Bioinformatics Advance Access published May 6, 2004

BIOINFORMATICS

A statistical framework for genomic data fusion

Gert R. G. Lanckriet[1], Tijl De Bie[2], Nello Cristianini[3], Michael I. Jordan[4] and William Stafford Noble[† 5]

[1]Department of Electrical Engineering and Computer Science, University of California, Berkeley, [2]Department of Electrical Engineering, ESAT-SCD, Katholieke Universiteit Leuven, Belgium, [3]Department of Statistics, University of California, Davis, [4]Division of Computer Science, Department of Statistics, University of California, Berkeley and [5]Department of Genome Sciences, University of Washington

ABSTRACT

Motivation: During the past decade, the new focus on genomics has highlighted a particular challenge: to integrate the different views of the genome that are provided by various types of experimental data.

Results: This paper describes a computational framework for integrating and drawing inferences from a collection of genome-wide measurements. Each data set is represented via a kernel function, which defines generalized similarity relationships between pairs of entities, such as genes or proteins. The kernel representation is both flexible and efficient, and can be applied to many different types of data. Furthermore, kernel functions derived from different types of data can be combined in a straightforward fashion. Recent advances in the theory of kernel methods have provided efficient algorithms to perform such combinations in a way that minimizes a statistical loss function. These methods exploit semidefinite programming techniques to reduce the problem of finding optimizing kernel combinations to a convex optimization problem. Computational experiments performed using yeast genome-wide data sets, including amino acid sequences, hydropathy profiles, gene expression data and known protein-protein interactions, demonstrate the utility of this approach. A statistical learning algorithm trained from all of these data to recognize particular classes of proteins—membrane proteins and ribosomal proteins—performs significantly better than the same algorithm trained on any single type of data.

Availability: Supplementary data at `http://noble.gs.washington.edu/proj/sdp-svm`.

Contact: noble@gs.washington.edu

INTRODUCTION

The recent availability of multiple types of genome-wide data provides biologists with complementary views of a single genome and highlights the need for algorithms capable of unifying these views. In yeast, for example, for a given gene we typically know the protein it encodes, that protein's similarity to other proteins, its hydrophobicity profile, the mRNA expression levels associated with the given gene under hundreds of experimental conditions, the occurrences of known or inferred transcription factor binding sites in the upstream region of that gene, and the identities of many of the proteins that interact with the given gene's protein product. Each of these distinct data types provides one view of the molecular machinery of the cell. In the near future, research in bioinformatics will focus more and more heavily on methods of data fusion.

Different data sources are likely to contain different and thus partly independent information about the task at hand. Combining those complementary pieces of information can be expected to enhance the total information about the problem at hand. One problem with this approach, however, is that genomic data come in a wide variety of data formats: expression data are expressed as vectors or time series; protein sequence data as strings from a 20-symbol alphabet; gene sequences are strings from a different (4-symbol) alphabet; protein-protein interactions are best expressed as graphs, and so on.

This paper presents a computational and statistical framework for integrating heterogeneous descriptions of the same set of genes. The approach relies on the use of kernel-based statistical learning methods that have already proven to be very useful tools in bioinformatics (Noble, 2004). These methods represent the data by means of a kernel function, which defines similarities between pairs of genes, proteins, etc. Such similarities can be quite complex relations, implicitly capturing aspects of the underlying biological machinery. One reason for the success of kernel methods is that the kernel function takes relationships that are implicit in the data and makes them explicit, so that it is easier to detect patterns. Each kernel function thus extracts a specific type of information from a given data set, thereby providing a partial description or view of the data. Our goal is to

[†] Corresponding author: Health Sciences Center, Box 357730, 1705 NE Pacific Street, Seattle, WA 98195, E-mail: noble@gs.washington.edu, Tel: (206) 543-8930, Fax: (206) 685-7301.

find a kernel that best represents all of the information available for a given statistical learning task. Given many partial descriptions of the data, we solve the mathematical problem of combining them using a convex optimization method known as semidefinite programming (SDP) (Nesterov and Nemirovsky, 1994; Vandenberghe and Boyd, 1996). This SDP-based approach (Lanckriet et al., 2004) yields a general methodology for combining many partial descriptions of data that is statistically sound, as well as computationally efficient and robust.

In order to demonstrate the feasibility of these methods, we apply them to the recognition of two important groups of proteins in yeast—ribosomal proteins and membrane proteins. The ribosome is a universal protein complex that is responsible for the translation of mRNA into the corresponding amino acid sequence via the universal genetic code. The structure of the ribosome has been solved (Schluenzen et al., 2000; Harms et al., 2001), although the precise roles of many auxiliary factors are not completely understood. Proteins that participate in the ribosome share similar sequence features and correlated mRNA expression patterns (Brown et al., 2000).

Membrane proteins are proteins that anchor in one of various membranes in the cell, including the plasma, ER, golgi, peroxisomal, vacuolar, cellular and mitochondrial inner and outer membranes. Many membrane proteins serve important communicative functions between cellular compartments and between the inside and the outside of the cell (Alberts et al., 1998). Classifying a protein as a membrane protein or not based on protein sequence is non-trivial and has been the subject of much previous research (Engleman et al., 1986; Krogh et al., 2001; Chen and Rost, 2002). This is a typical statistical learning problem in which a single type of feature derived from the protein sequence cannot provide the full story.

For both of these protein classes, we demonstrate that incorporating knowledge derived from the amino acid sequences, gene expression data and known protein-protein interactions significantly improves classification performance relative to our method trained on any single type of data. The SDP-based approach also performs better than a classifier trained using a naive, unweighted combination of kernels, and the method continues to perform well in the presence of artificially induced experimental noise.

We begin by outlining the main ideas of the kernel approach to pattern analysis, providing examples of kernels defined on yeast genome-wide data sets. We then describe how these kernels can be integrated using SDP to provide a unified description. Finally, we describe a series of computational experiments that demonstrate the validity and power of the kernel approach to data fusion for recognition of ribosomal and membrane proteins in yeast.

