Transductive learning with EM algorithm to classify proteins based on phylogenetic profiles

Roger A. Craig and Li Liao*

Department of Computer and Information Sciences,
University of Delaware,
Newark, DE 19716, USA
E-mail: rcraig@cis.udel.edu E-mail: lliao@cis.udel.edu
*Corresponding author

Abstract: We proposed a novel method for protein classification based on phylogenetic profiles. Each protein's profile was extended with extra bits encoding the phylogenetic tree structure and the likelihood, in the form of weights on profile indices, of the protein's functional family membership in each of the reference genomes. The extended profiles were then integrated as part of a kernel of a support vector machine, which was trained in a transductive learning scheme using the EM algorithm to update the weights. Classification accuracy was greatly increased when tested on the proteome of Saccharomyces cerevisiae using the MIPS classification as a benchmark.

Keywords: protein classification; transductive learning; support vector machines; SVMs; phylogenetic profiles; EM algorithm; data mining; bioinformatics.

Reference to this paper should be made as follows: Craig, R.A. and Liao, L. (2007) 'Transductive learning with EM algorithm to classify proteins based on phylogenetic profiles', *Int. J. Data Mining and Bioinformatics*, Vol. 1, No. 4, pp.337–351.

Biographical notes: Roger A. Craig is a PhD student in the Department of Computer and Information Sciences, University of Delaware. He is primarily interested in computer-based methods for the prediction and classification of proteins, regulatory RNAs, and their interacting partners.

Li Liao received his PhD degree from Peking University and is currently an Assistant Professor of Computer and Information Sciences at the University of Delaware. His research interests and experience span a wide range, including computer simulation of molecular systems, genome sequencing, protein homology detection and genome comparisons.

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 (Altschul et al., 1990; Smith and Waterman, 1981; Narra and Liao, 2005), were developed to compare sequence similarity for protein homology detection. The basis for this type of method is that homologous proteins, evolved from a common ancestral protein via mutations, are likely to remain similar in

sequence composition, while still playing 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%) identity. Over the past decade or so, various techniques have been developed for detecting distant protein homologues, for example, iterative search with refined profiles (Altschul et al., 1997), sophisticated probabilistic models, powerful statistical learning (Liao and Noble, 2003) and some hybrid approaches (Jaakola et al., 1999), to name a few.

The development of computational methods for predicting protein functions has 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 detection by identifying proteins that are related to one another because they are associated in a common structural complex, participate in common metabolic pathways or fuse into a single gene in some genomes (Enright et al., 1999). An important work in this line is the use of phylogenetic profiles for assigning gene functions based on evolutionary and/or coevolutionary patterns across species (Liberles et al., 2002; Marcotte et al., 1999; Pavlidis et al., 2001 and Pellegrini et al., 1999).

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, therefore their phylogenetic profiles consequently show similarity. For example, proteins P1 and P2 are two enzymes, respectively, for two consecutive reactions in a pathway, for this pathway to exist in a genome G, both P1 and P2 have to be present (under an assumption that no alternative enzymes are available for the same reactions). As a result, if we align the phylogenetic profiles (vectors) of protein P1 and P2 on top of each other, they both have ones (or zeros) in the column corresponding to genome G. Furthermore, that pathway, of which the proteins P1 and P2 are two component enzymes, essential for certain cellular functions may necessitate its existence (therefore the presence of both P1 and P2) in a group of evolutionary-related genomes. That is, such functional linkages exert some evolutionary pressure, which is reflected as correlations among the aligned phylogenetic vectors, not only within columns (the vertical direction) but also, perhaps more importantly, across different columns (the horizontal direction). To utilise the phylogenetic profiles for classifying proteins, similarity measures between the phylogenetic profiles have to be defined. Among the first proposed are edit distance and Euclidean distance (Marcotte et al., 1999). Although these simple measures generated useful classification and prediction, an important piece of information is missed out that is these reference genomes are not totally unrelated, rather they are correlated during evolution as represented in a phylogenetic tree. The recent focus has been given to incorporating the evolutionary relations that are represented in the phylogenetic tree of the genomes into profile similarity (Liberles et al., 2002; Vert, 2002).

