Protein Classification Using Transductive Learning On Phylogenetic Profiles

Roger Craig and Li Liao*
Department of Computer and Information Sciences
University of Delaware, Newark, DE 19716, USA

ABSTRACT Phylogenetic profiles of proteins – strings of ones and zeros encoding respectively the presence and absence of proteins in a group of genomes – have recently been used to identify homologous proteins and/or proteins that are functionally linked, such as participating in a metabolic pathway. We proposed a novel learning method for protein classification based on phylogenetic profiles, which takes into account both the phylogenetic tree structure and the likelihood of proteins presence in genomes. The method consists of a mechanism to extend the profiles with extra bits encoding the phylogenetic tree, whose interior nodes, representing hypothetical ancestral genomes, are scored in a way to reflect their chances of developing divergence in the descendants. The scoring scheme also incorporates the likelihood of proteins presence in genomes as weighting factors, which are collected from the training data initially and integrated as part of kernel of a support vector machine. In a transductive learning scheme, when the SVM is used for classifying test data, the weighting factors are updated iteratively using the predicted results. We tested our method on the proteome of *Saccharomyces cerevisiae* and used the MIPS classification as a benchmark. The results showed that the classification accuracy was greatly increased.

1. INTRODUCTION

Protein functional annotation remains a central task in genomics, and the computational efforts for this task have undergone several stages of development. Historically, most computational tools, such as BLAST [1] and Smith-Waterman [14], were developed to compare sequence similarity for protein homology detection. The basis for this type of methods is that homologous proteins evolved from a common ancestral protein via mutations are likely to remain similar in sequence composition, and at the same time still play the same functional role. However, the effectiveness of the methods that solely rely on sequence similarity can be seriously compromised when applied to proteins in the so-called twilight zone, namely, those proteins that are distantly homologous to one another and therefore share less (below 30%) similarity. Over the past decade or so, various techniques have been developed for detecting distant protein homologues, for example, iterative search with refined profiles [2], sophisticated probabilistic models, powerful statistical learning [10], and some hybrid approaches [6], to name a few.

The development of computational methods for predicting protein functions has been witnessed changes with new trends. On the one hand, there are efforts to make use of the non-sequential information, such as gene expression data, protein-protein interaction data, or data of other types. On the other hand, some methods in comparative genomics have gone beyond homology by identifying proteins that are related to one another because they are associated in a common structural complex, participate in common metabolic pathways, or because they fuse into a single gene in some genomes [4]. An important work in this line is the use of phylogenetic profiles for assigning gene functions based on evolutionary and/or co-evolutionary patterns across species [11, 12, 15, 16].

The phylogenetic profile of a protein is represented as a vector, where each component corresponds to a specific genome and takes a value of either one or zero: with one (zero) indicating the presence (absence) of a significant homology of that protein in the corresponding genome. As functionally linked proteins, e.g., in a structural complex or a metabolic pathway, tend to evolve in a correlated way, their phylogenetic profiles consequently show similarity. Simple measures for similarity between phylogenetic profiles were first proposed, such as edit distance and Euclidean distance. Although these simple measures generated useful classification and prediction, recent focus has been given to incorporating into profile similarity the evolutionary relations that are represented in the phylogenetic tree of the genomes [11, 18].

The importance of the phylogenetic tree in the study of phylogenetic profiles has been recognized in early work, e.g., Liberles et al in 2002 [11], where a method was proposed to utilize the historical evolutions of two proteins to account for their similarity (or dissimilarity). Evolutionary relationships

SAC'06, April, 23-27, 2006, Dijon, France.

* Corresponding author: lliao@cis.udel.edu

among organisms can be represented as a phylogenetic tree where leaves correspond to the current organisms and interior nodes correspond to hypothetical ancient organisms. So, rather than simply counting the presence and absence of the proteins in the current genomes like what is done for edit distance, a quantity called differential parsimony is calculated that minimizes the number of times when changes have to be made at tree branches to reconcile the two profiles; the smaller the differential parsimony, the more similar the two profiles are.

