Classification of genes using probabilistic models of microarray expression profiles

Paul Pavlidis
Columbia Genome Center
Columbia University
New York NY 10032
pp175@columbia.edu

Christopher Tang
Integrated Program in Cellular, Molecular, and Biophysical Studies
Columbia University
New York NY 10032
clt47@columbia.edu

William Stafford Noble* †
Department of Computer Science
Columbia University
New York NY 10027
noble@cs.columbia.edu

ABSTRACT

Microarray expression data provides a new method for classifying genes and gene products according to their expression profiles. Numerous unsupervised and supervised learning methods have been applied to the task of discovering and learning to recognize classes of co-expressed genes. Here we present a supervised learning method based upon techniques borrowed from biological sequence analysis. The expression profile of a class of co-expressed genes is summarized in a probabilistic model similar to a position-specific scoring matrix (PSSM). This model provides insight into the expression characteristics of the gene class, as well as accurate recognition performance. Because the PSSM models are generative, they are particularly useful when a biologist can identify *a priori* a class of co-expressed genes but is unable to identify a large collection of non co-expressed genes to serve as a negative training set. We validate the technique using expression data from *S. cerevisiae* and *C. elegans*.

Keywords

microarray expression analysis, probabilistic modeling, gene classification

1. INTRODUCTION

The field of bioinformatics and computational biology has traditionally focused upon the analysis of DNA and protein sequences. Compared to sequencing technology, microarray expression profiling is relatively new. However, the last several years have witnessed a flurry of algorithmic development (for reviews, see [3; 21]), which nonetheless barely manages to keep pace with the concurrent rapid production of microarray expression data. This paper draws an analogy between gene functional classification from microarray data and from biosequences. We apply tools developed for classifying DNA and protein sequences to microarray expression data.

The analogy between biological sequences and microarray expression data is not immediately evident. For example, a protein sequence can be represented as a string of letters drawn from an alphabet of twenty amino acids. The string can be of variable length, ranging from less than 50 amino acids to more than 5000. In contrast, microarray expression measurements are most commonly represented as a fixed-length vector of continuous values. For cDNA arrays, each value is the logarithm of the ratios of the estimated abundances of mRNA in two tissue samples. One microarray experiment produces on the order of 10 000 such ratios, each generally corresponding to a particular gene.

Despite the apparent difference between these two classes of data, it is possible to represent microarray expression data using position-specific scoring matrices (PSSMs) exactly like the ones used to model highly conserved motif regions in protein families [17] and in DNA promoter regions [2]. These models have a fully probabilistic interpretation, and also allow for the computation of accurate p values during database searches. We show that PSSMs can be used successfully to recognize classes of functionally related genes, learning only from positive examples of the class.

Previously, gene classification from microarray expression data was performed using a discriminative method known as a support vector machine [5]. A significant difference between the SVM technique and the probabilistic models introduced here lies in the SVM's ability to learn from both positive and negative examples. When the training set includes both positive and negative examples, it is clearly beneficial to employ a learning method, such as the SVM, which is capable of learning from the entire training set. However, in many situations, it may be easier for a biologist to identify positive members of a given class than to identify negative members, especially since negative observations are rarely published in the scientific literature. In these cases, generative modeling may be the only solution.

In the following section, we provide an outline of the proposed gene expression modeling techniques. This section is followed by a more detailed mathematical treatment. We then validate the models using microarray expression data from yeast and from the nematode *C. elegans*.

*Corresponding author: Department of Computer Science, Columbia University, 450 Computer Science Building, Mail Code 0401, 1214 Amsterdam Avenue, New York, NY 10027, Tel: (212) 939-7114, Fax: (212) 666-0140

†Formerly William Noble Grundy. See www.cs.columbia.edu/~noble/name-change.html

2. ALGORITHM

The goal of this work is to construct accurate probabilistic models of classes of similarly regulated genes, and to use these models to make predictions about the regulation of unannotated genes. We proceed in three steps. First, the continuous-valued expression measurements are converted to a discrete representation. Second, the discretized values for a given class of similarly regulated genes is used to construct a position-specific scoring matrix. Next, this matrix is used to search the expression profiles of the remaining, unclassified genes, yielding a p value for the match between the model and each profile in the database.

