PGM End-Term Project Report
Bayesian Structure Learning

Anshul Khantwal
MT16010, MTech CSE,
Indraprastha Institute of Information Technology, Delhi
India-110020
Email: anshul16010@iiitd.ac.in

Shagun Gupta
MT16052, MTech CSE,
Indraprastha Institute of Information Technology, Delhi
India-110020
Email: shagun16052@iiitd.ac.in

***Abstract*—Over the last two decades, study of basic structure of a classically understood signaling network that connects a number of key phosphorylated proteins in human T cell signaling, has been an area of study among the group of classical biochemistry and genetic analyst. In this report, we present our algorithm to construct a Bayesian network from the Multiparameter Single-Cell Dataset and infer various dependencies among the cellular proteins. We would be using BIC as our scoring function for the Bayesian network. The implementation will be evaluated over three datasets in-order to draw inferences.**

I. Data Preprocessing

We were provided with an Multiparameter Single-Cell Dataset that constituted of 14 excel files that reflected the effects of various reagents over the eleven proteins of the human T cells. Individual rows in each file signifies various human T cells that were used in experiments and eleven columns signify the number of key phosphorylated proteins present in the human T cells.

The given raw dataset was cleaned and prepocessed into three variants, namely, Western-Blot Dataset, Truncated Dataset and Combined Dataset. These were basically used individually for learning the Bayesian network as well as inferring the results.

A. Simulated Western-Blot Dataset

To create a Simulated Western-Blot dataset, the following steps were followed for each excel dataset file:

1) 20 cells were randomly sampled and averaged, until all the cells had been averaged. This yielded about 30 datapoints for each indidual condition.
2) Thus, a total of about 420 datapoints were obtained from 14 excel files.
3) These datapoints were then discretized on individual columns to fall in the range of three categorical labels, namely, low, medium and high.

B. Truncated Dataset

To create a Truncated dataset, the following steps were followed for each excel dataset file:

1) 30 cells were randomly sampled from each of the 14 conditions producing 420 data points in total.
2) These datapoints were then discretized on individual columns to fall in the range of three categorical labels, namely, low, medium and high.

The above process was repeated multiple times with different random seeds so as to generated different dataset samples.

C. Combined Dataset

To create a Combined dataset, the following steps were followed for each excel dataset file:

1) Outliers from the individual conditions that fell far away from three standard deviations from the mean were eliminated.
2) Datapoints were then discretized to fall in three levels using an agglomerative approach.
3) 600 datapoints were sampled from each conditions and were merged to form combined dataset.

The prepared dataset were given as an input to the proposed algorithm in order to perform Bayesian Network learning on the dataset and infer dependencies.

II. Proposed Algorithm

Bayesian networks basically provides a graphical representation of multivariate joint probability distributions. This representation consists of an directed acyclic graph whose nodes corresponds to random variables, each representing the measured amount of protein in the dataset. The goal of this algorithm was to search among possible graphs and select a graph that best describes the dependencies present in the Multiparameter Single-Cell Dataset.

The algorithm has been described in following key steps:

A. Problem Statement

Given a training set D_i and a scoring function, we would like to find a Bayesian network structure that would maximize the score over all the possible spaces of graph.

In our algorithm, we would be using Bayesian information criterion (BIC) score for scoring our graph and maximizing it over our search space. BIC provides us with advantage of score decomposition as well as easiness in its implementation.

$$\mathrm{score}_{BIC}(\mathcal{G} \ : \ \mathcal{D}) = M\sum_{i=1}^{n} \boldsymbol{I}_{\hat{P}}(X_i; \mathrm{Pa}_{X_i}) - M\sum_{i=1}^{n} \boldsymbol{H}_{\hat{P}}(X_i) - \frac{\log M}{2}\mathrm{Dim}[\mathcal{G}]$$

Here, BIC score is defined in terms of Mutual information of nodes, entropy of nodes and the dimension of the graph. Dimension term acts as a penalty term over the log likelihood score and thus enables in obtaining a true structure. BIC score of a network structure G decomposes into score a product of terms, one for each family. This property is crucial for decomposing the learning problem into independent subproblems.

