TRANSCRIPTION FACTOR DISCOVERY USING SUPPORT VECTOR MACHINES AND HETEROGENEOUS DATA

José F. Barbe[1], *Ahmed H. Tewfik*[1] *and Arkady B. Khodursky*[2]

[1]Department of Electrical and Computer Engineering
[2]Department of Biochemistry, Molecular Biology and Biophysics
University of Minnesota, {Minneapolis, MN 55455}[1]{St. Paul, MN 55108}[2]
pepe@umn.edu, tewfik@umn.edu, khodu001@umn.edu

ABSTRACT

In this work we analyze the suitability of expression and sequence data for discovery of co–regulatory relationships using Support Vector Machines. In addition, we try to assess the possibility of improving such results by heterogeneous data fusion and by estimating a probability of a correct classification.

As shown in other studies, we have found that transcription co–expression is a good estimator for genetic co–regulation. We also have found some evidence that operator site sequence motifs can be used to estimate co–regulation, but the kernels used for feature extraction did not achieve classification rates comparable to expression data.

Finally, the additional information provided by combining sequence and expression data can be exploited to estimate the probability of correct classification.

1. INTRODUCTION

The purpose of this work was to search for effective ways of discovering unknown regulon relationships using *Support Vector Machines* (SVM). We aim to combine expression and sequence data to increase the amount of available information, as suggested in [1, 2].

Qian *et al.* studied the same problem in [3], where they used SVM and expression data to discover transcription relationships. In their work the transcription relation was encoded in the data by concatenating the expression of the TF and the target gene. For negative training samples they used genes that did not have TF sequence motifs or by random pairing. Their results show a low error rate for SVM, in particular using Radial Basis Functions (RBF) kernel.

In [4], Pavlidis *et al.* look at the issue of determining regulation relationships based on TF motifs. For this purpose they use a Hidden Markov Model (HMM) whose transition probabilities are trained used the promoter sites of the known TFs. Using the Fischer Score a feature space can be created based on the parameters of the HMM. Finally, they train a SVM using the RBF kernel. While the results look promising, the authors acknowledge that locating highly conserved motifs is essential to the success of this classification method.

Our approach consists in the following: For the expression data we assume that co–expression signals co–regulation, therefore, we cluster based on the similarity of expression profiles. For the sequence data, we use Length Dependent String Kernels (LDSK) to extract the statistical features out of the sequence data instead of using HMMs. LDSK are computationally efficient and much less expensive than HMMs. Finally, we also study the possibility of assigning a probability score to each classification using the method proposed by Platt in [5].

2. PROKARYOTE GENE REGULATION

In prokaryotes regulation is performed basically in one way: by blocking the RNA *polymerase* from binding to the DNA sequence. This is achieved by a signaling sequence called operator site, located upstream of the regulated gene, that signals the TF to bind to the DNA and block the RNA *polymerase*. This, effectively stops the production of mRNA, which hold copies of the code of the regulated gene.

Operator sites of co-regulated genes show sequence patterns which are called motifs. In this work we will use LDSKs to search for new genes that show a similar motif, thus, belonging to the same regulation pathway.

The time evolution production of mRNA can be measured in the genome–wide expression profiles. Since the expression of co-regulated genes is turned on and off in unison we can expect that their expression profiles must behave with some sort of coherence (Co-expressed). This is the second source of information that we will use in this work for classification.

3. STATISTICAL LEARNING

For this work we use two learning formulations from Statistical Learning to perform our experimental work: k–*Nearest Neighbors* (KNN) and SVM. The purpose of KNN is to serve as a point of comparison for the rest of the experiments as shown later.

3.1. *k*–Nearest Neighbors

Given some random variable $\mathbf{x}_i \in \mathbb{R}^n$ with labels $y_i \in \mathbb{Z}$, KNN assumes that samples from the same source are clustered in the input space $\mathbb{R}^n$. Thus, KNN seeks the local

1-4244-0999-3/07/$25.00

boundary region that will give us the best performance for the given random process. The optimal classification is given when the estimated label is chosen equal to the majority of the labels inside in the local boundary.

