BIOINFORMATICS

Vol. 21 Suppl. 1 2005, pages i38–i46
doi:10.1093/bioinformatics/bti1016

Kernel methods for predicting protein–protein interactions

Asa Ben-Hur[1,*] and William Stafford Noble[1,2]

[1]Department of Genome Sciences and [2]Department of Computer Science and Engineering, University of Washington, Seattle, WA, USA

Received on January 15, 2005; accepted on March 27, 2005

ABSTRACT

Motivation: Despite advances in high-throughput methods for discovering protein–protein interactions, the interaction networks of even well-studied model organisms are sketchy at best, highlighting the continued need for computational methods to help direct experimentalists in the search for novel interactions.

Results: We present a kernel method for predicting protein–protein interactions using a combination of data sources, including protein sequences, Gene Ontology annotations, local properties of the network, and homologous interactions in other species. Whereas protein kernels proposed in the literature provide a similarity between single proteins, prediction of interactions requires a kernel between pairs of proteins. We propose a pairwise kernel that converts a kernel between single proteins into a kernel between pairs of proteins, and we illustrate the kernel's effectiveness in conjunction with a support vector machine classifier. Furthermore, we obtain improved performance by combining several sequence-based kernels based on k-mer frequency, motif and domain content and by further augmenting the pairwise sequence kernel with features that are based on other sources of data.

We apply our method to predict physical interactions in yeast using data from the BIND database. At a false positive rate of 1% the classifier retrieves close to 80% of a set of trusted interactions. We thus demonstrate the ability of our method to make accurate predictions despite the sizeable fraction of false positives that are known to exist in interaction databases.

Availability: The classification experiments were performed using PyML available at http://pyml.sourceforge.net. Data are available at: http://noble.gs.washington.edu/proj/sppi

Contact: asa@gs.washington.edu

1 INTRODUCTION

Most proteins perform their functions by interacting with other proteins.Therefore, information about the network of interactions that occur in a cell can greatly increase our understanding of protein function. Several experimental assays that probe interactions in a high-throughput manner are now available. These methods include the yeast two-hybrid screen and methods based on mass spectrometry (see von Mering *et al*., 2002 and references therein). The data obtained by these methods are partial: each experimental assay can identify only a subset of the interactions, and it has been estimated that for the organism with the most complete interaction network, namely yeast, only about half of the complete 'interactome' has been discovered (von Mering *et al*., 2002). In view of the very small overlap between interactions discovered by various high-throughput studies, some of them using the same method, the actual number of interactions is likely to be much higher. Computational methods are therefore required for discovering interactions that are not accessible to high-throughput methods. These computational predictions can then be verified by more labor-intensive methods.

A number of methods have been proposed for predicting protein–protein interactions from sequence. Sprinzak and Margalit (2001) have noted that many pairs of structural domains are over-represented in interacting proteins and that this information can be used to predict interactions. Several authors have proposed Bayesian network models that use the domain or motif content of a sequence to predict interactions (Deng *et al*., 2002; Gomez *et al*., 2003; Wang *et al*., 2005). The pairwise sequence kernel was independently proposed in a recent paper (Martin *et al*., 2005) with a sequence representation by 3mers. Other sequence-based methods use coevolution of interacting proteins by comparing phylogenetic trees (Ramani and Marcotte, 2003), correlated mutations (Pazos and Valencia, 2002) or gene fusion which works at the genome level (Marcotte *et al*., 1999). An alternative approach is to combine multiple sources of genomic information—gene expression, Gene Ontology (GO) annotations, transcriptional regulation, etc. to predict comembership in a complex (Zhang *et al*., 2004; Lin *et al*., 2004).

One can consider two variants of the interaction prediction problem: predicting comembership in a complex or predicting direct physical interaction. In this work, we focus on the latter task, and use interactions that are derived from the BIND database (Bader *et al*., 2001), which makes a distinction between experimental results that yield comembership in a complex and interactions that are more likely to be direct ones.

[*]To whom correspondence should be addressed.

Kernel methods, and in particular support vector machines (SVMs) (Schölkopf and Smola, 2002), have proven useful in many difficult classification problems in bioinformatics (Noble, 2004). The learning task we are addressing involves a relationship between pairs of protein sequences: whether two pairs of sequences are interacting or not. The standard sequence kernels (A kernel is a measure of similarity that satisfies the additional condition of being a dot product in some feature space; see Schölkopf and Smola, 2002 for details.) described in the literature measure similarity between single proteins. We propose a method for converting a kernel defined on single proteins into a pairwise kernel, and we describe the feature space produced by that kernel.

Our basic method uses motif, domain and *k*mer composition to form a pairwise kernel, and achieves better performance than simple methods based on BLAST or PSI-BLAST. However, because it is difficult to predict interactions from sequence alone, we incorporate additional sources of data. These include kernels based on similarity of GO annotations, a similarity score to interacting homologs in other species and the mutual-clustering coefficient (Goldberg and Roth, 2003) that measures the tendency of neighbors of interacting proteins to interact as well. Adding these additional data sources significantly improves our method's performance relative to a method trained using only the pairwise sequence kernel. Using kernel methods for combining data from heterogeneous sources of data allows us to use high-dimensional sequence data, whereas other studies on predicting protein–protein interactions (Zhang *et al*., 2004; Lin *et al*., 2004) use a low dimensional representations which are appropriate for any type of classifier.