The concept of minimizing the differences at the tree branches as a way to incorporate the evolutionary histories in comparing two phylogenetic profiles was generalized to include all evolutionary patterns and endowed with probabilistic formulation and interpretation [18]. An evolutionary pattern corresponds to a series of assignments of the gene's retention or loss at the branches of the phylogenetic tree such that the assignments match at the tree leaves with the profile of that gene. Due to the stochastic nature of the events where genes can be regained/added/lost during speciation, a probability is given to each such event, and the probability of an evolutionary pattern is therefore the product of probabilities of individual events at all tree branches, with an assumption that these events are independent of each other. The higher the probability an evolutionary pattern has, the more likely it explains the profile, in a probabilistic sense – why the protein (or rather its gene) is present or absent in current genomes as the result of evolution. Moreover, two profiles are considered similar to each other if they share many highly probable evolutionary patterns. Thus, the joint probabilities of all possible evolutionary patterns were summed to give a kernel function called tree kernel, which plays a role of dot product of profile vectors in some higher dimensional space, called feature space. As a test, this tree kernel was then used in a support vector machine to classify 2465 yeast proteins whose functions and classifications were known in the MIPS database (http://mips.gsf.de/genre/proj/yeast/). The method used some preset parameters: the probability that an existing gene is retained at a tree branch (i.e., speciation) is set at 0.9, and the probability that a new gene is created at a branch is set at 0.1. It was further assumed that such a distribution remains the same at all branches for all genes. Even with these crude assumptions, in the 3-fold cross validation experiments on those 2465 yeast genes, the tree kernel's classification accuracy already significantly exceeds that of a naïve kernel using just the ordinary dot product.

In this paper we propose a novel simple approach which incorporates both the phylogenetic tree and the likelihood of a protein's presence in current genomes in training a support vector machine. Particularly, our method does not require preset probabilities for gene retention and creation during speciation. Not only is it difficult to justify any *a priori* values for these probabilities and the associated assumptions as used in the tree kernel, it also turns out that their actual values do not seem to bear any influence to the classification accuracy of the tree kernel method – we tested with different settings varied from 0.9 to 0.1 for gene retention (and 0.1 to 0.9 for gene creation, correspondingly) at 0.1 intervals, and did not notice any significant changes in classification accuracy – this phenomenon is quite contrary to our intuition and worth further investigation for its own sake. Here we take a simple approach to incorporate the evolutionary history by encoding the phylogenetic tree as extra bits into the profiles [14]. In doing so, we label the interior nodes of the phylogenetic tree with scores. These scores ought to reflect, on the one hand, the tree topology, e.g., the number of branches at an interior node, representing its chances of developing divergence in descendants. On the other hand, these scores also ought to reflect the specific evolutionary history of the individual proteins (genes). Taking these into consideration, we computed the scores at tree branches (i.e., interior nodes) by averaging the scores at children nodes – in a recursive procedure that terminates when it reaches the leaf nodes, whose scores are simply the profile values of 1 or 0 when binary profiles are used, or real values as well, as shown later. To compare two profiles for similarity, we extend each profile with the extra bits that correspond to the interior nodes of the individually labeled phylogenetic tree and take values of the scores at these interior nodes. With the extended phylogenetic profiles, we input them to a polynomial kernel support vector machine for classification, hoping that these extra bits encoding the evolutionary history of individual genes embodied in the phylogenetic tree can help classify functionally linked genes.

The extension of phylogenetic profiles allows for incorporating more information into training the classifier. First, the scoring scheme can be refined using weighted averaging at the tree leaves, with the weighting factors reflecting the frequency of a presence (or absence) for any protein (not just the protein whose profile is being extended) at specific genomes. Such frequencies are collected from the training data. Therefore, the extended profiles now not only carry the information about each individual protein, but also some collective information about the proteome (as sampled in the training data). The scoring scheme can be further refined by collecting the weighting factors not just from the training data, but also from the predicted results of the testing data [3]. The profiles are thus updated with the predicted results, iteratively, in the spirit of transductive learning. The iteration will stop when a preset criterion is met; in our case, when a Kullback-Leibler distance of weighting factors (treated as probability distribution) between two iterations is smaller than a preset threshold. When tested with the same dataset and cross validation scheme as the tree kernel method in [18], our method, particularly when refined with the iterative weighting, has shown significantly superior performance.

