BMC Bioinformatics

BioMed Central

Proceedings

Protein Ranking by Semi-Supervised Network Propagation

Jason Weston*[1], Rui Kuang[2,4], Christina Leslie[2] and William Stafford Noble[3]

Address: [1]NEC LABS AMERICA, 4 Independence Way, Princeton, NJ, USA, [2]Center for Computational Learning Systems, Columbia University, Interchurch Center, 475 Riverside Dr., New York, USA, [3]Department of Genome Sciences, Department of Computer Science and Engineering, University of Washington, 1705 NE Pacific Street, Seattle, WA, USA and [4]Department of Computer Science, Columbia University, 1214 Amsterdam Avenue, New York, NY, USA

Email: Jason Weston* - jasonw@nec-labs.com; Rui Kuang - rkuang@cs.columbia.edu; Christina Leslie - cleslie@cs.columbia.edu; William Stafford Noble - noble@gs.washington.edu

* Corresponding author

from NIPS workshop on New Problems and Methods in Computational Biology
Whistler, Canada. 18 December 2004

Published: 20 March 2006

BMC Bioinformatics 2006, **7**(Suppl 1):S10 doi:10.1186/1471-2105-7-S1-S10

Abstract

Background: Biologists regularly search DNA or protein databases for sequences that share an evolutionary or functional relationship with a given query sequence. Traditional search methods, such as BLAST and PSI-BLAST, focus on detecting statistically significant pairwise sequence alignments and often miss more subtle sequence similarity. Recent work in the machine learning community has shown that exploiting the global structure of the network defined by these pairwise similarities can help detect more remote relationships than a purely local measure.

Methods: We review RankProp, a ranking algorithm that exploits the global network structure of similarity relationships among proteins in a database by performing a diffusion operation on a protein similarity network with weighted edges. The original RankProp algorithm is unsupervised. Here, we describe a semi-supervised version of the algorithm that uses labeled examples. Three possible ways of incorporating label information are considered: (i) as a validation set for model selection, (ii) to learn a new network, by choosing which transfer function to use for a given query, and (iii) to estimate edge weights, which measure the probability of inferring structural similarity.

Results: Benchmarked on a human-curated database of protein structures, the original RankProp algorithm provides significant improvement over local network search algorithms such as PSI-BLAST. Furthermore, we show here that labeled data can be used to learn a network without any need for estimating parameters of the transfer function, and that diffusion on this learned network produces better results than the original RankProp algorithm with a fixed network.

Conclusion: In order to gain maximal information from a network, labeled and unlabeled data should be used to extract both local and global structure.

Background

Pairwise sequence comparison is the "killer app" of bioinformatics. Algorithms like BLAST [1] and PSI-BLAST [2] allow a user to search a database of DNA or protein sequences using a single query sequence. The output of the search is a list of database sequences (called *targets*) that are ranked according to their similarity to the query. The similarities discovered by the algorithm may help the user infer functional properties of the query and target sequences; for example, a query sequence of unknown function that retrieves from the database a large collection of kinases is likely itself to be a kinase. This straightforward application is familiar to most molecular biologists. The web engine of the most popular pairwise sequence comparison algorithm, the BLAST server at the NCBI, runs 50,000 searches per day.

Early methods for detecting subtle sequence similarities were designed explicitly with respect to a simple model of molecular evolution. They measure similarities between protein pairs by computing the cost of mutation, insertion and deletion. The Smith-Waterman algorithm [3] is a provably optimal, quadratic time, dynamic programming algorithm to solve this problem, and BLAST is a linear time heuristic approximation algorithm [1].

More sophisticated solutions to this problem involve learning from data. In an example of this approach, an HMM or other generative model is constructed from a training set, which is accumulated iteratively from the target database. SAM-T98 [4] is an example of an iterative profile HMM method, and PSI-BLAST [2] is essentially a fast approximation of this approach. PSI-BLAST builds an alignment-based statistical model of a local region in the protein similarity network and then iteratively collects additional sequences from the database, adding them to the multiple alignment. The main idea is to characterize with a statistical model the family or superfamily that the query protein comes from. This model is then more capable than the original query sequence of finding other similar proteins. From a machine learning perspective, these methods can be described as *unsupervised learning* methods. While they learn from unlabeled database sequences in order to build a probabilistic model, they do not make use of known structural or functional information, which is available for a subset of the database of target proteins.