KERNEL METHODS

Kernel methods work by embedding data items (corresponding to genes, proteins, etc.) into a vector space $\mathcal{F}$, called a *feature space* (Cristianini and Shawe-Taylor, 2000; Schölkopf and Smola, 2002; Wahba, 1990; Vapnik, 1998, 1999). A key characteristic of kernel methods is that the embedding in feature space is generally defined implicitly, by specifying an inner product for the feature space. Thus, for a pair of data items, x_1 and x_2, denoting their embeddings as $\Phi(x_1)$ and $\Phi(x_2)$, respectively, we specify the inner product of the embedded data, $\langle \Phi(x_1), \Phi(x_2) \rangle$, via a *kernel function* $K(x_1, x_2)$. Any symmetric, positive semidefinite function is a valid kernel function, corresponding to an inner product in some feature space. Note that if all we require are inner products, then we do not need to have an explicit representation of the mapping Φ, nor do we even need to know the nature of the feature space. It suffices to be able to evaluate the kernel function.

Evaluating the kernel on all pairs of data points yields a symmetric, positive semidefinite matrix known as the *kernel matrix* or the *Gram matrix*. Intuitively, a kernel matrix can be regarded as a matrix of generalized similarity measures among the data points. The first stage of processing in a kernel method is to reduce the data by computing this matrix.

The reduction to a kernel matrix reflects the fact that kernel methods are generally based on linear statistical procedures in feature space. In particular, the classification algorithm that we use in this paper—known as a *support vector machine* (SVM, Boser et al., 1992)—forms a linear discriminant boundary in feature space. Consider a data set consisting of n pairs (x_i, y_i), where x_i is the ith data item (e.g., a protein sequence), and $y_i \in \{-1, 1\}$ is a label (e.g., membrane or non-membrane). Compute the $n \times n$ kernel matrix whose (i, j)th entry is $K(x_i, x_j)$. Given this matrix, and given the labels y_i, we can throw away the original data; the problem of fitting the SVM to data reduces to an optimization procedure that is based entirely on the kernel matrix and the labels.

Different kernel functions correspond to different embeddings of the data and thus can be viewed as capturing different notions of similarity. For example, in a space derived from amino acid sequences, two genes that are close to one another will have protein products with very similar amino acid sequences. This amino acid space would be quite different from a space derived from microarray gene expression measurements, in which closeness would indicate similarity of the expression profiles of the genes. In general, a single type of data can be mapped into many different feature spaces. The choice of feature space is made implicitly via the choice of kernel function.

For the tasks of ribosomal and membrane protein classification we experiment with seven kernel matrices

Table 1. Kernel functions. The table lists the seven kernels used to compare proteins, the data on which they are defined, and the method for computing similarities. The final kernel, K_{RND}, is included as a control. All kernel matrices, along with the data from which they were generated, are available at `noble.gs.washington.edu/proj/sdp-svm`.

Kernel	Data	Similarity measure
K_{SW}	protein sequences	Smith-Waterman
K_B	protein sequences	BLAST
K_{Pfam}	protein sequences	Pfam HMM
K_{FFT}	hydropathy profile	FFT
K_{LI}	protein interactions	linear kernel
K_D	protein interactions	diffusion kernel
K_E	gene expression	radial basis kernel
K_{RND}	random numbers	linear kernel

derived from three different types of data: four from the primary protein sequence, two from protein-protein interaction data, and one from mRNA expression data. These are summarized in Table 1.

Protein sequence: Smith-Waterman, BLAST and Pfam HMM kernels. A homolog of a membrane protein is likely also to be located in the membrane, and similarly for the ribosome. Therefore, we define three kernel matrices based upon standard homology detection methods. The first two sequence-based kernel matrices (K_{SW} and K_B) are generated using the BLAST (Altschul et al., 1990) and Smith-Waterman (SW) (Smith and Waterman, 1981) pairwise sequence comparison algorithms, as described previously (Liao and Noble, 2002). Both algorithms use gap opening and extension penalties of 11 and 1, and the BLOSUM 62 matrix. Because matrices of BLAST or Smith-Waterman scores are not necessarily positive semidefinite, we represent each protein as a vector of scores against all other proteins. Defining the similarity between proteins as the inner product between the score vectors (the so-called *empirical kernel map*, Tsuda, 1999) leads to valid kernel matrices, one for the BLAST score and one for the SW score. Note that including in the comparison set proteins with unknown labels allows the kernel to exploit this unlabeled data. The third kernel matrix (K_{Pfam}) is a generalization of the previous pairwise comparison-based matrices in which the pairwise comparison scores are replaced by expectation values derived from hidden Markov models in the Pfam database (Sonnhammer et al., 1997).

Protein sequence: FFT kernel. The fourth sequence-based kernel matrix (K_{FFT}) is specific to the membrane protein recognition task. This kernel directly incorporates information about hydrophobicity patterns, which are known to be useful in identifying membrane proteins. Generally, each membrane protein passes through the membrane several times. The transmembrane regions of the amino acid sequence are typically hydrophobic, whereas the non-membrane portions are hydrophilic. This specific hydrophobicity profile of the protein allows it to anchor itself in the cell membrane. Because the hydrophobicity profile of a membrane protein is critical to its function, this profile is better conserved in evolution than the specific amino acid sequence. Therefore, classical methods for determining whether a protein $\mathbf{p}_i$ (consisting of $|\mathbf{p}_i|$ amino acids) spans a membrane (Chen and Rost, 2002), depend upon its *hydropathy profile* $h(\mathbf{p}_i) \in \mathbb{R}^{|\mathbf{p}_i|}$: a vector containing the hydrophobicities of the amino acids along the protein (Engleman et al., 1986; Black and Mould, 1991; Hopp and Woods, 1981). The FFT kernel uses hydropathy profiles generated from the Kyte-Doolittle index (Kyte and Doolittle, 1982). This kernel compares the frequency content of the hydropathy profiles of the two proteins. First, the hydropathy profiles are pre-filtered with a low-pass filter to reduce noise:

$$h_f(\mathbf{p}_i) = f \otimes h(\mathbf{p}_i)$$

where $f = \frac{1}{4}(1\ 2\ 1)$ is the impulse response of the filter and $\otimes$ denotes convolution with that filter. After pre-filtering the hydropathy profiles (and if necessary appending zeros to make them equal in length—a commonly used technique not altering the frequency content), their frequency contents are computed with the Fast Fourier Transform (FFT) algorithm:

$$H_f(\mathbf{p}_i) = FFT(h_f(\mathbf{p}_i)).$$

The FFT kernel between proteins $\mathbf{p}_i$ and $\mathbf{p}_j$ is then obtained by applying a Gaussian kernel function to the frequency contents of their hydropathy profiles:

$$K_{FFT}(\mathbf{p}_i, \mathbf{p}_j) = \exp(-||H_f(\mathbf{p}_i) - H_f(\mathbf{p}_j)||^2/2\sigma)$$

with width $\sigma = 10$. This kernel detects periodicities in the hydropathy profile, a feature that is relevant to the identification of membrane proteins and complementary to the previous, homology-based kernels.

