BIOINFORMATICS ORIGINAL PAPER Vol. 24 no. 6 2008, pages 783–790
doi:10.1093/bioinformatics/btn028

Sequence analysis

SVM-HUSTLE—an iterative semi-supervised machine learning approach for pairwise protein remote homology detection

Anuj R. Shah[1,*], Christopher S. Oehmen[2] and Bobbie-Jo Webb-Robertson[2]

[1]Scientific Data Management and [2]Computational Biology and Bioinformatics, Pacific Northwest National Laboratory, Richland, WA, USA

Received on August 14, 2007; revised on January 15, 2008; accepted on January 16, 2008

Advance Access publication February 1, 2008

Associate Editor: Burkhard Rost

ABSTRACT

Motivation: As the amount of biological sequence data continues to grow exponentially we face the increasing challenge of assigning function to this enormous molecular 'parts list'. The most popular approaches to this challenge make use of the simplifying assumption that similar functional molecules, or proteins, sometimes have similar composition, or sequence. However, these algorithms often fail to identify *remote* homologs (proteins with similar function but dissimilar sequence) which often are a significant fraction of the total homolog collection for a given sequence. We introduce a Support Vector Machine (**SVM**)-based tool to detect homology using semi-supervised iterative learning (**SVM-HUSTLE**) that identifies significantly more remote homologs than current state-of-the-art sequence or cluster-based methods. As opposed to building profiles or position specific scoring matrices, SVM-HUSTLE builds an SVM classifier for a query sequence by training on a collection of representative high-confidence training sets, recruits additional sequences and assigns a statistical measure of homology between a pair of sequences. SVM-HUSTLE combines principles of semi-supervised learning theory with statistical sampling to create many concurrent classifiers to iteratively detect and refine, on-the-fly, patterns indicating homology.

Results: When compared against existing methods for identifying protein homologs (BLAST, PSI-BLAST, COMPASS, PROF_SIM, RANKPROP and their variants) on two different benchmark datasets SVM-HUSTLE significantly outperforms each of the above methods using the most stringent ROC_1 statistic with *P*-values less than 1e-20. SVM-HUSTLE also yields results comparable to HHSearch but at a substantially reduced computational cost since we do not require the construction of HMMs.

Availability: The software executable to run SVM-HUSTLE can be downloaded from http://www.sysbio.org/sysbio/networkbio/svm_hustle

Contact: anuj.shah@pnl.gov

1 INTRODUCTION

During the past 30 years, numerous efforts have been focused on computationally predicting the functions of novel proteins. A well-established practice is to compare a query sequence of unknown function with a well-characterized database of sequences. If two similar sequences are derived from a common ancestor, they often preserve a related function. Inferring this common ancestry between similar sequences, termed homology detection, provides a kind of boot-strapping method by which newly sequenced genomes can be characterized based on existing annotations.

One method of quantifying the degree of homology is finding an optimal alignment between two sequences. In practice, many proteins can be thought of as comprising a series of functional regions joined by natively disordered links (Dunker *et al.*, 2002). Mutations in the functional regions would tend to degrade the behavior of a protein, whereas mutations in disordered regions may not impact the function at all. As a result, one would expect that homologous proteins can be more rapidly detected by aligning these smaller pieces or 'local' regions. Alignment methods such as the Smith–Waterman algorithm (Smith and Waterman, 1981), the fast approximation to Smith–Waterman (FASTA) (Pearson, 1985) and Basic Local Alignment Search Tool (BLAST) (Altschul *et al.*, 1990) take advantage of local sequence similarity in proteins. However, these methods often fail to detect common ancestry when proteins share low residue similarity, i.e. when proteins are *remote* homologs, which occurs commonly (Rost, 1999).

The concept of protein families was introduced into the problem of remote homology detection as an approach to overcome the lack of sensitivity that arises with algorithms comparing sequences based on residue similarity. A protein family is a group of sequences that share a common evolutionary origin. A statistical/computational model can be built for each of these families or super-families. A protein of unknown function can then be compared to each of the protein families producing a measure of likelihood of the sequence originating from this family. These family-based methods help computational biologists identify nearly three times the number of homology relationships as identified by simple pairwise alignments (Park *et al.*, 1998). PSI-BLAST (Altschul *et al.*, 1997) and hidden Markov models (HMM) (Baldi *et al.*, 1994) are two examples of such generative approaches. PSI-BLAST builds an alignment-based statistical model from a set of proteins by defining a position specific scoring matrix (PSSM). It then adopts a semi-supervised algorithm to use this query-specific PSSM model to search the space to identify additional homologous sequences iteratively from the database, refining

*To whom correspondence should be addressed.

the model at each step. This comparison of profiles of a family of proteins against a database still fails to detect a large number of distant/weak homologs. Methods such as PROF_SIM (Yona and Levitt, 2002), COMPASS (Sadreyev *et al*., 2003) and HHSearch (Soeding, 2005) take the approach of PSI-BLAST one step further. Instead of comparing a profile against individual sequences within a database, these methods compare two profiles in a pairwise setting, yielding significantly more homologs than PSI-BLAST, HMM or profile-based methods. However one major short-coming of these methods is the computationally expensive multiple alignments that needs to be generated in order to generate profiles for a set of sequences.

