Learning Regulatory Network Models that Represent Regulator States and Roles

Keith Noto and Mark Craven

Department of Biostatistics and Medical Informatics
Department of Computer Sciences
University of Wisconsin
Madison, Wisconsin 53706, USA
noto@cs.wisc.edu, craven@biostat.wisc.edu

Abstract. We present an approach to inferring probabilistic models of gene-regulatory networks that is intended to provide a more mechanistic representation of transcriptional regulation than previous methods. Our approach involves learning Bayesian network models using both gene-expression and genomic-sequence data. One key aspect of our approach is that our models represent *states* of regulators in addition to their expression levels. For example, the state of a transcription factor may be determined by whether a particular small molecule is bound to it or not. Our models represent these states using hidden nodes in the Bayesian networks. A second key aspect of our approach is that we use known and predicted transcription start sites to determine whether a given transcription factor is more likely to act as an activator or a repressor for a given gene. We refer to this distinction as the *role* of a regulator with respect to a gene. Determining the roles of a regulator provides a helpful bias in learning accurate representations of regulator states. We evaluate our approach using sequence and expression data for *E. coli* K-12. Our experiments show that our models are comparable to, or better than, several baselines in terms of predictive accuracy. Moreover, they have more explanatory power than either baseline.

1 Introduction

A significant, central challenge in computational biology is to develop methods that can elucidate biological networks from high-throughput data sources. In recent years, numerous research groups have developed methods that address the tasks of inferring regulatory [1] and metabolic networks [2] from data. Such models of biological networks can have both predictive and explanatory value. To achieve a high level of explanatory value, a model should represent the *mechanisms* of the network in as much detail as possible. In this paper, we describe an approach to inferring regulatory networks from gene-expression and genomic sequence data. Our approach incorporates several innovations that attempt to provide a more mechanistic representation than those used in previous work in this area. Our research has focused on prokaryotic genomes, and thus we empirically evaluate our method using sequence and expression data for *E. coli* K-12 [3]. Our experiments show that our models are able to provide expression predictions which are almost as accurate, and sometimes more accurate than several baselines with less explanatory value.

E. Eskin, C. Workman (Eds.): RECOMB 2004 Ws on Regulatory Genomics, LNBI 3318, pp. 52–64, 2005.

There are numerous factors that make the task of inferring networks from high-throughput data sources a difficult one. First, the available data characterizing states of cells, such as microarray data, are incomplete; they characterize the states of cells under a range of conditions that is usually quite limited. Second, there are typically high levels of noise in some of the available data sources, such as microarray and protein-protein interaction data. Third, measurements are not available for important aspects of the biological networks under study. For example, most efforts at network inference have employed only gene-expression measurements of protein-coding genes and genomic sequence data. However, in many cases gene regulation, even at the level of transcription regulation, is controlled in part by small molecules (*e.g.* IPTG inactivates the lac repressor), changes in protein states such as phosphorylation (*e.g.* arcA is activated through phosphorylation), or expression of small RNAs (*e.g.* 6S RNA associates with and regulates RNA polymerase).

Probabilistic models of gene regulation [4–13] are appealing because they can, in part, account for the uncertainty inherent in available data, and the non-deterministic nature of many interactions in a cell. The method that we present here builds on recent work in learning probabilistic graphical models to characterize transcriptional regulation.

Our approach, which involves learning Bayesian networks [14] using both gene-expression data from microarrays and genomic sequence data, incorporates several innovations. First, our models include hidden nodes that can represent the *states* of transcription factors. It is often the case that *expression levels* of transcription factors alone are not sufficient to predict the expression levels of genes they regulate. Transcription factors may not bind to a particular DNA site unless (or except when) they have bound a specific small molecule or undergone some post-translational modification. Given only microarray and genomic sequence data, we cannot directly measure theses states. However, we can think of these states as latent variables and represent them using hidden nodes in our Bayesian networks.

A second significant innovation in our approach is that we use known and predicted transcription start sites to determine whether a given transcription factor is more likely to act as an activator or a repressor for a given gene. We refer to this distinction as the *role* of a regulator with respect to a gene. To do this, we take advantage of a detailed probabilistic model of transcription units that we have developed in previous work [15]. Depending on the relative positions of a transcription factor binding site and a known or predicted promoter, we get an indication as to whether the transcription factor is acting as an activator or a repressor in a given case. We use this information to guide the initialization of parameters associated with the hidden nodes discussed above.