RankProp

In this article we review RankProp, an unsupervised protein ranking algorithm. We also discuss some extensions to the algorithm that leverage the use of labeled data to make it a *semi-supervised learning* method. RankProp [5,6] works by defining a protein similarity network. In this network, nodes are proteins, and edges represent pairwise protein similarities identified using the PSI-BLAST algorithm. Given a query (a node in this network) the algorithm performs a diffusion operation on the graph. Each node is assigned an initial activation level. In subsequent iterations, the activation level of a node is given by the weighted combination of neighboring nodes plus a stability term based on its initial activation. The query node has its activation set to a constant value. Repeated iterations of this procedure result in a diffusion of the query's activation level across the network, until a fixed point is reached. The output of the algorithm is a list of target proteins, ranked by activation level. A target protein can achieve a high ranking by being connected to many proteins similar to the query, even if its direct connection to the query is not a strongly weighted edge. RankProp has been shown to significantly outperform BLAST and PSI-BLAST, when tested on its ability to recognize remote homologs in the SCOP [7] database of protein domain structures [5]. RankProp is inspired by two separate sources: first, diffusion techniques from machine learning [6], and second by the Google ranking algorithm, PageRank [8].

In semi-supervised learning, one is interested in using a large unlabeled set of data to help build a classification rule. Common scenarios for this situation include text and web page classification. These algorithms work by making the so-called cluster assumption: the classification rule does not change in regions of input space that are densely populated. In other words, the algorithm chooses a classification decision boundary that lies in a region of low density (see Figure 1). Clearly, the cluster assumption will not always hold true, but for many practical applications the assumption is reasonable. For example, for the problem of handwritten digit recognition, the region of space between the digits "2" and "0" is sparsely populated because people will write true digits more often than "in-between" digits that have no meaning. We know that the cluster assumption is very likely to be true for the protein ranking problem, because semi-supervised techniques have previously been successful in the protein classification problem [9,10]. Similar to semi-supervised classification methods, which capture cluster and manifold structure in the data, RankProp assumes that a protein lying in the same cluster or manifold with the query should be ranked higher, even if its similarity with the query using the original metric is not very high. An example is given in Figure 2.

An analogy can also be made between a protein database search, where one submits a query protein and is returned a ranked list of proteins, and a web search, where one enters a query word or phrase and retrieves a ranked list of web pages. RankProp is similar to PageRank [8], the ranking system used by Google. Both PageRank and RankProp are based upon constructing a network of object nodes with edges weighted by similarity between them. A diffusion operation is then performed to capture global properties of the network. PageRank ranks a web page more highly if it is linked with other highly ranked pages. See [6] for more details.

RankProp with labeled data

We hypothesize that extending RankProp to make use of both *unlabeled* and *labeled* data will provide a significant improvement in the resulting rankings, compared to the rankings produced by the original RankProp algorithm. In this context, labels come from 3D structural information. The 3D structure that a protein assumes after folding largely determines its function in the cell. It is far easier to

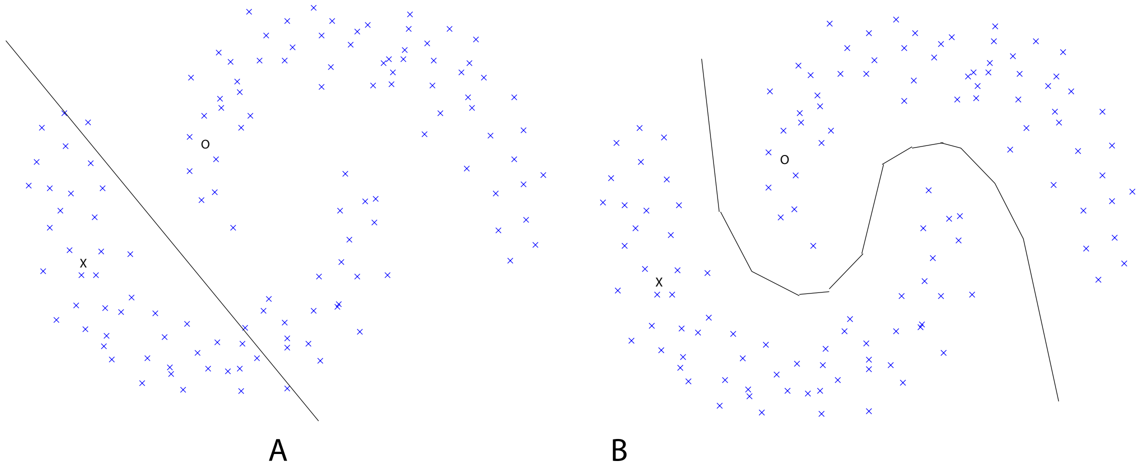

Figure 1

An example of the use of semi-supervised classification. (A) If only one example of each class is given (the large cross and circle), then one may choose an incorrect decision boundary between the two classes. (B) Given extra unlabeled data (small crosses), the decision boundary can be placed in a sparse region of the space, which can result in a more accurate classifier.