Prior to constructing the probability model, the gene expression measurements are discretized. Because the distribution of expression measurements is highly non-uniform, we use a binning procedure that takes into account the distribution of the data in each experiment [6]. The number b of bins is selected *a priori*, and the width of each bin is set so that, in an experiment consisting of n genes, each bin contains n/b measurements. The bin widths are selected independently for each experiment in the database.

The probabilistic model captures the expression pattern associated with a given class of similarly regulated genes. The model consists of a series of states, each of which is associated with one microarray experiment and contains a discrete probability distribution over the bins in that state. The resulting model can be represented as a matrix, in which each row corresponds to a bin, and each column corresponds to an experiment. Each entry in the matrix is the estimated log probability, for genes in the modeled class, of observing a given expression value in a particular experimental condition. An example of one such model is shown in Figure 1. The model is identical to the position-specific scoring matrices employed to model conserved sequence motifs. The presence of regions of high and low values in the model indicates that there are large differences between the expression levels of the genes in the class and those of the average gene, providing the basis for discrimination. A model generated for a representative random selection of genes would appear uniformly grey in the figure.

The accuracy of a motif PSSM for searching a sequence database can be improved by incorporating into the PSSM the background distribution of sequences in the database. This step, however, is not necessary for expression PSSMs because the discretization step already incorporates the background distribution. If the database to be searched is also used to compute bin widths, then the background distribution over bins in the discretized data is exactly uniform. Therefore, the log odds ratios and log probabilities in the PSSM would differ by a constant factor that would not affect the database search performance.

A significant benefit of employing probabilistic models is their ability to output interpretable scores. In particular, we can compute the log-odds score, which tells us how much more likely it is that a candidate gene profile was generated by the given model rather than being drawn from the background distribution. Furthermore, we can use established methods [1; 22] to convert these scores into p values, where the null hypothesis being tested is that the expression profile is randomly drawn from the background distribution. Either of these scoring schemes can be used to rank the sequences in an expression database, thereby identifying high-scoring genes that may belong to the class being modeled.

3. METHODS

3.1 Gene expression modeling

We begin with a matrix S of gene expression data, containing log expression ratios for n genes (rows) in m experimental conditions (columns). We have identified a subset $\hat{S}$ of genes that are expected, from prior knowledge, to share similar expression profiles. We want to build a matrix model $\hat{M}$ of gene expression profiles drawn from the training set. Assuming that the individual measurements are independent of one another, the probability $P(S_i|\hat{M})$ of the expression profile of gene i may be written as:

$$P(S_{i,1}|\hat{M})P(S_{i,2}|\hat{M})\dots P(S_{i,m}|\hat{M}). \quad (1)$$

Our model captures each term in this expression with one column of parameters in the matrix model. Thus, each matrix entry $\hat{M}_{j,k}$ is $P(x_k = j|\hat{M})$, the probability that gene expression measurement j is seen in experiment k. It can be shown [10] that the maximum likelihood estimate for model parameter $\hat{M}_{j,k}$ is given by the frequency with which expression value j occurs in the column k of the training set:

$$\hat{M}_{j,k} = \frac{E_k(j)}{\sum_{j'=1}^{b} E_k(j')} \quad (2)$$

where $E_k(j)$ is the number of times that expression measurement value j is observed in experimental condition k, and b is the total number of bins in the model.

The maximum likelihood estimate is reasonable in the presence of very large quantities of data. However, for smaller data sets or larger models, we need to incorporate prior knowledge. In biosequence analysis, the background distribution of amino acids or nucleotides in a large database can be incorporated into the above estimation formula as a single Dirichlet prior using a pseudocount technique [10]. Similarly, we derive our prior by constructing a second matrix model M from the entire data set S. As noted above, due to the data-specific binning procedure we employ, this background model M is uniform, with each entry $M_{j,k} = 1/b$. Therefore, the estimation formula for $\hat{M}$ is as follows:

$$\hat{M}_{j,k} = \frac{E_k(j) + A/b}{\sum_{j'=1}^{b} E_k(j') + A} \quad (3)$$

In the experiments reported here, the weight A on the prior is 10.