Protein interactions: linear and diffusion kernels. For the recognition of ribosomal proteins, protein-protein interactions are clearly informative, since all ribosomal proteins interact with other ribosomal proteins. For membrane protein recognition, we expect information about protein-protein interactions to be informative for two reasons. First, hydrophobic molecules or regions of molecules are probably more likely to interact with each other than with hydrophilic molecules or regions. Second, transmembrane proteins are often involved in signaling pathways, and therefore different membrane proteins are likely to interact with a similar class of molecules upstream and downstream in these pathways (e.g., hormones upstream or kinases downstream). The two protein interaction kernels are generated using medium- and

high-confidence interactions from a database of known interactions (von Mering et al., 2002). These interactions can be represented as an interaction matrix, in which rows and columns correspond to proteins, and binary entries indicate whether the two proteins interact.

The first interaction kernel matrix (K_{LI}) is comprised of linear interactions, i.e., inner products of rows and columns from the centered, binary interaction matrix. The more similar the interaction pattern (corresponding to a row or column from the interaction matrix) is for a pair of proteins, the larger the inner product will be.

An alternative way to represent the same interaction data is to consider the proteins as nodes in a large graph. In this graph, two proteins are linked when they interact and otherwise not. Kondor and Lafferty (2002) propose a general method for establishing similarities between the nodes of a graph, based on a random walk on the graph. This method efficiently accounts for all possible paths connecting two nodes, and for the lengths of those paths. Nodes that are connected by shorter paths or by many paths are considered more similar. The resulting *diffusion kernel* generates the second interaction kernel matrix (K_D).

An appealing characteristic of the diffusion kernel is its ability, like the empirical kernel map, to exploit unlabeled data. In order to compute the diffusion kernel, a graph is constructed using all known protein-protein interactions, including interactions involving proteins whose subcellular locations are unknown. Therefore, the diffusion process includes interactions involving unlabeled proteins, even though the kernel matrix only contains entries for labeled proteins. This allows two labeled proteins to be considered close to one another if they both interact with an unlabeled protein.

Gene expression: radial basis kernel. Finally, we also include a kernel constructed entirely from microarray gene expression measurements. A collection of 441 distinct experiments was downloaded from the Stanford Microarray Database (`genome-www.stanford.edu/microarray`). This data provides us with a 441-element expression vector characterizing each gene. A Gaussian kernel matrix (K_E) is computed from these vectors by applying a Gaussian kernel function with width $\sigma = 100$ to each pair of 441-element vectors, characterizing a pair of genes. Gene expression data is expected to be useful for recognizing ribosomal proteins, since their expression signatures are known to be highly correlated with one another. We do not expect that gene expression will be particularly useful for the membrane classification task. We do not need to eliminate the kernel *a priori*, however; as explained in the following section, our method is able to provide an *a posteriori* measure of how useful a data source is relative to the other sources of data.

KERNEL METHODS FOR DATA FUSION

Each of the kernel functions described above produces, for the yeast genome, a square matrix in which each entry encodes a particular notion of similarity of one yeast protein to another. Implicitly, each matrix also defines an embedding of the proteins in a feature space. Thus, the kernel representation casts heterogeneous data—variable-length amino acid strings, real-valued gene expression data, and a graph of protein-protein interactions—into the common format of kernel matrices.

The kernel formalism also allows these various matrices to be combined. Basic algebraic operations such as addition, multiplication and exponentiation preserve the key property of positive semidefiniteness, and thus allow a simple but powerful algebra of kernels (Berg et al., 1984). For example, given two kernel functions K_1 and K_2, inducing the embeddings $\Phi_1(x)$ and $\Phi_2(x)$, respectively, it is possible to define the kernel $K = K_1 + K_2$, inducing the embedding $\Phi(x) = [\Phi_1(x), \Phi_2(x)]$. Of even greater interest, we can consider parameterized combinations of kernels. In particular, given a set of kernels $\mathcal{K} = \{K_1, K_2, \ldots, K_m\}$, we can form the linear combination

$$K = \sum_{i=1}^{m} \mu_i K_i, \tag{1}$$

where the weights are constrained to be non-negative to assure positive semidefiniteness: $\mu_i \geq 0; i = 1, \ldots, m$. We consider this kind of kernel combination in this paper.

As we have discussed, fitting a kernel-based statistical classifier (such as the SVM) to data involves solving an optimization problem based on the kernel matrix and the labels. In particular, the SVM finds a linear discriminant in feature space that has maximal distance ("margin") between the members of the positive and negative classes. The algorithm for finding this optimal linear discriminant involves solving an optimization problem known as a *quadratic program*, a particular form of convex optimization problem for which efficient solutions are known (Nesterov and Nemirovsky, 1994).

