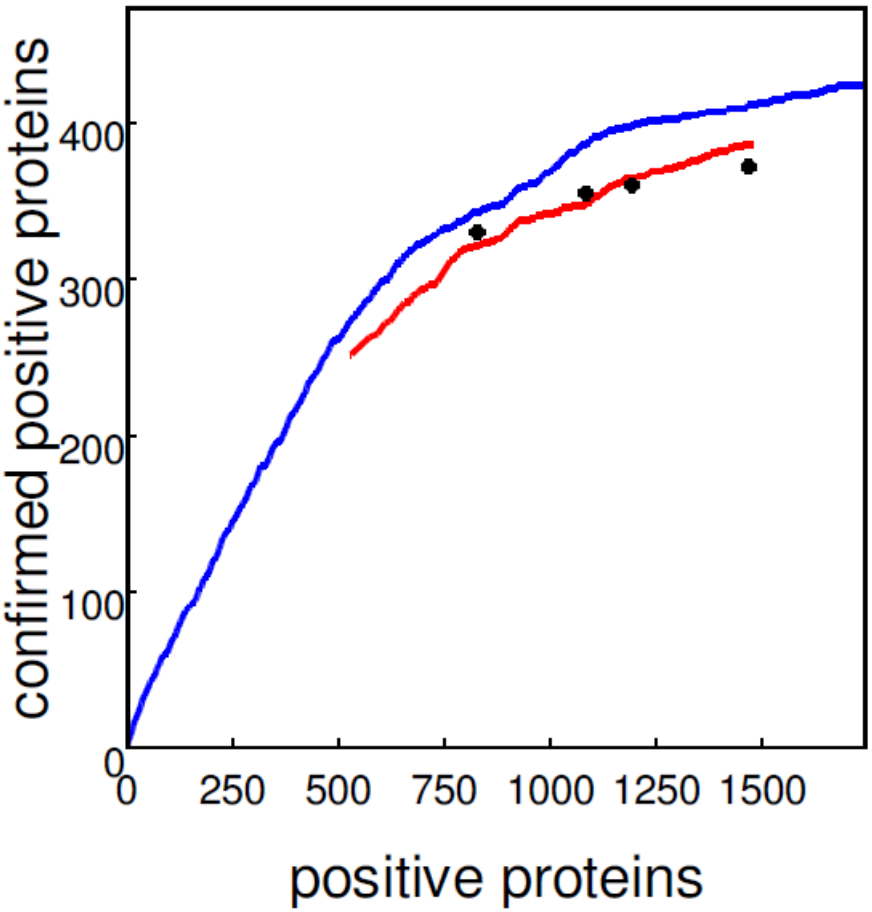


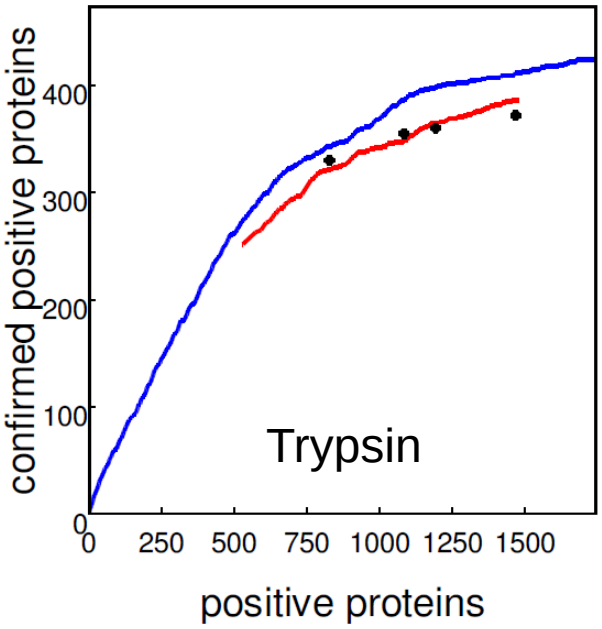
(D) Worm trypsin

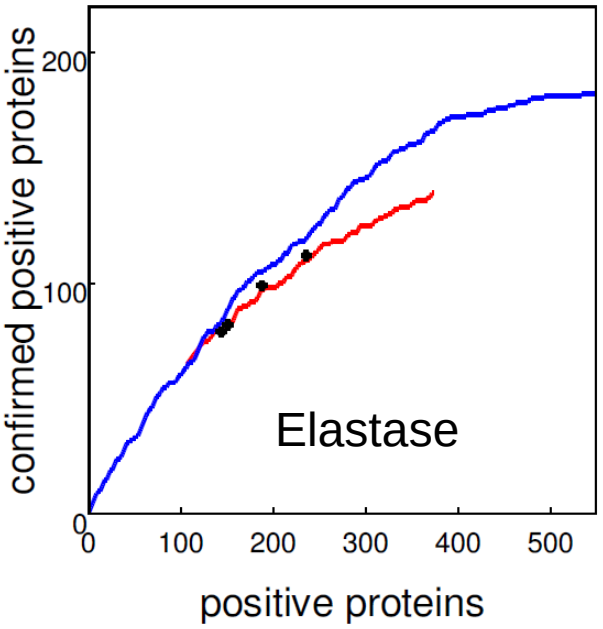


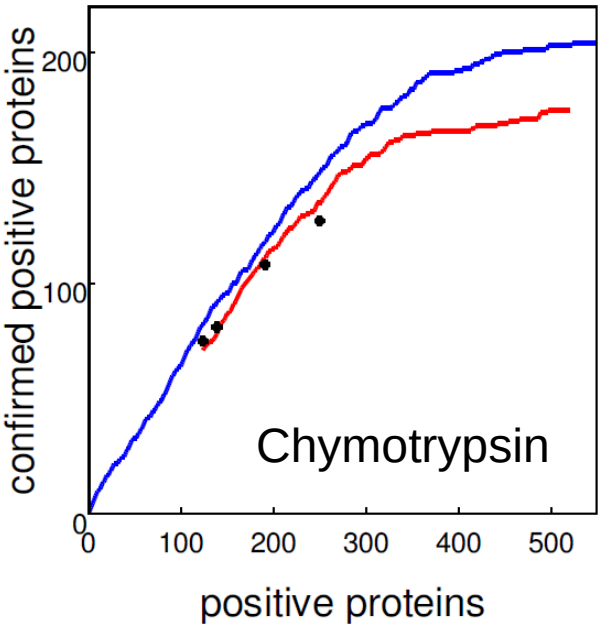


External gold standard





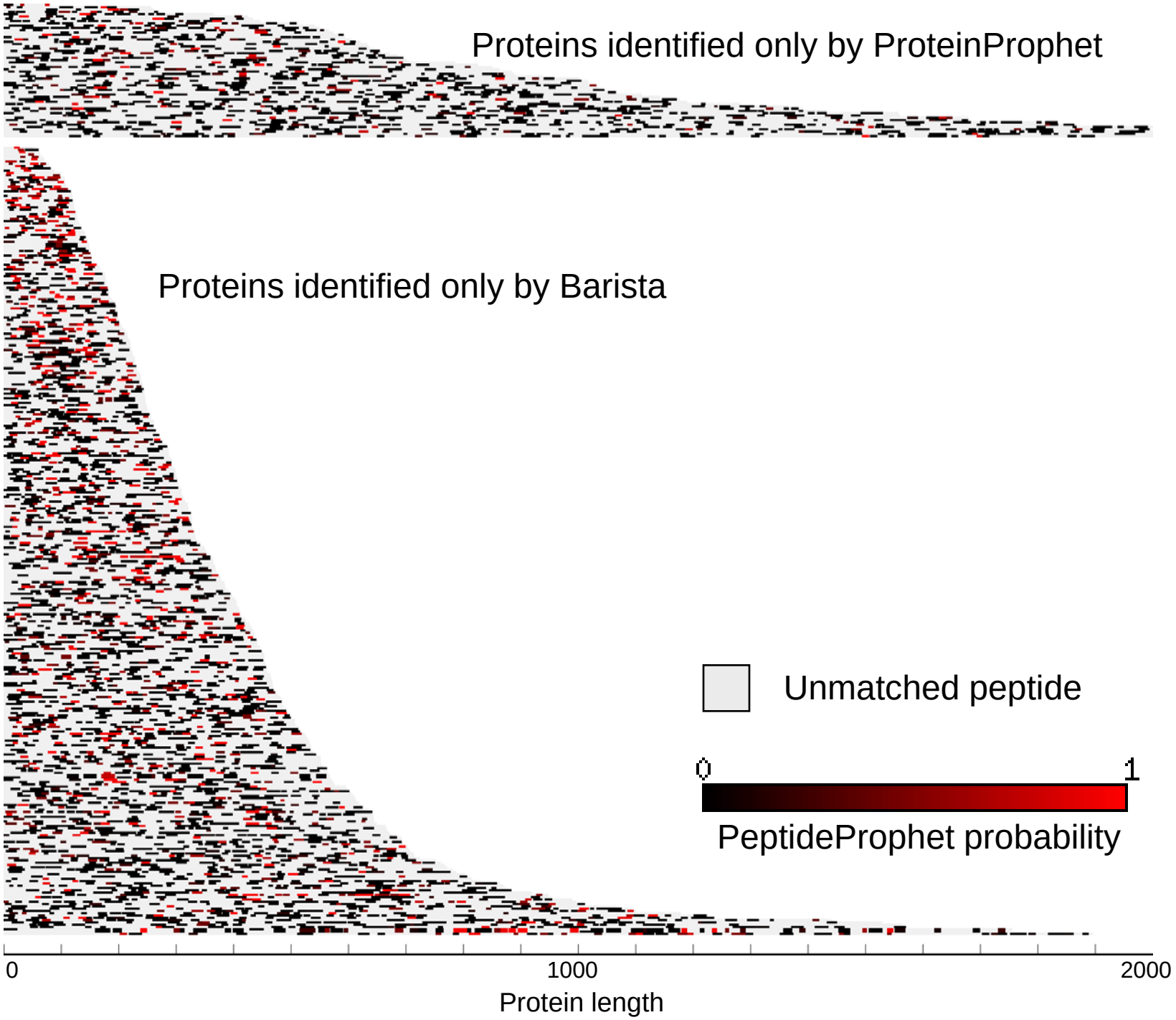




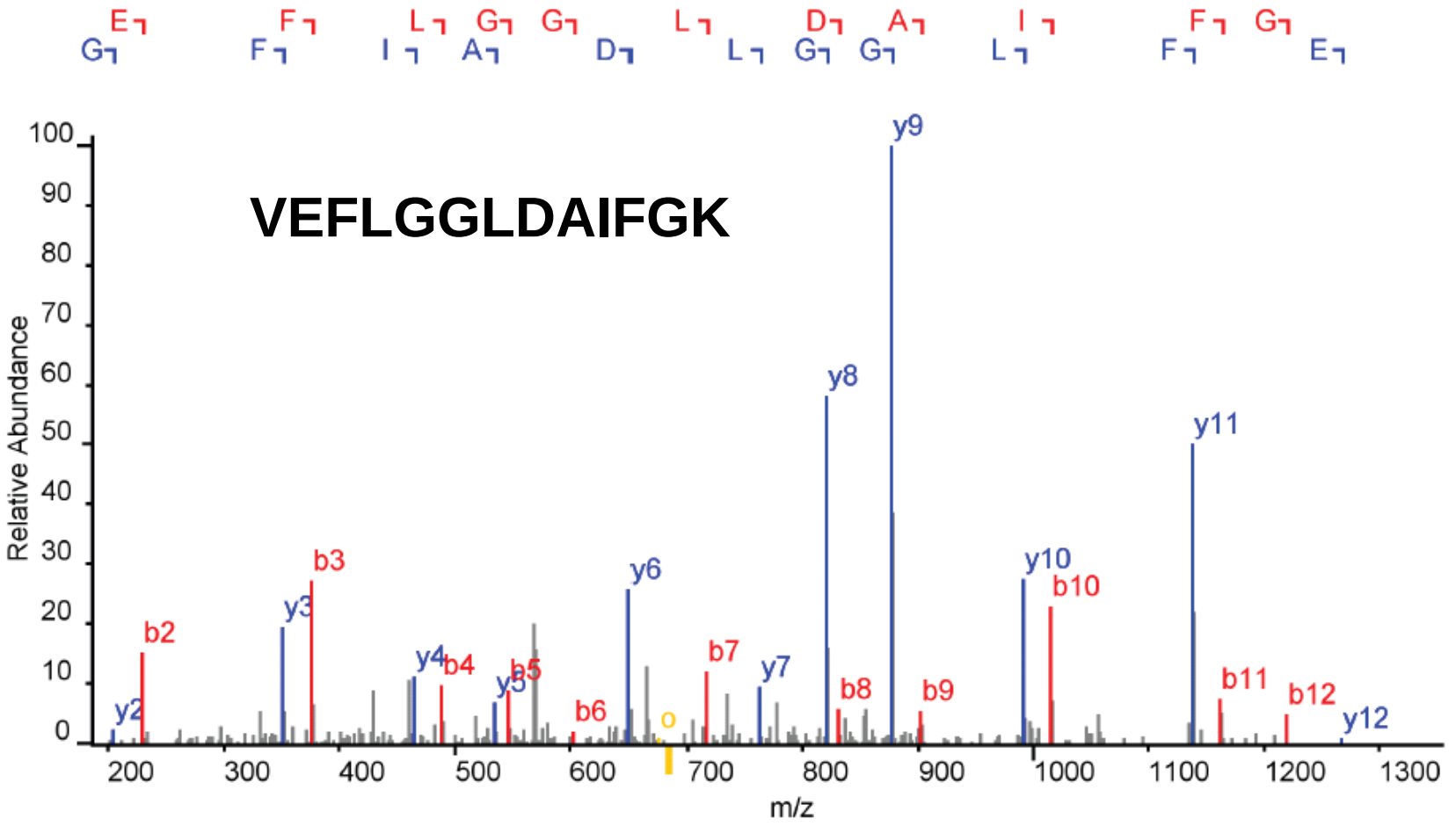
ProteinProphet

Barista

IDPicker


Proteins identified only by ProteinProphet
Proteins identified only by Barista
Unmatched peptide