The trained model is then used to search the complete set S of expression data. The match between a model $\hat{M}$ and an expression vector S_i can be evaluated using a log-odds score:

$$L(S_i) = log\frac{P(S_i|\hat{M})}{P(S_i|M)} = \sum_{k=1}^{m} log\frac{\hat{M}_{j,k}}{M_{j,k}} \quad (4)$$

The log-odds score can be used directly to rank the expression profiles in S. Expression profiles with high scores are likely to have been generated by the model.

The log-odds score $L(\cdot)$ can be rendered more interpretable by converting it to a p value or an E value. This conversion is accomplished by computing the probability density function (PDF) for scores produced by the model [1; 22]. The

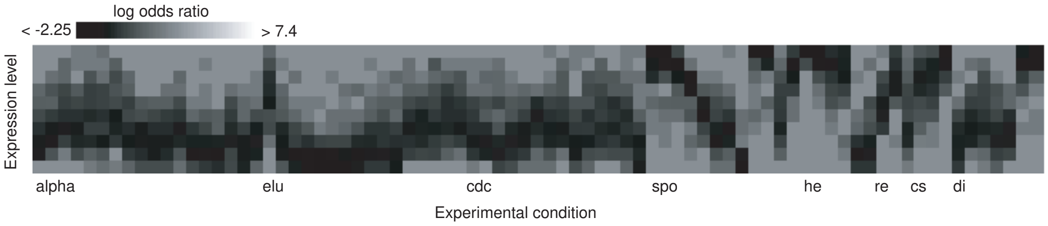

Figure 1: **Expression PSSM of yeast cytoplasmic ribosomal proteins.** Discretized expression levels appear along the y-axis, and experimental conditions appear along the x-axis. The shading in each square represents the magnitude of the corresponding model parameter. Labels on the x-axis represent the beginnings of experimental series: three cell division cycles — synchronized with α factor arrest (alpha) and centrifugal elutriation (elu), and measured using a temperature sensitive *cdc15* mutant (cdc) — sporulation (spo), heat shock (he), reducing shock (re), cold shock (co), and diauxic shift (di).

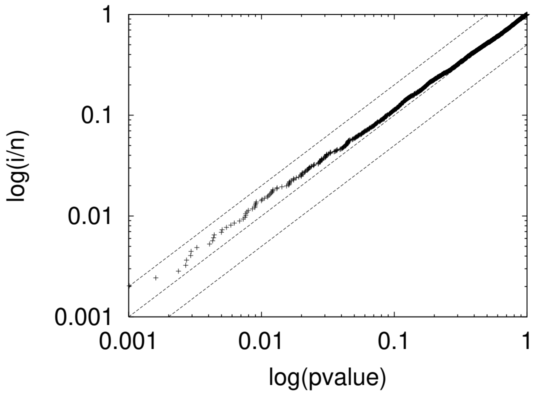

Figure 2: **Accuracy of the p value computation.** A set of p values are calculated for 2465 synthetic expression profiles, which were generated by shuffling expression values for each experiment in the real data. The p values are sorted and the resulting ranks recorded. The plot shows $i/2465$ as a function of p value, where i is the rank. The line $y = x$ represents the distribution of p values one would expect from random data, and the flanking lines represent a two-fold difference from the expected value.

computation of the PDF proceeds inductively via dynamic programming, as follows. First, the model entries are converted to positive integers, such that $\hat{L}_{j,k+1}$ is the integer log-odds score for bin j in experiment $k+1$. These integral scores correspond to indices x of an array A defining the PDF, which is filled inductively by noting that if we know the PDF $A^{(k)}$ for a model consisting of the first k experiments, we can calculate the PDF $A^{(k+1)}$ as

$$A^{(k+1)}(x) = \sum_{j=1}^{b} A^{(k)}(x - \hat{L}_{j,k+1}) M_{j,k}. \qquad (5)$$

This computation is exactly like the computation used in computing p values for sequence motifs, except that we use a position-specific background model M. The induction is initialized with $A^{(0)}(0) = 1$ and $A^{(0)}(x) = 0$ for $x \geq 1$. Iterating b times yields the PDF for a random expression profile matching the model, which is used to calculate a cumulative probablity distribution and thus p values. The