2. MATERIALS AND METHOD

2.1 Dataset

The data set used in this work is the same data set as in [15, 18], hence the performance improvement of our method over these existing methods can be conveniently assessed. Genes with accurate functional classifications were selected from the budding yeast *Saccharomyces cerevisiae* genome. To ensure adequate training and testing examples, only the functional classes that contain at least 10 genes were extracted from the several hundred classes in the Munich Information Center for Protein Sequences Comprehensive Yeast Genome Databases [13]. The resulting dataset contains 2465 genes in 133 functional classes. The binary profiles of these genes were built by BLAST search against each of the 24 genomes. Each bit in the profile for a gene was set to 0 if the E-value of the BLAST search for the

gene against the corresponding organism was larger 1, and was set to 1 if otherwise. The threshold value 1 was empirically set, and the resulting profiles yielded best classification performance in the tree kernel method. The profiles can also be directly E-value based, without converting to 0/1 with any threshold. The phylogenetic tree of these 24 genomes, shown in Figure 1, is the same as in [18].

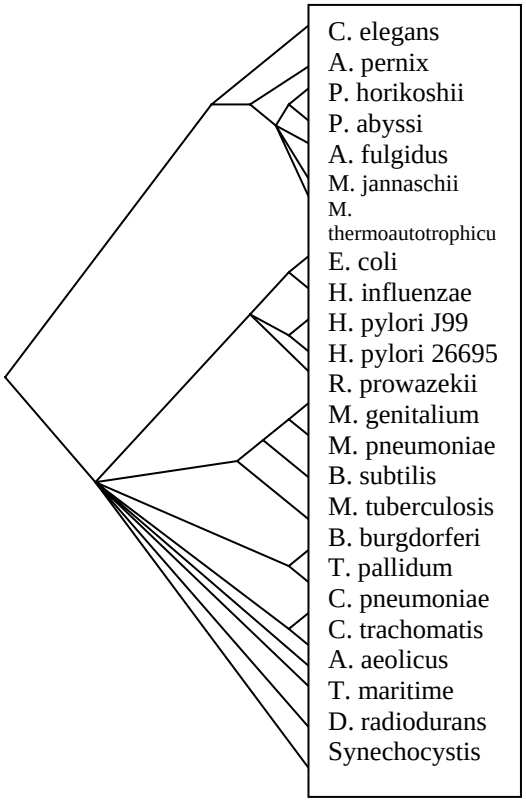

Figure 1. The 24 genomes and a phylogenetic tree of these genomes.

A 3-fold cross validation was adopted for the experiments. For each functional class, two third of its members are randomly selected as positive training examples, and the remaining third as positive testing examples. Genes not belonging in that class were randomly split into two thirds as negative training and one third as negative testing examples. This process is repeated 10 times for each functional class, and the classification accuracy, measured as receiver operating characteristic (ROC) score in this paper, is the average over these ten runs.

2.2 Tree Encoding and Profile Extension

As we argued above, the information encoded in the phylogenetic tree shall be incorporated into the profiles, and a two-step procedure is adopted as the following. In the first step a score is assigned to each interior node in the phylogenetic tree. Because each interior tree node is interpreted as an ancestral genome of the genomes at the descending nodes, a score therefore should reflect the degree of divergence (both biologically and pattern-wise) at these descendants. If these descendants are already assigned with scores, the average score will be assigned to the parent node. All interior nodes can be scored as such in a recursive procedure, such as a post-order tree traversal, as long as the leaf nodes are scored. The leaf nodes are scored according to the profile, namely, score 0 when the protein is absent, and score 1 when present. Once all interior nodes are assigned with scores, the second step is to flatten the score-clad tree into a vector − mapping of interior nodes into a vector is determined by a post-order tree traversal. The vector is then concatenated with the original profile vector to form an extended vector, which was called tree-encoded profile (TEP), which are 38 bit vectors, with the last 14 bits corresponding to the interior nodes of the phylogenetic tree in Figure 1. A schematically illustration of how a phylogenetic profiles is extended is given in Figure 2. Actually, as shown in the RESULTS section, direct use of the E-value based profiles gives better classification performance than converting into binary profiles with an arbitrary threshold, which inevitably incurs some loss of information.