Discriminative approaches use a machine learning technique such as Artificial Neural Networks, Genetic Algorithms and Support vector machines (SVM) (Vapnik, 1995, 1998) to learn a set of classification rules based on given training data. Protein sequences are represented as fixed-length vectors in a machine learning framework to classify positive and negative examples. A number of family-based discriminative approaches have been published in literature and the only difference is in the representation of the protein vectors (Atalay and Cetin-Atalay, 2005; Ben-Hur and Brutlag, 2003; Busuttil *et al*., 2004; Kuang *et al*., 2005a; Jaakkola *et al*., 2000; Leslie *et al*., 2003; Liao and Noble, 2003; Lingner and Meinicke, 2006; Ogul and Mumcuoglu, 2006; Rangwala and Karypis, 2005; Shah *et al*., 2007; Webb-Robertson *et al*., 2005; Hou *et al*., 2003, 2004). It has successfully been demonstrated that these discriminative approaches outperform the current generative approaches on a limited benchmark dataset consisting of typically 54 families (Zaki *et al*., 2003). Despite improved sensitivity for the task of remote homology detection, there are caveats to this approach: since the protein families used in family-based descriminative models are pre-determined, only proteins that belong to a known and trained family can be classified. Most discriminative methods answer the question 'Whether the given protein belongs to an existing family?' Thus new protein families or homologous sequence pairs outside these defined families, i.e. families for which models are pre-built, cannot be discovered. However, these approaches illustrate the power of applying machine learning methods to the problem of homology detection. The goal of this work is to develop the first non-family-based SVM method for sensitive homology detection.

Most biologists and informaticists to date use BLAST and PSI-BLAST as their preferred method for homology detection. Both of these algorithms use string matching to identify sequences from a database which are similar to a query sequence. The input query sequence is searched against a large gene/protein database to determine closest matches and a ranked list of sequences is produced as output. One of the defining characteristics of such a problem is the large ratio of negative examples (non-homologous sequences) to positive examples (homologous sequences). Recent algorithms such as RANKPROP (Weston *et al*., 2004) and MOTIFPROP (Kuang *et al*., 2005b) produce a ranking of proteins that are homologous to the query protein by starting with an initial similarity network produced by a base algorithm (BLAST/PSI-BLAST) and identify additional relationships via network propagation. The initial network has edges between similar proteins labeled with *E*-values (or a function of *E*-values) resulting from the base algorithm while the network propagation is achieved by means of a diffusion operation. These methods have demonstrated increased sensitivity over BLAST and PSI-BLAST.

We present an iterative semi-supervised approach to the problem of *pairwise* remote homology detection. This approach is analogous to PSI-BLAST, which iteratively builds a sequence position specific scoring matrix on a set of homologs (positive examples). But SVM-HUSTLE accounts for both positive and negative examples, so it has a substantially improved performance compared to PSI-BLAST. SVM-HUSTLE achieves this by using concurrent SVMs to iteratively search for homologs against a database with pre-computed sequence similarity scores and provides the answer to the question 'Which sequences are homologous to a given query?' Adopting a semi-supervised approach, SVM-HUSTLE treats the database as a source of both labeled and unlabeled data, progressively growing the set of positive training examples. At each step, the database is reclassified. SVM-Hustle does not need family or super family classification knowledge for a given query sequence and in this respect is similar to BLAST and PSI-BLAST and different from existing state-of-the-art SVM homology detection methods. Classification models within SVM-HUSTLE are built on-the-fly for the given input query using unlabeled data to refine the model. This semi-supervised learning approach leads to a set of classifiers that have higher sensitivity when detecting distant homologs compared to PSI-BLAST. In addition, we use Bayesian statistics to build a statistical measure of confidence related to a pair of sequences that accounts for both the homologous and non-homologous models.

2 METHODS

2.1 Representation of protein sequences in a support vector machine

Vapnik developed the theory behind the SVM as the basis for statistical binary classification for a wide range of application areas. (Vapnik, 1995, 1998). SVMs are machine learning algorithms designed with the intention of 'generalizing better'. The problem SVM algorithms address relates to the efficient learning of a classification rule from a set of exemplars. The goal is the classification of each candidate as being either 'member' or 'not a member' of a given class. In our case, we train the SVM to recognize patterns of sequence similarity which are indicative of homologous proteins. The SVM requires that input data be represented as a collection of fixed-length vectors.

Protein sequences by their very nature are variable length compositions of amino acids. We achieve the conversion to fixed-length vectors called protein vectorization as follows. The feature vector corresponding to a protein X is given by

$$Fx = [fx_1, fx_2, fx_3, \ldots fx_n] \quad (1)$$

where n is total number of proteins in the training set and $fx_i = E$-value of the Smith–Waterman score between the sequence X and the i-th training set sequence (Smith and Waterman, 1981; Liao and Noble, 2003). We use the E-values as they are a statistical measure of similarity as opposed to raw Smith-Waterman scores between sequences which may introduce a bias based on the lengths of the individual proteins. The Smith–Waterman scores are attained using a scoring matrix of BLOSUM62, gap opening penalty of 11 and gap extension penalty of 1.

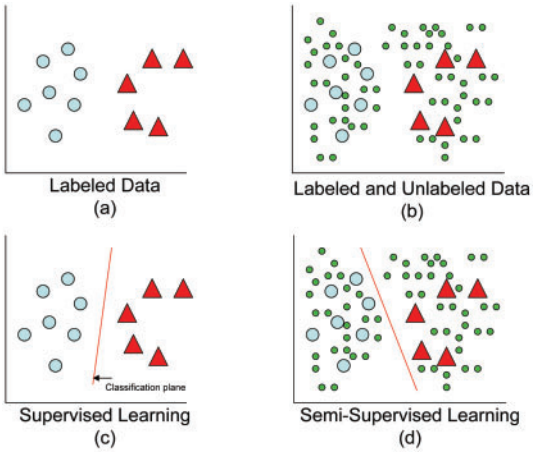

Fig. 1. Example of supervised and semi-supervised learning in binary classification problems. (**A**) Two class data is shown that needs to be classified using labeled data. (**B**) The same data is now captured along with unlabeled data represented in green dots. (**C**) A supervised learning technique may choose an incorrect decision boundary. (**D**) Given the extra unlabeled data points, the classification plane can be placed in region of sparse population, leading to better classification.