The importance of the phylogenetic tree in the study of phylogenetic profiles has been recognised in early work, e.g., Liberles et al. (2002), where a method was proposed to utilise the historical evolutions of two proteins to account for their similarity (or dissimilarity). Evolutionary relationships 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 minimises 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 minimising the differences at the tree branches as a way to incorporate the evolutionary histories in comparing two phylogenetic profiles was generalised to include all evolutionary patterns and endowed with probabilistic formulation and interpretation (Vert, 2002). 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 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 the 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 SVM 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 threefold 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 that incorporates both the phylogenetic tree (to account for correlations along the horizontal direction of aligned profiles) and the likelihood of a protein's presence in the current genomes (to account for correlations along the vertical direction of aligned profiles) into training a SVM. Particularly, our method does not require preset probabilities for gene retention and creation during speciation. Not only it is difficult to justify any a priori values for these probabilities and the associated assumptions as used in the tree kernel, but 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 0.1–0.9 for gene creation, correspondingly) at 0.1 intervals and did not notice any significant changes in classification accuracy – this phenomenon is quite contrary to the 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 (Narra and Liao, 2005). 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 extra bits that correspond to the interior nodes of the individually labelled phylogenetic tree and take values of the scores at these interior nodes (see Figure 2). We then input these extended phylogenetic profiles into a polynomial kernel SVM 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 also allows for incorporating correlations of proteins along the vertical direction of the aligned profiles into training the classifier. This is achieved by augmenting the scoring scheme with weighted averaging at the tree leaves, with the weighting factors reflecting the probability of a presence (or absence) for any protein (not just the protein whose profile is being extended) at specific current genomes. Such probabilities of proteins presence are collected using Maximum Likelihood (ML) methods from the training data, separately for the positive and negative examples. Therefore, the extended profile now carries not only the information about each individual protein's correlated presence at different genomes but also some collective information about the specific proteome (i.e., a column in the aligned profiles). In order for this refined scoring scheme to work with the testing examples as well, the weighting factors have to be known for the positive and negative testing examples before generating the extended profiles. Because whether a testing example is positive or negative is not known beforehand, we propose to solve this circular problem in a way of transductive learning – weighting factors are collected using Expectation Maximisation (EM) method from the predicted results of the testing data (Craig and Liao, 2005). The profiles are thus updated with the predicted results, iteratively, and the iteration will stop when a preset criterion is met, in our case, when a Kullback-Leibler (KL) 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 Vert (2002), our method, particularly when refined with the iterative weighting, has shown significantly superior performance.

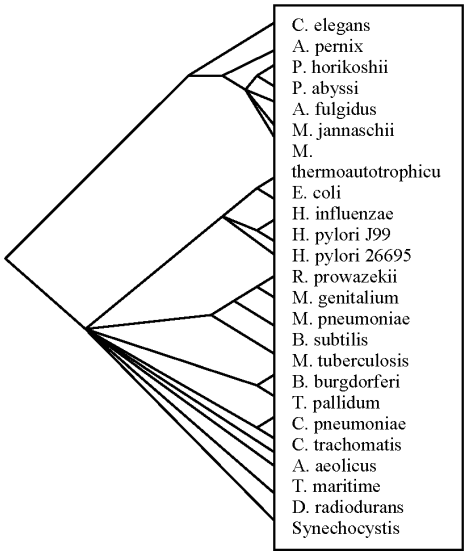
2 Overall approach

2.1 Data set

The data set used in this work is the same data set as in Pavlidis et al. (2001) and Vert (2002), 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 ten genes were extracted from the several hundred classes in the Munich Information Center for Protein Sequences Comprehensive Yeast Genome Databases (Mewes et al., 2002). 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 than 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 Vert (2002).

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



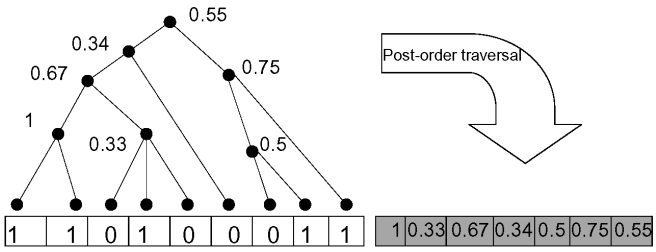
A three-fold cross validation was adopted for the experiments. For each functional class, two-third of its members is randomly selected as positive training examples, and the remaining third as positive testing examples. Genes not belonging in that class were randomly split into two-thirds as negative training and one-third as negative testing examples. This process is repeated ten 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 the protein is present. Once all interior nodes are assigned with scores, the second step is to flatten the

score-clad tree into a vector mapping of the 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 schematic illustration of how a phylogenetic profile 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.