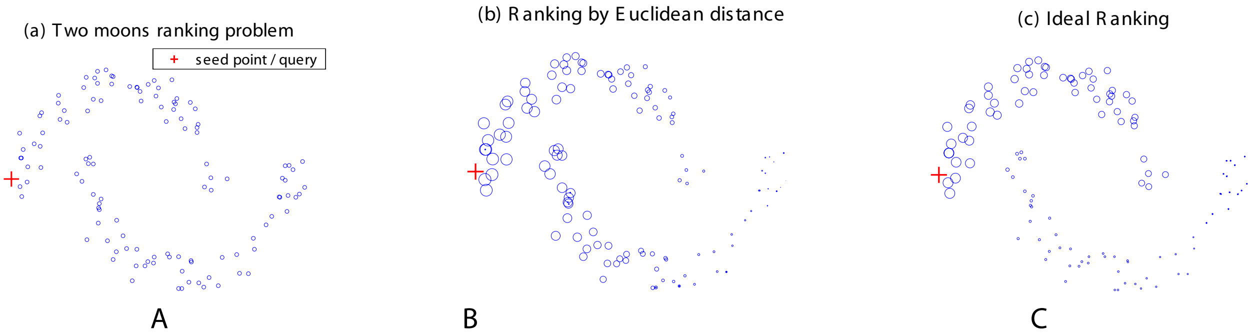

Figure 2

Semi-supervised learning for a ranking problem. (A) The query is a single point in input space, and the remaining points comprise the database one wishes to rank. (B) The ranking induced by Euclidean distance. Marking sizes are proportional to the ranking of each point. (C) The ideal ranking. Clearly, to find the optimal ranking we need to find the cluster/manifold structure in the data.

determine experimentally a protein's primary amino acid sequence than it is to discover its 3D structure. However, protein structure databases contain thousands of solved 3D structures. Thus, from a machine learning perspective, in addition to the large amount of unlabeled data — on the order of one million protein sequences — we also have available a small amount of labeled data — roughly 27,000 proteins with known 3D structures, organized into structural classes. We investigate three different ways of using this labeled data: (i) as a validation set to choose from competing similarity networks, (ii) to design edge weights that correspond exactly to the similarity measure of interest, the probability of homology, and (iii) to learn which similarity network to use on a per-query basis. Of these methods, the third method provides the best-performing algorithm across a range of evaluation metrics and superfamily sizes.

Results

Basic approach

The RankProp algorithm requires a protein similarity network as input. The protein similarity network represents the similarity between pairs of proteins by assigning a weight to each edge. The degree of similarity between two sequences is commonly summarized in an E-value, which is the expected number of times that this degree of sequence similarity would occur in a random database of the given size. RankProp bases its edge weights on E-values returned from PSI-BLAST searches, using a radial basis transfer function

$$W_{ij} = exp(-E_{ij}/\sigma), \qquad (1)$$

where E_{ij} is the E-value between protein *i* and *j*, and W_{ij} is the corresponding weight. In this way, edges between similar sequences are assigned large weights. The transfer function introduces a hyper-parameter σ, the radial basis width, which controls the importance of very similar proteins relative to distant ones.

We evaluate RankProp output using a 3D structure-based gold standard [7], measuring the extent to which known homologs occur above non-homologs in the ranked list. The protein network consists of 7329 SCOP domains and 101, 602 proteins from Swiss-Prot version 40. The SCOP domains were split into two portions: 379 superfamilies (4071 proteins) for training and 332 (2899 proteins) for testing (used as queries). For more details of the dataset see the Methods section.

We use receiver operating characteristic (ROC) curves to measure performance. The ROC score [11] is used to compare methods for a given query by measuring the area under a curve that plots true positives as a function of false positives for varying classification thresholds, where a true positive is an example that belongs to the same superfamily as the query, and a false positive is an example that is not in the same fold. The ROC$_n$ score computes this score up to the *n*th false positive [12]. A value of 1 implies that the algorithm successfully assigns all the true relationships higher scores than the false relationships. For a random ranking of this data, the expected ROC$_{50}$ score is close to 0 because most of the sequences are not related to the query.

Our experiments suggest that RankProp's ranking is superior to the ranking induced by the direct links in the original network, i.e. the ranking given by the PSI-BLAST algorithm, as shown in Figure 3 and Table 1.