2.3 Iterative Weighting in Transduction

A further refinement is attained by introducing weights in calculating the score $S(k)$ for any interior tree node k whose children nodes contain tree leaves:

$$S(k) = (1/|C|)\sum_{i \in C} S(i)\, W_{s(i)}(i) \qquad (1)$$

where C is the set of children nodes for node k. and $|C|$ is the size of the set C. For binary profiles, the scores $S(i)$ for leaf nodes $i \in C$ are 1 or -1. Note that the score zeros, in the original profile as indication of gene absence, are changed to -1 in order to make the effect of weighting non-zero, because a zero multiplied by any weighting factor is still zero. The weights $W_{\pm 1}(i)$ at a leaf i are collected as the frequency of absence (-1) and presence (+1) of proteins in genome i in the training data. The weight is always set to 1 if a node in C is not a leaf node. The weighting scheme may still be applicable even when the phylogenetic profiles are E-value based, namely, $S(i)$ for leaf nodes in Eq(1) are real value numbers. To do this, we first use a threshold E_0 on E-value as if for converting the profiles into binary, so as to collect the frequency based on $sign(E_0 - S(i))$ to be used as weighting factors $W_{sign(E0 - s(i))}(i)$. The Eq(1) is thus modified as the following.

$$S(k) = (1/L)\sum_{i \in L} S(i)\, W_{sign(E0 - s(i))}(i). \qquad (1')$$

The E-value based profiles using the weighted tree encoding scheme is shown to improve classification accuracy and is referred to as TEEWP later in the Results section.

Since the weighting factors reflect how likely proteins may be absent or present at a leaf position in the phylogenetic tree, and such collective information about the proteome (as sampled in the training data) helps distinguish proteins from different families, it therefore makes intuitive sense to collect the weighting factors for the positive training examples (family members) and for the negative training examples (non family members) separately. That is, when using Eq(1) or Eq(1') to extend a profile in the positive training set, $W_{\pm 1}(i)$ at a leaf i are collected from positive training examples only, whereas when using Eq(1) or Eq(1') to extend a profile in the negative training set, $W_{\pm 1}(i)$ at a leaf i are collected from negative training examples only. The difficulty with such a scheme of separate

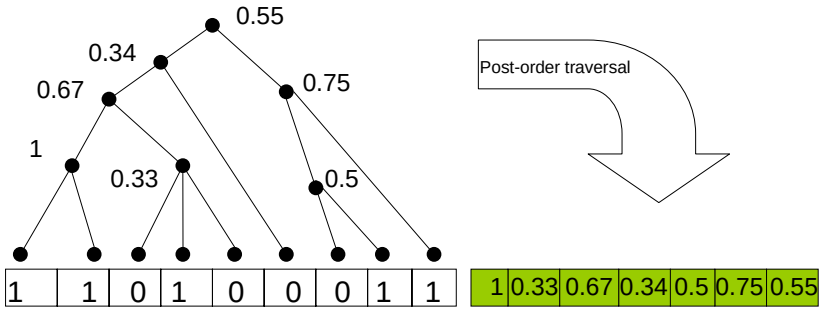

Figure 2. Schematic illustration of extending a phylogenetic profile (unshaded boxes) with extra bits (shaded boxes) encoding interior nodes of the phylogenetic tree.

weighting is: how should the profiles of the testing examples be weighted, since we do not know beforehand if a test example is positive or negative. To overcome this difficulty, we adopted the *transductive* learning paradigm [8, 19]. That is, we are allowed to look into the predictions made on the testing examples and collect weighting factors from the predicted positives and negatives respectively. We first rank the testing examples by their scores returned from the classifier as confidence score for the prediction. In SVMs, these scores termed as discriminant are values ranged [-1, 1], with 1 indicating a certain prediction for positive and -1 a certain prediction for negative. Like in other transductive learning applications [8], one piece of extra information allowed for use is the ratio ρ of true positives versus true negatives in the testing examples. This is feasible under the assumption that the class bias (i.e., ratio ρ) in the test set is the same as in the training set, although in reality this may not be exactly the case. We experimented with variations on ρ and found that the classification accuracy of the method is not sensitive to ρ ; the results are not reported in this paper for the sake of space. So, with the ranked list of n testing examples and ratio ρ, we simply take the top $n\rho$ as predicted positives and collect weighting factors W's. The rest are taken as predicted negatives and corresponding W's are calculated in a similar way. Then, the profiles of the testing examples are updated, depending on whether they are predicted positive or negative, with respective weighting factors. The updated profiles are fed to the classifier for classification. Once new predictions are made, we can update the weighting factors and reweight these profiles again. We do this iteratively, till some stopping criterion is met, which will be discussed in the next subsection. Although the weighting procedure only affects the last 14 bits in the profiles, the classification accuracy is greatly increased with this iterative weighting scheme, as shown in the RESULTS section.