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



The rationale for extending phylogenetic profiles to aid the learning and classification is strikingly similar to that of using a kernel function to map the input vectors into higher dimension feature space. Unlike the use of kernels where the mapping is implicit, here the mapping is given explicitly,

$$x \to \phi_T(x) \tag{1}$$

as prescribed above, with x standing for an 24-vector, $\phi_T(x)$ for the extended 38-vector and T stands for the phylogenetic tree. Such explicit mapping has the advantage of allowing for incorporation of domain-specific knowledge – the phylogenetic tree.

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:

$$S(k) = (1/|C|)\sum_{i \in C} S(i) W_{s(i)}(i) \tag{2}$$

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. As the weights can be interpreted as probability distribution of proteins presence in a genome, the frequency from counting gives the ML estimation of the probability distribution. 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 equation (2) are real value numbers. To do this, we first use a threshold E0 on

E-value as if for converting the profiles into binary, so as to collect the frequency based on sign($E0 - S(i)$), which will then be used as weighting factors $W_{\text{sign}(E0-s(i))}(i)$. The equation (2) is thus modified as the following.

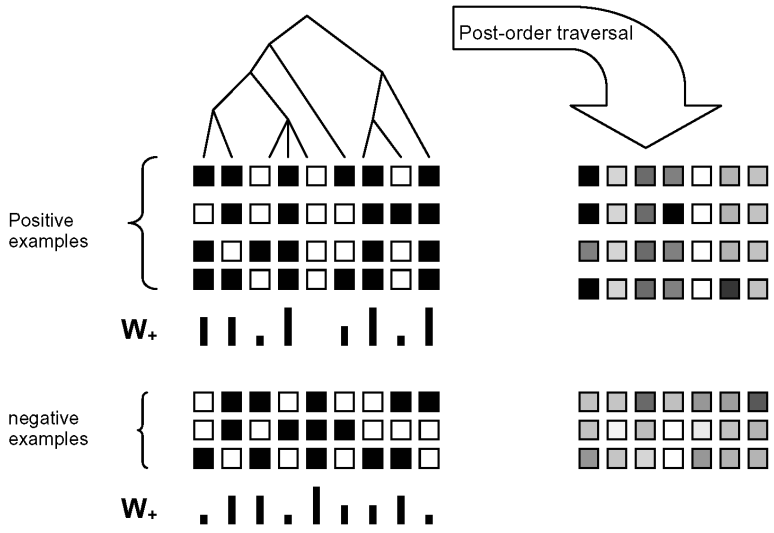
$$S(k) = (1/L)\sum_{i\in L} S(i)W_{\text{sign}(E0-s(i))}(i). \tag{2′}$$

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. The procedure for weighted extension, schematically illustrated in Figure 3, infuses into the mapping given in equation (1), some extra domain-specific information – W's, the probability distribution of presence and absence of proteins over genomes,

$$x \to \phi_{T,w}(x). \tag{3}$$

Since the weighting factor W's 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 family-specific weighting factors for the positive training examples (family members) and for the negative training examples (non-family members) separately. That is, when using equation (2) or (2′) to extend a profile in the positive training set, $W_{\pm 1}(i)$ at a leaf i is collected from positive training examples only, whereas when using equation (2) or (2′) to extend a profile in the negative training set, $W_{\pm 1}(i)$ at a leaf i is collected from negative training examples only. The difficulty with such a scheme of separate 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?

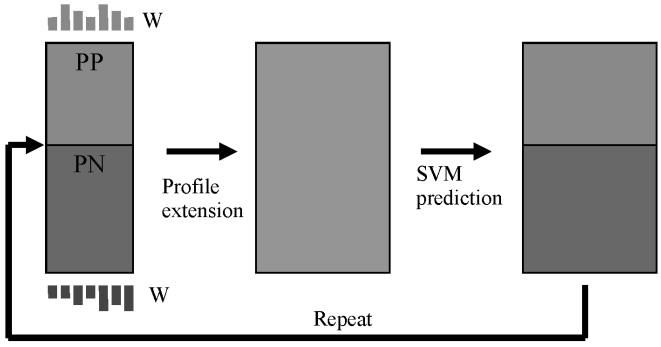
Figure 3 Schematic illustration of weighted extension of phylogenetic profiles. Black squares indicate 1 and empty squares indicate 0. The weighting factors $W+$ are represented by vertical bars. The height of the vertical bars indicates the frequency of having a black square in a given column. The extended bits are grey-shaded indicating real number values