The specific form of SVM that we use in this paper is the *1-norm soft margin support vector machine* (Boser et al., 1992; Schölkopf and Smola, 2002). An SVM forms a linear discriminant boundary in the feature space $\mathcal{F}$: $f(x) = \mathbf{w}^T \Phi(x) + b$, where $\mathbf{w} \in \mathcal{F}$ and $b \in \mathbb{R}$. Given a labeled sample $S_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, a 1-norm soft margin SVM optimizes with respect to $\mathbf{w}$ and b so as to maximize the distance ("margin") between the positive and negative class, allowing misclassifications (therefore

"soft margin"):

$$\min_{\mathbf{w},b,\xi} \ \mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i \qquad (2)$$
$$\text{subject to } \ y_i(\mathbf{w}^T\Phi(x_i)+b) \geq 1-\xi_i n$$
$$\xi_i \geq 0, \qquad i=1,\ldots,n$$

where C is a regularization parameter, trading off error against margin. By considering the dual problem corresponding to Equation (2), one can prove (Schölkopf and Smola, 2002) that the weight vector can be expressed as $\mathbf{w} = \sum_{i=1}^{n}\alpha_i\Phi(x_i)$, where the support values α_i are solutions of the following dual *quadratic program* (QP):

$$\max_{\alpha} \ 2\alpha^T\mathbf{e} - \alpha^T\text{diag}(\mathbf{y})K\text{diag}(\mathbf{y})\alpha \qquad (3)$$
$$\text{subject to } \ 0 \leq \alpha \leq C, \ \ \alpha^T\mathbf{y} = 0,$$

where $\mathbf{y} = (y_1 \ y_2 \ \cdots \ y_n)^T$ and $\text{diag}(\mathbf{y})$ is a diagonal matrix with entries given by the elements of $\mathbf{y}$. An unlabeled data item x_{new} can subsequently be classified by computing the linear function

$$f(x_{new}) = \mathbf{w}^T\Phi(x_{new}) + b = \sum_{i=1}^{n}\alpha_i K(x_i, x_{new}) + b.$$

If $f(x_{new})$ is positive, then we classify x_{new} as belonging to class $+1$; otherwise, we classify x_{new} as belonging to class -1.

In Lanckriet et al. (2004), we show that for a fixed trace of K, the classification performance is bounded by a function of the optimum achieved in Equation (3): the smaller, the better the guaranteed performance. Thus, whereas in the standard SVM formulation K is a given kernel matrix, we can in fact *learn* an optimal kernel matrix by parameterizing K and minimizing Equation (3) with respect to these kernel parameters. More concretely, we consider the parameterization in Equation (1) with additional trace and positive semidefiniteness constraints. Plugging this into Equation (3) and minimizing with respect to μ_i gives:

$$\min_{\mu_i}\max_{\alpha} \ 2\alpha^T\mathbf{e} - \alpha^T\text{diag}(\mathbf{y})\left(\sum_{i=1}^{m}\mu_i K_i\right)\text{diag}(\mathbf{y})\alpha$$
$$\text{subject to } \ 0 \leq \alpha \leq C, \ \ \alpha^T\mathbf{y} = 0,$$
$$\text{trace}\left(\sum_{i=1}^{m}\mu_i K_i\right) = c,$$
$$\sum_{i=1}^{m}\mu_i K_i \succeq 0,$$

where c is a constant. Again considering the Lagrangian dual problem, we can show that this problem of finding optimal μ_i and α_i reduces to a convex optimization problem known as a *semidefinite program (SDP)*:

$$\min_{\mu_i,t,\lambda,\gamma\geq 0} \ t \qquad (4)$$
$$\text{subject to } \ \text{trace}\left(\sum_{i=1}^{m}\mu_i K_i\right) = c,$$
$$\sum_{i=1}^{m}\mu_i K_i \succeq 0,$$
$$\begin{pmatrix} Y(\mu) & \mathbf{e}+\gamma+\lambda\mathbf{y} \\ (\mathbf{e}+\gamma+\lambda\mathbf{y})^T & t-2C\delta^T\mathbf{e} \end{pmatrix} \succeq 0,$$

where we let $Y(\mu) = \text{diag}(\mathbf{y})(\sum_{i=1}^{m}\mu_i K_i)\text{diag}(\mathbf{y})$. SDP can be viewed as a generalization of linear programming, where scalar linear inequality constraints are replaced by more general linear matrix inequalities (LMIs): $F(\mathbf{x}) \succeq 0$, meaning that the matrix F has to be in the cone of positive semidefinite matrices, as a function of the decision variables $\mathbf{x}$. Note that the first LMI constraint in Equation (4), $K = \sum_{i=1}^{m}\mu_i K_i \succeq 0$, emerges very naturally because the optimal kernel matrix must indeed come from the cone of positive semidefinite matrices. Linear programs and semidefinite programs are both instances of convex optimization problems, and both can be solved via efficient interior-point algorithms (Vandenberghe and Boyd, 1996).

In this paper, the weights μ_i are constrained to be non-negative and the K_i are positive semidefinite and normalized ($[K_i]_{jj} = 1$) by construction; thus $K \succeq 0$ is automatically satisfied. In that case, we can show that the SDP in Equation (4) reduces to a *quadratically constrained quadratic program (QCQP)*, which is a special case of SDP that can be solved more efficiently:

$$\max_{\alpha,t} \ 2\alpha^T\mathbf{e} - ct \qquad (5)$$
$$\text{subject to } \ t \geq \frac{1}{n}\alpha^T\text{diag}(\mathbf{y})K_i\text{diag}(\mathbf{y})\alpha,$$
$$\alpha^T\mathbf{y} = 0,$$
$$0 \leq \alpha \leq C,$$

for $i = 1,\ldots,m$. Thus, by solving a QCQP, we are able to find an adaptive combination of kernel matrices—and thus an adaptive combination of heterogeneous information sources—that solves our classification problem. The output of our procedure is a set of weights μ_i and a discriminant function based on these weights. We obtain a classification decision that merges information encoded in the various kernel matrices, and we obtain weights μ_i that reflect the relative importance of these information sources.

EXPERIMENTAL DESIGN

In order to test our kernel-based approach in the setting of yeast protein classification, we use as a gold standard the annotations provided by the MIPS Comprehensive Yeast Genome Database (CYGD) (Mewes et al., 2000). The CYGD assigns 1125 yeast proteins to particular complexes, of which 138 participate in the ribosome. The remaining approximately 5000 yeast proteins are unlabeled. Similarly, CYGD assigns subcellular locations to 2318 yeast proteins, of which 497 belong to various membrane protein classes, leaving approximately 4000 yeast proteins with uncertain location.

