Modeling Protein Interactions Using Bayesian Networks

Sabeek Pradhan, Shayne Longpre, and Varun Vijay

December 12, 2014

1 Introduction

ALL (Acute Lymphoblastic Leukemia) is a common form of cancer in children. About 15% of childhood ALL cases involve a genetic mutation known as CRLF2 Translocation, in which the CRLF2 gene is found in the "wrong" chromosome. This translocation severely worsens the patient's prognosis (Moorman et. al.), but researchers do not know why this is the case. Understanding why the CRLF2 translocation worsens a patient's prognosis could aid the development of new treatments for ALL patients with this translocation.

In our project, we sought to use recent advances in the analysis of disease features at the molecular level, to reveal the variation of disease features as a function of differences in the level and activity of molecules (in our case, of proteins). This work has been facilitated by the development of new experimental techniques such as Mass Cyclometry (Bendall et al.). The method did not yield informative results when used on cells in basal state (Irish et. al.), indicating the need to "potentiate signalling" to initiate cell activity. The dataset we have obtained uses several drugs, which stimulate or inhibit certain proteins, and may provide better insights when analyzed.

2 Dataset

Our data set consisted of cell samples from 12 children with ALL, half of whom exhibit CRLF2 translocation. For each patient, 16-18 different samples of cells were drawn, and a combination of drugs was applied to each sample. For each sample, our data matrix had cells as rows, proteins as columns, and each cell represented the protein level.

This data was provided to us by Dr. Karen Sachs at Stanford's Center for Clinical Sciences Research (CCSR).

3 Features and Output

We decided to model the data using Bayesian networks in order to expose the structure of protein interactions. The nodes would be proteins, along with added nodes to represent patients, drugs and the translocation factor. The added nodes were restricted to be roots since they could not be influenced by proteins.

4 Methodology

4.1 Pre-processing the Data

Initially, the data was log-normally distributed, with a large positive skew. This presented challenges for our model learning and scoring algorithms, so we normalized the data to make it approximate a Gaussian

distribution. However, at the suggestion of Dr. Sachs, we did not simply take the logarithm of the data, lest we run into problems as values approached zero. Instead, we applied the following transformation:

$$x := \text{asinh}\left(\frac{x}{5}\right)$$

This allowed us to normalize the data without running into issues with near-0 values.

4.2 Discretization

The majority of our optimizations focused on different ways to discretize the data. Any algorithm that ran on continuous data would require some type of regression to determine which other nodes had a significant impact on the node in question and should thus have edges. However, regression analysis is highly sensitive to feature selection. Since the mechanisms for the interactions between these proteins has yet to be explored in depth, any attempts at feature selection would be somewhat arbitrary. Further, there is no guarantee that the mechanisms of interaction are even of a linear form; if no linear regression could adequately describe the relationships then learning a network on continuous data would be an exercise in futility. Lastly, Dr. Sachs suggested that such biological data can often be modeled as discrete even though it is technically continuous.

We used two different methods to discretize our data: K-means and equi-depth partitions. In each case, we ran the discretizations on each protein individually. As per Dr. Sachs's suggestions, we ran our discretization methods with 3 and 6 bins. The partition boundaries for each of the four discretizations over a subset of the proteins are shown below:

	3 K-Means		3 Equi-Depth	
	1-2	2-3	1-2	2-3
CD20	0.261	1.330	-0.112	-0.027
CD34	0.646	2.047	-0.056	0.442
CD45	0.337	1.436	-0.089	0.068

	6 K-Means					6 Equi-Depth				
	1-2	2-3	3-4	4-5	5-6	1-2	2-3	3-4	4-5	5-6
CD20	-0.082	0.115	0.536	1.263	2.471	-0.155	-0.112	-0.070	-0.027	0.100
CD34	0.155	0.706	1.406	2.191	3.104	-0.127	-0.056	0.055	0.442	1.498
CD45	0.039	0.377	0.880	1.576	2.521	-0.1431	-0.0890	-0.0347	0.068	0.336

The top row tells us which columns correspond to which discretization technique, and the number of buckets. The second row tells us which boundary partition is being represented. For example, 1-2 means we are splitting the first and second bucket by the value indicated below. The first column corresponds to specific proteins, and the internal values are protein level boundaries.