2 Approach

In this section, we first describe how we use Bayesian networks to represent various aspects of transcriptional regulation networks. A Bayesian network consists of two components: a qualitative one (the *structure*) in the form of a directed acyclic graph whose nodes correspond to the random variables, and a quantitative component consisting of a set of *conditional probability distributions* (CPDs). We then discuss how we learn both the structure and the parameters of our networks.

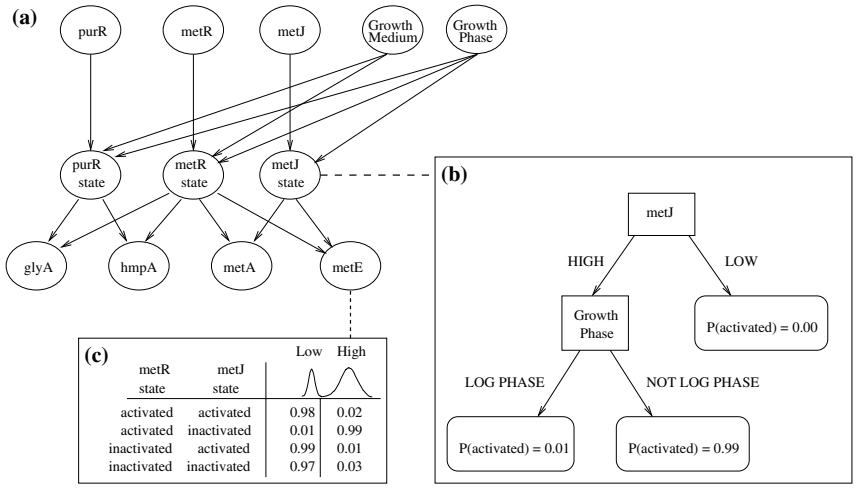


Fig. 1. (a) An example network with three regulators (purR, metR, and metJ), two cellular condition variables (Growth Medium and Growth Phase), and four regulated gene variables (glyA, hmpA, metA, and metE). (b) A possible CPD-Tree for the hidden node metJ-state. (c) A possible CPT for the regulatee node, metE, whose expression states are defined by a two-Gaussian mixture.

2.1 Network Architecture

Our models contain four distinct types of variables on three distinct levels. An example is shown in Fig. 1 (a). On the top level, there are nodes that represent the expression of regulators (genes whose products regulate other genes), and also nodes that represent the cellular conditions under which various gene-expression measurements were collected. On the bottom level, there are nodes representing the expression of genes known or predicted to be influenced by the regulators on the top level (we refer to these genes as *regulatees*). On the middle level, there are hidden nodes, one paired with each regulator node. These hidden nodes represent the "states" of the corresponding regulators. The parents of each hidden node are selected from a set of candidates that includes both the corresponding regulator expression node and the cellular condition nodes. The parents of each regulatee node are the hidden nodes corresponding to the regulators known or predicted to have a regulatory influence over that gene.

Each hidden node has two possible values, which can be interpreted as "activated" and "inactivated." As discussed in Section 1, regulators, such as transcription factors, are often activated or inactivated by *effectors*, such as small molecules. Although we do not have data that will allow us to directly detect the effectors for specific regulators, the network-learning algorithm can use cellular condition nodes as surrogates for these effectors. Consider for example, the transcription factor CAP which is activated by the small molecule cAMP. Our data do not contain cAMP measurements, but our method may learn that the absence of glucose in the growth medium is predictive of when CAP is activated. Thus the method has learned that glucose absence is a good surrogate for cAMP.

2.2 Representing Gene Expression States

We represent the expression levels of genes using a Gaussian mixture model [16]. We assume that most genes have multimodal expression-level distributions, with each mode corresponding to an "expression state" of the gene. Each Gaussian in the mixture represents the range of expression values for one state of the gene. Fig. 2 shows the mixture model inferred by our method for the metE gene.