Class	Size
S. cerevisiae classes	
Tri-carboxylic acid cycle	17
Respiration	30
Cytoplasmic ribosome	121
Proteasome	35
Histones	11
Helix-turn-helix proteins	16
C. elegans classes	
Ribosomal	34
Proteasome subunit	28

Table 1: **Gene functional classifications.**

accuracy of the resulting p values computed is demonstrated in Figure 2. The E value, which is the expected number of times that a match with a given score will appear in a random database of a particular size, can be derived from the p value by multiplying the p value by the number of entries in the database.

3.2 Experimental validation

We validate the methods described above using two publicly available cDNA microarray gene expression data sets. We test our method using cross-validation, and compare the resulting recognition performance to two other gene functional classification algorithms, a simple k-nearest neighbor technique and the SVM classifier.

The first data set is the widely-studied set of yeast expression measurements produced by Eisen *et al.* [11]. This gene expression matrix contains 79 experiments over a variety of experimental conditions. The complete data set contains 2465 genes. Training labels were extracted from the MIPS yeast genome database (MYGD) [18]. We use six classes described previously [5]. These include five classes expected to be learnable and the helix-turn-helix proteins, included as a negative control.

The second data set is an amalgam of *C. elegans* data from three previous studies. In the first study [13], mRNA expression levels were measured over the course of eight time-points during the life cycle of the nematode. This data was collected using Affymetrix chip technology, and expression levels were quantitated in units of parts-per-million detected on the surface of the microarray chip. For each gene, the vector of eight expression measurements was normalized to a mean of zero and a variance of one. Data from the other two experiments [16; 20] were collected using spotted glass technology. For each gene, the log expression ratios from 54 experimental conditions were combined and normalized to have a mean of zero and a variance of one. Genes were eliminated from the data set if they did not appear in all experimental conditions or if they lacked functional annotation in the WormPD database [8; 7]. The final data set consists of 6801 genes measured under 62 experimental conditions. For this data, training labels were extracted from WormPD for the classes listed in Table 1. Both the yeast and *C. elegans* expression matrices are discretized as described above, using 10 bins for each experiment. The yeast gene classifications are derived from the MIPS Yeast Genome Database [18] (www.mips.biochem.mpg.de/proj/yeast). The *C. elegans* classifications are derived from Proteome's WormPD [8; 7] (www.proteome.com/databases).

We compare the performance of the techniques described above to two other classifiers. The first is the support vector machine algorithm [23; 9]. Briefly, the SVM learning algorithm constructs a maximum-margin hyperplane that separates the negative and positive examples in a training set. The margin is soft, in the sense that it permits some misclassifications, as might be expected in a noisy data set. The hyperplane is calculated in an implicit feature space, whose dimensionality depends upon the choice of kernel function used. We use the radial basis kernel function $K(\tilde{\mathbf{X}}, \tilde{\mathbf{Y}}) = exp(-||\tilde{\mathbf{X}} - \tilde{\mathbf{Y}}||^2/2\sigma^2)$, which has previously been shown to perform well on the yeast data set [5]. Once the separating hyperplane is constructed, the SVM can be used to predict the classifications of previously unseen examples. We use a particularly simple SVM optimization algorithm that was introduced by Jaakkola *et al.* [14]. Our implementation is freely available on the web (www.cs.columbia.edu/compbio/svm). For a more complete explanation of our SVM methods, see [5; 4] and the accompanying web page (www.cse.ucsc.edu/research/compbio/genex).

The second classifier is a k-nearest neighbor technique, similar to that used by Golub *et al.* [12]. For a given test vector, the k closest members of the training set are located using the Pearson correlation coefficient [24] as similarity metric. The predicted label of the test vector is determined by a vote among the k neighbors. In this work, we use $k = 3$.

We evaluate the classification algorithms using three-fold cross-validation. The labeled training set is split into three subsets of equal size. Positive and negative examples are split evenly among the three subsets. Training is performed using two of the subsets, and testing is performed on the remaining subset. This train/test procedure is repeated three different ways, and the results are combined to produce one complete set of predictions for the entire training set.