B. Select a Random Graph

A graph was selected at random with specified number of nodes. This helped in starting our searching and selecting the next best possible move.

C. Select the Next Best Possible Operation

Given a random graph, following operations are possible, namely, insertion of new edge, deletion of already existing edge and reversal of an existing edge. In this step, we would like to search in these three possible spaces and find out the best possible operation that would maximize the score of the graph.

Let $\mathcal{G}$ be a network structure and score *be a decomposable score.*

- *If o is "Add $X \to Y$," and $X \to Y \notin \mathcal{G}$, then*

$$\delta(\mathcal{G} \ : \ o) = \mathrm{FamScore}(Y, \mathrm{Pa}_Y^{\mathcal{G}} \cup \{X\} \ : \ \mathcal{D}) - \mathrm{FamScore}(Y, \mathrm{Pa}_Y^{\mathcal{G}} \ : \ \mathcal{D}).$$

- *If o is "Delete $X \to Y$" and $X \to Y \in \mathcal{G}$, then*

$$\delta(\mathcal{G} \ : \ o) = \mathrm{FamScore}(Y, \mathrm{Pa}_Y^{\mathcal{G}} - \{X\} \ : \ \mathcal{D}) - \mathrm{FamScore}(Y, \mathrm{Pa}_Y^{\mathcal{G}} \ : \ \mathcal{D}).$$

- *If o is "Reverse $X \to Y$" and $X \to Y \in \mathcal{G}$, then*

$$\begin{aligned}\delta(\mathcal{G} \ : \ o) \ = \ & \mathrm{FamScore}(X, \mathrm{Pa}_X^{\mathcal{G}} \cup \{Y\} \ : \ \mathcal{D}) + \mathrm{FamScore}(Y, \mathrm{Pa}_Y^{\mathcal{G}} - \{X\} \ : \ \mathcal{D}) \\ & - \mathrm{FamScore}(X, \mathrm{Pa}_X^{\mathcal{G}} \ : \ \mathcal{D}) - \mathrm{FamScore}(Y, \mathrm{Pa}_Y^{\mathcal{G}} \ : \ \mathcal{D}).\end{aligned}$$

The above figure describes the three operations and their relative delta scores. This reflects the change in overall score of the two graphs. The operation with the maximum delta score is applied on the graph.

D. Convergence to Bayesian Network

The above step is iterated till we reach the point of convergence. Convergence is defined by the notion of graph being isomorphic to the previous graph from previous iteration.

The above procedure provides us with a single Bayesian network corresponding to a particular random graph. The algorithm do gets effected by the problem of finding a graph with a maximal score in its local maxima. The problem is countered by repeating the above procedure multiple times with different random graphs and extracting the best possible Bayesian Network from them.

III. Implementation Details

The above algorithm was implemented in Python programming language on Windows 10 OS and Intel i7 Processor with 16GB RAM.

In-order to represent a graphical structure, NetworkX package was used. NetworkX is a Python language software package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

The dataset was cleaned and preprocessed in generateSimulatedWesternBlotDataset, generateTruncatedDataset and generateCompleteDataset methods using Pandas package to read the dataset and were stored in a pickle format.

familyScore method is responsible for calculating the family score of a node given its parents. learnGraph method learns an individual Bayesain Network from a random graph. We had used two driver programs that were responsible for learning the Bayesian network from different datasets using different convergence approach. learnDataset driver program was meant to consider the best graph from a collection of generated graphs on the basis of maximal BIC score whereas learnDatasetSupportScore used to select only those edges that were above a particular support confidence among the collection of generated graphs.

IV. Results

The algorithm was applied on three datasets and learned network structures were analyzed. On most of the occasions, the algorithm was found to work better with combined dataset when compared to western-blot and truncated datasets. Combined dataset being large in number of datapoint was able to capture the dependencies more efficiently.