The primary input to the classification algorithm is a collection of kernel matrices from Table 1. For membrane protein classification, for comparison with the SDP/SVM learning algorithm, we consider several classical biological methods that are commonly used to determine whether a Kyte-Doolittle plot corresponds to a membrane protein, as well as a state-of-the-art technique using hidden Markov models (HMMs) to predict transmembrane helices in proteins (Krogh et al., 2001). The first method relies on the observation that the average hydrophobicity of membrane proteins tends to be higher than that of non-membrane proteins, because the transmembrane regions are more hydrophobic. We therefore define f_1 as the average hydrophobicity, normalized by the length of the protein. We will compare the classification performance of our statistical learning algorithm with this metric.

Clearly, however, f_1 is too simplistic. For example, protein regions that are not transmembrane only induce noise in f_1. Therefore, an alternative metric filters the hydrophobicity plot with a low-pass filter and then computes the number, the height and the width of those peaks above a certain threshold (Chen and Rost, 2002). The filter is intended to smooth out periodic effects. We implement two such filters, choosing values for the filter order and the threshold based on Chen and Rost (2002). In particular, we define f_2 as the area under the 7th-order low-pass filtered Kyte-Doolittle plot and above a threshold value 2, normalized by the length of the protein. Similarly, f_3 is the corresponding area using a 20th-order filter and a threshold of 1.6.

Finally, the Transmembrane HMM (TMHMM) web server (`www.cbs.dtu.dk/services/TMHMM`) is used to make predictions for each protein. In Krogh et al. (2001), transmembrane proteins are identified by TMHMM using three different metrics: the expected number of amino acids in transmembrane helices, the number of transmembrane helices predicted by the N-best algorithm, and the expected number of transmembrane helices. Only the first two of these metrics are provided in the TMHMM output. Accordingly, we produce two lists of proteins, ranked by the number of predicted transmembrane helices (T_{PH}) and by the expected number of residues in transmembrane helices (T_{ENR}).

Each algorithm's performance is measured by randomly splitting the data (without stratifying) into a training and test set in a ratio of 80/20. We report the receiver operating characteristic (ROC) score, which is the area under a curve that plots true positive rate as a function of false positive rate for differing classification thresholds (Hanley and McNeil, 1982; Gribskov and Robinson, 1996). The ROC score measures the overall quality of the ranking induced by the classifier, rather than the quality of a single point in that ranking. An ROC score of 0.5 corresponds to random guessing, and an ROC score of 1.0 implies that the algorithm succeeded in putting all of the positive examples before all of the negatives. In addition, we select the point on the ROC curve that yields a 1% false positive rate, and we report the rate of true positives at this point (TP1FP). Each experiment is repeated 30 times with different random splits in order to estimate the variance of the performance values.

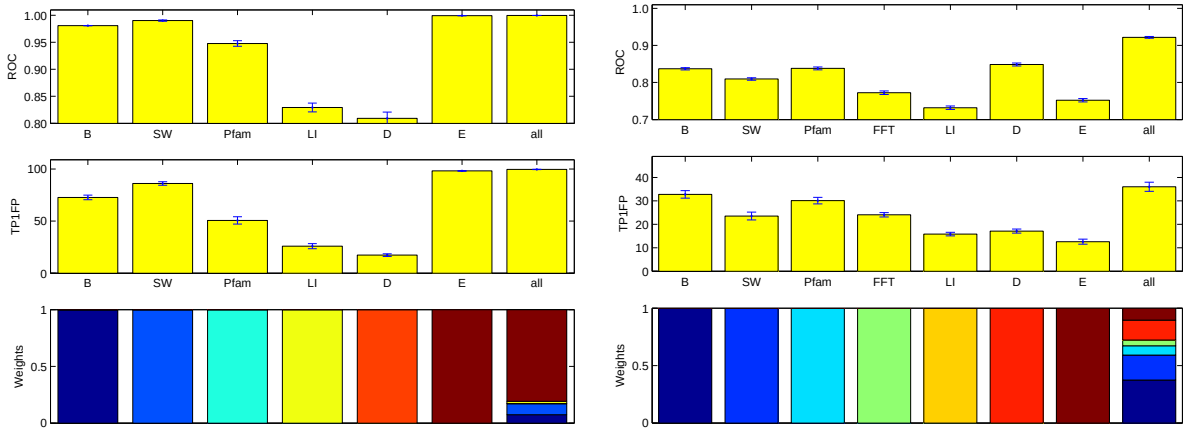
RESULTS

We performed computational experiments that study the performance of the SDP/SVM approach as a function of the number of data sources, compare the approach to a simpler approach using an unweighted combination of kernels, study the robustness of the method to the presence of noise, and for membrane protein classification, compare the performance of the method to classical biological methods and state-of-the-art techniques for membrane protein classification.

Ribosomal Protein Classification

Figure 1(A) shows the results of training an SVM to recognize the cytoplasmic ribosomal proteins, using various kernel functions. Very good recognition performance can be achieved using several types of data individually: the Smith-Waterman kernel yields an ROC of 0.9903 and a TP1FP of 86.23%, and the gene expression kernel yields corresponding values of 0.9995 and 98.31%. However, combining all six kernels using SDP provides still better performance (ROC of 0.9998 and TP1FP of 99.71%). These differences, though small, are statistically significant according to a Bonferroni corrected Wilcoxon signed rank test.

For this task, the SDP approach performs no better than the naive approach of combining all six kernel matrices in an unweighted fashion. Note, however, that the SDP solution also provides an additional explanatory result, in the form of the weights assigned to the kernels. These weights are illustrated in Figure 1(A) and suggest that, as expected, the cytoplasmic ribosomal proteins are best defined by their expression profiles and, secondarily, by



(A) Ribosomal proteins

(B) Membrane proteins

Fig. 1. Combining data sets yields better classification performance. The height of the bars in the upper two plots are proportional to the ROC score (top) and the percentage of true positives at one percent false positives (middle), for the SDP/SVM method using the given kernel. Error bars indicate standard error across 30 random train/test splits. In the lower plots, the heights of the colored bars indicate the relative weights of the different kernel matrices in the optimal linear combination. These results in tabular form, along with percent accuracy measurements, are given in the online supplement.

their sequences. An additional benefit offered by SDP over the naive approach is its robustness in the presence of noise. In order to illustrate this effect, we omit the expression kernel from the combination and add six kernels generated from Gaussian noise ($K_{R1\ldots R6}$). This set of kernels degrades the performance of the naive combination, but has no effect on the SDP/SVM. With six additional random kernels ($K_{R7\ldots R12}$) the benefit of optimizing the weights is even more apparent (see Table 2 and the online supplement).