Before introducing our solution to overcome this difficulty, it is worthwhile to examine the effectiveness of using the family-specific W's in the mapping equation (3). For that, let us assume that we can cheat for a while, namely, the W's for the family members are collected from all positive examples (both training and testing) and then are used to map the profiles of the family members into 38-vectors. Similarly, the W's for non-family members are collected from all negative examples (both training and testing) and then are used to map the profiles of these non-family members into 38-vectors. Such extended 38 vectors are then input for classification, using the same SVM (Section 2.5) and cross-validation scheme (Sections 2.1 and 3) as used for other cases in this study, and are shown to produce the best classification accuracy (the top curve in Figure 5, with details explained in Section 3). Now that we have demonstrated the very significant role of the family-specific W's in data representation, let us address the difficulty we face in using them for the testing examples where we do not know a priori their labels as needed for collecting W's and then applying them in turn for the mapping to the extended profiles. In other words, the simple ML estimation for W's cannot be applied, because of the latent information, i.e., the labels of these testing examples.

To overcome this difficulty, we adopted the transductive learning paradigm (Joachims, 1999b; Vapnik, 1998) and combined it with EM procedure. That is, we are allowed to look into the predictions made on the testing examples (E-step) and collect weighting factors from the Predicted Positives (PP) and negatives, respectively (M-step). In E-step, 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 (Joachims, 1999b), one piece of extra information allowed for use is the ratio ρ of true positives vs. 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 PPs and collect weighting factors W's. The rest are taken as Predicted Negatives (PNs), and the corresponding W's are calculated in a similar way. Then, the profiles of the testing examples are updated, depending on whether they are PP or negative, with the 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. The iterative procedure is shown schematically in Figure 4. 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. The EM procedure presented here is quite similar to the Baum-Welch algorithm used in hidden Markov models when training data are unlabelled, but they differ in a significant way: Baum-Welch algorithm is applied during the model training, whereas the EM procedure here is applied in a transductive learning paradigm.

Figure 4 Schematic illustration of the EM procedure to obtain the weighting factors W for Predicted Positive examples (PP) and Predicted Negative (PN) examples. The profile extension step is detailed in Figure 3. The rectangle in the middle is shaded uniformly to indicate that the labels of the testing examples are assumed to be unknown until predicted by the SVM



2.4 Stopping criterion

In order to put this method into practical use, a criterion must be established to stop the iterative reweighting procedure. 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., and KL distance to measure the change brought about on W's between two consecutive iterations. For the jth bit (i.e., leaf j) of the 24 bits, calculate the KL distance as

$$d_j = \sum_{a=-1,+1} \{W_a(j) \log[W_a(j) / W'_a(j)]\}, \tag{4}$$

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

$$d = \left(\sum_{j=1, \text{ to } 24} d_j \right) / 24, \tag{5}$$

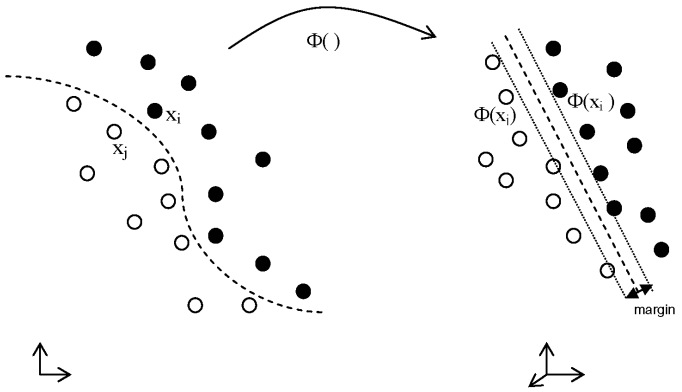
for each iteration to see if it converges and at least shows a trend of convergence. We also monitor the classification accuracy at each iteration. We hope to see that, over iterations, while the classification accuracy increases the average KL distance decreases. Then, an empirical threshold can be 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 6.