2 KERNELS FOR PROTEIN–PROTEIN INTERACTIONS

SVMs and other kernel methods derive much of their power from their ability to incorporate prior knowledge via the kernel function. Furthermore, the kernel approach offers the ability to easily apply kernels to diverse types of data, including fixed-length vectors (e.g. microarray expression data), variable-length strings (DNA and protein sequences), graphs and trees. In this work, we employ a diverse collection of kernels described in this section.

2.1 Pairwise kernels

The kernels proposed in the literature for handling genomic information, e.g. sequence kernels such as the motif and Pfam kernels presented later in the section, provide a similarity between pairs of sequences, or more generally, a similarity between a representation of a pair of proteins. Therefore, such kernels are not directly applicable to the task of predicting protein–protein interactions, which requires a similarity between two pairs of proteins. Thus, we want a function $K((X_1, X_2), (X_1', X_2'))$ that returns the similarity between proteins X_1 and X_2 compared with proteins X_1' and X_2'. We call a kernel that operates on individual genes or proteins a 'genomic kernel', and a kernel that compares pairs of genes or proteins a 'pairwise kernel'. Pairwise kernels can be computed either indirectly, by way of an intermediate genomic kernel, or directly using features that characterize pairs of proteins.

The most straightforward way to construct a pairwise kernel is to express the similarity between pairs of proteins in terms of similarities between individual proteins. In this approach, we consider two pairs to be similar to one another when each protein of one pair is similar to one protein of the other pair. For example, if protein X_1 is similar to protein X_1', and X_2 is similar to X_2', then we can say that the pairs (X_1, X_2) and (X_1', X_2') are similar. We can translate these intuitions into the following pairwise kernel:

$$K((X_1, X_2), (X_1', X_2')) = K'(X_1, X_1')K'(X_2, X_2') + K'(X_1, X_2')K'(X_2, X_1'),$$

where $K'(\cdot, \cdot)$ is any genomic kernel. This kernel takes into account the fact that X_1 can be similar to either X_1' or X_2'.

An alternative to the above approach is to represent a pair of sequences (X_1, X_2) explicitly in terms of the domain or motif pairs that appear in it. This representation is motivated by the observation that some domains are significantly over-represented in interacting proteins (Sprinzak and Margalit, 2001). A similar observation holds for sequence motifs as well. Given a pair of sequences X_1, X_2 represented by vectors $\mathbf{x}_1, \mathbf{x}_2$, with components $x_i^{(1)}, x_i^{(2)}$ we form the vector $\mathbf{x}_{12}$ with components $x_i^{(1)}x_j^{(2)} + x_i^{(2)}x_j^{(1)}$. We can now define the explicit pairwise kernel:

$$K((X_1, X_2), (X_1', X_2')) = K'(\mathbf{x}_{12}, \mathbf{x}_{12}'), \tag{1}$$

where $\mathbf{x}_{12}$ is the pairwise representation of the pair (X_1, X_2), and $K'(\cdot, \cdot)$ is any kernel that operates on vector data. It is straightforward to check that for a linear kernel function, the pairwise and explicit pairwise kernels are identical. The explicit representation can be used in order to rank the relevance of motif pairs with respect to the classification task. This ranking is accomplished, e.g. by sorting the motif pairs according to the magnitude of the corresponding weight vector components.

2.2 Sequence kernels

We use three sequence kernels in this work: the spectrum kernel (Leslie *et al*., 2002), the motif kernel (Ben-hur and Brutlag, 2003) and the Pfam kernel (Gomez *et al*., 2003). The feature space of these kernels is a set of sequence models, and each component of the feature space representation measures the extent to which a given sequence fits the model. The spectrum kernel models a sequence in the space of all *k*mers, and its features count the number of times each *k*mer appears in the sequence.

The sequence models for our motif kernel are discrete sequence motifs, providing a count of how many times a discrete sequence motif matches a sequence. To compute the motif kernel we used discrete sequence motifs from the eMotif database (Nevill-Manning *et al*., 1997). Yeast ORFs contain occurrences of 17 768 motifs out of a set of 42 718 motifs.

Finally, the Pfam kernel uses a set of hidden Markov models (HMMs) to represent the domain structure of a protein, and is computed by comparing each protein sequence with every HMM in the Pfam database (Sonnhammer *et al*., 1997). Every such protein–HMM comparison yields an E-value statistic. Pfam version 10.0 contains 6190 domain HMMs; therefore, each protein is represented by a vector of 6190 log E-values. This Pfam kernel has been used previously to predict protein–protein interactions (Gomez *et al*., 2003), though not in conjunction with the pairwise kernel described above.

For all three sequence kernels we use a a normalized linear kernel, $K(x, y)/\sqrt{K(x, x)K(y, y)}$; in the case of the Pfam kernel we first performed an initial step of centering the kernel.

2.3 Non-sequence kernels

An alternative to using the pairwise kernel is the following:

$$K((X_1, X_2), (X_1', X_2')) = K'(X_1, X_2)K'(X_1', X_2'). \quad (2)$$

This kernel is appropriate when similarity within the pair is directly related to the likelihood that a pair of proteins interact. In fact, this is a valid kernel even if K' is not a kernel, because in this formulation K' is simply a feature of the pair of proteins. Consider GO annotations, for example: a pair of proteins is more likely to interact if the two proteins share similar annotations. In addition to GO annotation we also consider local properties of the interaction network, and homologous interactions in other species. We summarize these properties as a vector of scores $\mathbf{s}(X_1, X_2)$, such that the kernel for the non-sequence data can be any kernel appropriate for vector data:

$$K_{\text{non-seq}}((X_1, X_2)), (X_1', X_2')) = K'(\mathbf{s}(X_1, X_2), \mathbf{s}(X_1', X_2')), \quad (3)$$

where we chose to use a Gaussian kernel for K'.