We can formulate KNN for the two–class classification, *i.e.* $y_i \in \{-1, +1\}$, problem as:

$$\hat{y} = \begin{cases} +1 & \sum_{i=1}^{n} y_i K_k(\mathbf{x}_0, \mathbf{x}_i) > 0 \\ -1 & \text{otherwise} \end{cases} \quad (1)$$

where the training points are the pairs $(\mathbf{x}_i, y_i)$, $\hat{y}$ is the estimated label for $\mathbf{x}_0$ and $K_k(\mathbf{x}_0, \mathbf{x}_i) = 1$ if $\mathbf{x}_i$ is one of the k closest points and 0 otherwise. To avoid confusion we select $k = 2m - 1$ for $m \in \mathbb{Z}^+$.

To find the optimal value of k we can perform k-fold cross validation on the training data until the empirical risk, given by the number of misclassifications, converges to some minimum.

3.2. Support Vector Machines

SVM is a type of hyperplane based binary classifier, that uses the *maximum margin paradigm* to achieve optimal prediction generalization from the training data.

The problem amounts to finding the hyperplane $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$ that best describes the labels of the training data and, at the same time, provides the maximum distance between convex hulls that enclose the data of each class. Then the label for a new point $\mathbf{x}$ can be assigned by the function:

$$f_{w,b} = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b) \quad (2)$$

Non-linear data can be linearized by mapping the space to a higher dimensional space:

$$\begin{aligned} \mathbf{\Phi}: \quad & \mathcal{X} \to \mathcal{H} \\ & x \mapsto \mathbf{x} := \mathbf{\Phi}(x) \end{aligned} \quad (3)$$

Better computational efficiency is achieved by taking advantage of the kernel trick: $\mathrm{K}(x, y) = \langle \mathbf{\Phi}(x), \mathbf{\Phi}(y) \rangle$.

In the case that we have non separable data, we introduce slack variables (ξ_i) that allow misclassifications for some of the training points and a global cost variable (C) which penalizes misclassifications. The value of C is tuned during the training process.

The solution to this optimization problem is given by solving (4) for α:

$$\max_{\alpha \in \mathbb{R}^m} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \quad (4)$$

subject to $0 \leq \alpha_i \leq \frac{C}{m}$ and $\sum_{i=1}^{m} \alpha_i y_i = 0$ for $i = 1, \ldots, m$.

3.2.1. Numerical Kernels

In the literature there are some well known kernels defined for $\mathbb{R}^n$ spaces. In the present work, we use the following:

Radial Basis Functions (RBF)

$$k(x, x') = \exp\left(- \frac{\|x - x'\|^2}{2\sigma^2} \right) \quad (5)$$

Linear

$$k(x, x') = \langle x, x' \rangle \quad (6)$$

3.2.2. String Kernels

Given that $\mathcal{X}$ is a non-empty set, we can define a map $\mathbf{\Phi}$ to some Hilbert Space where the inner product function is defined. We can exploit the previous characteristic to perform classification based on string data. A string based kernel can be the following [6]:

$$k(x, x') := \sum_{s \in \mathcal{A}^*} \mathtt{num}_s(x)\, \mathtt{num}_s(x') w_s \quad (7)$$

where $\mathcal{A} = \{\mathtt{A}, \mathtt{T}, \mathtt{G}, \mathtt{C}\}$ is the alphabet (For DNA sequence Classification), $\mathcal{A}^*$ is the subset of non-empy strings, $x \in \mathcal{A}^n$, $\mathtt{num}_s(x)$ is the number of times the substring s appears in x and w_s is a weight associated with it.