2.5 SVM and kernels

The classifier used here is a SVM with a polynomial kernel. As a powerful statistical learning method, SVMs, originally proposed by Vapnik (1998), have recently been applied with remarkable success in bioinformatics problems, including remote protein homology detection, microarray gene expression analysis and protein secondary structure prediction (Schoelkopf et al., 2004).

The basic idea of SVMs is simple; it is to find a hyperplane that separates two classes of objects, as represented as points in a vector space, with the maximum margin to the boundary lines, such a hyperplane ensures good generalisation – unseen data are then classified according to their location with respect to the hyperplane. The power of SVMs comes partly from the data representation, where an entity, e.g., a protein, is represented by a set of attributes. However, how those attributes contribute to distinguishing a true positive (filled dot in Figure 5) from a true negative (empty circle) may be quite complex. In other words, the boundary line between the two classes, if depicted in a vector space, can be highly non-linear (dashed line in the left panel of Figure 5). The SVMs method will find a non-linear mapping that transform the data from the original space, called input space, into a higher dimensional space, called feature space, where the data can be linearly separable (right panel of Figure 5).

Figure 5 Schematic illustration of non-linear mapping of data from input space to feature space, where a maximum margin hyperplane is found, for a SVM



In general, the mapping can be quite complex and the dimension can be very (infinitely) high in order for the mapped data to be linearly separable. The trick of SVMs is the use of kernel functions, which define how the dot product between two points in the feature space, which is the only quantity needed to solve the quadratic programming problem for finding the maximum margin hyperplane in the feature space. The use of kernel functions avoids explicit mapping to high-dimensional feature space; high dimensionality often poses difficult problems for learning such as over-fitting, thus termed the curse of dimensionality. Two commonly used generic kernels are Gaussian Radial Basis Function (RBF) and polynomial functions. For vectors x and y, Gaussian RBF is defined as

$$K(x, y) = \exp[-(|x - y|^2 / c)], \tag{6}$$

and the polynomial kernel is defined as

$$\text{Polynomial } K(x, y) = [1 + s(x \cdot y)]^d, \tag{7}$$

where, c, s and d are parameters adjustable in the software package SVM Light (Joachims, 1999a). Both kernels are experimented with the default values for c, s and d, and the polynomial kernel yielded better results that are reported in this paper.

It is worth noting that, overall, our method can be viewed either as a hybrid that employs both an explicit mapping equation (3) and a generic kernel equation (7) in tandem or as a conventional SVM but with a specially engineered kernel that is composed of two parts, equations (3) and (7).

3 Results

The results of the threefold cross-validation experiments on the dataset of 2465 yeast genes are summarised in Figures 6 and 7. The function prediction for each class of the 133 classes is measured by ROC score. ROC score is the normalised area under a curve that plots the true positives as a function of false positives for varying classification thresholds (Gribskov and Robinson, 1996). ROC50 scores are ROC scores that are calculated by integrating the area up to the 50th false positive. The ROC50 scores are in a range of [0, 1], with 1 for a perfect classification. Recall that, for each functional class, two-thirds 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. To ensure the results are robust to the data preparation procedure, we repeat the experiments over ten random runs. In each run, the training and testing data are independently prepared. ROC50 scores reported in this study are all averaged over ten random runs.

In Figure 6, 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 (Vert, 2002) 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 5, the performance of random tree with ten 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 6, 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 7 (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 optimisation problem with a well-defined objective function, the KL distance of weighting factors seems to be a useful pragmatic substitute for that purpose.

