
Predicting the Time Course of Gene Expression in a Relational Learning Framework

Moritz Take
Stefan Kramer

TAKE@INFORMATIK.UNI-FREIBURG.DE
KRAMER@IN.TUM.DE

Technische Universität München, Institut für Informatik/I12, Boltzmannstr. 3, D-85748 Garching b. München, Germany

Abstract

In gene regulation prediction, the task is to predict the regulation of a gene under specific condition from binding site information, the state of regulators, and additional information. Previous work showed that relational learning techniques (in particular boosted first-order decision trees) are particularly useful for this kind of task, as (1) they can be adapted to complex datasets flexibly, and (2) they are able to integrate different types of data from diverse sources. This extended abstract describes our first steps towards refining this model for the prediction of the time course of gene expression.

1. Introduction

One of the central goals in computational and systems biology is to understand the mechanisms of gene transcriptional regulation on a system-wide level. The efforts are often based on high-throughput genomic data of model organisms such as *S. cerevisiae*. The goal of this work is to learn a predictive model of gene regulation. Our starting point is the model of [Middendorf et al., 2004], where the presence of transcription factor binding sites (motifs) in the gene's regulatory region and the expression levels of regulators (e.g., transcription factors or protein kinases) are used to predict gene regulation. In a recent paper, [Fröhler & Kramer, 2007] showed how an ILP approach can be used to integrate information from various sources (e.g., functional categorizations and protein-protein interactions) to improve the performance of this model substantially. This extended ab-

stract sketches a further refinement of the model to take into account the time course of gene expression.

2. Data and Results

This work is based on the *S. cerevisiae* data by [Gasch et al., 2000]. As stated above, the goal is to learn a prediction model for the regulatory response of genes under different environmental conditions. In its most basic version, we have three different predicates, `gene(GeneId, CondId, Level)`, `hasTFBS(GeneId, BsId)`, and `expression(RegId, CondId, RegLevel)`. `gene(GeneId, CondId, Level)` gives the expression level for each gene under a specific experimental condition. As in the study by Middendorf *et al.*, gene expression is discretized and mapped onto three distinct values +1 (up-regulated), 0, and -1 (down-regulated). The learning task is to predict the expression level for a given gene under a certain condition, given some background information (see below). The dataset contains information about 1,411 genes, 173 experimental conditions, and 54,183 instances. The relation `hasTFBS(GeneId, BsId)` holds information about the binding sites for each gene found in its regulatory region, taken from the TRANSFAC database. The regulatory region of each gene is represented as a vector containing the binding sites for this gene. Given this information, we can identify subsets of genes according to binding motifs, which are assumed to share regulation behavior. The relation `expression(RegId, CondId, RegLevel)` gives the expression levels of regulators (e.g., transcription factors or protein kinases) under certain experimental conditions. In our representation, the regulators are a subset of all genes. In the experiments described in this paper, we used a set of 53 different regulators. The goal of the application is to predict the expression level for a given gene under a given condition (predicate `gene(GeneId, CondId, Level)`), in terms of the predicates `hasTFBS(GeneId,`