2.2 Semi-supervised learning

In supervised learning, the SVM 'learns' classification rules from both positive and negative exemplars. In unsupervised learning, patterns are formed based on clustering techniques. Semi-supervised learning is a training strategy that combines information from labeled data (i.e. data that is known to fall into one category or the other) and the unlabeled data which often clusters around labeled data. Semi-supervised learning functions are based on so-called cluster assumption: classification rules remain unchanged in areas of dense exemplars. Using clustering to associate unlabeled data with the labeled training data often improves coverage of the vector space improving the accuracy of classification.

Figure 1 illustrates the potential difference between learning using a semi-supervised technique as opposed to a supervised technique. The supervised algorithm may develop a classification rule that could cause incorrect classification of incoming exemplars when labeled data sparsely covers the vector space. Adding unlabeled data refines the detail of the separating hyperplane placing the decision boundary in areas of low density generating a more accurate classifier.

Semi-supervised learning is generally applied when unlabeled data is abundant while labeled data is small e.g. web page classification. For a detailed literature survey of semi-supervised learning interested readers can refer to Zhu (2006). Weston *et al.* (2006), recently extended the RANKPROP algorithm employing a semi-supervised learning technique to the problem of remote homology detection. RANKPROP exploits the global structure of similarity relationships among proteins in a database by performing a diffusion operation on a similarity network with weighted edges.

The initial similarity network, which is given as input to the algorithm, is formed by connecting proteins that are homologous using an auxiliary approach such as PSI-BLAST and the edge labels are based on a transfer function applied to the probabilistic *E*-values from PSI-BLAST searches. Multiple strategies to incorporate knowledge from unlabeled examples in the RANKPROP algorithm are suggested by (Weston *et al.*, 2006). In each case they achieve better rankings than a supervised approach.

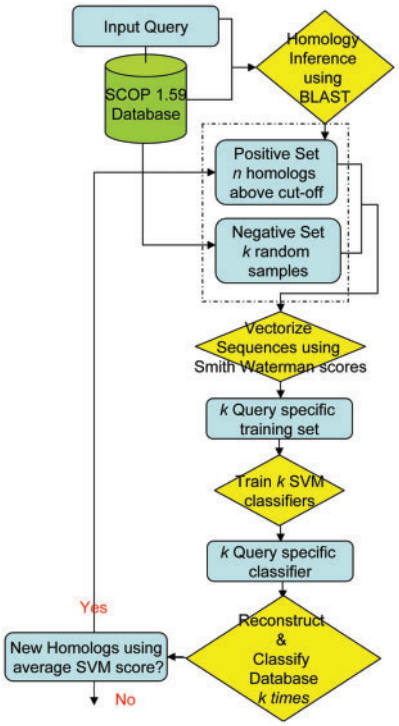

Fig. 2. SVM-HUSTLE algorithm flowchart.

2.3 SVM-HUSTLE: combining semi-supervised learning and SVM methodology

SVM-HUSTLE is a new algorithm that employs a semi-supervised SVM model to iteratively identify homologs to a query sequence from a database. The SVM-HUSTLE methodology can be described by six steps detailed in the flow chart of the algorithm in Figure 2. A novel component of this algorithm is the selection of the negative class (i.e. non-homologs) by sampling, to better cover the vector space. The number of non-homologs is much larger than the homologous class and as such we employ a random sampling algorithm to attain a representative negative class of proportionate size to the positive class. For example, in the benchmark dataset used for demonstration purposes there are 7329 protein sequences, for which we know the homologies. This dataset contains more than 53 million protein pairs. Of these pairs only 1%, (591 383), are homologous. To account for this imbalance in positive and negative examples, we iteratively repeat the entire sampling and classification step k number of times and use the average score for the subsequent iteration.

(1) ***Identify Seed Positive Training Set.*** An initial set of n homologous proteins are identified using an auxiliary method (BLAST in this case) against the database D consisting of 7329 proteins. This produces a basis set consisting of protein sequences that are similar to the original sequence with high statistical confidence (E-value <0.1). These n protein sequences form the initial positive training set (Pos) for SVM-HUSTLE. This is the only stage in the algorithm where sequence alignment scores are used.

(2) ***Randomly sample k Negative Training Sets.*** Select k (number of concurrent SVM classifiers) random sets of proteins from D of size that is an integral multiple of n for k SVM constructions. These negative training examples are represented as an array *Neg[k]*. Proteins in the negative training sets must be (i) not in

the high- or medium-confidence BLAST output, and (ii) only in one negative training set.

(3) ***Train SVM Classifiers.*** Train a set of *SVM[k]* classifiers where each training set has the same positive training examples (*Pos*) and a different set of negative training examples as defined by *Neg[k]*. The train and test set protein sequences are now vectorized using the approach discussed in Section 2.1.

(4) ***Vectorize the database.*** The database *D* is vectorized *k* times, once for each trained SVM. For each classifier, the proteins in *D* are converted to vector form using the positive train set and one of the negative train sets (*Neg[k]*). There will be k different vectorized databases—one for each of the classifiers. Each instance of the vectorized database is different since they all have different negative sequences.

(5) ***Classify protein pairs.*** Classify each protein in *D* and use the SVM discriminant score as a measure of classification confidence, select the top (best average score) *a*n* homologs for the given query protein q. The parameter '*a*' corresponds to the ratio of protein sequences selected as positive training examples per iteration.