4. RESULTS

The results show the utility of modeling gene expression data using PSSMs. Figures 3 and 4 summarize the performance of the three classification techniques. Each plot is a receiver operating characteristic curve, plotting true positives as a function of false positives for varying classification thresholds. Because the k-nearest neighbor technique has no threshold, it appears as a single point on each plot. The helix-turn-helix class is included as a non-learnable control class, and as expected, none of the three classifiers performs well for this class. In every other class, the PSSM technique leads to significant learning. Except for the ribosomal classes (discussed below), the E value threshold yields a result that is close to or better than the k-nearest neighbor classification. As expected, neither the probability model nor the k-nearest neighbor algorithm, which are trained only from positive examples, performs as well as the SVM technique, which is trained from positive and negative examples. In most cases, however, the difference between PSSM and SVM techniques is not large, especially near the SVM threshold.

In several classes, the conservative E value threshold of 10^{-4} yields a large number of false positives. This effect is most notable in the yeast cytoplasmic ribosomal class, in which there are 61 false positives above threshold. We know from Figure 2 that the p value computation produces accurate results for data that has been shuffled within each experiment. We can conclude, therefore, that the independence assumption built into the PSSM model is inappropriate in this case.

We have verified that correlations across experiments lead to the p value scaling problem. We computed Pearson correlation coefficients among all pairs of the 79 experimental conditions in the yeast data. We then eliminated one experiment from each highly correlated pair until the maximum correlation coefficient between pairs of remaining experiments was 0.7. The resulting set of 62 experimental vectors yields 29 false positive ribosomal proteins at the 10^{-4} E value threshold. Unfortunately, this data reduction technique is not a panacea, because the reduced training set leads to a corresponding reduction in sensitivity. For the ribosomal class, the result is a reduction in true positives from 120 to 117. In other classes, removing correlated features has a uniformly detrimental effect on the overall error rate.

5. DISCUSSION

Our experiments show clearly that the SVM technique provides more accurate recognition performance than the probabilistic models presented here. In a sense, however, this comparison is unfair. One motivation for developing these generative models is to cope with situations in which a biologist has identified *a priori* a class of co-expressed genes but is unable to identify a large collection of non co-expressed genes to use as a negative training set. In our experience, this type of situation arises regularly in the analysis of microarray data. In the absence of negative examples, the PSSM technique would learn as shown above, whereas the SVM would learn nothing at all.

In addition to learning only from positive examples, generative modeling has other benefits. For example, the model shown in Figure 1 provides insight into the expression characteristics of the class of genes being modeled. In contrast, because the SVM operates in an implicit, high-dimensional space, the relevant features in that space are difficult to examine. Furthermore, the probabilistic underpinnings of the proposed PSSM model provide a principled way to deal with missing data and to learn simultaneously from other types of data.

The k-nearest neighbor technique also performs well for many of the classes we investigated. This technique has the benefit of simplicity, but lacks the probabilistic interpretation of a generative model. In addition, the k-nearest neighbor