2.3.1 A GO kernel Proteins that are not present in the same cellular component or that participate in different biological processes are less likely to interact. We represent this prior knowledge using a kernel that measures the similarity of the GO (Gene Ontology Consortium, 2000) annotations of a pair of proteins, one kernel for each of the three GO hierarchies. The feature space for the GO kernel is a vector space with one component for each node in the directed acyclic graph in which GO annotations are represented. Let the annotations (nodes in the GO graph) assigned to protein p be denoted by $\mathcal{A}_p$. Note that, in GO, a single protein can be assigned several annotations. A component of the vector corresponding to node a is non-zero if a or a parent of a is in $\mathcal{A}_p$.

We consider two ways in which to define the dot product in this space. When the non-zero components are set equal to 1, then when each protein has a single annotation, and the annotatinos are on a tree, the dot product between two proteins is the height of the lowest common ancestor of the two nodes. An alternative approach assigns annotation a a score of $-\log p(a)$, where $p(a)$ is the fraction of proteins that have annotation a. We then score the similarity of annotations a, a' as $\max_{a''\in\text{ancestors}(a)\cap\text{ancestors}(a')} -\log p(a'')$. In a tree topology, this score is the similarity between the deepest common ancestor of a and a', because the node frequencies are decreasing along a path from the root to any node. The score is a dot product with respect to the infinity norm on the annotation vector space. This also holds when the proteins have more than one annotation and the similarity between their annotations is defined as the maximum similarity between any pair of annotations. When one of the proteins has an unknown GO annotation, the kernel value is set to 0.

2.3.2 Interactions in other species It has been shown that interactions in other species can be used to validate or infer interactions (Yu *et al*., 2004): the existence of interacting homologs of a given pair of proteins implies that the original proteins are more likely to interact. We quantify this observation with the following homology score for a pair of proteins (X_1, X_2):

$$h(X_1, X_2) = \max_{i\in\mathcal{H}(X_1), j\in\mathcal{H}(X_2)} I(i, j) \times \min(l(X_1, X_i), l(X_2, X_j)),$$

where $\mathcal{H}(X)$ is the set of non-yeast proteins that are significant BLAST hits of X, $I(i, j)$ is an indicator variable for the interaction between proteins i and j, and $l(X_k, X_i)$ is the negative of the log E-value provided by BLAST when comparing protein k with protein i in the context of a given sequence database. We used interactions in human, mouse, nematode and fruit fly to score the interactions in yeast.

2.3.3 Mutual clustering coefficient Protein–protein interaction networks tend to be 'cliquish'; i.e. the neighbors of interacting proteins tend to interact. Goldberg and Roth (2003) quantified this cohesiveness using the mutual clustering coefficient (MCC). Given two proteins u, v, their MCC can be quantified, by the Jaccard coefficient $|N(v) \cup N(u)|/|N(v) \cap N(u)|$, where $N(x)$ is the set of neighbors of a protein x in an interaction network. In our classification experiments we performed cross-validation where the MCC in each cross-validation fold is computed with respect to the interactions that occur in the training set of that particular fold.

2.4 Combining kernels

Given a genomic kernel K, we denote by $K_{\text{p}}(K)$ the pairwise kernel that uses K. When several genomic kernels are available, the final kernel can be defined as $\sum_i K_{\text{p}}(K_i)$ or as $K_{\text{p}}(\sum_i K_i)$. Using $K_{\text{p}}(\sum_i K_i)$ mixes features between the individual kernels, while the feature space for $\sum_i K_{\mathrm{p}}(K_i)$ includes pairs of features that originate from the same genomic kernel. In practice, the results from these two different approaches were very close, and the mixing approach was used because of its lower memory requirement. A Gaussian or polynomial kernel can be introduced at several stages: instead of the linear genomic kernel as: $\exp(-\gamma(K_{\mathrm{p}}(P, P) - 2K_{\mathrm{p}}(P, P') + K_p(P', P'))$, where P, P' are two pairs of proteins. We have not tried introducing a non-linear kernel at the level of the genomic kernel; a Gaussian kernel at the level of the pairwise kernel performed similar to the 'linear' pairwise kernel, despite the high dimensionality of the resulting feature space. The results reported in this paper are computed using 'linear' pairwise kernels.

2.5 Incorporating interaction reliability in training

Several studies of protein–protein interaction data have noted that different experimental assays produce varying levels of false positives and have proposed methods for finding which interactions are likely to be reliable (von Mering *et al*., 2002; Sprinzak *et al*., 2003; Deane *et al*., 2002) (see Section 3.1 for details). We incorporate this knowledge about the reliability of protein–protein interactions into the training procedure using the SVM soft-margin parameter C (Schölkopf and Smola, 2002). This parameter puts a penalty on patterns that are misclassified or are close to the SVM decision boundary. Each training example receives a value of C that depends on its reliability. For a training set with an equal number of positive and negative examples we use two values: C_{high} for interactions believed to be reliable and for negative examples; C_{low} for positive examples that are not known to be reliable.