Leslie et al. propose in [7] a string kernel for protein classification, called *K Spectrum* kernel (KSPEC) which measures similarities between strings by looking at frequencies of substrings of length k:

$$\begin{aligned} \mathbf{\Phi}: \quad & \mathcal{X} \to \mathbb{R}^{l^k} \\ & x \mapsto \mathbf{x} := \mathbf{\Phi}(x) = \mathtt{num}_a(x)_{a \in \mathcal{A}^k} \end{aligned} \quad (8)$$

$$k(x, y) = \langle \mathbf{\Phi}(x), \mathbf{\Phi}(y) \rangle \quad (9)$$

In equation (8) for string x we create a vector $\in \mathbb{R}^{l^k}$, where each dimension holds the amount of times a given substring of length k appears in the string. To measure the similarity between strings we calculate the inner product between those vectors (9).

We also used the *Constant* kernel for strings which assigns $w_s = |s|$ in equation (7).

Finally, *Vishwatan* and *Smola* propose in [6] a computationally efficient method, based on *Suffix Trees*, to perform the calculation of LDSKs.

3.2.3. Probability Estimation

Platt [5] proposes a method to estimate the probability of correct classification. This method assumes that the Cumulative Distribution Function (CDF) $P[y = 1|f]$, *i.e.* the conditional probability of belonging to class 1 given the decision function, can be approximated by a sigmoid function. This observation comes after doing an empirical analysis of the results of SVM with real data.

If the previous assumption proves to be correct for our data, then it is safe to use this algorithm.

4. EXPERIMENTAL SETTING

To perform classification we used libSVM v2.82 [8], which is a freely available implementation of the SVM algorithm. For string kernels we used SASK, a freely available implementation of an algorithm proposed in [9], which is a variation of the Suffix Tree algorithm mentioned earlier; this algorithm achieves improved performance by using *Suffix Arrays*.

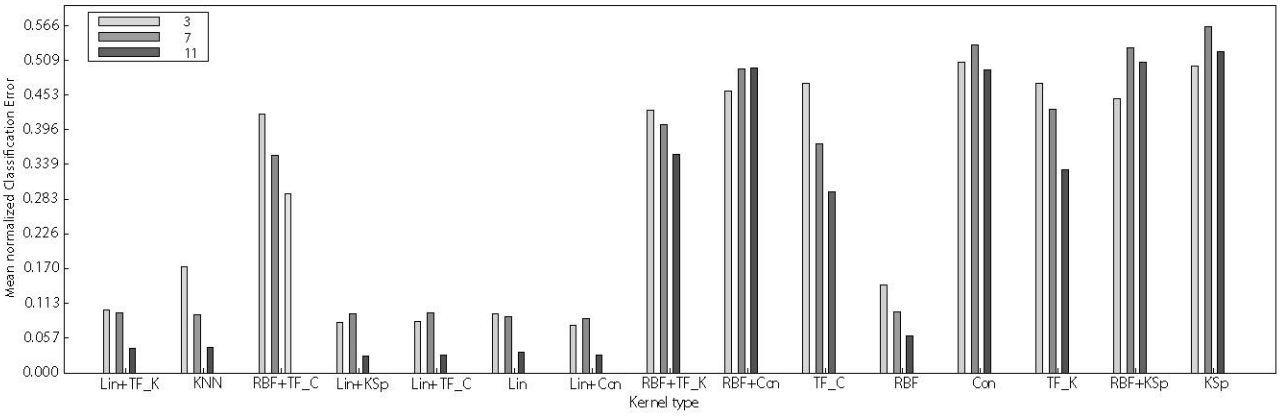

Figure 1. Average *Mean Normalized Error Rate* across 100 experiments with the different classifiers used in the experiment for the *PhoB* dataset. KSp and Con refer to sequence data classification with K-Spectrum and the Constant Kernel. TF_K and TF_C refer to the same classifier using TF sequence selected with PSSM. The legend refers to using 3, 7 and 11 of the training samples from the labeled dataset (With a similar amount of negatives, randomly sampled).