So far, we have described RankProp as a purely unsupervised approach. However, we would also like to make use

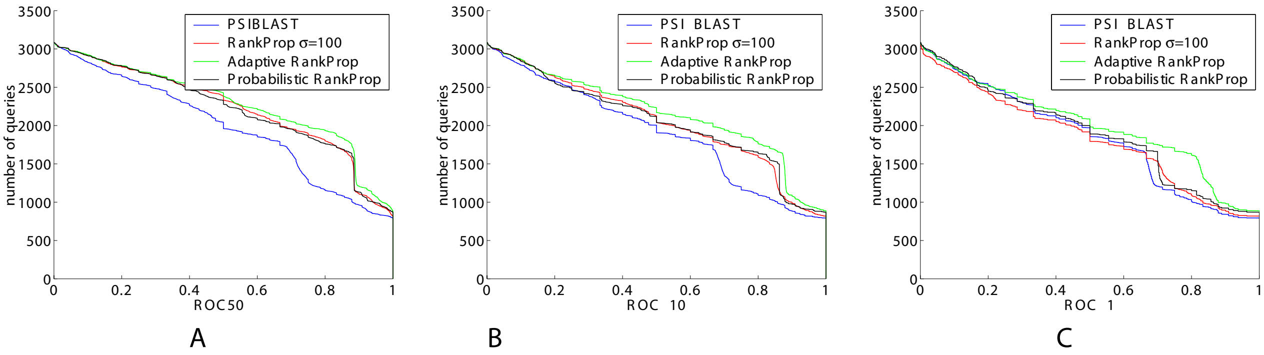

Figure 3

Comparison of PSI-BLAST and RankProp variants. Each figures plots the number of test set queries for which a given method achieves a ROC$_n$ score threshold. Figures (A), (B) and (C) use ROC$_{50}$, ROC$_{10}$ and ROC$_1$ scores, respectively. Corresponding mean ROC$_n$ scores are given in Table 1. The variants of RankProp are described in the text. Previous work [5] used RankProp with σ = 100.

Table 1: Comparison of PSI-BLAST and RankProp variants. The table lists ROC_1, ROC_{10} and ROC_{50} scores, averaged across 2899 SCOP domains in the test set, for PSI-BLAST and five variants of the RankProp algorithm.

	ROC_1	ROC_{10}	ROC_{50}
PSI-BLAST	0.5973	0.6167	0.6406
RankProp $\sigma = 10$	0.5964	0.6658	0.7169
RankProp $\sigma = 100$	0.5924	0.6671	0.7249
RankProp $\sigma = 1000$	0.6040	0.6661	0.6999
Probabilistic RankProp	0.6145	0.6660	0.7195
Adaptive RankProp	0.6510	0.7000	0.7420

of the labeled data available, which can be done by learning some aspect of the network with the available labels. In the following subsections, we consider three ways to achieve this goal.

Model selection of radial basis width

RankProp takes as input a weighted protein similarity network. Clearly, the quality of the rankings produced by RankProp depends critically on the quality of the initial weights. If we can parameterize the weights in some way, then these parameters can be inferred using labeled data.

Our first method for making use of labeled data simply learns the radial basis width parameter σ from Equation (1). This approach requires running RankProp with each sequence in a labeled training set for each value of σ. The ROC_n scores of the resulting rankings can then be used to select an optimal value for σ. This procedure was performed in the original RankProp paper [5]. Table 1 shows that selecting an appropriate value of σ can significantly affect the performance of the RankProp algorithm.

The probability of homology network

The RankProp algorithm with the transfer function (1) requires that the user specify the radial basis width parameter σ in advance. However, selecting an appropriate value for σ is difficult because the resulting edge weights have no clear semantics. Perhaps the most intuitive choice of edge weight between two proteins is the probability that the two proteins are structurally related, since our final measurement of success is the ranking performance based on the same structural relation.

Our second method for making use of labeled data uses this probabilistic formulation. In particular, we suggest an empirical approach for estimating edge probabilities from labeled data. In order to perform superfamily detection by network propagation, the most natural weight to assign to an edge between proteins i and j is the probability that i and j belong to the same superfamily. Using a labeled set of proteins, we discretize the range of possible E-values, and for each resulting E-value bin, we compute the frequency of pairs of proteins in that bin being in the same superfamily. Figure 4 compares the resulting empirical mapping to the original mapping for various values of σ. This probabilistic method is parameterless and yields a transfer function that is similar to the best performing value $\sigma = 100$ [5] of the original algorithm.