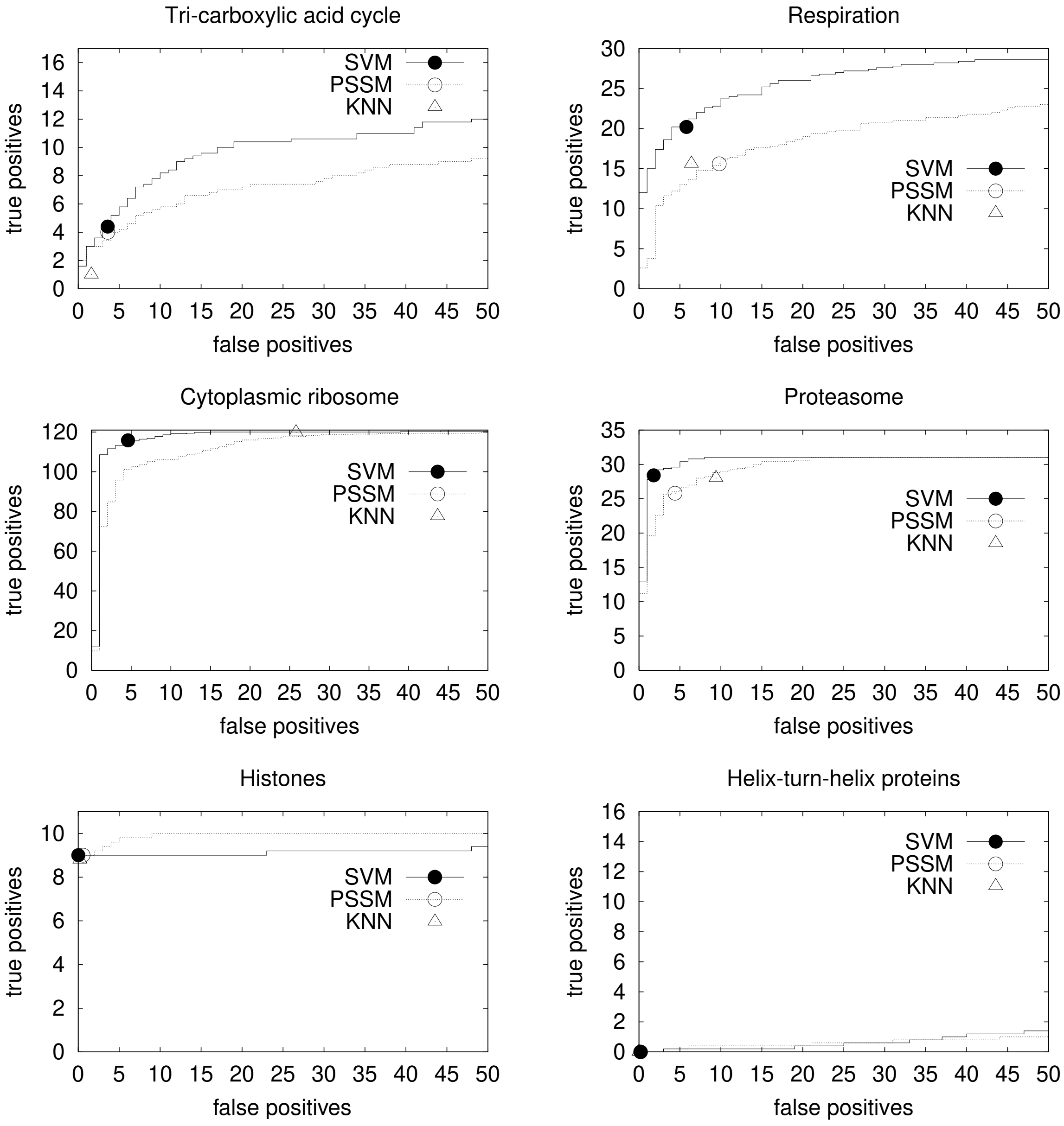

Figure 3: **Receiver operating characteristic curves for yeast MYGD classes.** Each plot is the result of combining the predictions from three-fold cross-validation, and averaging the resulting false positive and true positive counts across five repetitions. Triangles represent average true positives and false positives given by the k-nearest neighbor technique. Filled circles represent the average number of true and false positives selected by the SVM; open circles represent the same average for the two-dimensional PSSM using an E value threshold of 10^{-4}. In the histones and helix-turn-helix classes, all three points overlap. In the ribosomal class, the open circle for the PSSM technique is off the edge of the plot at (61,120). The helix-turn-helix class is a control, not expected to exhibit significant learning with any method.