4.1. Material

For our experiments we used expression and sequence data from *Escherichia Coli K12*, provided by one of the co-authors. The genome–wide expression measured the evolution in time of two different regulatory processes: *LexA*, which is activated when the bacteria sustains DNA damage; *PhoB*, which is triggered when the bacteria undergoes Phosphate starvation.

For the sequence data we used a portion of the raw string of the data located in the upstream regions of the genes and also a localized portion of the string where the score given by the PSSM was maximum; for the training data we always used the known TF binding site.

After processing the data we had 19 known genes for the first data set and 15 for the second; in the same way we had 2762 and 2567 unlabeled genes respectively. The expression data was measured experimentally and was available as vector of 7 and 8 time points, respectively.

From RegulonDB [10], a manually curated database of *E. Coli*, we obtained the list of the known members of the regulon (Positive Class) and Position Specific Scoring Matrices (PSSM) for discovery of TF binding sites.

4.2. Procedure

There is no certain way to establish that a given gene is not part of a regulon (Negative Class). Nevertheless the size of a regulon on average is very small (Less than 50 genes) compared to the overall size of the entire genome (More than 2500). Therefore the likelihood of picking a gene that belongs to positive class by randomly sampling the unlabeled data set is very small ($\ll 0.02$). We used this assumption to balance the training data, *i.e.*, we sampled randomly a number of genes from the unlabeled data set and added them to the training set as the negative class.

To determine a lower bound for the amount of necessary information for acceptable classification performance, we performed experiments with 25%, 50% and 75% of the known data used for training and the remaining used for the test.

We repeated our experiments a 100 times and each time the data used for training was selected randomly (From both the labeled and unlabeled data sets), to determine the robustness of the classifier against variations in the training data.

The training was performed by searching through the parameter space for the parameters that would minimize the classification error using k–fold Cross Validation (We used k between 3 and 5 depending on the size of the training set). The parameters and their respective search space were: C for the SVM, $[2^{-5}, 2^{15}]$; σ for the RBF, $[2^{-15}, 2^{3}]$; K for the KSPEC, $[2, 6]$; the length of the raw string, $[50, 300]$; and the linear combination coefficients for the heterogeneous kernels $\{.25, .50, .75\}$.

Finally, we also included a KNN classifier as a point of reference for SVM. Classification with KNN was done only with expression data. The training method was similar to SVM and the only parameter trained was k which was $\{3, 5, 7, \ldots, |\texttt{training\_set}|/2\}$.

5. RESULTS

Due to lack of space we will discuss the results that we deemed more interesting.

In figure 1 we can see an overall comparison of the performance of the classifiers on average. The first thing to note is that classification methods using expression data show a good performance; the sequence based methods don't work as good.

Within the expression data, KNN offers very competitive results, in spite of its simplicity. SVM methods offer the best performance and, within these, the Linear Kernel offers good results for low amounts of training data. This could be explained by the fact that dimensionality of the input space is low relative to the amount of training data (n$\backsim$8), effectively making this a linearly separable problem.

The classification performance of the LDSK was improved if we only used the portions of the sequence that

had a high score using PSSM. Nevertheless, relative to expression data the improvement was not meaningful.

We also performed combinations of all the sequence and expression kernels to see if the classification performance was improved. In the best case, when the Linear kernel is used for expression, the results are the same. When using RBF the classification error worsens.

In order to analyze if the Probability estimation method mentioned in section 3.2.3 was viable with our data set, we also plotted the CDF of $P[y = 1|f]$. Due to space restrictions we are not showing those graphs, but the CDF showed a behavior that could be approximated by a sigmoid as predicted by Platt [5]. Given that the assumption holds, we considered that it was safe to use this algorithm.

Data fusion did not improve the classification performance as mentioned previously, however we found that, in a consistent manner, the Heterogeneous classification that included the Linear Kernel for the expression data assigned a high probability value to all our test data (See figure 2). This was consistent across all of our experiments.