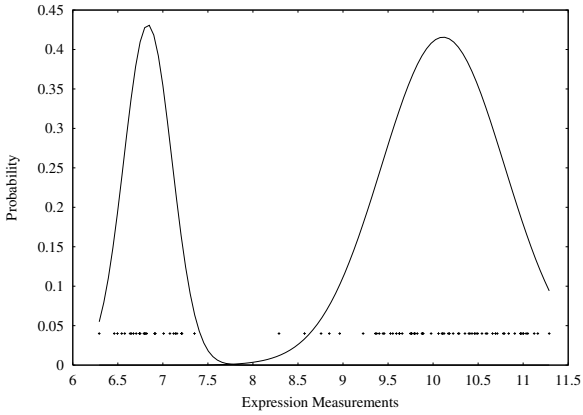


Fig. 2. Expression measurements for the gene metE, and a two-Gaussian mixture which describes its states. Expression measurements are plotted near the x-axis.

We use cross validation to choose the number of Gaussians in each mixture. Let $\mathbf{x}$ be the set of expression values for a given gene. For each fold, i, of cross validation, we divide $\mathbf{x}$ into two subsets, training data $\mathbf{x}_i^{'}$ and held-aside data $\mathbf{x}_i^{''}$. We use an expectation-maximization (EM) algorithm to optimize the parameters, Φ_i (the mean, variance and weight of each Gaussian in the mixture). However, we constrain the parameters so that the Gaussians are sufficiently far apart to ensure they each cover a separate range of expression values. Specifically, each Gaussian must have the highest probability density at each expression level within two standard deviations of its mean. The EM algorithm will attempt to optimize Φ_i as $argmax_{\Phi_i} P(\mathbf{x}_i^{'}|\Phi_i)$, but only a local optimum is guaranteed. We then use the held-aside data to calculate the score for this fold as $score_i = P(\mathbf{x}_i^{''}|\Phi_i)$. We repeat this process for one, two, or three Gaussians in a mixture. We choose the number of Gaussians associated with the highest score, $\sum_i score_i$, provided that a pairwise, two-tailed *t*-test determines the improvement to be statistically significant over the scores obtained from mixtures with fewer Gaussians. Once the number of Gaussians has been settled, we use the EM algorithm to optimize the final parameters Φ as $argmax_{\Phi} P(\mathbf{x}|\Phi)$, and consider each individual Gaussian to represent an expression state for that gene. If our method selects a mixture model with only one Gaussian for a given gene (*i.e.* there is only one expression state of this gene in the training set), then the gene is not included in the network model. In our experiments, this is the case for roughly half of the genes considered. About 90% of the remaining genes have two expression states, and 10% have three.

In order for our network to learn from data, we must express these data in terms of values of the variables in our network. For the regulator and regulatee nodes, these are expression states of the genes, but our evidence consists of expression measurements. Since the expression states are described by Gaussians over the expression values, it is straightforward to calculate a probability distribution over the expression states for each gene, given an expression measurement. These probability distributions are what we use as the evidence concerning the states of genes. Because of the constraints on the distance between Gaussians and the fact that they are positioned using the data, most such distributions will clearly favor a single Gaussian (gene expression state) over any other.

2.3 Representing Conditional Probability Distributions

As shown in Fig. 1 (b), we use trees to represent the conditional probability distributions for the hidden nodes in our networks. Each tree represents the distribution over values (*i.e.* states) of the corresponding hidden node, conditioned on the values of the node's parents. Recall that the candidate parents for each hidden node consist of the regulator expression level as well as the complete set of cellular condition nodes. We use trees to represent these CPDs for three key reasons. First, we assume that only a few of the candidate parents are relevant to modeling the regulator state, and thus we want the model to be able to select a small number of parents from a fairly large candidate pool. Second, trees provide descriptions of regulator states that are readily comprehensible and thus they can lend insight into the mechanisms which determine a regulator's behavior. Third, the trees can account for cases in which there is context-sensitive independence in determining a hidden node's probability distribution. In the sample tree shown in Fig. 1 (b), note how "Growth Phase" is only relevant if the regulator metJ has expression "HIGH." Note also how "Growth Medium" is not chosen as a parent at all.

Using our current data set, each regulatee in the network has a relatively small number of parents (between one and four), and we expect each parent to be relevant, so we use conventional *conditional probability tables* (CPTs) for the regulatee nodes, as shown in Figure 1 (c). The CPT for each regulatee node represents the distribution over the possible expression states of the particular gene, conditioned on the possible states of its parents.