Among the 30 train/test splits, seven proteins are consistently mislabeled by SDP/SVM (see online supplement). These include one, YLR406C (RPL31B), that was previously misclassified as non-ribosomal in an SVM-based study using a smaller microarray expression data set (Brown et al., 2000). In order to better understand the seven false negatives, we separated out the kernel-specific components of the SVM discriminant score. In nearly every case, the component corresponding to the gene expression kernel is the only one that is negative (data not shown). In other words, these seven proteins show atypical expression profiles, relative to the rest of the ribosome, which explains their misclassification by the SVM. Visual inspection of the expression matrix (online supplement) verifies these differences.

Finally, the trained SVM was applied to the set of approximately 5000 proteins that are not annotated in CYGD as participating in any protein complex. Among these, the SVM predicts that 14 belong in the cytoplasmic ribosomal class (see online supplement). However, nine of these predictions correspond to questionable ORFs, each of which lies directly opposite a gene that encodes a ribosomal protein. In these cases, the microarray expression data for the questionable ORFs undoubtedly reflect the strong pattern of expression from the corresponding ribosomal genes. Among the remaining five proteins, two (YNL119W and YKL056C) were predicted to be ribosomal proteins in a previous SVM-based study (Brown et al., 2000). YKL056C is particularly interesting: it is a highly conserved, ubiqitous protein homologous to the mammalian translationally controlled tumor protein (Gross et al., 1989) and to human IgE-dependent histamine-releasing factor.

Membrane Protein Classification

The results of the first membrane protein classification experiment are summarized in Figure 1(B). The plot illustrates that SDP/SVM learns significantly better from the heterogeneous data than from any single data type. The mean ROC score using all seven kernel matrices (0.9219 ± 0.0024) is significantly higher than the best ROC score using only one matrix (0.8487 ± 0.0039 using the diffusion kernel). This improvement corresponds to a change in TP1FP of 18.91%, from 17.15% to 36.06% and a change in test set accuracy of 7.36%, from 81.30% to 88.66%.

As expected, the sequence-based kernels yield good individual performance. The value of these kernels is evidenced by their corresponding ROC scores and by the relatively large weights assigned to the sequence-based

Table 2. Classification performance on the cytoplasmic ribosomal class, in the presence of noise or improper weighting. The table lists the percentage of true positives at one percent false positives (TP1FP) and the ROC score for several combinations of kernels. The first three lines of results were obtained using SDP-SVM, and the last three lines by setting the weights uniformly. Columns 1 through 5 report the average weights for the potentially informative kernels (averaged over the training/test splits), column 6 contains the average weight for a first set of 6 random kernels (averaged over the 6 kernels and the training/test splits) and column 7 similarly for an additional set of 6 random kernels. Each random kernel was generated by computing inner products on randomly generated 400-element vectors, in which each vector component was sampled independently from a standard normal distribution. In the table, a hyphen indicates that the corresponding kernel is not considered in the combination.

K_{SW}	K_{PF}	K_{LI}	K_B	K_D	$K_{R1\ldots R6}$	$K_{R7\ldots R12}$	TP1FP	ROC
5.08	0.31	0.22	0.39	0.00	–	–	$88.21 \pm 1.73\%$	0.9933 ± 0.0011
5.07	0.31	0.22	0.39	0.00	0.01	–	$88.19 \pm 1.60\%$	0.9932 ± 0.0011
5.06	0.30	0.22	0.38	0.01	0.02	0.01	$88.08 \pm 1.65\%$	0.9932 ± 0.0010
1.00	1.00	1.00	1.00	1.00	–	–	$75.20 \pm 2.38\%$	0.9906 ± 0.0012
1.00	1.00	1.00	1.00	1.00	1.00	–	$59.66 \pm 3.03\%$	0.9791 ± 0.0017
1.00	1.00	1.00	1.00	1.00	1.00	1.00	$42.87 \pm 2.59\%$	0.9620 ± 0.0027

kernels by the SDP. These weights are as follows: $\mu_B = 2.62$, $\mu_{SW} = 1.52$, $\mu_{Pfam} = 0.57$, $\mu_{FFT} = 0.35$, $\mu_{LI} = 0.01$, $\mu_D = 1.21$ and $\mu_E = 0.73$.[‡] Thus, two of the three kernel matrices that receive weights larger than 1 are derived from the amino acid sequence.

The results also show that the interaction-based diffusion kernel is more informative than the expression kernel. The diffusion kernel yields an individual ROC score which is significantly higher than the expression kernel, and the SDP also assigns a larger weight to the diffusion kernel (1.21) than to the expression kernel (0.73). Accordingly, removing the diffusion kernel reduces the percentage true positives at one percent false positives from 36.06% to 34.52%, whereas removing the expression kernel has a smaller effect, leading to a TP1FP of 35.88%. Further description of the results obtained when various subsets of kernels are used is provided in the online supplement.

In order to test the robustness of our approach, we performed a second experiment using four real kernels—K_B, K_{SW}, K_D, and K_E—and four Gaussian noise kernels $K_{R1\ldots R4}$. Using all eight kernels, SDP assigns values to the random kernels weights that are close to zero. Therefore, the overall performance, as measured by TP1FP or ROC score, remains virtually unchanged. In contrast, the performance of the uniformly weighted kernel combination, which was previously competitive with the SDP combination, degrades significantly in the presence of noise, from TP1FP of 33.87% down to 26.24%. Thus, the SDP approach provides a kind of insurance against the inclusion of noisy or irrelevant kernels.

We also compared the membrane protein classification performance of the SDP/SVM method with that of several other techniques for membrane protein classification. The ROC and TP1FP for these methods are listed in Table 4. The results indicate that using learning in this context dramatically improves the results relative to the simple hydropathy profile approach. The SDP/SVM method also improves, though to a lesser degree, upon the performance of the state-of-the-art TMHMM model. However, the comparison to TMHMM is somewhat problematic, for several reasons. First, TMHMM is provided as a pre-trained model. As such, a cross-validated comparison with the SDP/SVM is not possible. In particular, some members of the cross-validation test sets were almost certainly used in training TMHMM, making its performance estimate too optimistic. On the other hand, TMHMM aims to predict membrane protein topology across many different genomes, rather than in a yeast-specific fashion. Despite these difficulties, the results in Table 4 are interesting because they suggest that an approach that exploits multiple genome-wide data sets may provide better membrane protein recognition performance than a sequence-specific approach.