Figure 6 Histogram of ROC50 scores for various experiments on the 133 families

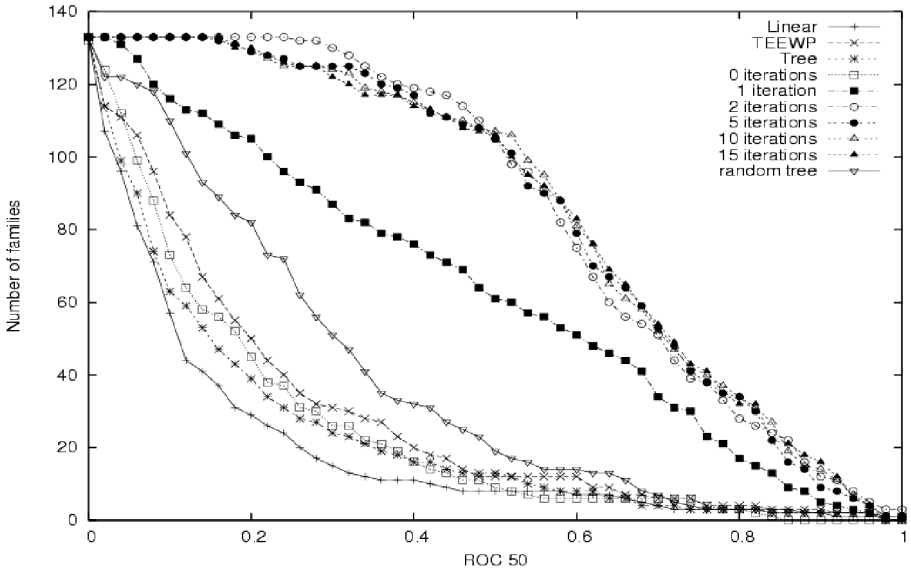
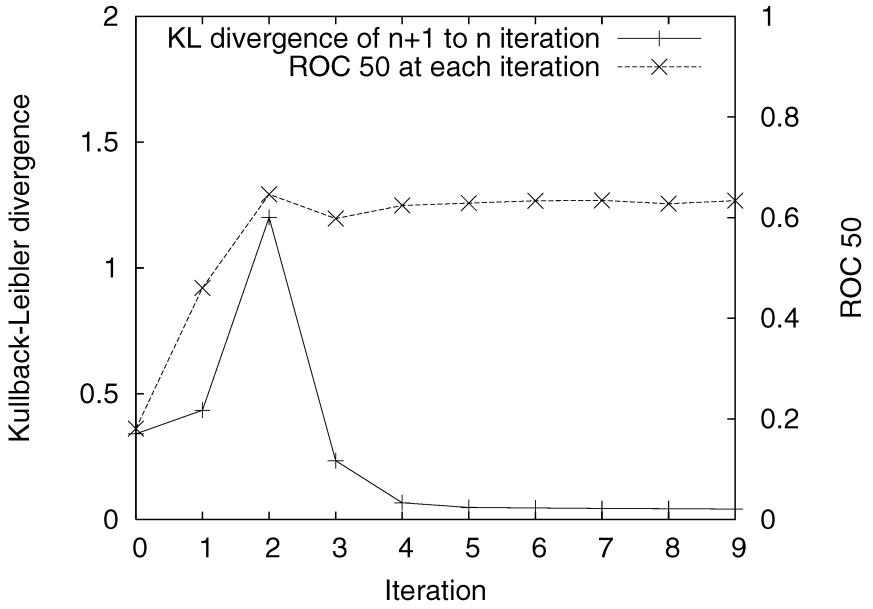


Figure 7 The KL distance and average ROC50 scores over ten iterations



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 1) 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).

Table 1 Performance of various methods for the 15 selected families

Name	*No. of ORFs*	*Naïve*	*Tree*	*TEEWP*	*IR, N = 0*	*IR, N = 1*	*IR, N = 2*	*IR, N = 15*
Amino acid transporters	25	0.740	0.810	0.749	0.856	0.969	0.956	*0.978*
Fermentation	34	0.680	0.730	*0.980*	0.755	0.915	0.919	0.979
ABC transporters	28	0.640	0.870	*1.000*	0.826	0.958	0.966	0.973
C-compound, carbohydrate transport	42	0.590	0.680	0.754	0.801	0.865	0.879	*0.922*
Amino acid biosynthesis	118	0.370	0.460	0.448	0.511	0.663	0.784	*0.827*
Amino acid metabolism	204	0.350	0.320	0.412	0.424	0.479	0.590	*0.673*
TCA pathway	25	0.330	0.480	0.396	0.550	0.845	*0.941*	0.892
Transport facilitation	310	0.330	0.280	0.344	0.397	0.535	*0.720*	0.673
Organisation of plasma membrane	144	0.310	0.300	0.454	0.386	0.577	0.738	*0.791*
Amino acid degradation (catabolism)	35	0.300	0.520	0.651	0.444	0.774	*0.853*	0.831
Lipid and fatty-acid transport	16	0.290	0.520	0.685	0.408	0.715	*0.865*	0.848
Homeostasis of other cations	23	0.260	0.330	0.000	0.488	0.853	0.793	*0.921*
Glycolysis and gluconeogenesis	35	0.250	0.660	0.607	0.495	0.799	0.891	*0.928*
Metabolism	1062	0.240	0.200	0.250	0.240	0.211	*0.735*	0.317
Cellular import	101	0.200	0.270	0.163	0.204	0.234	*0.701*	0.617
tRNA modification	17	0.150	0.320	0.913	0.283	0.790	*0.921*	0.914
Average	–	0.377	0.484	0.550	0.504	0.699	*0.828*	0.818