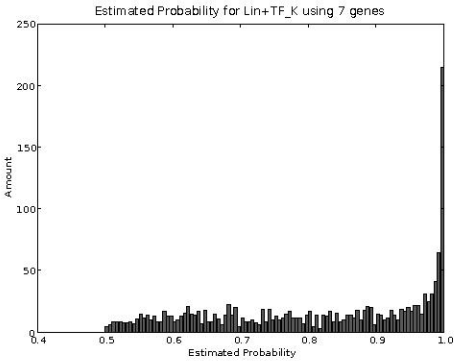

Figure 2. Histogram of the estimated probabilities for 100 independent experiments using the *PhoB* dataset, a combined kernel of Linear+KSPEC and 7 genes from the labeled set for training.

6. CONCLUSION

In the previous sections we have discussed the viability of expression and sequence data as means to perform TF discovery and possible ways to improve the performance of our SVM classifier.

Our previous results confirm, once again, that co–expression is a good estimator of co–regulation. While the results obtained from sequence data alone are not as good, by reducing our feature search to sequence areas that have a high probability using PSSMs we were able to improve the classification error rate, lending credence that motifs might be good also for classification regulon members.

On the other hand LDSKs seem less suitable to extract the features from the motif sequence that will allow SVM to perform a good classification. This is due in part to the fact that sequence data may present high order statistical interdependencies and LDSKs only measure correspondence of sequence substrings without regarding the relative position between them.

While fusing the kernels did not yield improved classification they did allow us to obtain meaningful results from the probability estimation algorithm. In particular, the best results where obtained by fusing the features extracted using a linear kernel with the expression data and any of the string kernels used.

Currently we are working in incorporating a Fisher Score Kernel as proposed in [4] to compare against the LDSKs and to see if we can obtain improved results by using a kernel that is capable of extracting more complex features.

7. REFERENCES

[1] P. Pavlidis, J. Weston, J. Cai, and W. N. Grundy, "Gene functional classification from heterogeneous data," in *Proceedings of the Fifth International Conference on Computational Molecular Biology*. ACM, 2001, pp. 242–248.

[2] G. R. G. Lanckriet, T. De Bie, N. Cristianini, M. I. Jordan, and W. S. Noble, "A statistical framework for genomic data fusion," *Bioinformatics*, vol. 20, no. 16, pp. 2626–2635, 2004.

[3] J. Qian, J. Lin, N. M. Luscombe, H. Yu, and M. Gerstein, "Prediction of regulatory networks: genome-wide identification of transcription factor targets from gene expression data," *Bioinformatics*, vol. 19, no. 15, pp. 1917–1926, 2003.

[4] P. Pavlidis, T. Furey, M. Liberto, D. Haussler, and W. Grundy, "Promoter region-based classification of genes," in *Proceedings of the Pacific Symposium on Biocomputing*, 2001.

[5] J. C. Platt, "Probabilistic Outputs for Support Vector Machines and Comparison to Regularized Likelihood Methods," in *Advances in Kernel Methods - Support Vector Learning*, A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, Eds., 2000, pp. 61–74.

[6] S. Vishwanathan and A. J. Smola, "Fast kernels for string and tree matching," in *Kernels and Bioinformatics*, K. Tsuda, B. Schölkopf, and J. Vert, Eds., Cambridge, MA, 2004, MIT Press.

[7] C. Leslie, E. Eskin, and W. S. Noble, "The spectrum kernel: A string kernel for svm protein classification.," in *Pacific Symposium on Biocomputing*, 2002, pp. 566–575.

[8] C.-C. Chang and C.-J. Lin, "Libsvm : a library for support vector machines.," Tech. Rep., Department of Computer Science, National Taiwan University, 2006.

[9] C. H. Teo and S. Vishwanathan, "Fast and Space Efficient String Kernels using Suffix Arrays," in *International Conference on Machine Learning*, 2006.

[10] H. S. *et al*, "RegulonDB (version 5.0): Escherichia Coli K-12 transcriptional regulatory network, operon organization, and growth conditions," *Nucleic Acids Res.*, vol. 34, Jan 2006.