3 METHODS

3.1 Interaction data

We focus on the prediction of physical interactions in yeast and use interaction data from the BIND database (Bader *et al*., 2001). BIND includes published interaction data from high-throughput experiments as well as curated entries derived from published papers. The advantage of BIND is that it provides an explicit distinction between direct physical interactions and comembership in a complex.

3.1.1 Positive and negative examples We use physical interactions from BIND as positive examples, for a dataset comprising 10 517 interactions among 4233 yeast proteins (downloaded July 9, 2004). We eliminated self interactions from the dataset since such interactions do not require a pairwise kernel, and the GO and MCC features are not appropriate in this case. As negative examples we select random, non-interacting pairs from the 4233 interacting proteins; the number of negative examples was taken as equal to the number of positive examples. In view of the large number of protein pairs compared with the number of interactions, such a set of negative examples is likely to contain very few proteins that interact.

High-throughput protein–protein interaction data contain a large fraction of false positives, estimated to be up to 50% in some experiments (von Mering *et al*., 2002). Therefore, we prepared a set of BIND interactions that are expected to have a low rate of false positives. We use these reliable interactions in two ways. We evaluate the performance of our method on the reliable interactions because they are more likely to reflect the true performance of the classifier. We also use reliability to set the value of the SVM soft-margin parameter as discussed in Section 2.5. 'Gold standard' interactions can be derived from several sources:

- Interactions corroborated by interacting yeast paralogs. Deane *et al*. (2002) find 2829 interactions from the DIP database that are supported by their paralogous verification method (PVM). The estimated false positive rate of this method is 1%.
- Interactions that are supported by interacting homologs in multiple species are likely to be correct (Yu *et al*., 2004).
- Interactions that are discovered by different experimental assays were estimated to be correct 95% of the time (Sprinzak *et al*., 2003).
- Highly reliable methods, e.g. interactions derived from crystallized complexes.

We do not use PVM-validated interactions because they contain several biases.

- The test set is biased toward interactions that can be easily discovered by sequence similarity.
- The list of PVM-validated interactions cannot be used as-is to set the SVM soft-margin parameter in training because this may incorporate information about interactions that are in the test set.

Also, we do not include interactions validated by interacting homologs in other species, since that information is included in the data as a feature. Therefore, for the purpose of assessing performance we use a list of 750 interactions that were validated by high-quality or multiple assays. For setting the SVM soft-margin parameter we augment the 750 interactions with PVM-validated interactions that are computed on the basis of the training data alone. Training is performed on all interactions so that sensitivity is not sacrificed.

3.2 BLAST/PSI-BLAST based ranking

We compare our method with a simple ranking method that assigns a candidate interaction a score based on its similarity to the interacting pairs in the training set. Specifically, let $l(X, X')$ denote the negative log of the E-value assigned by PSI-BLAST (BLAST) when searching X against X' in the context of a large database of sequences, and let $I(i, j)$ be an indicator variable for the interaction between proteins i and j.

Table 1. ROC scores for the various methods computed using 5-fold cross-validation

Method	Kernel	ROC score	ROC$_{50}$ score
BLAST	—	0.74	0.18
PSIBLAST	—	0.78	0.11
Non-sequence	$K_{\text{non-seq}}$	0.95	0.37
Motif	$K_{\text{p}}(K_{\text{motif}})$	0.76	0.17
Pfam	$K_{\text{p}}(K_{\text{Pfam}})$	0.78	0.20
Spectrum ($k = 3$)	$K_{\text{p}}(K_{\text{spec}})$	0.81	0.05
Motif + Pfam	$K_{\text{p}}(K_{\text{motif}} + K_{\text{Pfam}})$	0.82	0.22
Motif + Pfam + spectrum	$K_{\text{p}}(K_{\text{motif}} + K_{\text{Pfam}} + K_{\text{spec}})$	0.86	0.17
All kernels	$K_{\text{feat}} + K_{\text{p}}(K_{\text{motif}} + K_{\text{Pfam}} + K_{\text{spec}})$	0.97	0.44
All + reliability	$K_{\text{feat}} + K_{\text{p}}(K_{\text{motif}} + K_{\text{Pfam}} + K_{\text{spec}})$	**0.97**	**0.58**

Training data include all BIND physical interactions. ROC scores are computed on reliable interactions that do not include PVM-validated interactions. The BLAST and PSIBLAST methods rank interactions according to Equation (4). The 'kernel' column of the table shows which kernel was used in conjunction with the SVM classifier. The notation $K_{\text{p}}(K_{\text{g}})$ denotes that the pairwise kernel was derived from a genomic kernel K_{g}. The $K_{\text{non-seq}}$ is a Gaussian kernel over the non-sequence features; in each method it participates in, the width of the Gaussian was determined by cross-validation as part of the classifier's training. The all-reliable method uses information on reliability to set the SVM soft-margin parameter as described in Section 2.5.

$l(X, X')$ is positive for significant matches and increases as the quality of the match increases. The score for a query (X_1, X_2) is defined as:

$$\max_{i \in \mathcal{P}, j \in \mathcal{P}} I(i, j) \min(l(X_1, X_i), l(X_2, X_j)), \qquad (4)$$

where $\mathcal{P}$ is the set of all proteins in the training set. In these experiments, we use PSI-BLAST scores computed in the context of the Swiss-Prot database (version 40, containing 101 602 proteins).