2.4 Learning Network Parameters

Recall that each hidden variable is binary, and we refer to the possible values as "activated" and "inactivated". Since these states are unobserved, we cannot calculate the CPDs for the hidden and regulatee nodes directly. Instead, we set their parameters with an EM algorithm wherein we refine the CPDs iteratively until they converge to a local optimum which is consistent with the observed training data. Let $s^e_{R,i}$ represent the observed expression state of the ith regulator in experiment e, and let $s^e_{r,i}$ represent the observed expression state of the ith regulatee. Similarly, let $s^e_{c,i}$ denote the state of the ith cellular-condition variable in experiment e. We use Θ to represent all of the parameters of a Bayesian network, including the parameters (and structure) of each CPD tree

and of each CPT. The EM algorithm adjusts the parameters trying to maximize the joint probability of the expression states across all experiments, given the cellular conditions:

$$\hat{\Theta} = \arg\max_{\Theta} \prod_{e} P(s^e_{R,1}, ..., s^e_{R,m},\ s^e_{r,1}, ..., s^e_{r,n} \mid s^e_{c,1}, ...s^e_{c,k},\ \Theta).$$

Here m is the number of regulators in the network, n is the number of regulatees, and k is the number of cellular-condition variables. We assume that we are always given the values of the cellular-condition variables, and thus our model represents the probability of the expression states conditioned on these values. The details of the E-step and M-step in this context are as follows.

E-Step: Let $S^e_{h,i}$ represent the (unobserved) state of hidden node i in experiment e (we use uppercase S to denote random variables in the Bayesian network and lowercase s to denote particular values the variables can take). In the E-Step, we compute the expected distribution over values of $S^e_{h,i}$ for each e and each i, given the observed expression states of the regulators and regulatees, and the observed cellular conditions:

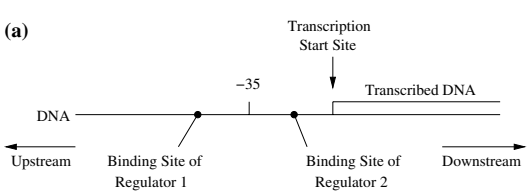
$$P(S^e_{h,i} | s^e_{R,1}, ..., s^e_{R,m},\ s^e_{r,1}, ..., s^e_{r,n},\ s^e_{c,1}, ...s^e_{c,k},\ \Theta).$$

This is computed as a Bayesian network query using *variable elimination* [17]. If the probability that a certain $S^e_{h,i}$ takes on the value "activated" is 0.7, then $S^e_{h,i}$ is treated as being 70% an "activated" data value and 30% an "inactivated" data value.

M-Step: Once these expected values are calculated, we use our now complete set of data to recalculate the network parameters. Let $\Theta_{h,i}$ refer to the CPD parameters for the ith hidden node and let $\Theta_{r,i}$ refer to the CPT parameters for the ith regulatee node. In the M-step, we attempt to maximize:

$$\prod_{e} P(s^e_{R,1}, ..., s^e_{R,m},\ s^e_{r,1}, ..., s^e_{r,n},\ s^e_{h,1}, ..., s^e_{h,k} \mid s^e_{c,1}, ...s^e_{c,k},\ \Theta).$$

where $s^e_{h,1}, ..., s^e_{h,k}$ denotes the expectations for the hidden nodes calculated in the E-step. Each CPD-tree for hidden variable i, represented by $\Theta_{h,i}$, is re-grown by selecting a variable to split on (regulator expression state or cellular condition variable) which separates the set of expected values for the hidden variable, $\{s^e_{h,i}\}$, over all experiments e, into two subsets, $S^{\mathbf{left}}_{h,i}$ and $S^{\mathbf{right}}_{h,i}$, such that the classification error when considering the values in $S^{\mathbf{left}}_{h,i}$ and $S^{\mathbf{right}}_{h,i}$ to be either "activated" or "inactivated" is minimized. This process recurs on both subsets until no candidate split will further separate the data. A probability distribution over $S_{h,i}$ is then calculated for each leaf in the tree based on $\{s^e_{h,i}\}$ for the experiments e contained in that leaf's subset. Subtrees with a common ancestor that have nearly the same probability distributions over $S_{h,i}$ are pruned using an approach based on *minimum description length* (MDL), similar to one developed by Mehta *et al.* [18]. These pruned trees may not maximize the probability of the hidden states exactly, as do their unpruned counterparts. In practice, however, pruning speeds up convergence without sacrificing accuracy. Each regulatee CPT, $\Theta_{r,i}$, is also recalculated in the standard way during the M-step using $\{s^e_{r,i}\}$ and $\{s^e_{\pi(i)}\}$, where $\pi(i)$ is the set of parent nodes of regulatee node i.