2.4 Stopping Criterion

In order to put this method into practical use, a criterion must be established to stop the iterative procedure for reweighting. Since the weighting factor $W_{\pm 1}(i)$ at a leaf i can be interpreted as probability distribution over two possible outcomes, we use relative entropy, a.k.a., Kullback-Leibler (KL) distance, to measure the change brought about on W's between two iterations.

For the j-th bit (i.e., leaf j) of the 24 bits, calculate the KL distance as

$$d_j = \sum_{a = -1, +1} \{ W_a(j) \log [W_a(j)/ W'_a(j)]\}, \quad (2)$$

where $W_a(j)$ is the weighting factor at leaf j, with a= "+1" or "-1", and $W'_a(j)$ is the weighting factor at leaf j calculated from the previous iteration. We monitor the average KL distance over the 24 original bits

$$D = (\sum_{j=1, \text{ to } 24} d_j) / 24, \quad (3)$$

for each iteration to see if it converges and at least shows a trend of convergence. We also monitor the classification accuracy at each iteration. We hope to see that, over iterations, while the classification accuracy increases, the average KL distance decreases. Then, an empirical threshold can be set on the average KL distance to stop the reweighting procedure, in order to achieve some desired classification accuracy. The result is reported in Figure 4.

2.5 Kernel

The classifier here is a support vector machine with a polynomial kernel. The polynomial kernel for vectors x and y is defined as

$$K(x, y) = [1+ s D(x, y)]^d \quad (4)$$

where s and d are two parameters adjustable in the software package *SVM Light* [6]. D(x, y) in Eq(4) is the dot product of vectors x and y, $D(x,y) = x \cdot y$. The default values for s and d are used.

3. RESULTS

The results of the 3-fold cross validation experiments on the dataset of 2465 Yeast genes are summarized in Figures 3 and 4. The function prediction for each class of the 133 classes is measured by receiver operating characteristic (ROC) score. ROC score is the normalized area under a curve that plots the true positives as a function of false positives for varying classification thresholds [5]. ROC50 scores are ROC scores that are calculated by integrating the area up to the 50th false positives. The ROC 50 scores are in a range of [0, 1], with 1 for a perfect classification. Recall that, for each functional class, two third of its members are randomly selected as positive training examples, and the remaining third as positive testing

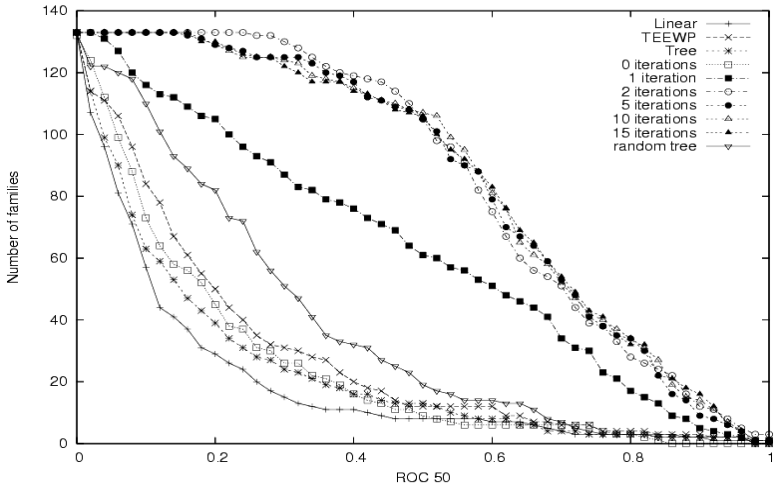

Figure 3. Histogram of ROC50 scores for various experiments on the 133 families.