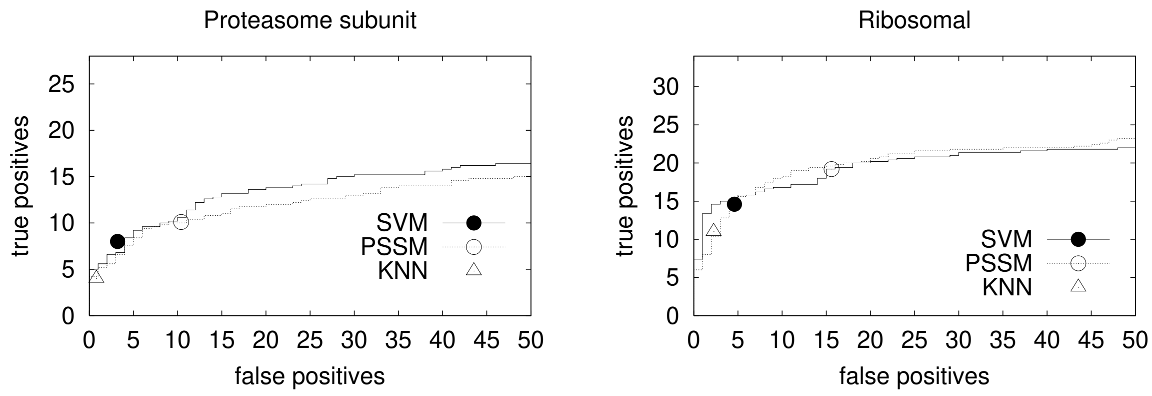

Figure 4: **Receiver operating characteristic curves for *C. elegans* WormPD classes.** See Figure 3 caption for description.

algorithm does not allow the user easily to select a classification threshold, since the output is a single classification rather than a ranking.

A significant assumption underlying the PSSM model is the independence of states in the model. This assumption is clearly unrealistic, both in modeling sequence motifs and in modeling expression profiles. The expression data sets we used consisted of time series data, which leads to particularly high levels of correlation between expression levels between experimental conditions, making the independence assumption especially problematic. For some classes of co-expressed genes, this leads to improperly scaled p values. The simple models we demonstrate here may yield more accurate p values when applied to non-time series data, as suggested by the improvement in the false positive rate we observed after removing highly correlated features. We can also weaken the independence assumption and improve the accuracy of the p values by including inter-experimental dependencies explicitly in the probabilistic model. Toward this end, we have experimented with a larger matrix model, in which the probabilities at each state are conditional upon the observation at the previous state. The resulting model has three dimensions, rather than two: the entry in position (i, j, k) is the probability of observing expression bin i, given that we are in experimental condition k and that we observed bin j in condition $k - 1$. This richer model is capable of capturing more detail, at the expense of requiring more training data. Because the training sets employed here are relatively small, the increased model size leads to a significant decrease in overall classification performance. We are currently developing a more sophisticated set of prior probabilities for use with higher-order PSSM models.

The conversion of continuous gene expression measurements to discrete values is not required in order to build a probabilistic model of gene classes. Indeed, discretizing has the drawback of potentially erasing fine structure in the data. For example, two distinct peaks of expression values could be lumped together in a single bin. Instead, models could be derived using a mixture of continuous distributions for each experiment. Though building such models is straightforward, we chose to use discrete distributions for our initial studies because the form of the underlying continuous distributions is not obvious *a priori*. In future, however, we plan to experiment with continuous density models.

In this work, we have applied the PSSM method to modeling classes of co-expressed genes. However, the technique could also be used to model tissue classes that share expression profiles. In this situation, however, the number of model parameters would be extremely large. For data sets of realistic size, it will be necessary to either reduce the number of genes under consideration [6], or to cluster the genes and tie the model parameters within the resulting gene clusters. Jaakkola *et al.* [15] have demonstrated how to combine the accurate recognition performance of the SVM with the interpretability and other benefits of a generative probability model. In this technique, called the Fisher kernel method, the positive training examples are used to construct a probability model, which is then used as the kernel function for a support vector machine. The Fisher kernel method has been applied successfully to protein homology detection [14] and promoter classification [19]. In future work, we plan to apply this method to generative models of gene expression.

Acknowledgments

WSN is funded by an Award in Bioinformatics from the PhRMA Foundation, and by National Science Foundation grants DBI-0078523 and IIS-0093302.

6. REFERENCES

[1] T. L. Bailey and M. Gribskov. Methods and statistics for combining motif match scores. *Journal of Computational Biology*, 5:211–221, 1998.

[2] D. Barrick, K. Villanueba, J. Childs, R. Kalil, T. D. Schneider, C. E. Lawrence, L. Gold, and G. D. Stormo. Quantitative analysis of ribosome binding sites in *E. coli*. *Nucleic Acids Research*, 22(7):1287–1295, 1994.