3.3 Figures of merit

Throughout this paper we evaluate the quality of a predictive method using two different metrics. Both metrics—the area under the receiver operating characteristic curve (ROC score), and the normalized area under that curve up to the first 50 false positives (ROC$_{50}$ score)—aim to measure both sensitivity and specificity by integrating over a curve that plots true positive rate as a function of false positive rate. We include both metrics in order to account for two different types of scenarios in which a protein–protein interaction prediction method might be employed.

In the first scenario, imagine that you have developed a low-throughput method for detecting whether a given pair of proteins interacts. Rather than testing your method on randomly selected pairs of proteins, you could use a predictive algorithm to identify likely candidates. In this case, you would start from the top of the ranked list of predictions, testing pairs until you ran out of time or money, or until the success rate of the predictor was too low to be useful. In this scenario, a predictor that maximizes the quality of the high-confidence interactions i.e. that maximizes the ROC$_{50}$ score, is going to be most useful.

In the second, more common scenario, you are interested in a particular biological system. You run the predictive algorithm, and you check your favorite set of proteins to see whether they participate in any predicted interactions. In this case, you do not care about the high-confidence interactions above; instead, you would like to be sure that the complete set of predictions is of high quality. In this case you are interested in the ROC score of the classifier.

4 RESULTS

We report, in this section, the results of experiments in predicting protein–protein interactions using an SVM classifier with various kernels, and compare these with a simple method based on BLAST or PSI-BLAST. All the experiments were performed using the PyML machine learning framework available at http://pyml.sourceforge.net. We begin this section with results obtained using the various kernels and kernel combinations, followed by a discussion of the choice of negative examples, and a section that shows the effects of choosing a non-redundant set of proteins.

4.1 Main results

We report results that are computed using 5-fold cross-validation on all BIND physical interactions. The SVM soft-margin parameter was not optimized—we used the default low value for this parameter to account for the noise in the data. The ROC/ROC$_{50}$ curve is then computed for those reliable interactions that were not obtained using the PVM method as discussed in Section 3.1. The ROC statistics that summarize these experiments are reported in Table 1 and the selected ROC curves are shown in Figure 1.

Our basic method uses a pairwise kernel based on one of several sequence kernels—the motif, Pfam and spectrum kernels. The performance of the motif and Pfam kernels is comparable, with a slight advantage for the Pfam kernel (the ROC scores are 0.76 and 0.78 and ROC$_{50}$ scores are 0.17 and 0.20). The spectrum kernel (using $k$mers of length 3) achieves a higher ROC score of 0.81, but its ROC$_{50}$ score is significantly lower

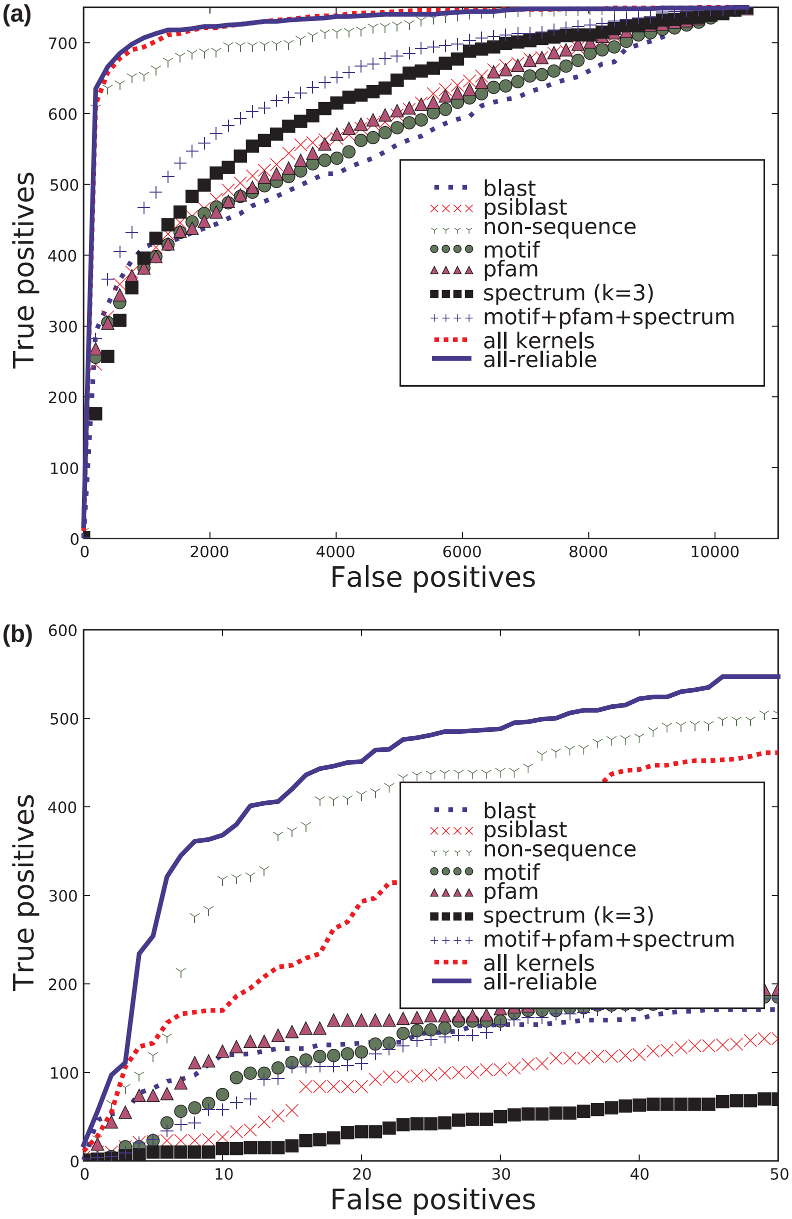

Fig. 1. ROC (**a**) and ROC$_{50}$ (**b**) curves for several methods. Best performance is obtained using a kernel that combines all the kernels presented in the paper. Additional results are summarized in Table 1, along with a description of the methods.