Overall, the probabilistic network provides performance comparable or superior to all values of σ we tried, as shown in Figure 3 and Table 1, measured using ROC_1, ROC_{10} and ROC_{50} scores. However, the improvement is less convincing at the ROC_{10} and ROC_1 levels, i.e., when the very highest ranked proteins become increasingly important. It is not surprising that RankProp has similar ROC_1 performance to PSI-BLAST, because examples very close to the query using the original similarity metric are usually already highly ranked.

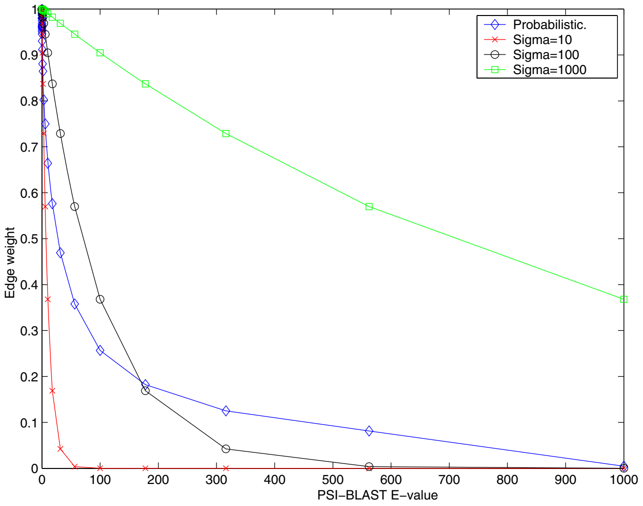

Figure 4

Comparison of transfer functions for converting PSI-BLAST E-values to edge weights. The figure plots edge weight as a function of E-value for various transfer functions. The original RankProp algorithm assigns edge weights using the function exp($-D_{ij}/\sigma$), where σ had to be chosen *a priori*. Curves for $\sigma = 10, 100, 1000$ are shown. Also shown is the curve produced by the probabilistic approach, in which the edge weight is the empirical probability of two proteins belonging to the same superfamily.

Although probabilistic RankProp does not outperform the original RankProp when used with the best choice of σ, the simple scheme of chosing σ via cross validation is very costly computationally. Given that the probabilistic network yields performance that is as good as the best choice of σ, we feel that it is a useful technique.

Adaptive model selection of radial basis width

We observed that the optimal value of σ for *a particular query* depends on the local density of the protein similarity network around that query. This effect can be seen in Figure 5. Here, we estimate the network density around a query by the number of strongly weighted (E-value less than 0.01) edges from the query to its neighbors. The figure shows that larger values of the radial basis width σ produce better error rates for queries in more densely populated areas of the network. In other words, RankProp appears to perform better on smaller superfamilies for small σ, and better on larger superfamilies for large σ.

This observation suggests our third strategy for making use of labeled data: *given the density of the network in the region of each query, use the labeled training data to learn which value of σ gives optimal performance*. Accordingly, we choose several values of σ (10,100, and 1000) and solve three corresponding regression problems to predict the ROC_n score given a histogram of number of hits with E-value less than *t* as input for each σ. In this experiment we use ROC_1, but in principle we could optimize for any error measure. Then, given the query, we choose the value of σ that is predicted to give the best ROC_1 score by the regressions. We call this method "adaptive RankProp." The results, given in Figure 3 and Table 1, show improved ROC_1, ROC_{10} and ROC_{50} scores. We note that the supplementary material of [5] also suggests to adapt the width σ per query, but there a hand-built rule of thumb was suggested, rather than choosing the width by learning from data.

Discussion

In this article we reviewed the RankProp algorithm and suggested some ways of using labeled data to further improve the basic algorithm. Based on our experiments, we advocate making use of all available information — in this case using both labeled and unlabeled data — to achieve the best possible results.

The basic way to use labeled data with the ranking problem is to optimize the parameters of the model of choice, which in this case is the protein similarity network. In the previous sections, we defined three possible parameterizations and then optimized them, in each case yielding good results. However, many other parameterizations are possible. For example, one could build a network based upon several measures of similarity, including pairwise sequence similarity metrics (BLAST, PSI-BLAST), common motif occurrences (MotifProp [13]), and predicted secondary structure similarity. The relative weights of these measures could then be learned. Another possibility is to put label information explicitly into the network: if two proteins are known to be homologs, then the edge weight can be set to one, and if they are known *not* to be homologs, then it can be set to zero. However, some preliminary experiments (not shown) indicated that this approach does not improve ranking performance.