(b)

Regulator 1 (Activator)	Regulator 2 (Repressor)	Low	High
activated	activated	0.7	0.3
activated	inactivated	0.2	0.8
inactivated	activated	0.9	0.1
inactivated	inactivated	0.5	0.5

Fig. 3. (a) An example promoter configuration with one regulator binding site on each side of the -35 position. (b) A CPT for the gene that has been initialized based on the configuration of these two regulator binding sites.

2.5 Initializing Network Parameters

The EM algorithm described above will converge to a local optimum. In order to guide the network to converge to a good solution, we initialize the CPT for each regulatee based on prior knowledge about the roles of each its regulators. Specifically, for each regulatee we consider the relative location of the transcription start site, which is either known or predicted [15], and the binding sites for the regulators, which are also either known or predicted. We tentatively designate as *activators* those regulators that bind strictly upstream of the regulatee's promoter (which is estimated to extend 35 nucleotides upstream of its transcription start site), and we tentatively designate all other regulators as *repressors*. In the CPT for each regulatee, we assign a higher probability to the *highest* expression state when the putative activators are in the "activated" state, and we assign a higher probability to the *lowest* expression state when the putative repressors are in the "activated" state. We put more weight on this effect for repressors, which are believed to have a more stringent control on expression, and we put more weight on this effect when the regulatee's transcription start site is known than when it is only predicted. This initialization process is illustrated in Fig. 3.

3 Empirical Evaluation

In this section we present experiments designed to evaluate our Bayesian networks that (a) use hidden nodes to represent regulator states, (b) attempt to compensate for missing regulators, and (c) have the parameters associated with these nodes initialized to reflect their predicted roles (activator or repressor) with respect to individual genes.

3.1 Experimental Data and Methodology

We initialize the topology of our networks using 64 known and predicted *E. coli* regulators and their 296 known and predicted regulatees. The known instances are from TRANSFAC [19] and EcoCyc [20], and the predicted instances are based on binding sites predicted from cross-species comparison [21]. Our gene-expression data comes from a set of 90 Affymetrix microarray experiments [22]. Each array is annotated with experimental conditions, and the data are normalized using the *robust multiarray averaging* (RMA) technique [23].

We divide the microarray experiments into sets for which all of the annotated cellular conditions are identical (we call these replicate sets). From the original 90 experiments, there are 42 of these sets. The largest contains five experiments, and the others are copied so that each replicate set contains exactly five experiments. To assess model accuracy, we use leave-one-out cross-validation on each replicate set. That is, we hold one set of five identical experiments aside, train the model on the remaining experiments, and then evaluate the network on the held-aside data. For each testing example, we provide the network with expression levels of the regulators and the values of the cellular conditions, and then calculate a probability distribution over the possible expression states for each regulatee.

We evaluate the accuracy of our models using three measures. First, we calculate *classification error* as the extent to which the network predicts the incorrect expression states for each regulatee and experiment. Instead of calculating this error using "hard predictions" of the expression state of each regulatee, we take into account our uncertainty in each predicted expression state as well as the uncertainty in the discretization of each gene. In particular, we calculate classification error as follows:

$$\text{class error} = 100\% \times \left(1 - \frac{1}{E}\frac{1}{n}\sum_{e=1}^{E}\sum_{i=1}^{n}\sum_{d} P_{\Theta}(S_{r,i}^{e} = d) P_{\Phi}(S_{r,i}^{e} = d | x_{r,i}^{e})\right).$$