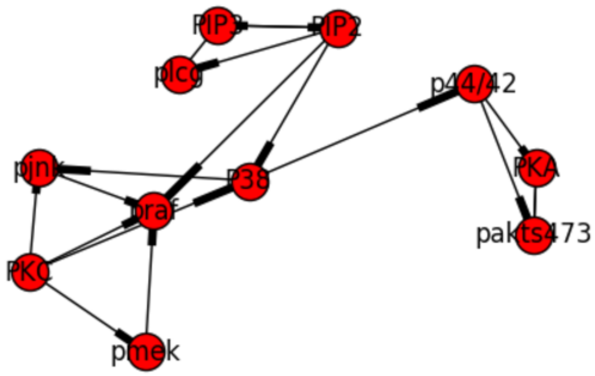

Fig. 1. Bayesian Network obtained from Combined Dataset using Support Confidence Criterion of 40%

Fig. 1 and Fig. 2 describes the graph generated using our algorithm on the Combined dataset. About 50% of the expected dependencies were captured in the Fig. 1 whereas only 21.42% were captured in the Fig. 2. These network structures were generated by averaging over 100 Graphs and capturing the best network using Support Confidence Criterion and Maximal BIC Score Criterion respectively.

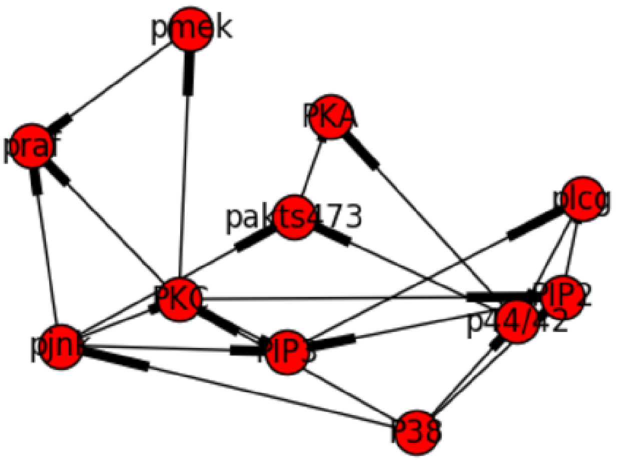

Fig. 2. Bayesian Network obtained from Combined Dataset using Maximal BIC Score Criterion

Fig. 3 and Fig. 4 describes the graph generated using our algorithm on the Simulated Western-Blot dataset. About 21.42% of the expected dependencies were captured in the Fig. 3 whereas only 28.57% were captured in the Fig. 4. These network structures were generated by averaging over 100 Graphs and capturing the best network using Support Confidence Criterion and Maximal BIC Score Criterion respectively.

Fig. 5 and Fig. 6 describes the graph generated using our algorithm on the Truncated dataset. About 21.42% of the expected dependencies were captured in the Fig. 5 whereas only 21.42% were captured in the Fig. 6. These network structures were generated by averaging over 100 Graphs and capturing the best network using Support Confidence Criterion and Maximal BIC Score Criterion respectively.

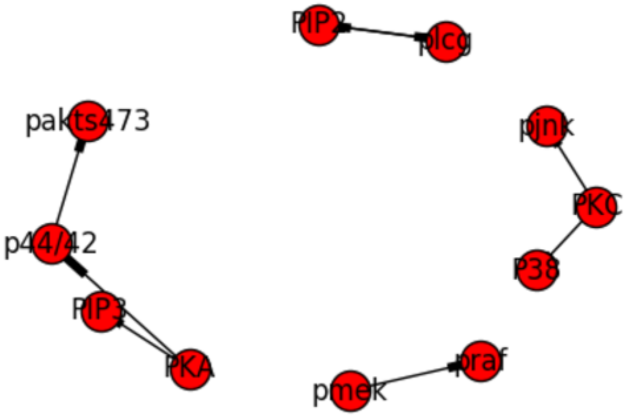

Fig. 3. Bayesian Network obtained from Simulated Western-Blot Dataset using Support Confidence Criterion of 40%