Appearing in working notes of the ICML07 workshop on the *Induction of Process Models*, June 24, Corvallis, OR, USA, 2007. Copyright 2007 by the author(s)/owner(s).

```

predicted.willBeExpressed(A,B,[-1]) :- isExpressed(YIR017C,B,-1), !.
% 319.0/372.0=0.85752688172043
willBeExpressed(A,B,[-1]) :- isExpressed(YNL216W,B,1), !.
% 86.0/113.0=0.761061946902655
willBeExpressed(A,B,[-1]) :- isExpressed(YDR277C,B,-1), !.
% 71.0/101.0=0.702970297029703
willBeExpressed(A,B,[-1]) :- isExpressed(YDR028C,B,-1), !.
% 86.0/125.0=0.688
willBeExpressed(A,B,[-1]) :- isExpressed(YBR240C,B,-1), !.
% 95.0/148.0=0.641891891891892
willBeExpressed(A,B,[-1]) :- isExpressed(YBR279W,B,-1), !.
% 85.0/135.0=0.62962962962963
willBeExpressed(A,B,[-1]) :- isExpressed(YGL035C,B,-1), !.
% 91.0/146.0=0.623287671232877
willBeExpressed(A,B,[-1]) :- isExpressed(YOR090C,B,-1),belongsToFuncat(A,10.03), !.
% 82.0/110.0=0.7454545454545454
willBeExpressed(A,B,[1]) :- isExpressed(YOR090C,B,-1), !.
% 112.0/201.0=0.557213930348259
willBeExpressed(A,B,[-1]) :- isExpressed(YJL164C,B,1),belongsToFuncat(A,10.01.03), !.
% 102.0/118.0=0.864406779661017
willBeExpressed(A,B,[-1]) :- isExpressed(YJL164C,B,1),belongsToFuncat(A,41), !.
% 87.0/117.0=0.743589743589744
willBeExpressed(A,B,[-1]) :- isExpressed(YJL164C,B,1),belongsToFuncat(A,18.01), !.
% 74.0/107.0=0.691588785046729
willBeExpressed(A,B,[-1]) :- isExpressed(YJL164C,B,1),belongsToFuncat(A,10.03.01.01.09), !.
% 79.0/114.0=0.692982456140351
willBeExpressed(A,B,[-1]) :- isExpressed(YJL164C,B,1),hasTFBS(A,Y$Y30_01), !.
% 74.0/116.0=0.637931034482759
willBeExpressed(A,B,[-1]) :- isExpressed(YJL164C,B,1),belongsToFuncat(A,01.03), !.
% 91.0/156.0=0.5833333333333333
willBeExpressed(A,B,[1]) :- isExpressed(YJL164C,B,1),hasTFBS(A,Y$BAP2_01), !.
% 127.0/129.0=0.984496124031008
willBeExpressed(A,B,[-1]) :- isExpressed(YJL164C,B,1),belongsToFuncat(A,34.11.03.13), !.
% 70.0/112.0=0.625
willBeExpressed(A,B,[1]) :- isExpressed(YJL164C,B,1),belongsToFuncat(A,16.19.01), !.
% 113.0/115.0=0.982608695652174
...
    
```

Table 1. Tilde decision tree predicting the next state of a regulator depending on previous states and other information

BsId) and expression(RegId, CondId, RegLevel). Additionally to the two basic predicates, we enrich the representation with further predicates:

- `assignedToFuncat(GeneId, FunCatId)`, containing all annotated FunCat [Ruepp et al., 2004] terms for gene `GeneId` and all parent FunCat terms of this term,
- `hasPPI(GeneId, GeneId)`, containing binary protein-protein interactions from the MIPS database [Mewes et al., 1997] for each gene whose expression state is to be predicted, and `hasTFPPI(RegId, RegId)` containing binary protein-protein interactions of the regulators themselves.

In our experiments, we tested boosted Tilde decision trees [Blockeel & De Raedt, 1998] and alternating decision trees [Freund & Mason, 1999] on the data. To see the value of different types of data, we included results with/without FunCat terms, with/without protein-protein interactions, and with/without information on transcription factor binding sites. The experiments showed that all of those data contribute to some extent to the overall performance. The best

overall predictive accuracy using the Middendorf representation was 91.2 %, with a different set of transcription factors it is even possible to predict up- or down-regulation with an accuracy of 92.8 %. Moreover, it is possible to extract the functional categories affected by the experimental conditions, together with important transcription factor binding sites and transcription factors for these categories from the induced trees. For a detailed description of the data, and a qualitative discussion of results, we have to refer to the long paper [Fröhler & Kramer, 2007].

3. Time Series: Data and Preliminary Results

The classification model developed above is static in the sense that the time course of gene expression is not taken into account explicitly. However, as the regulators are a subset of all genes, the vector of the regulators' states can be both input and output of the model. Therefore, one might consider applying it iteratively to obtain a sequence of successive gene expression states. In this way, a static model would be applied in a way that a sequence of predicted states is obtained. The obvious problem with this approach is

that the classifier only captures a static association of expression patterns. For a more adequate representation, the current state of regulators has to be provided as input for the prediction of subsequent states.