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 the previous methods (Vert, 2002). 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 combined with a EM procedure that allows for the use of prediction from the previous iteration. It differs from the standard transductive learning (Joachims, 1999b; Vapnik, 1998) in that, the classifier, a SVM 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 the 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 optimisation problem will be pursued as future research.

References

Altschul, S., Gish, W., Miller, W., Myers, E. and Lipman, D. (1990) 'Basic local alignment search tool', *Journal of Molecular Biology*, Vol. 215, pp.403–410.

Altschul, S., Madden, T., Schaffer, A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D. (1997) 'Gapped blast and psi-blast: a new generation of protein database search programs', *Nucleic Acids Research*, Vol. 25, pp.3389–3420.

Craig, R. and Liao, L. (2005) '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.

Enright, A.J., IIiopoulos, I., Kyrpides, N.C. and Ouzounis, C.A. (1999) 'Protein interaction maps for complete genome based on gene fusion events', *Nature*, Vol. 403, pp.86–90.

Gribskov, M. and Robinson, N. (1996) 'Use of receiver operating characteristic analysis to evaluate sequence matching', *Computers and Chemistry*, Vol. 10, pp.25–33.

Jaakola, T., Diekhans, M. and Haussler, D. (1999) 'Using the fisher kernel method to detect remote protein homologies', *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pp.95–114.

Joachims, T. (1999a) 'Making large-scale svm learning practical', in Scholkopf, B., Burges, C. and Smola, A. (Eds.): *Advances in Kernel Methods –Support Vector Learning*, MIT Press, Cambridge, Massachusetts, pp.169–184.

Joachims, T. (1999b) 'Transductive inference for text classification using support vector machines', *Proceedings of the International Conference on Machine Learning (ICML)*.

Liao, L. and Noble, W.S. (2003) '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.

Liberles, D.A., Thoren, A., vonHeijne, G. and Elofsson, A. (2002) 'The use of phylogenetic profiles for gene predictions', *Current Genomics*, Vol. 3, pp.131–137.

Marcotte, E.M., Pellegrini, M., Thompson, M.J., Yeates, T.O. and Eisenberg, D. (1999) 'A combined algorithm for genome-wide prediction of protein function', *Nature*, Vol. 402, pp.83–86.

Mewes, H.W., Frishman, D., Güldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgenstern, B., Münsterkoetter, M., Rudd, S. and Weil, B. (2002) 'MIPS: a database for genomes and protein sequences', *Nucleic Acids Research*, Vol. 30, pp.31–34.

Narra, K. and Liao, L. (2005) '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, pp.58–63.

Pavlidis, P., Weston, J., Cai, J. and Grundy, W.N. (2001) Gene functional classification from heterogeneous data', *The Proceedings of the Fifth International Conference on Computational Biology*, pp.242–248.

Pellegrini, M., Marcotte, E.M., Thompson, M.J., Eisenberg, D. and Yeates, T.O. (1999) 'Assigning protein functions by comparative genome analysis: protein phylogenetic profiles', *Proc. Natl. Acad. Sci.*, USA, Vol. 96, pp.4285–4288.

Schoelkopf, B., Tsuda, K. and Vert, J-P. (Eds.) (2004) *Kernel Methods in Computational Biology*, MIT Press, Cambridge, Massachusetts.

Smith, T.F. and Waterman, W.S. (1981) 'Identification of common molecular subsequences', *Journal of Molecular Biology*, Vol. 147, pp.195–197.

Vapnik, V. (1998) *Statistical Learning Theory*, John Wiley and Sons, New York.

Vert, J.P. (2002) 'A tree kernel to analyze phylogenetic profiles', *Bioinformatics*, Vol. 18, pp.S276–S284.