Here $P_{\Theta}(S_{r,i}^{e} = d)$ is shorthand for $P(S_{r,i}^{e} = d | s_{R,1}^{e}, ..., s_{R,m}^{e}, s_{c,1}^{e}, ..., s_{c,k}^{e}, \Theta)$ which is the probability, as predicted by the Bayesian network, that the ith regulatee is in state d for experiment e, given the expression values of the regulators and the cellular conditions. $P_{\Phi}(S_{r,i}^{e} = d | x_{r,i}^{e})$ represents the probability that the regulatee is in state d given expression measurement $x_{r,i}^{e}$, according to the Gaussian mixture model for this gene. Second, we compute the *average squared error*, where the error is the difference between the actual expression value and the means of the Gaussians representing each expression state, weighted by the predicted distribution over these states:

$$\text{ASE} = \frac{1}{E}\frac{1}{n}\sum_{e=1}^{E}\sum_{i=1}^{n}\sum_{d} P_{\Theta}(S_{r,i}^{e} = d)(\mu_{r,i,d} - x_{r,i}^{e})^2.$$

Here, $\mu_{r,i,d}$ is the mean of the Gaussian for state d in the mixture model for the ith regulatee. Third, we calculate the joint *log probability* of all test-set expression values, again taking into account our uncertainty in each predicted expression state and in discretization of each gene:

$$\text{log probability} = \sum_{e=1}^{E}\sum_{i=1}^{n}\log\left(\sum_{d} P_{\Theta}(S_{r,i}^{e} = d) P_{\Phi}(S_{r,i}^{e} = d | x_{r,i}^{e})\right).$$

We apply pairwise, two-tailed *t*-tests to test the statistical significance of differences between methods.

3.2 Experiment 1: The Value of Representing Regulator States

In order to test the value of including hidden nodes that represent regulator states, we compare against two baselines, examples of which are shown in Fig. 4. The first baseline employs Bayesian networks that have nodes representing the expression levels of

Fig. 4. Examples of the two baseline networks used in Experiment 1. These are the counterpart baselines for the network shown in Fig. 1.

regulators and regulatees, but which do not have hidden nodes representing regulator states. The second baseline augments the first by also incorporating cellular condition nodes, but it too does not have hidden nodes.

Table 1 shows the totals over all test folds for all three measures. For each of these measures, our models are more accurate than the baseline models which do not have hidden or cellular condition nodes. The differences are statistically significant at a confidence of over 99% for classification error and average squared error. The difference in overall probability is not statistically significant at a confidence of more than 95%. The baseline networks that include cellular condition nodes provide slightly more accurate predictions than our models with hidden nodes. However, we argue that these baseline models have a significant limitation in they do not provide a very mechanistic description of regulatee expression. That is, they do not directly represent the states of regulators and how these states govern the expression of regulatees. Thus, they have less explanatory power than our models.

Table 1. Predictive accuracy for the models with hidden nodes and the two baselines.

Model Variant	Classification Error	Average Squared Error	Log Probability
Full Model	16.59%	0.59	-12,066
Without Hidden Nodes	12.42	0.51	-12,193
Without Hidden or Cellular Condition Nodes	22.16	0.75	-13,363

Note also that our models show an improvement in overall log probability when compared to each of these baselines. Since the overall probability is a product of regulatee expression probabilities, an incorrect prediction with a probability very close to zero can have an unbounded effect on the final measurement. We hypothesize that our models make fewer of these extreme probability predictions because the regulatees are constrained by binary-valued parents.

3.3 Experiment 2: Discovering Missing Regulators

It is certainly the case that some relevant regulators are not represented in our networks. In this section, we consider a simple approach that dynamically adds hidden nodes to the networks. This approach tries to identify sets of regulatees for which a network makes incorrect predictions on many of the same training examples. After first training

a network using the EM approach described earlier, we recursively cluster regulatees that share at least 50% of training examples incorrectly predicted by either regulatee (or cluster). A new hidden node is created for each cluster and this node becomes a parent of the regulatees in the cluster. The network is then re-trained using the EM approach. This procedure may be iterated a number of times.

Table 2 shows the resulting accuracy with up to three iterations of this procedure. Each iteration decreases the classification error of the models, and each decrease is statistically significant at a confidence above 97%. For the average squared error measure, only the decrease in error from the original model to any of the other three is statistically significant (at a confidence of 95% or greater). The application of this procedure improves the overall probability, but it does not continue to increase over multiple iterations. The differences in overall probability are not statistically significant.

Table 2. Predictive accuracy for the models with added hidden nodes.