An implementation of RankProp is now available on the Santa Cruz Gene Sorter, http://genome.ucsc.edu, featuring a pre-computed network of human genes. We plan to extend this implementation to use a larger database, and establish a separate web server capable of processing new queries, rather than operating on a pre-defined network.

Finally, one important difference between RankProp and existing methods such as BLAST and PSI-BLAST is that RankProp does not return an E-value or other confidence measure along with each ranked protein. Defining such a confidence measure is the subject of our current research.

Conclusion

The RankProp algorithm uses global network information to give improved protein rankings by performing diffusion on a graph built with PSI-BLAST similarity scores. PSI-BLAST improves upon BLAST by incorporating unlabeled data into its search algorithm, but advanced machine learning techniques appear to extract extra information useful for this task. In this article, we showed how labeled data can be used to further improve the unsupervised diffusion technique by learning various parameters of the similarity network. These results may have implications for other ranking problems in bioinformatics as well, as long as a suitable similarily network can be defined.

Methods

Data Preparation

We tested the quality of the protein rankings produced by RankProp, using as a gold standard the human-annotated SCOP database of protein 3D structural domains [7]. SCOP has been used as a gold standard in many previous studies (e.g., [14-16]). Sequences were extracted from verson 1.59 of the database, purged using ASTRAL [17] so that no pair of sequences shares more than 95% identity. The resulting collection of 7329 SCOP domains was split into two portions: 379 superfamilies (4071 proteins) for training and 332 (2899 proteins) for testing. Note that training and testing sequences never come from the same superfamily.

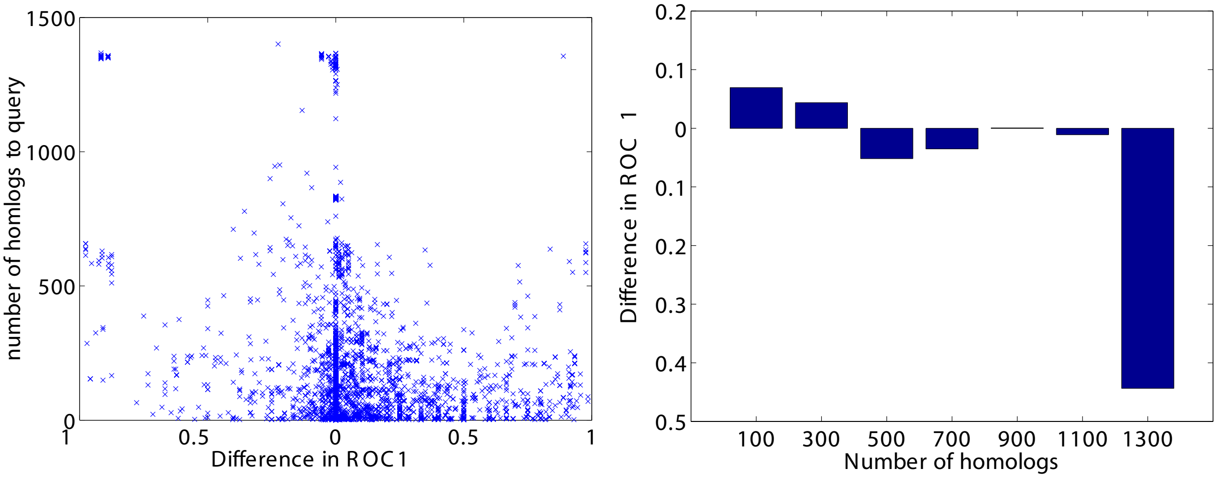

Figure 5

The optimal radial basis width σ depends upon the number of close homologs to the query. In the figure, each point corresponds to one query sequence from the training set. For each query, the y-axis coordinate is the number of target sequences that PSI-BLAST assigns an E-value less than 0.01, and the x-axis coordinate is the difference in ROC$_1$ between RankProp with parameters $\sigma = 10$ and $\sigma = 1000$. The same analysis is shown in bar chart form (right). Each bar indicates the mean difference in ROC for all queries with number of homologs falling into a bin of size 200. Similar results (not shown) arise for other values of σ that we tried. Here, the error measure is the ROC$_1$ score.

The SCOP database is organized hierarchically into classes, folds, superfamilies and families. For the purposes of this experiment, two domains that come from the same superfamily are assumed to be homologous, and two domains from different folds are assumed to be unrelated. For pairs of proteins in the same fold but different superfamilies, their relationship is uncertain, and so these pairs are not used in evaluating the algorithm.

In all the experiments reported here, the SCOP database was concatenated with 101,602 proteins from Swiss-Prot version 40. Using this larger database benefits both PSI-BLAST and RankProp.