than that of the Pfam and motif kernels. The higher ROC score can be explained by the fact that the motif and the Pfam methods are limited in their sensitivity by the motifs and the domain models available. However, when such models offer a good description of a sequence, their predictions are likely to be more accurate, which is reflected in the much higher ROC$_{50}$ scores of these methods. Each of the pairwise kernels by itself is not doing much better than BLAST or PSI-BLAST, but once they are combined, they offer improved performance. We note that using a spectrum kernel with *k*mers of length 4 did not improve the performance of the method.

We now explore the effect of adding to the sequence kernels, a kernel based on three types of non-sequence data—GO annotations, the homology score and the MCC. For the non-sequence features, we first standardized the data (subtracted the mean of each feature and divided by the standard deviation), and used a Gaussian kernel whose width was determined by cross-validation.

Combining the non-sequence features with the pairwise sequence kernel yielded better performance than any method by itself in both performance metrics. Furthermore, setting the soft-margin parameter of the SVM according to the reliability of the interactions provided another significant boost to the performance. Its ROC and ROC_{50} scores were 0.98 and 0.58, respectively; at a false positive rate of 1% the classifier retrieves ~80% of the trusted interactions. In this experiment we did not try to optimize the ratio between the two soft margin constants, and used $C_{\mathrm{low}} = 0.01C_{\mathrm{high}}$.

The main contribution to the gain in performance comes from the GO-process kernel feature. Its ROC score by itself is 0.68 on all the BIND interactions and 0.95 when limiting to the reliable positive examples. The difference between the two numbers is probably due to the sizable fraction of false interactions in the BIND dataset. In the following subsection we point out scenarios where the GO data are not useful. The ROC score for the MCC feature was 0.68 on all BIND interactions and 0.53 when computed on the reliable interactions. The large difference for the MCC feature is a result of the fact that the MCC requires a large number of interactions to be useful. At a BLAST cutoff of $1\mathrm{e}^{-10}$, 329 interactions from BIND were supported by interactions from other species, as opposed to 49 negative examples. The ROC score for this feature by itself is low since it is sparse, i.e. is informative for a small number of interactions.

4.2 The role of GO annotations

In order to understand the difference in the role of the sequence kernels and the non-sequence kernels, we compared the two kernels on the task of distinguishing between physically interacting proteins pairs and those that are members of the same complex. In this case, the negative examples are chosen as protein pairs that are known to belong to the same complex but are not known to physically interact. This set of negative examples is likely to be more noisy than the non-interacting set, because complexes that are not accessible by yeast two-hybrid probably contain many physical interactions. But still, the motif-pairwise method achieves an ROC score of 0.78, very close to the value obtained with non-interacting negative examples. In this task, a classifier based on the non-sequence kernel fails with an ROC score of 0.5. This is due to the fact that cocomplexed proteins, such as physically interacting proteins, tend to have similar GO annotations and network properties, where as the motif and Pfam rely on a signal that is often directly related to the interaction site itself (Wang *et al*., 2005). Similar observations can be made for other features used to predict cocomplexed proteins, such as gene-expression data.

4.3 Choosing negative examples

Recall that examples of non-interacting proteins were chosen as random pairs of interacting proteins. To test the stability of our results with respect to the choice of negative examples, we ran a set of experiments using 10 different randomly selected sets of non-interacting proteins. Predictions were made using the motif kernel. The standard deviation of the resulting ROC scores was 0.003, showing good stability.

Significant attention has been paid to the problem of selecting gold standard interacting protein pairs for the purposes of training and validating predictive computational methods (Jansen *et al*., 2003). However, less emphasis has been placed on the choice of non-interacting protein pairs. In this study, we selected negatives uniformly at random. We find that this strategy leads to consistent behavior and avoids bias.

The possibility for bias due to the method of constructing negative examples is evidenced by results reported in a related paper (Martin *et al*., 2005). In this work, the authors report that a pairwise spectrum kernel provides highly accurate predictions of yeast interactions using a dataset studied in Jansen *et al*. (2003). The positive examples in this dataset satisfy our criteria of trusted interactions, and one might conclude that the use of highly reliable interactions is the reason for the success of the predictive method. However, we found that the method of choosing negative examples has a strong effect on the performance: the negative examples from Jansen *et al*. (2003) were chosen as pairs of proteins that are known to be localized in different cellular compartments. This makes these protein pairs much less likely to interact than randomly selected pairs, but the selection constraints impose a bias on the resulting distribution that makes the overall learning task easier [note that this is less likely to affect the results of non-sequence based methods, such as the one used by Jansen *et al*. (2003)]. To illustrate this effect, we created datasets with negative examples taken as pairs whose GO component similarity, as measured by our kernel, is below a given threshold. The performance of the resulting classifier varied as we varied this threshold (Table 2). This constrained selection method was tested with the spectrum and motif kernels using both the BIND interaction data and a set of trusted interactions similar to the one used by Martin *et al*. (2005) extracted from DIP and MIPS (Mewes *et al*., 2000; Xenarios *et al*., 2002). For the spectrum kernel, the ROC (ROC_{50}) scores varied from 0.87 (0.08) to 0.97 (0.46) on the DIP/MIPS data and from 0.77 (0.04) to 0.95 (0.36) on the BIND data, as the threshold was lowered from 0.5 to 0.04. Similarly, although slightly less pronounced, results were obtained for the motif pairwise kernel.