(6) ***Iterate.*** Repeat steps 2–5 until no new samples are classified as positive or for a maximum number of iterations (*m*).

SVM-HUSTLE uses semi-supervised learning to build multiple query-specific models of homology for the incoming query sequence. These models are then subjected to supervised learning using SVMs. The database is re-created using these query-specific models and re-classified using each distinct SVM. Finally, an averaging mechanism is used that helps select the best possible matches for a protein sequence at each step. This combination of semi-supervised learning and supervised learning gives SVM-HUSTLE much of its classification power.

Figure 3 represents a schematic of the internal operation of SVM-HUSTLE after kernel transformations. The labeled and unlabeled examples can be laid out in two dimensions as shown. Constructing a single hyperplane based on the initial set of positive examples and randomly selected negative examples may not lead to correct classification in all cases. However, constructing multiple hyperplanes (*k*) would lead to an aggregate classification plane; one that will yield more accurate and sensitive results. It is possible that a true homolog can be recruited as a negative training example (as indicated by an underlined triangle in Fig. 3B and C) in one instance of the classification runs, however, the chances are slim since the ratio of positive to negative examples is very small. Additionally, using the averaging approach will also account for such discrepancies.

As its final output, SVM-HUSTLE produces a ranked listing of protein sequences that are homologous to a given query sequence with statistical scores signifying the similarity. SVM-HUSTLE does not require prior knowledge of protein family memberships for classification, so it can accurately and sensitively detect homologs which do not fall into any known protein family.

3 RESULTS AND DISCUSSION

3.1 Performance comparison with supervised and semi-supervised homology detection methods

We compare SVM-HUSTLE with PSI-BLAST, MotifProp, RankProp and their variants (k-mer MotifProp, E-Motif MotifProp etc). Comparisons between the pairwise approach of SVM-HUSTLE and family-based classification methods such as SVM-pairwise and its descendents cannot be performed as these methods rely on a family-based classification that identifies homologies within a set of pre-defined families.

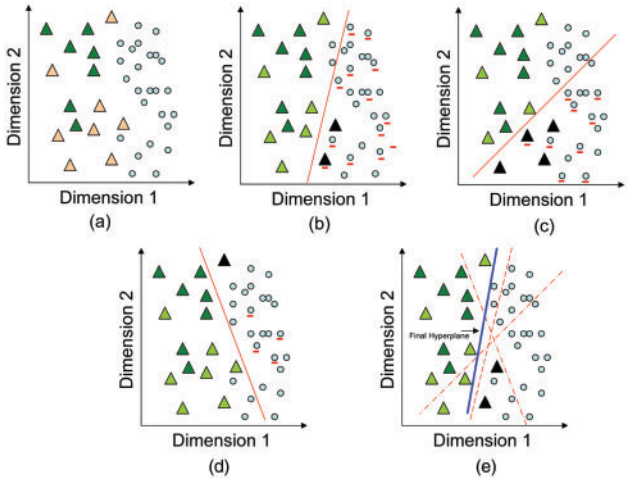

Fig. 3. Internal operation of the SVM-HUSTLE algorithm represented in two dimensions. The triangles represent the homologs of a query sequence while the dots are the unlabeled data samples. Depending on the initial set of positive examples (dark green) and randomly selected negative examples (triangles and dots underlined in red) a hyperplane can be drawn in any of the three positions as illustrated in the figure. Figure (**A**) represents the data that is considered linearly separable in space after kernel transformations. Figures (**B**), (**C**) and (**D**) are examples of a hyperplane in two-dimensional space that separates the data based on separate random selection of the negative class. In each of these figures, the black triangles represent incorrectly classified data, the lime green triangles are correctly classified based on the hyperplane selection and the red underlined black triangles are true homologs recruited by chance as a negative training example. Figure (**E**) shows the final classification (hyperplane in dark blue) after taking the average of the other three hyperplanes illustrating the necessity of averaging the classifiers.

The comparison is made on a set of 7329 sequences, which were extracted from version 1.59 of the SCOP database, purged by using the website http://astral.berkeley.edu so that no pair of sequences shares more than 95% identity. This benchmark dataset has been used in a number of previous studies (Kuang *et al.*, 2005b; Weston *et al.*, 2004, 2005). Following a similar procedure, the dataset was divided into 4246 training sequences and 3083 test sequences. SVM-HUSTLE identifies remote homologs on-the-fly and as such does not require a pre-defined training set. The SCOP 1.59 training set of 4246 sequences has no overlap in superfamily membership with the 3083 test sequences and as such inclusion in the database D does not provide more positive examples, only negative examples. Similar to PSI-BLAST, the training data is included in the database used to build the model, but homology metrics are only generated for the test data to allow comparison with the current state-of-the-art. No prior knowledge of family or superfamily is used in the training of SVM-HUSTLE. Initially all data is considered unlabeled and the initial positive training set is constructed using BLAST results.

The performance of RANKPROP, MOTIFPROP and their variants were assessed by their ROC scores which were directly obtained from the authors of the algorithms. PSI-BLAST was allowed to run for a maximum of six iterations, using a default *E*-value threshold of 0.005 and the BLOSUM62 scoring matrix.

As suggested by (Schaffer *et al*., 2001) this results in the most optimal performance of PSI-BLAST. Too few iterations results in an incomplete position-specific scoring matrix. Too many iterations result in the recruitment of large numbers of false positives, skewing the scoring matrix in favor of false positives.