[‡] For ease of interpretation, we scale the weights such that their sum is equal to the number m of kernel matrices.

Table 4. Comparison of membrane protein recognition methods. Each row in the table corresponds to one of the membrane protein recognition methods described in the text: three methods that apply filters directly to the hydrophobicity profile, two methods based upon the TMHMM model, and the SDP/SVM approach. For each method, the ROC and TP1FP are reported.

Method	ROC	TP1FP
f_1	0.7345	16.70%
f_2	0.7504	13.48%
f_3	0.7879	21.93%
T_{PH}	0.7362	30.02%
T_{ENR}	0.8018	31.38%
SDP/SVM	0.9219	36.06%

DISCUSSION

We have described a general method for combining heterogeneous genome-wide data sets in the setting of kernel-based statistical learning algorithms, and we have demonstrated an application of this method to the problems of classifying yeast ribosomal and membrane proteins. The performance of the resulting SDP/SVM

Table 3. Classification performance on the membrane proteins, in the presence of noise or improper weighting. The table lists the percentage true positives at one percent false positives (TP1FP) and the ROC score for several combinations of kernels. The first two lines of results were obtained using SDP-SVM, and the last two lines were obtained using a uniform kernel weighting. Columns 1 through 8 report the average weights for the respective kernels (averaged over the training/test splits). A hyphen indicates that the corresponding kernel is not considered in the combination.

K_B	K_{SW}	K_D	K_E	K_{R1}	K_{R2}	K_{R3}	K_{R4}	TP1FP	ROC
1.81	1.05	0.73	0.42	–	–	–	–	$35.71 \pm 2.13\%$	0.9196 ± 0.0023
3.30	1.98	1.31	0.79	0.08	0.17	0.21	0.17	$34.14 \pm 2.09\%$	0.9145 ± 0.0026
1.00	1.00	1.00	1.00	–	–	–	–	$33.87 \pm 2.20\%$	0.9180 ± 0.0026
1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	$26.24 \pm 1.39\%$	0.8627 ± 0.0033

algorithm improves upon the SVM trained on any single data set or trained using a naive combination of kernels. Moreover, the SDP/SVM algorithm's performance consistently improves as additional genome-wide data sets are added to the kernel representation and is robust in the presence of noise.

? have presented a kernel-based approach to data fusion that is complementary to that presented here. In their approach, canonical correlation analysis (CCA) is used to select features from the space defined by a second kernel, and can be generalized to operate with more than two kernels. Thus, whereas the SDP approach combines different sources into a joint representation, kernel CCA separates components of a single kernel matrix, identifying the most relevant ones.

Semidefinite programming is viewed as a tractable instance of general convex programming, because it is known to be solvable in polynomial time, whereas general convex programs need not be (Nesterov and Nemirovsky, 1994). In practice, however, there are important computational issues that must be faced in any implementation. In particular, our application requires the formation and manipulation of $n \times n$ kernel matrices. For genome-scale data, such matrices are large, and naive implementation can create serious demands on memory resources. However, kernel matrices often have special properties that can be exploited by more sophisticated implementations. In particular, it is possible to prove that certain kernels necessarily lead to low-rank kernel matrices, and indeed low-rank matrices are also often encountered in practice (Williams and Seeger, 2000). Methods such as incomplete Cholesky decomposition can be used to find low-rank approximations of such matrices, without even forming the full kernel matrix, and these methods have been used successfully in implementations of other kernel methods (Bach and Jordan, 2002; Fine and Scheinberg, 2001). Time complexity is another concern. The worst-case complexity of the SDP in (4) is $O(n^{4.5})$ (Lanckriet et al., 2004), although it can be solved in $O(n^3)$, as a QCQP, under reasonable assumptions. In practice, however, this complexity bound is not necessarily reached by any given class of problem, and indeed time complexity has been less of a concern than space complexity in our work thus far. Moreover, the low-rank approximation tools may also provide some help with regards to time complexity. Nonetheless, running time issues are a concern for deployment of our approach with higher eukaryotic genomes, and new implementational strategies may be needed.

Kernel-based statistical learning methods have a number of general virtues as tools for biological data analysis. First, the kernel framework accommodates not only the vectorial and matrix data that are familiar in classical statistical analysis, but also more exotic data types such as strings, trees, graphs and text. The ability to handle such data is clearly essential in the biological domain. Second, kernels provide significant opportunities for the incorporation of more specific biological knowledge, as we have seen with the FFT kernel and the Pfam kernel. Third, the growing suite of kernel-based data analysis algorithms require only that data be reduced to a kernel matrix; this creates opportunities for standardization. Finally, as we have shown here, the reduction of heterogeneous data types to the common format of kernel matrices allows the development of general tools for combining multiple data types. Kernel matrices are required only to respect the constraint of positive semidefiniteness, and thus the powerful technique of semidefinite programming can be exploited to derive general procedures for combining data of heterogeneous format and origin.

We thus envision the development of general libraries of kernel matrices for biological data, such as those that we have provided at `noble.gs.washington.edu/proj/sdp-svm`, that summarize the statistically-relevant features of primary data, encapsulate biological knowledge, and serve as inputs to a wide variety of subsequent data analyses. Indeed, given the appropriate kernel matrices, the methods that we have described here are applicable to problems such as the prediction of protein metabolic, regulatory and other functional classes, the prediction of protein subcellular locations, and the prediction of protein-protein interactions.

Finally, while we have focused on the binary classification problem in the current paper, there are many possible extensions of our work to other statistical learning problems. One notable example is the problem of *trans-*

duction, in which the classifier is told *a priori* the identity of the points that are in the test set (but not their labels). This approach can deliver superior predictive performance (Vapnik, 1998), and would seem particularly appropriate in gene or protein classification problems, where the entities to be classified are often known *a priori*.