[3] A. Brazma and J. Vilo. Gene expression data analysis. *FEBS Letters*, 23893:1–8, 2000.

[4] M. P. S. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. Sugnet, M. Ares, and D. Haussler. Support vector machine classification of microarray gene expression data. Technical Report UCSC-CRL-99-09, University of California, Santa Cruz, June 1999.

[5] M. P. S. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. Sugnet, T. S. Furey, J. M. Ares, and D. Haussler. 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, 2000.

[6] A. Califano, G. Stolovitsky, and Y. Tu. Analysis of gene expression microarrays for phenotype classification. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 75–85, 2000.

[7] M. C. Costanzo, M. E. Crawford, J. E. Hirschman, J. E. Kranz, P. Olsen, L. S. Robertson, M. S. Skrzypek, B. R. Braun, K. L. Hopkins, P. Kondu, C. Lengieza, J. E. Lew-Smith, M. Tillberg, and J. I. Garrels. YPD[tm], PombePD[tm], and WormPD[tm]: model organism volumes of the BioKnowledge[tm] library, an integrated resource for protein information. *Nucleic Acids Research*, 29(1):75–79, 2001.

[8] M. C. Costanzo, J. D. Hogan, M. E. Cusick, B. P. Davis, A. M. Fancher, P. E. Hodges, P. Kondu, C. Lengieza, J. E. Lew-Smith, C. Lingner, K. J. Roberg-Perez, M. Tillberg, J. E. Brooks, and J. I. Garrels. The Yeast Proteome Database (YPD) and *Caenorhabditis elegans* Proteome Database (WormPD): comprehensive resources for the organization and comparison of model organism protein information. *Nucleic Acids Research*, 28(1):73–76, 2000.

[9] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge UP, 2000.

[10] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis*. Cambridge UP, 1998.

[11] M. Eisen, P. Spellman, P. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences of the United States of America*, 95:14863–14868, 1998.

[12] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[13] A. A. Hill, C. P. Hunter, B. T. Tsung, G. Tucker-Kellog, and E. L. Brown. Genomic analysis of gene expression in *C. elegans*. *Science*, 290(5492):809–812, 2000.

[14] T. Jaakkola, M. Diekhans, and D. Haussler. Using the Fisher kernel method to detect remote protein homologies. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 149–158, Menlo Park, CA, 1999. AAAI Press.

[15] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems 11*, San Mateo, CA, 1998. Morgan Kauffmann.

[16] M. Jiang, J. Ryu, M. Kiraly, K. Duke, V. Reinke, and S. K. Kim. Genome-wide analysis of developmental and sex-regulated gene expression profiles in *Caenorhabditis elegans*. *Proceedings of the National Academy of Sciences of the United States of America*, 98(1):218–223, 2001.

[17] C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wootton. Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. *Science*, 262(5131):208–214, 1993.

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

[19] P. Pavlidis, T. S. Furey, M. Liberto, D. Haussler, and W. N. Grundy. Promoter region-based classification of genes. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 151–163, 2001.

[20] V. Reinke, H. E. Smith, J. Nance, J. Wang, C. V. Doren, R. Begley, S. J. Jones, E. B. Davis, S. Scherer, S. Ward, and S. K. Kim. A global profile of germline gene expression in *C. elegans*. *Molecular Cell*, 6:605–616, 2000.

[21] A. Schulze and J. Downward. Analysis of gene expression by microarrays: cell biologist's gold mine or minefield? *Journal of Cell Science*, 113:4151–4156, 2000.

[22] R. L. Tatusov, S. F. Altschul, and E. V. Koonin. Detection of conserved segments in proteins: iterative scanning of sequence databases with alignment blocks. *Proceedings of the National Academy of Sciences of the United States of America*, 91(25):12091–12095, 1994.

[23] V. N. Vapnik. *Statistical Learning Theory*. Adaptive and learning systems for signal processing, communications, and control. Wiley, New York, 1998.

[24] J. H. Zar. *Biostatistical Analysis*. Prentice Hall, 1998.