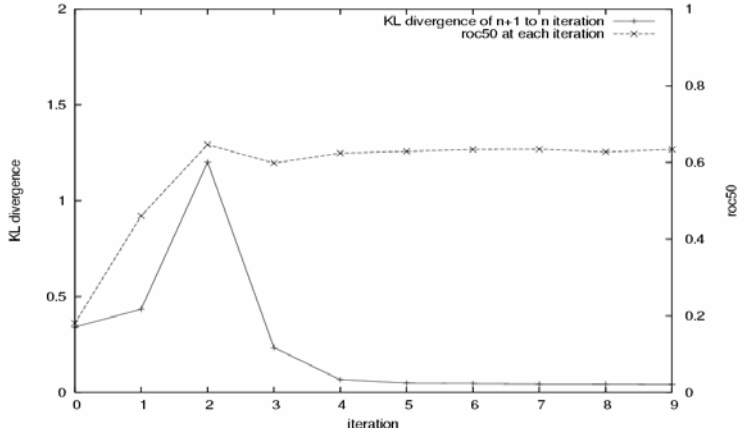

Figure 4. The KL distance and average ROC50 scores over 10 iterations.

examples. Genes not belonging in that class were randomly split into two thirds as negative training and one third as negative testing examples. To ensure the results are robust to the data preparation procedure, we repeat the experiments over 10 random runs. In each run, the training and testing data is independently prepared. ROC50 scores reported in this study are all averaged over 10 random runs.

In Figure 3, histograms show the performance over 133 classes: the number of classes (Y-axis) that were classified with accuracy better than a given ROC50 score (X-axis). Therefore, a higher curve indicates more accurate prediction. For comparison purpose, histograms for a linear kernel, the "tree" kernel [18], and tree encoded E-value weighted profiles (TEEWP) are also shown in Figure 3. It is easily seen that, although TEEWP has already outperformed the tree kernel, the iterative weighting scheme improved the accuracy greatly and achieved a superior performance. To verify the gained performance is indeed due to incorporating phylogenetic tree, we repeat the transductive learning but with a randomly permutated tree. As shown in Figure 3, the performance of random tree with 10 iterations is much worse. In addition, we notice that the overall performance gain is very significant with the first several iterations, and then tend to quickly converge with more iterations. In Figure 4, this trend of convergence is shown with the average ROC50 score of 133 classes (the left Y-axis) going up and reaching a plateau as the number iteration increases. Also shown in Figure 4 (the right Y-axis) is that the average KL distance of weighting factors between iterations is diminishing, a sign that the weighting factors stop picking up new information from more iterations. Although, at present, the iterative weighting scheme is not yet formulated as an optimization problem with a well defined objective function, the KL distance of weighting factors seems to be a useful pragmatic substitute for that purpose.

In addition, the 15 functional classes where the naïve linear kernel had reported best ROC50 scores among that of the 133 classes are examined in details (see Table 1s in supplemental data at http://liao.cis.udel.edu/trans_tk), and in all 15 classes, our method outperformed both naïve linear kernel and tree

kernel. Except two classes (fermentation and ABC transporters), the best ROC50 scores are consistently achieved by the transductive learning method, with the highest performance improvement over that of naïve linear kernel being 514% (in the class of TRNA modification).

4. DISCUSSION

We presented a novel approach that extends the phylogenetic profiles with extra bits encoding the phylogenetic tree and classifies proteins based on the weighted phylogenetic profiles in a transductive manner. The approach gives superior performance as tested in classifying the yeast genome, as compared to previous methods [18]. The superior performance is believed to come partly from the use of domain specific information about the genome -- the frequencies of protein's absence and presence in a given genome as opposed to other genomes (corresponding to leaves in the phylogenetic tree). In order to incorporate such domain specific information into the phylogenetic profiles for proteins whose class membership is yet to be predicted, we proposed a self-consistent, transductive type learning, which allows for the use of prediction from the previous iteration. It differs from the standard transductive learning [8, 19] by that, the classifier, which is a support vector machine in this case, is not retrained; only the phylogenetic profiles of testing examples are reweighted, although reweighting affects the feature space and thus can also be viewed as part of kernel. One apparent advantage over the standard transductive learning is thus the speed gained from avoiding retraining the classifier. The empirical results show the trend of quick convergence. Identifying an objective function and formulating the iterative reweighting as an optimization problem will be pursued as future research.