Acknowledgments

WSN is supported by a Sloan Foundation Research Fellowship and by National Science Foundation grants DBI-0078523 and ISI-0093302. MIJ and GL acknowledge support from ONR MURI N00014-00-1-0637 and NSF grant IIS-9988642. TDB is a Research Assistant with the Fund for Scientific Research, Flanders (F.W.O.–Vlaanderen).

REFERENCES

Alberts, B., D. Bray, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter (1998). *Essential Cell Biology: An Introduction to the Molecular Biology of the Cell*. Garland Science Publishing.

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

Bach, F. and M. I. Jordan (2002). Kernel independent component analysis. *Journal of Machine Learning Research 3*, 1–48.

Berg, C., J. Christensen, and P. Ressel (1984). *Harmonic Analysis on Semigroups: Theory of Positive Definite and Related Functions*. New York: Springer.

Black, S. D. and D. R. Mould (1991). Development of hydrophobicity parameters to analyze proteins which bear post- or cotranslational modifications. *Anal. Biochem. 193*, 72–82.

Boser, B. E., I. Guyon, and V. Vapnik (1992). A training algorithm for optimal margin classifiers. In *Computational Learing Theory*, pp. 144–152.

Brown, M. P. S., W. N. Grundy, D. Lin, N. Cristianini, C. Sugnet, T. S. Furey, J. M. Ares, and D. Haussler (2000). Knowledge-based analysis of microarray gene expression data using support vector machines. *Proceedings of the National Academy of Sciences of the United States of America 97*(1), 262–267.

Chen, C. and B. Rost (2002). State-of-the-art in membrane protein prediction. *Applied Bioinformatics 1*(1), 21–35.

Cristianini, N. and J. Shawe-Taylor (2000). *An Introduction to Support Vector Machines*. Cambridge, U.K.: Cambridge University Press.

Engleman, D. M., T. A. Steitz, and A. Goldman (1986). Identifying nonpolar transbilayer helices in amino acid sequences of membrane proteins. *Ann. Rev. Biophys. Biophys. Chem. 15*, 321–353.

Fine, S. and K. Scheinberg (2001). Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research 2*, 243–264.

Gribskov, M. and N. L. Robinson (1996). Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. *Computers and Chemistry 20*(1), 25–33.

Gross, G., M. Gaestel, H. Bohm, and H. Bielka (1989). cDNA sequence coding for a translationally controlled human tumor protein. *NAR 17*(20), 8367.

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

Harms, J., F. Schluenzen, R. Zarivach, A. Bashan, S. Gat, I. Agmon, H. Bartels, F. Franceschi, and A. Yonath (2001). High resolution structure of the large ribosomal subunit from a meshophilic eubacterium. *Cell 107*, 679–688.

Hopp, T. P. and K. R. Woods (1981). Prediction of protein antigenic determinants from amino acid sequences. *Proc. Natl. Acad. Sci. USA 78*, 3824–3828.

Kondor, R. I. and J. Lafferty (2002). Diffusion kernels on graphs and other discrete input spaces. In C. Sammut and A. Hoffmann (Eds.), *Proceedings of the International Conference on Machine Learning*. Morgan Kaufmann.

Krogh, A., B. Larsson, G. von Heijne, and E. L. L. Sonnhammer (2001). Predicting transmembrane protein topology with a hidden Markov model: Application to complete genomes. *Journal of Molecular Biology 305*(3), 567–580.

Kyte, J. and R. F. Doolittle (1982). A simple method for displaying the hydropathic character of a protein. *Journal of Molecular Biology 157*, 105–132.

Lanckriet, G. R. G., N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan (2004). Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research 5*, 27–72.

Liao, L. and W. S. Noble (2002). Combining pairwise sequence similarity and support vector machines for remote protein homology detection. In *Proceedings of the Sixth Annual International Conference on Computational Molecular Biology*, pp. 225–232.

Mewes, H. W., D. Frishman, C. Gruber, B. Geier, D. Haase, A. Kaps, K. Lemcke, G. Mannhaupt, F. Pfeiffer, C. Sch¨uller, S. Stocker, and B. Weil (2000). MIPS: a database for genomes and protein sequences. *Nucleic Acids Research 28*(1), 37–40.

Nesterov, Y. and A. Nemirovsky (1994). *Interior Point Polynomial Methods in Convex Programming: Theory and Applications*. Philadelphia, PA: SIAM.

Noble, W. S. (2004). *Kernel methods in computational biology*, Chapter Support vector machine applications in computational biology. MIT Press. In press.

Schluenzen, F., A. Tocilj, R. Zarivach, J. Harms, M. Gluehmann, D. Janell, A. Bashan, H. Bartels, I. Agmon, F. Franceschi, and A. Yonath (2000). Structure of functionally activated small ribosomal subunit at 3.3 å resolution. *Cell 102*, 615–623.

Sch¨olkopf, B. and A. Smola (2002). *Learning with Kernels*. Cambridge, MA: MIT Press.

Smith, T. F. and M. S. Waterman (1981). Identification of common molecular subsequences. *Journal of Molecular Biology 147*(1), 195–197.

Sonnhammer, E., S. Eddy, and R. Durbin (1997). Pfam: a comprehensive database of protein domain families based on seed alignments. *Proteins 28*(3), 405–420.

Tsuda, K. (1999). Support vector classification with asymmetric kernel function. In M. Verleysen (Ed.), *Proceedings ESANN*, pp. 183–188.

Vandenberghe, L. and S. Boyd (1996). Semidefinite programming. *SIAM Review 38*(1), 49–95.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley-Interscience.

Vapnik, V. N. (1999). *The Nature of Statistical Learning Theory* (2nd ed.). Springer.

von Mering, C., R. Krause, B. Snel, M. Cornell, S. G. Olivier, S. Fields, and P. Bork (2002). Comparative assessment of large-

scale data sets of protein-protein interactions. *Nature 417*, 399–403.

Wahba, G. (1990). *Spline Models for Observational Data*. SIAM.

Williams, C. K. I. and M. Seeger (2000). Effect of the input density distribution on kernel-based classifiers. In P. Langley (Ed.), *Proceedings of Seventeenth International Conference on Machine Learning (ICML 2000)*. San Francisco, CA: Morgan Kaufmann.