In order to compare the performance of SVM-HUSTLE against the above methods, SVM-HUSTLE was run on each of the 3083 test set protein sequences on a LINUX cluster using four nodes with two processors per node with different values of parameters defined in Section 2.3 namely:

(1) k—number of concurrent SVM classifiers,

(2) a—ratio of protein sequences selected as positive training examples at each iteration and

(3) m—maximum number of iterations.

SVM-HUSTLE could not be run on three sequences out of the total 3083 sequences in the test set as BLAST failed to produce homolog hits with E-values better than 0.10. In its current form SVM-HUSTLE cannot be run on sequences with no BLAST hits, however an option would be to run an exhaustive sequence similarity algorithm such as Smith–Waterman or Needleman–Wunsch to extract a starting homologous set of protein sequences. For the current experiments, these three sequences were eliminated from all methods for comparison with SVM-HUSTLE. The network propagation methods overcome this issue by building a network despite the discovery of only a self-homolog. However, a single positive example of a sequence being homologous to itself is not adequate to train a SVM.

The primary statistic used for the statistical comparison of homology detection methods is a Receiver Operating Characteristic (ROC) (Hanley and McNeil, 1982) curve. A ROC curve yields a measure of the rate of false positives versus the true positives i.e. it is a plot of the true positive rate against the false positive rate for the different possible cut-offs of a diagnostic test. The area under a ROC plot is 1 for an ideal classifier, indicating that one can infer all of the correct protein homologies (identified from the SCOP classification) without having to tolerate any false positives.

The ROC_n score computes this score up to the n-th false positive (Gribskov and Robinson, 1996). Figure 5 outlines the pseudo code for an ROC score calculation. The ROC_1 values were calculated for each of the test set sequences to gage the relative performance of homology detection methods in the context of a database search where the vast majority of sequences are not homologous. In this case, it is important to get most of the correct homologs before a small number (rather than a small percentage) of false positives are classified as homologs. Essentially, the ROC_1 value equates to less than one error per query. Figure 4 shows the comparison of SVM-HUSTLE with the other approaches using the ROC_1 curve. The graph plots the total number of queries for which a given method exceeds a ROC score threshold. As shown by Figure 4, our results suggest that SVM-HUSTLE outperforms each of the algorithms with P-values less than 1e-20 in a two-tailed signed rank test (Salzberg, 1997).

Table 1 reports the average ROC_n scores across all test set queries for a given method. All results of SVM-HUSTLE are reported for $k = 100$, $a = 1$, $m = 10$ and a BLAST cut-off of 0.10 for initial homolog set. The rationale behind selection of a generous cut-off for the BLAST search is to allow more diverse sequences to be recruited in the initial positive train set.

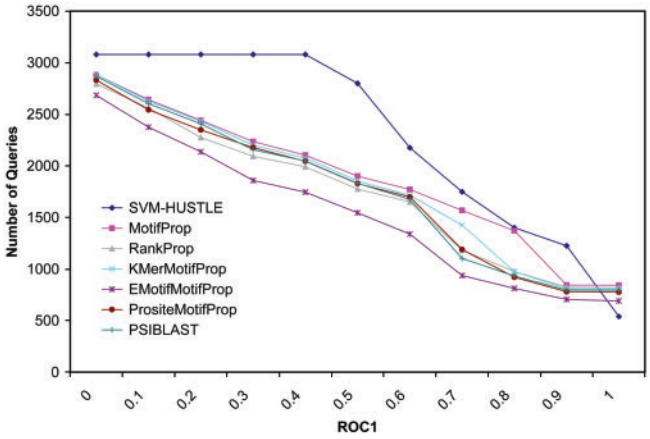

Fig. 4. Comparison of the ROC scores of SVM-HUSTLE, MotifProp, RankProp, k-mer MotifProp, E-Motif MotifProp, Prosite-MotifProp and PSI-BLAST for test queries. The graph plots the total number of queries for which a given method exceeds an ROC_1 score threshold indicated on the horizontal axis. The higher the curve, the more queries in the set having high ROC_1 scores for a given method, and hence a better classifier. SVM-HUSTLE outperforms each of the current-state-of-the-art-methods at almost all ROC_1 values.

```
Function calculate_ROC
Input:
        Parameter 1: SVM scores test sequences
        Parameter 2: True classification of test sequences
Output: ROC score

Step 1) Sort the SVM scores of the test sequences and get a sorted list of class labels (1 or -1) in a single column

tp=0 /* Initialize true positive */
fp=0 /* Initialize false positive */
roc=0 /* Initialize ROC score */

for each of the sorted label do
if (label=1) then
                tp=tp+1
else {
                fp=fp+1
                roc=roc+tp
                }

if (tp=0) then roc=0
else if (fp=0) then roc=1
else roc=roc/(tp*fp)
```

Fig. 5. Psuedo code for ROC computation.

Table 1. ROC_n ($n = 1, 10, 50$) scores for each method averaged over all queries

Method	SVM-HUSTLE	MotifProp	RankProp	PSI-BLAST
ROC 1	0.7445	0.6405	0.5927	0.5978
ROC 10	0.7818	0.6629	0.6674	0.6172
ROC 50	0.8122	0.6876	0.7249	0.6411

3.2 Performance comparison with profile-profile methods

In this experiment, we compare SVM-HUSTLE against profile-profile based remote homology detection methods such as

COMPASS (Sadreyev *et al.*, 2003), PROF_SIM (Yona and Levitt, 2002) and HHSearch (Soeding, 2005). These programs are shown to be significantly more sensitive than PSI-BLAST and have been applied to identify novel homologies.