0
1
PeptideProphet probability
0
1000
2000
Protein length

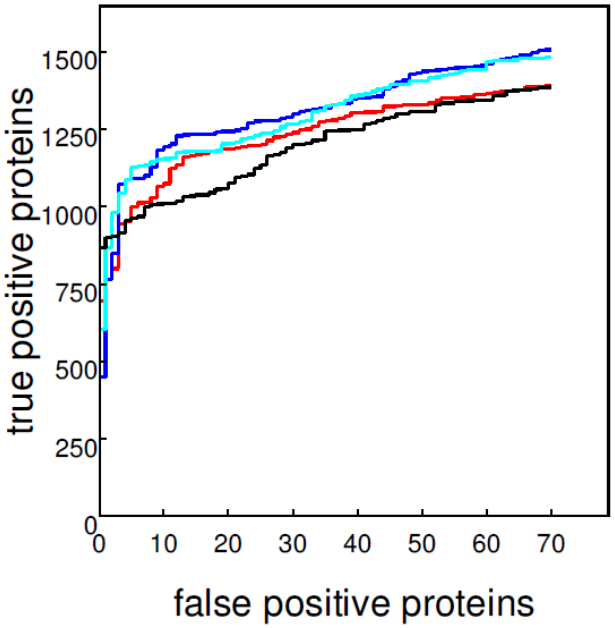
One-hit wonder



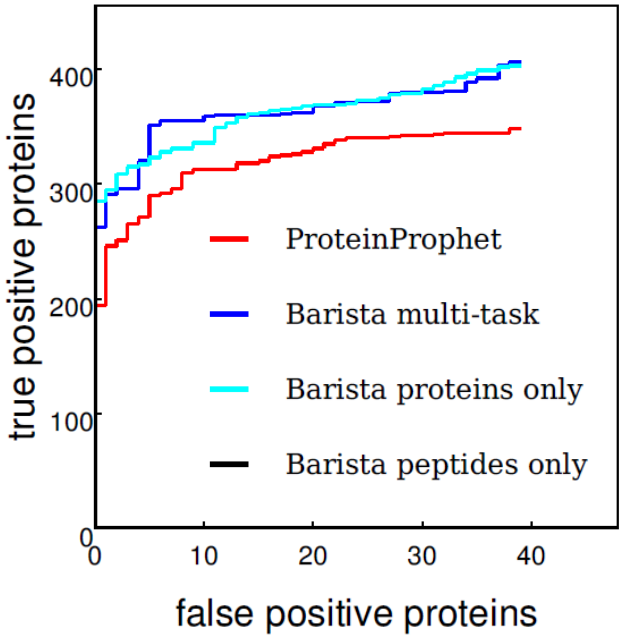
MVNVK|VEFLGGLDAIFGK|QR|VHK|IK|MDK|EDPVTVGDLIDHIVSTMINNPNDVSIFIEDDSIRPGIITLINDTDWELEGEK|DYILEDGDIISFTSTLHGG

Multi-task results

Peptide level evaluation



Protein level evaluation



- At the peptide level, multi-tasking improves relative to either single-task optimization.
- At the protein level, multi-tasking improves only relative to peptide level optimization.

Conclusions

- Barista solves the protein identification in a single, direct optimization.
- Barista takes into account weak matches and normalizes for the total number of peptides in the protein.
- Multi-task learning allows for the simultaneous optimization of peptide- and protein-level rankings.

Take-home messages

- Generative models and discriminative, direct optimization techniques are both valuable.
- Developing application-specific algorithms often provides better results than using out-of-the-box algorithms.

Machine Learning in Computational Biology workshop

MLCB

- Affiliated with NIPS
- Whistler, BC, Canada
- December 11-12, 2009
- Unpublished or recently published work.
- 6-page abstracts due September 27.

http://www.mlcb.org