4.3 Network Learning Algorithm

The number of possible Bayesian networks scales hyperexponentially with the number of nodes in the final graph. Thus, with 64 nodes in our graph, it would have been impossible to exhaustively search through all possible graphs. Instead, we had to run a heuristic based search function. We chose to use Greedy Hill Climbing, which works as follows:

1. Randomly initialize the network structure.
2. Iterate through possible "steps", where each step involves adding or remove an edge, and execute the step with the most positive effect.
3. Repeat until convergence.

At each step, we scored our network using BDe (Bayesian Dirichlet likelihood equivalence–see below), which provides a conjugate prior for discrete data on a multinomial distribution. It penalizes both poor fit and too many parameters. The BDe score of the final network gave us a measure of training error. Our measure of test error was the BDe score computed with respect to a test sample that the network did not see before .

Our implementation of the Greedy Hill Climbing algorithm was done using the Biolearn package, a set of Java files provided by Dr. Dana Pe'er at Columbia University.

4.4 Sampling and Model Averaging

Due to the size of the dataset, we were forced to run our algorithms on samples drawn from the data. We implemented a sampling script that randomly drew 500 data points from each original data file (multithreading was necessary to make the script run in reasonable time). We then trained our network on this sampled file.

Unfortunately, as the Figure 1 shows, training our networks on single samples provided unsatisfactory results. The networks were extremely dense and contained many edges that would appear only in the network from that particular sample. This is partially because the networks overreacted to minor peculiarities within the sample file they were generated from, leading to overfitting. Another contributing factor was the fact that, as mentioned above, it is impossible to exhaustively search through all possible Bayesian networks. Since the GreedyHillClimbing algorithm we used makes only incremental adjustments to the network, it runs the risk of getting stuck at local optima, which depend on the random initialization of the network.

To ameliorate this, we implemented model averaging. With model averaging, we trained Bayesian networks on 100 different samples. We then counted the number of networks any given edge appeared in and included in our final network all the edges that appeared in at least 90 networks. This greatly improved our network's density and predictive power.

4.5 Scoring

To score our networks, both while generating them and to analyze our final results, we used the Bayesian Dirichlet likelihood equivalence function (BDe for short). The BDe score is a negative number where values closer to 0 indicate better networks. A brief description of the BDe function is below.

The BDe scoring method assumes that the distribution of the parameters $p(\theta_D|G)$ is Dirichlet for some complete acyclic graph in a space D. We also assume that the network fulfils some other criteria, including parameter independence, parameter modularity and likelihood equivalence, in order to make the score computable. Then, for any Bayesian networks in D that fulfils these assumptions, the joint probability of the network with data T is

$$P(B,T) = P(B) \times \prod_{i=1}^{n} \prod_{j=1}^{q_i} \left(\frac{\Gamma(N'_{ij})}{\Gamma(N_{ij} + N'_{ij})} \times \prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk} + N'_{ijk})}{\Gamma(N'_{ijk})} \right)$$

where $\Pi_{X_i}$is the set of parents of node i, $w_{ij}$is the jth configuration of Π_{X_i}i.e. one of the many possible sets of parents, $q_i$is the number of possible configurations, $N_{ij}$is the number of instances in which variable $X_i$takes its kth value, and $N'_{ij} = N' \times P(X_i = x_{ij}, \Pi_{x_i} = w_{ij}|G)$, the equivalent sample size, expresses the strength of our belief in the distribution. $\Gamma(n) = (n-1)!$is the Gamma function.

The BDe score is then $BDe(B,T) = log(P(B,T))$. As the expression indicates, the BDe balances the likelihood of the network ($P(B)$), which is a reflection of the number of assumptions made, and the rest of the expression, which reflects how well it fits the data.

5 Results and Discussion

We used as our oracle the score of a run on a single sample and as our oracle the score of a run on the data set it was trained on. We then sampled new test sets of identical sizes to the training test sets for the test score. The scores[1] are as follows:

Test Name	Training Score	Test Score
Averaged 3 K-means	-3,435,170	-3,442,390
Averaged 3 Equal Buckets	-3,353,890	-3,361,960
Averaged 6 K-means	-1,894,420	-1,899,090
Averaged 6 Equal Buckets	-1,911,120	-1,915,730
Sample run 3 K-means	-15,588,800	-15,596,800
Sample run 3 Equal Buckets	-16,800,100	-16,814,300
Sample run 6 K-means	-9,134,700	-9,141,650
Sample run 6 Equal Buckets	-9,199,450	-9,205,700

The graph for our 6 K-means run, which was our best performer, is in Figure 2. The graphs for our 6 Equal Buckets, 3 K-means, and 3 Equal Buckets are in Figures 3, 4, and 5, respectively.