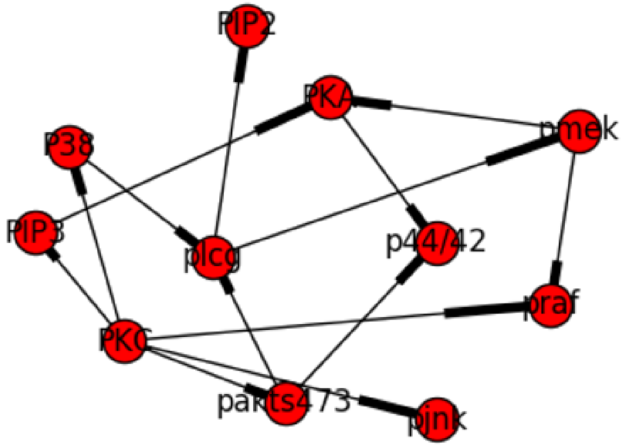

Fig. 4. Bayesian Network obtained from Simulated Western-Blot Dataset using Maximal BIC Score Criterion

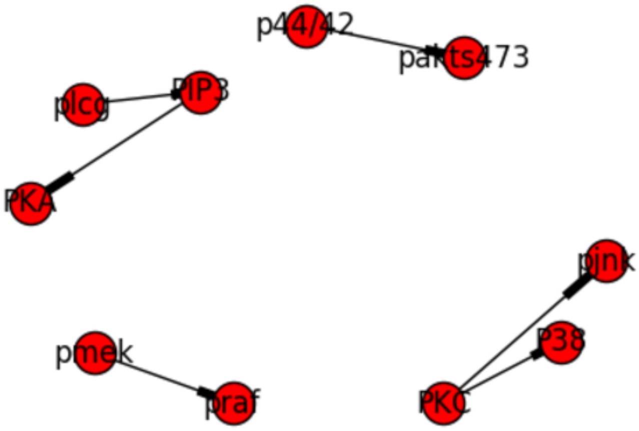

Fig. 5. Bayesian Network obtained from Truncated Dataset using Support Confidence Criterion of 40%

Some edges were captured in the reverse direction of the expected. When these reversed edges were taken into account for determining the accuracy in prediction, an increment was observed. About 78.57% accuracy was observed in combined dataset whereas 35.7% accuracy in Western-Blot dataset and about 42.85% accuracy in Truncated dataset using support confidence of 40% on edges.

Expected Edges	14
Predicted Edges	7
Reversed Edges	4
Not Reported Edges	3

TABLE I. STATISTICS OF REPORTED EDGES ON BAYESIAN NETWORK GENERATED USING COMBINED DATASET

Table I listed the counts of edges in a Bayesian network generated using confidence support of 40% on combined dataset. 7 edges were predicted correctly out of 14 expected edges. About 4 edges were found in their reversed direction.

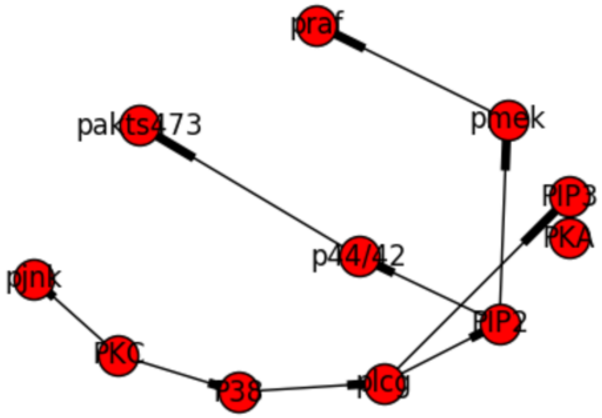

Fig. 6. Bayesian Network obtained from Truncated Dataset using Maximal BIC Score Criterion

Only 3 edges were missed in this dataset. This was an motivating statistic for our analysis.

When individual conditions from individual excel files were used as a dataset to learn Bayesian network, quite a few dependencies were evident in them also.

V. Conclusion

The implementation was found to perform decently well over the Multiparameter Single-Cell Dataset. BIC score was effective in finding the relevance of a graphical structure due to its decomposability property. The implementation was able to capture a decent percentage of the expected dependencies. In order to further enhance the accuracy of our model, we would be interested in applying other scoring functions to our implementation as an future improvement.

References

[1] K. Sachs, O. Perez, D. Pe’er, D. A. Lauffenburger, and G. P. Nolan, “Causal protein-signaling networks derived from multiparameter single-cell data,” *Science*, vol. 308, no. 5721, pp. 523–529, 2005.