PSI-BLAST

PSI-BLAST (v 2.2.2) was used for comparison with RankProp. PSI-BLAST was run with default parameters, including the BLOSUM 62 matrix, but with an E-value threshold of 10,000 for reporting results. PSI-BLAST was allowed to run a maximum of six iterations, which previous work indicates is sufficient for good performance [16], using the default E-value threshold of 0.005 for inclusion in the model.

RankProp

The protein similarity network for RankProp was built using the same version of PSI-BLAST as above. In the network K used by RankProp, the weight K_{ij} associated with a directed edge from protein i to protein j is exp(-$S_j(i)/\sigma$), where $S_j(i)$ is the E-value assigned to protein i given query j. For efficiency, the number of outgoing edges from each node is capped at 1000, unless the number of target sequences with E-values less than 0.05 exceeds 1000.

Given the similarity network, the RankProp algorithm can then be described as follows:

```
1. Initialization: $y_1(0) = 1; y_i(0) = 0$
2. for $t = 0,1, 2,...$ do
3.     for $i = 2$ to $m$ do
4.         $y_i(t+1) \leftarrow K_{1i} + \alpha \sum_{j=2}^{m} K_{ji} y_j(t)$
5.     end for
6. until convergence
```

7. **Termination**: Let y_i^* denote the limit of the sequence $\{y_i(t)\}$. Then y_i^* is the ranking score of the i^{th} point (largest ranked first).

Given a set of objects (in this case, proteins) $X = \{x_1, ..., x_m\}$, let x_1 be the *query* and $x_2, ..., x_m$ be the database (*targets*) that we would like to rank. Let K be the matrix of object-object similarities, i.e., K_{ij} gives a similarity score between x_i and x_j, with K normalized so that $\sum_{j=2}^{m} K_{ji} = 1$ for all i. For computational efficiency, we set $K_{1i} = K_{i1}$ for all i, so that we can compute weights involving the query using a single execution of PSI-BLAST. Let y_i, $i = 2, ..., m$, be the initial ranking "score" of a target. In practice, for efficiency, the algorithm is terminated after a fixed number I of iterations, and $y_i(I)$ is used as an approximation of y_i^*. In our experiments, RankProp was run for $I = 20$ iterations, which experiments in the supplement to [5] show brings the algorithm very close to convergence. The parameter $\alpha \in [0,1]$ is set *a priori* by the user. For $\alpha = 0$, no global structure is found, and the algorithm's output is just the ranking according to the original distance metric. All our experiments use $\alpha = 0.95$, looking for clear structure in the data. However, in principle this hyperparameter could be selected using labeled data as well.

RankProp with probability of homology network

It is possible to define a similarity network for RankProp without resorting to the adjustment of free parameters. This is accomplished by making use of labeled data. To perform superfamily detection using RankProp (performing network propagation) the most natural weight to assign to an edge from protein i to protein j is the probability that i and j belong to the same superfamily. We suggest computing exactly this probability from labeled data. The method requires a training set of m proteins with known labels y_i, and a matrix D of PSI-BLAST E-values for these proteins, where $D_{ij} = S_j(i)$ is the E-value between proteins i and j using j as the query and i as the target. We then compute a histogram of empirical frequencies for the PSI-BLAST E-values. More specifically, we choose bin centers v_k, and compute n_k, the number of times D_{ij} falls into the bin centered at v_k, and s_k, the number of times that the latter occurs when i and j are in the same superfamily. We then compute s_k/n_k, the empirical probability belonging to the superfamily of interest for the bin. The mapping $\hat{p}(x)$ that converts a PSI-BLAST E-value to a probability of homology is created is done by locating the two closest bins and using linear interpolation on the estimated probabilities. We (arbitrarily) choose the bin centers $v = (10^{-20}, 10^{-15}, 10^{-10}, 10^{-9.5}, ..., 10^{-4.5}, 10^{-4}, 10^{-3.75}, ..., 10^{3})$.

The resulting map is given in Figure 4 and compared to the $exp(-S_j(i)/\sigma)$ function of the original RankProp algorithm for different values of σ. The results show that it is as good as or better than any choice of σ. Although it does not improve over the best choice of σ, this method provides a very straightforward and computationally efficient method for building a strongly performing network. In comparison, choosing the value of σ by validation set is far slower to compute, as it involves running RankProp many times. The method described in this section simply requires one pass through the matrix of E-values generated from PSI-BLAST to compute its network.