In general, the runs with 6 discretization bins outperformed the runs with 3. Further, the graphs for the two runs with 3 bins closely resembled each other, as did the runs with 6 bins. This suggests that the networks with 6 bins were capturing different (and more complex) relationships than the networks with only 3.

From a biological perspective, there was an edge from Translocation to the CD45 protein in all our runs. This indicates that whether or not a patient has the CRLF2 translocation has a very significant impact on their CD45 level. Thus, CD45 could be a key part of the mechanism by which patients with the CRLF2 translocation have worse prognoses than those without the translocation. Similar effects were found for the proteins CD20, CD43, and CD58 (which were found in almost all networks with 3 bins) and for the proteins CD38, and Pax5 (which were found in a majority of the networks with 6 bins). Further research is necessary to determine why these proteins are so influenced by the presence of a CRLF2 translocation and what impact they could have on a patient's survival chances. Dr. Sachs and her colleagues will be analyzing our results and may hopefully be able to suggest new areas of research based off our findings.

6 References

Characterization of Patient Specific Signaling via Augmentation of Bayesian Networks with Disease and Patient State Nodes, K. Sarchs, A. J. Gentles, Y. Ryan, S. Itani, J Irish, G. P. Nolan, 31st Annual International Conference of the IEEE EMBS (2009)

IGH@ translocations, CRLF2 deregulation, and microdeletions in adolescents and adults with acute lymphoblastic leukemia, A.V Moorman, C. Schwab, H.M. Ensor, L.J. Russell, H. Morrison, L. Jones, D. Masic, B. Patel, J.M. Rowe, M. Tallman, A. H. Goldstone, A. K. Fielding, C. J. Harrison. J Clin Oncol. 2012;30(25):3100–8.

Scoring function for learning Bayesian networks, A. M. Carvalho, Technical report, INESC-ID Tec. Rep. 54/2009, 2009

Single-cell mass cytometry of differential immune and drug responses across a human hematopoietic continuum, S.C. Bendall, E.F. Simonds, P. Qiu, A.D. Amir, P.O. Krutzik, R. Finck, R.V. Burggner, R. Melamed, A. Trejo, S.K. Plevritis, G.P. Nolan

Single cell profiling of potentiated phospho-protein networks in cancer cells, J. M. Irish, R. Hovland, P.O. Krutzik, O. Bruserud, G.P. Nolan, Cell (2004)

[1] Note: After the poster session, we discovered several bugs in our scoring function, which is why the BDe scores are very different here than they were on our poster.

7 Figures

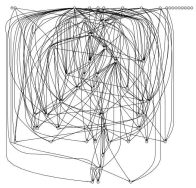

Figure 1: A noisy, dense Bayesian network generated from a single sample

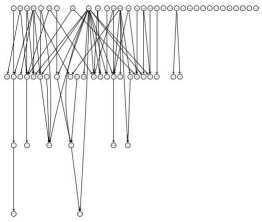

Figure 2: The Bayesian network generated by the Averaged 6 K-means algorithm

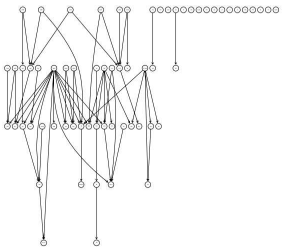

Figure 3: The Bayesian network generated by the Averaged 6 Equal Buckets algorithm

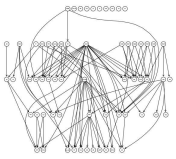

Figure 4: The Bayesian network generated by the Averaged 3 K-means algorithm

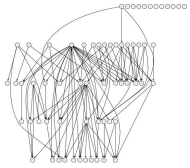

Figure 5: The Bayesian network generated by the Averaged 3 Equal Buckets algorithm