To compare the results sequences were extracted from the SCOP 1.63 database using the website http://astral.berkeley.edu so that no pair of sequences shares more than 20% identity (SCOP-20). This results in a total of 3691 sequences, the homologies for which are obtained from their SCOP classifications. This benchmark dataset, previously used in (Soeding, 2005), has only 41 505 true positive homolog pairs while there are 1.08×10^7 false positives. The results for the above profile-profile comparison algorithms are extracted from (Soeding, 2005) while SVM-HUSTLE was set up to run each of the 3691 sequences with varying parameters. We report the results of SVM-HUSTLE for $k = 60$, $a = 1$ and $m = 10$ for this experiment. To include sufficient number of homologs in the initial training set, the BLAST cut-off was increased to 1.0 since this dataset has hardly any sequence similarity. In order to recruit sequences in a semi-supervised technique, we augment the 3691 protein sequences with sequences from the previous 7329 benchmark dataset. This results in a total of 8272 unique sequences. However, the test statistics are calculated only on the original 3691 SCOP-20 dataset.

As previously report in (Soeding, 2005) at a 10% error rate (FP/(TP + FP) = 10%) BLAST detects only 908 homologous pairs which is 2.2% of the total number of homologs. PSI-BLAST performs slightly better and results in a correct identification of 17.7% while HMMER finds 18.7%. PROF_SIM and COMPASS find 24.9 and 34%, respectively. HHSearch algorithms perform much better than any of the existing methods identifying 40–50% of the homologs, based on usage of additional knowledge such as the correlation score and predicted secondary structure. SVM-HUSTLE achieves an identification rate of 48.5%, correctly identifying 20 153 of the 41 505 homologs. Modification of the configuration parameters may result in higher percentage of homologs being found, e.g. increasing the BLAST cut-off from 1 to 10 and increasing the number of concurrent SVM's results in an additional 5% increase in homolog detection. These results are comparable to HHSearch while they outperform both PROF_SIM and COMPASS yielding almost twice the number of homolog detection.

Our results indicate the importance of using semi-supervised learning in combination with a supervised approach. First, Using the SCOP 7329 dataset as a source of unlabeled data (in a semi-supervised setting) increases the homology detection capability of SVM-HUSTLE. Second, SVM-HUSTLE can be easily reproduced and does not require the complicated profile–profile alignments or the alignments of a large number of variable length sequences in order to generate HMM profiles.

3.3 Assessing statistical significance of a SVM-HUSTLE score

Application of sequence homology algorithms traditionally involves a statistical score that assesses how likely an alignment with a score that good or better could have emerged by chance, under a specified null distribution (commonly an *E*-value).

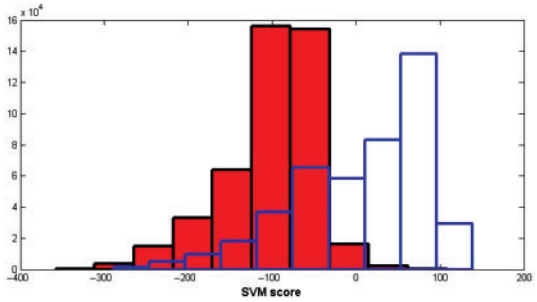

Fig. 6. The histogram of the SVM-HUSTLE scores for the non-homolgous pairs is shown in red and the SVM-HUSTLE scores for the homologous pairs is overlaid with a blue outline.

Unlike a *P*-value, which gives the level of significance for the null hypothesis (the two sequence are not homologous), we calculate the posterior odds. The posterior odds allows us to account for both the non-homologous ($\bar{H}$) and homolgous (H) models. Specifically, by Bayes formula we can compute the probability that the two sequences are not homologous given the SVM-HUSTLE score, s_{jk}, of a pair of sequences $R^{(j)}$ and $R^{(k)}$:

$$P\left(\bar{H}|R^{(j)}, R^{(k)}\right) = P\left(\bar{H}|s_{jk}\right) = \frac{P(s_{jk}|\bar{H})P(\bar{H})}{P(s_{jk}|\bar{H})P(\bar{H}) + P(s_{jk}|H)P(H)}, \tag{2}$$

which is the Bayesian analog to a *P*-value. This metric is dependent upon the knowledge of the distributions of the SVM scores associated with the $\bar{H}$ and H models. We use the raw SVM-HUSTLE values attained in the model evaluation to estimate these distributions. Figure 6 gives the histogram of the SVM-HUSTLE scores for the homologous pairs and an equal number of randomly selected non-homologs. The two distributions are fairly well separated as expected by the ROC scores shown in Figure 4 and Table 1.

In Figure 6, both distributions appear slightly skewed to the right, however a normal probability plot shows that the negative set ($\bar{H}$) does follow a normal distribution quite well. However, significant curvature in the normal probability plot of the positive set (H) suggests an alternate distribution, such as a Gamma. However, the Gamma distribution requires that $s \in [0;\infty]$. This is easily solved by simply adding a constant to all SVM-HUSTLE scores $q_{jk} = s_{jk} + \mathrm{C}$, which doesn't change the distribution, only the mean. The scaled SVM values (q) for the H model with $C = 400$ follows a Gamma distribution with a scale parameter of ~22.98 and a shape parameter of ~17.70, Figure 7A:

$$P\left(s_{jk}|H\right) \sim \Gamma\left(s_{jk}, 22.98, 17.7\right).$$

For $\bar{H}$, the scaled SVM-HUSTLE scores follow a normal distribution with a mean of ~299.37 and a standard deviation of ~54.27, Figure 7B:

$$P\left(s_{jk}|\bar{H}\right) \sim N\left(s_{jk}, 299.27, 54.27\right).$$

For our implementation of SVM-HUSTLE, we give the statistical probability that the sequence pair belongs to the null model, Equation (1).