4.4 The dependence on interacting paralogs

The yeast genome contains a large number of duplicated genes. Since we are using a sequence-based method to predict interactions, we need to determine to what extent the performance depends on the presence of interacting paralogs. We therefore performed an experiment in which the training set and test set do not contain proteins whose BLAST E-values are more significant than a given threshold. In this case we performed 2-fold cross-validation instead of 5-fold cross-validation. For the pairwise motif–Pfam–spectrum kernel the ROC score decreased from 0.86 with no constraint to 0.81 when the training and test sets did not contain proteins whose BLAST E-values were better than 0.1. The ROC score

Table 2. The dependence of the performance of the spectrum pairwise method on the similarity between localization annotations in negative examples

Dataset	Threshold	ROC	ROC_{50}
BIND	0.50	0.77	0.04
	0.10	0.89	0.15
	0.07	0.91	0.21
	0.05	0.92	0.25
	0.04	0.95	0.36
DIP/MIPS	0.5	0.87	0.08
	0.1	0.94	0.22
	0.07	0.95	0.32
	0.05	0.96	0.34
	0.04	0.97	0.46

Enforcing the condition that no two proteins in the set of negative examples have a GO similarity that is less than a given threshold puts a constraint on the distribution of negative examples. This constraint makes it easy for the classifier to distinguish between positive and negative examples, and the effect gets stronger as the threshold becomes smaller. We performed the experiment on the BIND interaction dataset and on a dataset of reliable interactions derived from DIP and MIPS interactions.

for the PSI-BLAST (BLAST) method went down from 0.78 (0.74) to 0.62 (0.62). This illustrates that the kernel combination is less dependent on the presence of interacting paralogs than BLAST or PSI-BLAST.

5 DISCUSSION

In this paper we presented several kernels for prediction of protein–protein interactions and used them in combination for improved performance. The concern regarding the pairwise kernel is the high dimensionality of its feature space, which is quadratic in the number of features of the underlying kernel. We considered an alternative kernel which uses summation instead of the multiplication used in the expression for the pairwise kernel, similar to the work of Gomez *et al*. (2003). The performance of the summation kernel is not as good as the corresponding pairwise kernel, showing the advantage of using pairs of features.

When training a classifier to predict protein–protein interactions, there is a balance between placing in the training set only trusted interactions as opposed to trying to maximize the number of positive examples by adding interactions about which we are less sure. When using a sequence-based approach, as we have done here, the sensitivity of the method may depend on the richness of the training set. We have shown in this paper that we are able to use a larger set of noisy data while still achieving a good performance. As an alternative to training on a dataset that includes false positive interactions we plan to first apply a step of filtering the interaction data on the basis of features of trusted interactions, in order to maximize the number of interactions that can be considered reliable.

We also made no attempt to purge from our dataset examples that contain missing data (missing GO annotations). When trying to make predictions on unseen data, these data will contain missing data and so, the method is more likely to generalize if presented with examples containing missing data during training.

During the time of writing this paper we found that the pairwise approach was proposed by Martin *et al*. (2005). They used only the spectrum kernel, whereas here we considered several sequence kernels. We found that the spectrum kernel works better than the motif and Pfam kernels according to the ROC metric, but the spectrum kernel does not work as well as the motif and Pfam kernels according to the ROC_{50} metric. Apparently, the signal that the spectrum kernel generates is not as specific as that of the other kernels.

In addition, we have illustrated that pairwise sequence kernels can be successfully combined with non-sequence data. In this work, we have not attempted to learn the weights of the various kernels as done by Lanckriet *et al*. (2004). This is an avenue for future work, although solving the resulting semi-definite programming problem promises to be computationally expensive, owing to the large training sets involved. We also plan to consider additional sources of data such as gene expression and transcription factor binding data, which have also been shown to be informative in predicting protein–protein interactions (Zhang *et al*., 2004).

ACKNOWLEDGEMENTS

The authors thank Doug Brutlag, David Baker, Ora Schueler-Furman and Trisha Davis for the helpful discussions. This work is funded by NCRR NIH award P41 RR11823, by NHGRI NIH award R33 HG003070, and by NSF award BDI-0243257. W.S.N. is an Alfred P. Sloan Research Fellow.

REFERENCES

Bader,G.D. Donaldson,I., Wolting,C. Ouellette,B.F., Pawson,T. and Hogue,C.W. (2001) BIND—the biomolecular interaction network database. *Nucleic Acids Res.*, **29**, 242–245.

Ben-hur,A. and Brutlag,D. (2003) Remote homology detection: a motif based approach. *Bioinformatics*, **19** (Suppl 1), i26–i33.

Deane,C. Salwinski,L., Xenarios,I. and Eisenberg,D. (2002) Two methods for assessment of the reliability of high throughput observations. *Mol. Cell. Proteomics*, **1**, 349–356.