Iterations	Classification Error	Average Squared Error	Log Probability
0 (Original model)	16.59%	0.59	-12,066
1	14.23	0.53	-11,586
2	13.65	0.51	-11,987
3	13.34	0.51	-12,004

Notice that this technique improves all three of our measurements, and that the classification error approaches that of the baseline without hidden nodes shown in Table 1, yet still provides models that explain relevant regulatory mechanisms.

3.4 Experiment 3: The Value of Initializing Regulator Roles

Recall that, before we train our network, we initialize the CPTs of the regulatee nodes based on the relative positions of known and predicted regulator binding sites and known or predicted promoters. We hypothesize that this initialization process will guide the EM algorithm toward a better solution. To evaluate the effectiveness of this technique, we compare the accuracy of our approach to a variant in which we initialize the parameters randomly. We also apply the technique of adding hidden nodes as described in the previous experiment because this increases the parameter space, and, one would expect, the importance of a good initialization.

The results of this experiment are shown in Table 3. Our initialization technique improved both the classification error and the average squared error, and the improvement

Table 3. Predictive accuracy for models with promoter-based parameter initialization and random initialization.

Initialization	Classification Error	Average Squared Error	Log Probability
Using Promoter Data	13.34%	0.51	-12,004
Random	14.19	0.54	-11,893

is statistically significant at a confidence above 96% for both measures. The technique did not improve the overall probability, however the decrease is not statistically significant. Repeating the experiment using random initialization many times on the same fold of cross validation, we estimate the standard deviation of the classification error at about 0.63% and of the average squared error at about 0.028. Notice that our initialization technique is an improvement over random initialization of at least this much. The standard deviation for log probability is estimated at about 25.53.

4 Conclusion

In addressing the problem of inferring models of transcriptional regulation, we have developed an approach that is able to learn to represent the states of regulators (*i.e.* whether a transcription factor is activated or not) as well as their roles (*i.e.* whether a transcription factor acts as an activator or repressor for a given gene). We have empirically evaluated our approach using gene-expression and genomic-sequence data for *E. coli* K-12. Our experiments show that both of these aspects of our approach result in models with a high level of predictive accuracy.

There are a number of extensions to our approach that we plan to investigate in future research. First, in keeping with our goal of learning more mechanistic representations, we plan to extend our models to account for additional types of regulatory mechanisms, such as riboswitches [24]. Second, we plan to adjust our approach so that we can relax some of the simplifying assumptions we have made in our initial work, such as the assumptions that genes have only one transcription start site, regulators have only one binding site in a given promoter region, and genes have distinct modes in their expression-level distributions. Third, we plan to extend our method so that the process of adding candidate regulators to a network involves looking for evidence of these regulators (*e.g.* transcription factor binding sites) in the genomic sequence. All of these proposed extensions are aimed at advancing the theme of learning models that exploit multiple sources of data, and attempt to provide mechanistic descriptions of regulatory relationships.

Acknowledgments

This research was supported in part by NLM training grant 5T15LM005359, NSF grant IIS-0093016, and NIH grant R01-LM07050-01. The authors would like to thank Joseph Bockhorst and Jesse Davis for helpful comments on an earlier draft of this paper.

References

1. de Jong, H.: Modeling and simulation of genetic regulatory systems: A literature review. Journal of Computational Biology **9** (2002) 67–103
2. King, R., Whelan, K., Jones, F., Reiser, P., Bryant, C., Muggleton, S., Kell, D., Oliver, S.: Functional genomic hypothesis generation and experimentation by a robot scientist. Nature **427** (2004) 247–252