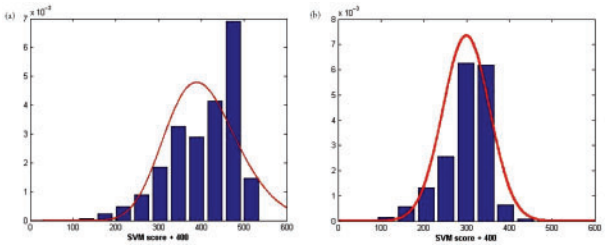

Fig. 7. The fitted distribution of the (**A**) positive H model is a Gamma and the (**B**) negative $\bar{H}$ model is a Gaussian.

3.4 Computational costs and scalability

Our results clearly demonstrate the improved performance characteristics of SVM-HUSTLE when detecting remote homologs. This increased sensitivity in homology detection does however come with increase in computational cost with respect to PSI-BLAST. A query specific training set is generated with similar positive exemplars and randomly sampled multiple negative exemplars. Since a query specific model is built for each incoming query, the target database D has to be reconstructed (re-vectorized) using this model, for each concurrent SVM classifier. The reconstruction cost of the database is the biggest bottleneck for SVM-HUSTLE.

The database reconstruction cost is directly impacted by the parameter k; the number of concurrent SVM classifiers, which linearly increases the time-to-solution for a given query. Increasing the number of concurrent classifiers introduces greater variability in the negative set that is randomly selected. As a result, we can better represent the space of negative exemplars in the training space while a smaller value of k only covers part of the database. We experimented with multiple values of this parameter and report results for $k=60$. Our evaluation indicates a statistically significant increase in accuracy values when moving from $k=4$ to $k=100$. The transition from $k=40$ to $k=80$ is significant and as a result we settle for the average case where $k=60$. A formal estimation of the best value of k would be part of future work on SVM-HUSTLE.

Depending on configuration parameters for SVM-HUSTLE, the database would be reconstructed in the worst case (complete 10 iterations, 60 samples) for m (number of iterations) $\times k$ (concurrent SVM classifiers) times, 600 in our case. The best case scenario for SVM-HUSTLE would be when $m=1$ which would amount to k reconstructions. If a computing facility is chosen such that there are multiple processors and the processor to classifier ratio is unity, the computation can be effectively expected to complete in time proportional to the size of the database. As the database grows in size (number of sequences), the database reconstruction cost can be prohibitive in nature and can slow down classification. Currently, SVM-Hustle requires specialized hardware to scale to large databases, such as nr. An additional challenge with large datasets is that in the case where a query has a large number of homologs, e.g. over 10 000, the size of the SVM training set would be prohibitive. However this is easily circumvented by setting a maximum limit on the initial positive train set as well as the BLAST cut-off for selection of training examples. We are currently developing a web service that would allow searches against various large sequence databases, public or user-defined. This would offer access to the specialized hardware required to run the algorithm on larger datasets. Additionally, an executable that can be used on datasets of less than 10 000 proteins is available at http://www.sysbio.org/sysbio/networkbio/svm_hustle.

The average time to solution for a single query in both experiments using SVM-HUSTLE is in seconds once the initial setup is completed. A large number of the queries from the test set (85%) complete well within the average time however queries that have a large number of training examples require more SVM optimization cycles and take around 1–2 min or more for convergence. Depending on the size of the database, users looking for a small set of homologs may be able to retrieve their results in a very short time.

As mentioned, the primary computational bottleneck is in creating the protein vectors on the fly using the Smith-Waterman algorithm to reconstruct the database. An alternative and efficient way of building the SVM classifier would be to use a protein representation that is independent of the training set sequences. We are currently investigating the potential of using fixed-length motif representation of protein sequences based on motif content (Ben-Hur and Brutlag, 2003; Hou *et al.*, 2003). These motif-based representations would allow SVM-HUSTLE to produce rank-lists of proteins within seconds without requiring the reconstruction phase.

4 CONCLUSIONS

The accurate annotation of newly sequenced proteins precedes much of the meaningful work that can be done in understanding molecular systems. For example, the analysis of microarray data and co-regulation in general, is much more meaningful when the genes are observed in a functional context. This relies heavily on one's ability to accurately and sensitively identify homologous proteins which are of distant evolutionary origin and hence, have a low degree of sequence similarity.

Our approach to incorporate semi-supervised learning coupled with concurrent training of SVM classifiers, though computationally expensive compared to PSI-BLAST, improves the sensitivity of remote homology detection by a significant margin using stricter thresholds of the ROC measure. More importantly, SVM-HUSTLE does not rely on knowing the superfamily classifications of query sequences making it possible to find homologs of proteins which do not fit into any known family.

SVM-HUSTLE is a pairwise sequence homology detection algorithm that builds models from representative high-confidence training sequences on the fly in a semi-supervised fashion. A typical setting for the use of SVM-HUSTLE would be to search for homologous proteins from a database for a given query. SVM-HUSTLE yields significantly better results than network propagation algorithms such as the RANKPROP and MOTIFPROP on our benchmark datasets. The performance is also comparable to HHSearch which requires the

computation of multiple alignments and/or structural models of the database.

ACKNOWLEDGEMENTS

This research was supported by the Data-intensive Computing for Complex Biological Systems project funded by the Office of Advanced Scientific Computing Research, and under the Laboratory Directed Research and Development Program at the Pacific Northwest National Laboratory, (PNNL). PNNL is a multi-program national laboratory operated by Battelle for the US DOE under contract DE-AC06-76RL01830. This research was performed in part using the Molecular Science Computing Facility (MSCF) in the William R. Wiley Environmental Molecular Science Laboratory (EMSL), a national scientific user facility sponsored by the US DOE, OBER and located at PNNL. In addition, the authors would like to thank the Dr Jason Weston, Dr Christina Leslie, Dr Rui Kuang, Dr Li Liao and Dr William Stafford Noble for the public access to their data and results.