Deng,M., Mehta,S., Sun,F. and Chen,T. (2002) Inferring domain–domain interactions from protein–protein interactions. *Genome Res.*, **12**, 1540–1548.

Gene Ontology Consortium (2000) Gene ontology: tool for the unification of biology. *Natl Genet.*, **25**, 25–29.

Goldberg,D. and Roth,F. (2003) Assessing experimentally derived interactions in a small world. *Proc. Natl Acad. Sci. USA*, **100**, 4372–4376.

Gomez,S.M., Noble,W.S. and Rzhetsky,A. (2003) Learning to predict protein–protein interactions. *Bioinformatics*, **19**, 1875–1881.

Jansen,R., Yu,H., Greenbaum,D., Kluger,Y., Krogan,N.J., Chung,S., Emili,A., Snyder,M., Greenblatt,J.F. and Gerstein,M. (2003) A Bayesian networks approach for predicting protein–protein interactions from genomic data. *Science*, **302**, 449–453.

Lanckriet,G.R.G., Deng,M., Cristianini,N., Jordan,M.I. and Noble,W.S. (2004) Kernel-based data fusion and its application to protein function prediction in yeast. In Altman,R.B., Dunker,A.K., Hunter,L., Jung,T.A. and Klein,T.E. (eds), *Proceedings of the Pacific Symposium on Biocomputing*, World Scientific, Singapore, pp. 300–311.

Leslie,C., Eskin,E. and Noble,W.S. (2002) The spectrum kernel: A string kernel for SVM protein classification. In Altman,R.B., Dunker,A.K., Hunter,L., Lauderdale,K. and Klein,T.E. (eds), *Proceedings of the Pacific Symposium on Biocomputing*, New Jersey. World Scientific, Singapore, pp. 564–575.

Lin,N., Wu,B., Jansen,R., Gerstein,M. and Zhao,H. (2004) Information assessment on predicting protein–protein interactions. *BMC Bioinformatics*, **5**, 154.

Marcotte,E.M., Pellegrini,M., Ng,H.-L., Rice,D.W., Yeates,T.O. and Eisenberg,D. (1999) Detecting protein function and protein–protein interactions from genome sequences. *Science*, **285**, 751–753.

Martin,S., Roe,D. and Faulon J.-L. (2005) Predicting protein–protein interactions using signature products. *Bioinformatics*, **21**, 218–226.

Mewes,H.W., Frishman,D., Gruber,C., Geier,B., Haase,D., Kaps,A., Lemcke,K., Mannhaupt,G., Pfeiffer,F., Schüller,C., Stocker,S. and Weil,B. (2000) MIPS: a database for genomes and protein sequences. *Nucleic Acids Res.*, **28**, 37–40.

Nevill-Manning,C.G., Sethi,K.S., Wu,T.D. and Brutlag,D.L. (1997) Enumerating and ranking discrete motifs. In *Proceedings of the Fifth International Conference on Intelligent Systems for Molecular Biology*, pp. 202–209.

Noble,W.S. (2004) Support vector machine applications in computational biology. In Schoelkopf,B., Tsuda,K. and Vert,J.-P. (eds), *Kernel Methods in Computational Biology*. MIT Press, Cambridge, MA, pp. 71–92.

Pazos,F. and Valencia,A. (2002) *In silico* two-hybrid system for the selection of physically interacting protein pairs. *Proteins*, **47**, 219–227.

Ramani,A. and Marcotte,E. (2003) Exploiting the co-evolution of interacting proteins to discover interaction specificity. *J. Mol. Biol.*, **327**, 273–284.

Schölkopf,B. and Smola,A. (2002) *Learning with Kernels*. MIT Press, Cambridge, MA.

Sonnhammer,E., Eddy,S. and Durbin,R. (1997) Pfam: a comprehensive database of protein domain families based on seed alignments. *Proteins*, **28**, 405–420.

Sprinzak,E. and Margalit,H. (2001) Correlated sequence-signatures as markers of protein–protein interaction. *J. Mol. Biol.*, **311**, 681–692.

Sprinzak,E., Sattath,S. and Margalit,H. (2003) How reliable are experimental protein–protein interaction data? *J. Mol. Biol.*, **327**, 919–923.

von Mering,C., Krause,R., Snel,B., Cornell,M., Olivier,S.G., Fields,S. and Bork,P. (2002) Comparative assessment of large-scale data sets of protein–protein interactions. *Nature*, **417**, 399–403.

Wang,H., Segal,E., Ben-Hur,A., Koller,D. and Brutlag,D.L. (2005) Identifying protein–protein interaction sites on a genome-wide scale. In Lawrence K. Saul, Yair Weiss and Léon Bottou (eds), *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, pp. 1465–1472.

Xenarios,I., Salwinski,L., Duan,X.Q.J., Higney,P., Kim,S.M. and Eisenberg,D. (2002) DIP: the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions. *Nucleic Acids Res.*, **30**, 303–305.

Yu,H., Luscombe,N., Lu,H., Zhu,X., Xia,Y., Han,J., Bertin,N., Chung,S., Vidal,M. and Gerstein,M. (2004) Annotation transfer between genomes: protein–protein interlogs and protein–DNA regulogs. *Genome Res.*, **14**, 1107–1118.

Zhang,L., Wong,S., King,O. and Roth,F. (2004) Predicting co-complexed protein pairs using genomic and proteomic data integration. *BMC Bioinformatics*, **5**, 38–53.