3. Blattner, F.R., Plunkett, G., Bloch, C.A., Perna, N.T., Burland, V., Riley, M., Collado-Vides, J., Glasner, J.D., Rode, C.K., Mayhew, G.F., Gregor, J., Davis, N.W., Kirkpatrick, H.A., Goeden, M.A., Rose, D.J., Mau, B., Shao, Y.: The complete genome sequence of Escherichia coli K-12. Science **277** (1997) 1453–1474
4. Friedman, N., Linial, M., Pe'er, D.: Using Bayesian networks to analyze expression data. Journal of Computational Biology **7** (2000) 601–620
5. Hartemink, A., Gifford, D., Jaakkola, T., Young, R.: Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. In: Proceedings of the Fifth Pacific Symposium on Biocomputing, Kohala Coast, HI, World Scientific Press (2001) 422–433
6. Hartemink, A., Gifford, D., Jaakkola, T., Young, R.: Combining location and expression data for principled discovery of genetic regulatory networks. In: Proceedings of the Fifth Pacific Symposium on Biocomputing, Lihue, HI, World Scientific Press (2002) 437–449
7. Pe'er, D., Regev, A., Elidan, G., Friedman, N.: Inferring subnetworks from peturbed expression profiles. Bioinformatics **17** (2001) 215S–224S
8. Segal, E., Yelensky, R., Koller, D.: Genome-wide discovery of transcriptional modules from DNA sequence and gene expression. Bioinformatics **19** (2003) i273–i282
9. Segal, E., Shapira, M., Regev, A., Pe'er, D., Botstein, D., Koller, D., Friedman, N.: Module networks: Identifying regulatory modules and their condition-specific regulators from gene expression data. Nature Genetics **34** (2003) 166–176
10. Tamada, Y., Kim, S., Bannai, H., Imoto, S., Tashiro, K., Kuhara, S., Miyano, S.: Estimating gene networks from gene expression data by combining Bayesian network model with promoter element detection. Bioinformatics **19** (2003) ii227–ii236
11. Yoo, C., Cooper, G.: Discovery of gene-regulation pathways using local causal search. In: Proceedings of the Annual Fall Symposium of the American Medical Informatics Association. (2002) 914–918
12. Yoo, C., Thorsson, V., Cooper, G.: Discovery of causal relationships in a gene-regulation pathway from a mixture of experimental and observational DNA microarray data. In: Proceedings of the Fifth Pacific Symposium on Biocomputing, Lihue, HI, World Scientific Press (2002) 498–509
13. Ong, I., Glasner, J., Page, D.: Modelling regulatory pathways in E. coli from time series expression profiles. Bioinformatics **18** (2002) S241–S248
14. Pearl, J.: Probabalistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Mateo, CA (1988)
15. Bockhorst, J., Qiu, Y., Glasner, J., Liu, M., Blattner, F., Craven, M.: Predicting bacterial transcription units using sequence and expression data. Bioinformatics **19** (2003) i34–i43
16. Xing, E., Jordan, M., Karp, R.: Feature selection for high-dimensional genomic microarray data. In: Proceedings of the Eighteenth International Conference on Machine Learning. (2001)
17. D'Ambrosio, B.: Inference in Bayesian networks. AI Magazine **20** (1999) 21–36
18. Mehta, M., Rissanen, J., Agrawal, R.: MDL-based decision tree pruning. In: Proceedings of the First International Conference on Knowledge Discovery and Data Mining, AAAI Press (1995) 216–221
19. Wingender, E., Chen, X., Fricke, E., Geffers, R., Hehl, R., Liebich, I., Krull, M., Matys, V., Michael, H., Ohnhäuser, R., Prüß, M., Schacherer, F., Thiele, S., Urbach, S.: The TRANSFAC system on gene expression regulation. Nucleic Acids Research **29** (2001) 281–283
20. Karp, P., Riley, M., Saier, M., Paulsen, I., Collado-Vides, J., Paley, S., Pellegrini-Toole, A., Bonavides, C., Gama-Castro, S.: The EcoCyc database. Nucleic Acids Research **30** (2002) 56–58

21. McCue, L.A., Thompson, W., Carmack, C.S., Lawrence, C.: Factors influencing the identification or transcription factor binding sites by cross-species comparison. Genome Research **12** (2002) 1523–1532
22. Glasner, J., Liss, P., III, G.P., Darling, A., Prasad, T., Rusch, M., Byrnes, A., Gilson, M., Biehl, B., Blattner, F., Perna, N.: ASAP, a systematic annotation package for community analysis of genomes. Nucleic Acids Research **31** (2003) 147–151
23. Irizarry, R., Hobbs, B., Collin, F., Beazer-Barclay, Y., Antonellis, K., Scherf, U., Speed, T.: Exploration, normalization, and summaries of high density oligonucleotide array probe level data. Biostatistics **4** (2002) 249–264
24. Nudler, E., Mironov, A.: The riboswitch control of bacterial metabolism. Trends in Biochemical Sciences **29** (2004) 11–17