RankProp with adaptive training

In adaptive RankProp, one chooses a different value of σ for building the protein similarity network per test example. To implement this approach, a supervised machine learning method is used to predict the best choice of σ for a given query. The choice is made by using a regression function to predict the ROC score than one would achieve on that query for a given value of σ. We learn a separate regression function for each possible choice of σ and choose the value of σ with the highest predicted ROC score.

The input to each regression problem is a 5-dimensional vector, where the features count the number of E-values returned by PSI-BLAST using the given query that are less than 1e-10, 1e-5, 0.1, 1, and 10, respectively. The regression output is the predicted ROC_1 score on a validation set when RankProp is trained with the given value of σ. Both input and output features can be generated for a training set, so the regression can be learned, and then applied to a new test example.

We subtracted the mean from the outputs and normalized the inputs to have mean zero and standard deviation one, and used linear least squares to learn the regression.

Authors' contributions

JW developed and implemented the RankProp algorithms, and drafted the manuscript. RK helped design and implement some of the experiments, and revised part of the manuscript. WSN and CL coordinated the research, helped design the experiments and assisted with drafting the manuscript.

Acknowledgements

We would like to thank Andre Elisseeff and Denyong Zhou for their help with this work, and Jim Kent and Mark Diekhans for their help with implementing the RankProp search capability for the human genome browser. This work is supported by the NSF (awards EIA-0312706 and DBI-0243257) and the NIH (GM74257-01). WSN is an Alfred P. Sloan Research Fellow.

References

1. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ: **A basic local alignment search tool.** *Journal of Molecular Biology* 1990, **215:**403-410.
2. Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ: **Gapped BLAST and PSI-BLAST: A new generation of protein database search programs.** *Nucleic Acids Research* 1997, **25:**3389-3402.
3. Smith T, Waterman M: **Identification of common molecular subsequences.** *Journal of Molecular Biology* 1981, **147:**195-197.
4. Karplus K, Karchin R, Barrett C, Tu S, Cline M, Diekhans M, Grate L, Casper J, Hughey R: **What is the value added by human intervention in protein structure prediction?** *Proteins* 2001, **45(S5):**86-91.
5. Weston J, Elisseeff A, Zhou D, Leslie C, Noble WS: **Protein ranking: from local to global structure in the protein similarity network.** *PNAS USA* 2004, **101:**6559-6563.
6. Zhou D, Weston J, Gretton A, Bousquet O, Schoelkopf B: **Ranking on Data Manifolds.** *Neural Information Processing Systems* 2003.
7. Murzin AG, Brenner SE, Hubbard T, Chothia C: **SCOP: A structural classification of proteins database for the investigation of sequences and structures.** *Journal of Molecular Biology* 1995, **247(4):**536-540.
8. Page L, Brin S, Motwani R, Winograd T: **The PageRank Citation Ranking: Bringing Order to the Web.** *Tech. rep., Stanford Digital Library Technologies Project* 1998. [citeseer.ist.psu.edu/page98pagerank.html]
9. Weston J, Leslie C, Zhou D, Elisseeff A, Noble WS: **Cluster Kernels for Semi-supervised Protein Classification.** *Advances in Neural Information Processing Systems 17* 2003.
10. Kuang R, Ie E, Wang K, Wang K, Siddiqi M, Freund Y, Leslie C: **Profile-based string kernels for remote homology detection and motif extraction.** *Computational Systems Biology Conference* 2004.
11. Hanley JA, McNeil BJ: **The meaning and use of the area under a receiver operating characteristic (ROC) curve.** *Radiology* 1982, **143:**29-36.
12. Gribskov M, Robinson NL: **Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching.** *Computers and Chemistry* 1996, **20:**25-33.
13. Kuang R, Weston J, Noble WS, Leslie C: **Motif-based Protein Ranking with Network Propagation.** *submission to Bioinformatics* 2005.
14. Park J, Karplus K, Barrett C, Hughey R, Haussler D, Hubbard T, Chothia C: **Sequence comparisons using multiple sequences detect three times as many remote homologues as pairwise methods.** *Journal of Molecular Biology* 1998, **284(4):**1201-1210.
15. Jaakkola T, Diekhans M, Haussler D: **Using the Fisher kernel method to detect remote protein homologies.** In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology Menlo Park, CA: AAAI Press*; 1999:149-158.
16. Schäffer AA, Aravind L, Madden TL, Shavirin S, Spouge JL, Wolf YI, Koonin EV, Altschul SF: **Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements.** *Nucleic Acids Research* 2001, **29(14):**2994-3005.
17. Brenner SE, Koehl P, Levitt M: **The ASTRAL compendium for sequence and structure analysis.** *Nucleic Acids Research* 2000, **28:**254-256.