Conflict of Interest: none declared.

REFERENCES

Altschul,S.F. *et al*. (1990) A basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Altschul,S.F. *et al*. (1997) Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucl. Acids Res.*, **25**, 3389–3402.

Atalay,V. and Cetin-Atalay,R. (2005) Implicit motif distribution based hybrid computational kernel for sequence classification. *Bioinformatics*, **21**, 1429–1436.

Baldi,P. *et al*. (1994) Hidden Markov models of biological primary sequence information. *Proc. Natl Acad. Sci.*, **91**, 1059–1063.

Ben-Hur,A. and Brutlag,D. (2003) Remote homology detection: a motif based approach. *Bioinformatics*, **19**, i26–i33.

Busuttil,S. *et al*. (2004) Support vector machines with profile-based kernels for remote protein homology detection. *Genome Informatics*, **15**, 191–200.

Dunker,A.K. *et al*. (2002) Intrinsic disorder and protein function. *Biochemistry*, **41**, 6573–6582.

Gribskov,M. and Robinson,N.L. (1996) Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. *Comp Chem.*, **20**, 25–33.

Hanley,J.A. and McNeil,B.J. (1982) The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, **143**, 29–36.

Hou,Y. *et al*. (2003) Efficient remote homology detection using local structure. *Bioinformatics*, **19**, 2294–2301.

Hou,Y. *et al*. (2004) Remote homology detection using local sequence-structure correlations. *Proteins: Structure, Function and Bioinformatics*, **57**, 518–530.

Jaakkola,T. *et al*. (2000) A discriminative framework for detecting remote protein homologies. *J. Comput. Biol.*, **7**, 95–114.

Kuang,R. *et al*. (2005a) Profile-based string kernels for remote homology detection and motif extraction. *J. Bioinform. Computat. Biol.*, **3**, 527–550.

Kuang,R. *et al*. (2005b) Motif-based protein ranking by network propagation. *Bioinformatics*, **21**, 3711–3718.

Leslie,C. *et al*. (2003) Mismatch string kernels for discriminative protein classification. *Bioinformatics*, **1**, 1–10.

Liao,L. and Noble,W.S. (2003) Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships. *J. Comput. Biol.*, **10**, 857–868.

Lingner,T. and Meinicke,P. (2006) Remote homology detection based on oligomer distances. *Bioinformatics*, **22**, 2224–2231.

Ogul,H. and Mumcuoglu,E.U. (2006) A discriminative method for remote homology detection based on n-peptide compositions with reduced amino acid alphabets. *J. Mol. Biol.*, **284**, 1202–1210.

Park,J. *et al*. (1998) Sequence comparisons using multiple sequences detect three times as many remote homologues as pairwise methods. *J. Mol. Biol.*, **284**, 1202–1210.

Pearson,W.R. (1985) Rapid and sensitive sequence comparisons with FASTP and FASTA. *Methods Enzymol.*, **183**, 63–98.

Rangwala,H. and Karypis,G. (2005) Profile based direct kernels for remote homology detection and fold recognition. *Bioinformatics*, **21**, 4239–4247.

Rost,B. (1999) Twilight zone of protein sequence alignments. *Protein Eng.*, **12**, 85–94.

Sadreyev,R.I. *et al*. (2003) Profile-profile comparisons by COMPASS predict intricate homologies between protein families. *Protein Sci.*, **12**, 2262–2272.

Salzberg,S.L. (1997) On comparing classifiers: pitfalls to avoid and recommended approach. *Data Mining Knowledge Discovery*, **1**, 317–328.

Schaffer,A.A. *et al*. (2001) Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements. *Nucl. Acids Res.*, **29**, 2994–3005.

Shah,A.R. *et al*. (2007) Integrating subcellular location for improving machine learning models of remote homology detection in eukaryotic organisms. *Comput Biol. Chem.*, **31**, 138–142.

Smith,T. and Waterman,M. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.

Soeding,J. (2005) Protein homology detection by HMM-HMM comparison. *Bioinformatics*, **21**, 951–960.

Vapnik,V.N. (1995) *The nature of Statistical Learning Theory*. Springer, New York.

Vapnik,V.N. (1998) *Statistical Learning Theory. Adaptive and Learning Systems for Signal Processing, Communications, and Control*. Wiley, New york.

Webb-Robertson,B.-J.M. *et al*. (2005) SVM-BALSA: remote homology detection based on Bayesian sequence alignment. *Comput. Biol. Chem.*, **29**, 440–443.

Weston,J. *et al*. (2004) Protein ranking: from local to global structure in the protein similarity network. *Proc. Natl Acad. Sci.*, **101**, 6559–6563.

Weston,J. *et al*. (2006) Protein ranking by semi-supervised network propagation. *BMC Bioinformatics*, **7** (**Suppl I**), S10.

Weston,J. *et al*. (2005) Semi-supervised protein classification using cluster kernels. *Bioinformatics*, **21**, 3241–3247.

Yona,G. and Levitt,M. (2002) Within the twilight zone: a sensitive profile-profile comparison tool based on information theory. *J. Mol. Biol.*, **315**, 1257–1275.

Zaki,N.M. *et al*. (2003) A comparative analysis of protein homology detection methods. *J. Theor.*, **5**.

Zhu,X. (2006) *Semi-supervised Learning Literature Survey*. University of Wisconsin, Madison.