5. REFERENCES

[1] Altschul, S., Gish, W., Miller, W., Myers, E. and Lipman, D. Basic local alignment search tool. *Journal of Molecular Biology*, vol. 215, pp. 403-410, 1990.

[2] Altschul, S., Madden, T., Schaffer, A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D. Gapped blast and psi-blast: A new generation of protein database search programs. *Nucleic Acids Research* vol. 25, pp. 3389-3420, 1997.

[3] Craig, R. and Liao L. Iterative Weighting of phylogenetic Profiles increases Classification Accuracy. To appear in *The Proceedings of International Conference on Machine Learning and Applications*. (Los Angeles, California, December, 2005).

[4] Enright, A.J., IIiopoulos, I., Kyrpides, N.C. and Ouzounis, C.A. Protein interaction maps for complete genome based on gene fusion events. *Nature*, vol. 403, pp. 86-90, 1999.

[5] Gribskov, M. and Robinson, N. Use of receiver operating characteristic analysis to evaluate sequence matching. *Computers and Chemistry*, vol. 10, pp. 25-33, 1996.

[6] Jaakola, T., Diekhans, M., and 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*. 1999, pp. 95-114.

[7] Joachims, T. Making large-scale svm learning practical. *Advances in kernel Methods – Support Vector Learning*. Scholkopf, B., Burges, C., and Smola A. (eds), MIT Press, 1999. pp. 169-184.

[8] Joachims, T. Transductive Inference for Text Classification using Support Vector Machines. In *Proceedings of the International Conference on Machine Learning (ICML), 1999.*

[9] Leslie, C., Eskin, E., Cohen, A., Weston, J., and Noble, W. Mismatch String Kernels for Discriminative Protein Classification. *Bioinformatics* 20(4), pp. 467-76, 2004.

[10] Liao, L. and Noble, W.S. Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships. The *Journal of Computational Biology*, vol. 10, pp. 857- 868, 2003.

[11] Liberles, D. A., Thoren, A., vonHeijne, G., and Elofsson, A. The use of phylogenetic profiles for gene predictions. *Current Genomics*, vol. 3, pp. 131-137, 2002

[12] Marcotte, E.M., Pellegrini, M., Thompson, M.J., Yeates, T.O. and Eisenberg, D. A combined algorithm for genome-wide prediction of protein function. *Nature*, vol. 402, pp. 83-86, 1999.

[13] Mewes, H.W., Frishman, D., Güldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M.,Morgenstern, B**.,** Münsterkoetter, M., Rudd, S., and Weil, B. MIPS: a database for genomes and protein sequences. *Nucleic Acids Research*, vol. 30, pp. 31-34, 2002.

[14] Narra, K. and Liao L. Use of Extended Phylogenetic Profiles with E-values and Support Vector Machines for Protein Family Classification", *International Journal of Computer and Information Science*, vol. 6, No. 1, 2005.

[15] Pavlidis, P., Weston, J., Cai, J., and Grundy, W.N. Gene functional classification from heterogeneous data. In *The Proceedings of the Fifth International Conference on Computational Biology*. pp. 249-255.

[16] Pellegrini, M., Marcotte, E.M., Thompson, M.J., Eisenberg, D., and Yeates, T.O. Assigning protein functions by comparative genome analysis: Protein phylogenetic profiles. *Proc. Natl. Acad. Sci. USA*, vol. 96, pp. 4285-4288, 1999.

[17] Smith, T.F. and Waterman, W.S. Identification of common molecular subsequences. *Journal of Molecular Biology*, vol. 147, pp. 195-197, 1981.

[18] Vert, J.P. A tree kernel to analyze phylogenetic profiles. *Bioinformatics*, vol. 18 pp. S276-S284, 2002.

[19] V. Vapnik, *Statistical Learning Theory*, Wiley, 1998.