To explicitly represent the time dimension, we have to extract information from the Gasch data as follows: From 173 different conditions, we have $n = 79$ conditions being part of a time series. In total, we find $k = 13$ different time series in the database. Therefore, we have $(n - 1) - (k - 1) = 66$ different state transitions in the Gasch data. The time intervals between state transitions vary largely, from 5 minutes to 240 minutes, with the majority of 58 between 5 and 30 minutes. From the full list of 475 regulators, gene expression data is available for 448. Additionally, 21,128 ground facts of background knowledge are provided as described above.

For the formulation of the new learning task, a new relation `willBeExpressed(GeneId, ConditionID, Level)` is introduced, linking the previous condition of a gene to its expression in the next state. Applying Tilde to this data again, we obtain 73 % predictive accuracy and 0.8 AUC in ten-fold cross-validation. An excerpt from the first-order decision tree learned by Tilde can be found in Table 1.

4. Discussion and Future Work

We extended our recently proposed model [Fröhler & Kramer, 2007] to explicitly represent time and state transitions. The first remaining problem to be addressed is the varying length of the time intervals. Second, the problem formulation has to be modified to include unchanged states. Third, a closer look at subsequent states of regulators in the time series showed that they only change in 5-6 % of the cases (considering two states only). To gain a better understanding of the structure of the data, we visualized them in a plot (see Figure 1) depicting over-expressed states in red, under-expressed states in green, and unchanged states in black. The plot suggests that the states are changing sufficiently often, such that the prediction of state changes should be, at least theoretically, possible.

The approach described in this abstract is similar to the one by [Ong et al., 2007], where logical clauses representing state transitions are learned from this dataset. The authors focus on 19 genes from the DNA damage checkpoint pathway and predict the relative change of gene expression. If a relative change of +0.3 is observed, the gene is considered as upregulated, vice versa, a relative change of -0.3 indicates downregula-

tion compared to the previous state. A similar representation might circumvent some of the problems addressed in this abstract.

References

- [Blockeel & De Raedt, 1998] Blockeel, H., Raedt, L.D.: Top-down induction of first-order logical decision trees. *Artificial Intelligence* **101**(1-2) (1998) 285–297.
- [Freund & Mason, 1999] Freund, Y., Mason, L.: The Alternating Decision Tree Learning Algorithm. *Proc. of the 16th International Conference on Machine Learning*, Morgan Kaufmann (1999).
- [Fröhler & Kramer, 2007] Fröhler, S., Kramer, S.: Inductive logic programming for gene regulation prediction. *Machine Learning*, to appear (2007).
- [Gasch et al., 2000] Gasch, A.P., Spellman, P.T., Kao, C.M., Carmel-Harel, O., Eisen, M.B., Storz, G., Botstein, D., Brown, P.O.: Genomic Expression Programs in the Response of Yeast Cells to Environmental Changes. *Mol.Biol.Cell* **11**(12) (2000) 4241–4257.
- [Mewes et al., 1997] Mewes, H., Albermann, K., Heumann, K., Liebl, S., Pfeiffer, F.: MIPS: a database for protein sequences, homology data and yeast genome information. *Nucl. Acids Res.* **25**(1) (1997) 28–30.
- [Middendorf et al., 2004] Middendorf, M., Kundaje, A., Wiggins, C., Freund, Y., Leslie, C.: Predicting genetic regulatory response using classification. *Bioinformatics* **20**(suppl.1) (2004) i232–240.
- [Ong et al., 2007] Ong, I.M., Topper, S.E., Page, D., Santos Costa, V.: Inferring Regulatory Networks from Time Series Expression Data and Relational Data via Inductive Logic Programming. *Proc. of the 16th International Conference on Inductive Logic Programming*, Springer (2007).
- [Ruepp et al., 2004] Ruepp, A., Zollner, A., Maier, D., Albermann, K., Hani, J., Mokrejs, M., Tetko, I., Guldener, U., Mannhaupt, G., Munsterkotter, M., Mewes, H.W.: The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucl. Acids Res.* **32**(18) (2004) 5539–5545.



Figure 1. Visualization of time series of expression states of regulators (time series separated by white bars). Expression states are shown for successive time steps (horizontally) for known or putative regulators (vertically). The upper plot shows the discretized expression values, the lower plot visualizes the changes in expression from one time step to the next.