Time Series Gene Expression Prediction using Neural Networks with Hidden Layers

Michael R. Smith[*], Mark Clement[†], Tony Martinez[‡] and Quinn Snell[§]
Department of Computer Science, Brigham Young University, Provo, UT 84602, USA
Email: [*]msmith@axon.cs.byu.edu, {[†]clement, [‡]martinez, [§]snell}@cs.byu.edu

I. Introduction

With the advancement in DNA microarray technologies, inferring genetic regulatory networks (GRNs) is critical to understanding cellular processes and complex gene interactions. The microarray analysis provides time series data points measuring gene expression levels which are then used to model how genes interact with each other. Modeling and simulating GRNs can potentially be used in areas such as drug testing and cancer treatment. Modeling GRNs poses a difficult problem since only a limited amount of noisy, high dimensional data is provided and GRNs involve non-linear relationships between genes with feedback loops. There are also potentially many genes or other implicit factors that affect the gene expression level which may not be measured in the microarray analysis.

A large body of research has been performed in an attempt to characterize GRNs using various models. However, the models have been constrained to be interpretable so that a GRN can be inferred from the model. Boolean networks have been used because of their simplicity and their ability to deal with noisy data [1] but lose information by having a binary representation of the genes. Ordinary differential equations attempt to incorporate biological knowledge and derive kinetic equations for gene interactions [2]. However, it is assumed that all components of the system and the interactions between them are known and all necessary constants can be measured or estimated which is not always the case. Bayesian networks have also been used to model GRNs [3] to deal with the stochastic aspects of gene expression and to deal with noisy data, but they minimize the dynamical aspects of gene regulation.

We examine modeling GRNs with NNs. Earlier work has used NNs to model GRNs [4] by having each node in the NN represent a gene while the weight between the nodes represents the strength of the interaction. Thus, a GRN could be inferred from the weights of the network. Using NNs to models GRNs was extended to recurrent NNs (RNNs), a class of NNs that have feedback loops [5]. The RNNs model the feedback that occurs naturally in gene interactions. However, the recurrence also make the RNNs difficult to train. A common approach used for training RNNs is backpropagation through time (BPTT) which unfolds the network a certain number of time steps to train the feedback weights.

Hidden layers in NNs have been omitted to keep the model interpretable. Without a hidden layer, the NN is only capable of modeling first order relationships. For example, suppose we were modeling the first gene (G1) in a two gene network. Let us assume a Boolean model for the example. Suppose G1 is high when in the previous time step G1 is low and G2 is high or when G1 is high and G2 is low. Otherwise G1 is low (logical XOR). A NN with without a hidden layer is incapable of modeling this problem because the outputs are not linearly separable. The hidden layer is necessary to transform the problem space so that the problem is linearly separable after the transformation.

This research examines modeling GRNs with neural networks (NNs) and recurrent NNs that have a hidden layer for gene expression level prediction rather than inferring the GRN. Predicting the gene expression levels can be an indication of how well the NN has modeled the GRN. One purpose for a predictive model is the generation of additional data points. A simpler model can then use these synthetic data points in addition to the actual data points to infer the GRN from the additional data. The predicted data points also allow a researcher to examine the way a GRN reacts to perturbations *in silico* without having to perform the actual experiments which are often time consuming and expensive. Modeling the GRN with a NN with a hidden layer will be more descriptive than one without a hidden layer, but the model will not be as interpretable and thus the GRN will not be able to be inferred by simply examining the weights. The advantage of using NNs is that they are well suited to modeling complex relationships and handling noisy data such as that in microarray data. We find that the NNs with hidden layers result in lower error in gene expression prediction.

NNs with hidden layers have been used with microarray data for classification, but not for gene expression level prediction [6]. To our knowledge using NNs with hidden layers to predict of gene expression levels of a GRN has not been studied.

Another problem with traditional GRN algorithms is their need for large volumes of time series data. In this research, we combine data points with variable elapsed time to create additional data points for training that incorporate temporal information. Traditionally, the data at time t is used to predict the expression levels at time t+1. To make more use of the available data, we combine each pair of data points from the time series data and include a new attribute of the time that has elapsed. A data point is a set of gene expression levels at a specific time as given in the microarray data. The NNs will have one additional input node representing the size of the time step. For example, for microarray data with 5 time steps, traditional training algorithms will have a difficult time training. By combining the data points, we can train the network much more effectively by including the original time steps (1,2), (2,3), (3,4), (4,5) plus the time steps between (1,3), (1,4), (1,5), (2,1),…,(3,5). We find that this approach aids NNs with a hidden layer, but does results in higher error for NNs without a hidden layer. The resultant expression levels are smoothed as well, suggesting that some noise filtering has been done. In terms of error values, the NN with a hidden layer is comparable to an Elman net [7] architecture for a recurrent NN.

II. Experimental Methodology

To examine the impact of using NNs with hidden layers and combining data points we use a NN without a hidden layer (perceptron), a NN with a hidden layer, and a RNN. The perceptron and NNs models are trained using backpropagation and the RNN is trained using backpropagation through time (BPTT) [8] and Elman nets [7] on two data sets. The first data set was a synthetic data set from the DREAM 3 competition [9]. This data set was used because there truth value is known and consists of 4 perturbations to a 10 gene network measured every 10 seconds for a duration of 200 seconds. We evaluated the models using the first perturbation as the test set and the remaining three perturbations as the training data. The second data set was the SOS DNA repair network in bacterium *Escherichia coli* data set [10] which is a real world data set and was used to compare our work against other models. The SOS system consists of about 30 genes with four experiments being conducted with different light intensities. Each experiment measures eight major genes sampled every 6 minutes for 50 time steps. The models are evaluated on the test set using root mean squared error (RMSE), correlation coefficient, mean error, and graphically to visually determine goodness.

III. Results

The RMSE and correlation coefficient on the test set for each model on the DREAM data set is given in Table I where “time” means that the model was trained using data that was augmented using our method for combining data points. The values in bold indicate the best values for the RMSE and the correlation coefficient. Based on the RMSE and the correlation coefficient, the neural network with hidden layers trained using the modified data set and the Elman net perform the best. From this, adding a hidden layer (NN) does improve the model compared to the perceptron model as the RMSE decreases and correlation coefficient increases. Also, the combination of the data points to create a temporal aspect also improves the performance of the NN model and degrades the performance of the perceptron model.

Fig. 1 shows the gene expressions levels for the first experiment from the SOS data set as well as the gene expressions levels predicted by the perceptron, NN, NN with time, BPTT, and Elman net models. The x-axis represents time and the y-axis represents the gene expression level for each gene. The perceptron model is abrupt and falsely predicts fluctuations in the gene expression levels. The NN model follows the actual expression level very closely and is slightly smoother than the target signal, possibly filtering noise. The NN with time smooths the expression levels even more, providing the overall trend of the time-series data. The BPTT and Elman net models are abrupt, exaggerating slight fluctuations in the target gene expression levels. The Elman net model is able to converge to the steady state, but BPTT is never able to.

Model	RMSE	Correlation
Perceptron	0.0153	0.923
Perceptron time	0.0197	0.882
NN	0.0118	0.928
NN time	**0.0106**	**0.935**
BPTT	0.0148	0.9141
Elman	**0.0103**	**0.937**

TABLE I
Comparison of the RMSE on the DREAM data set.

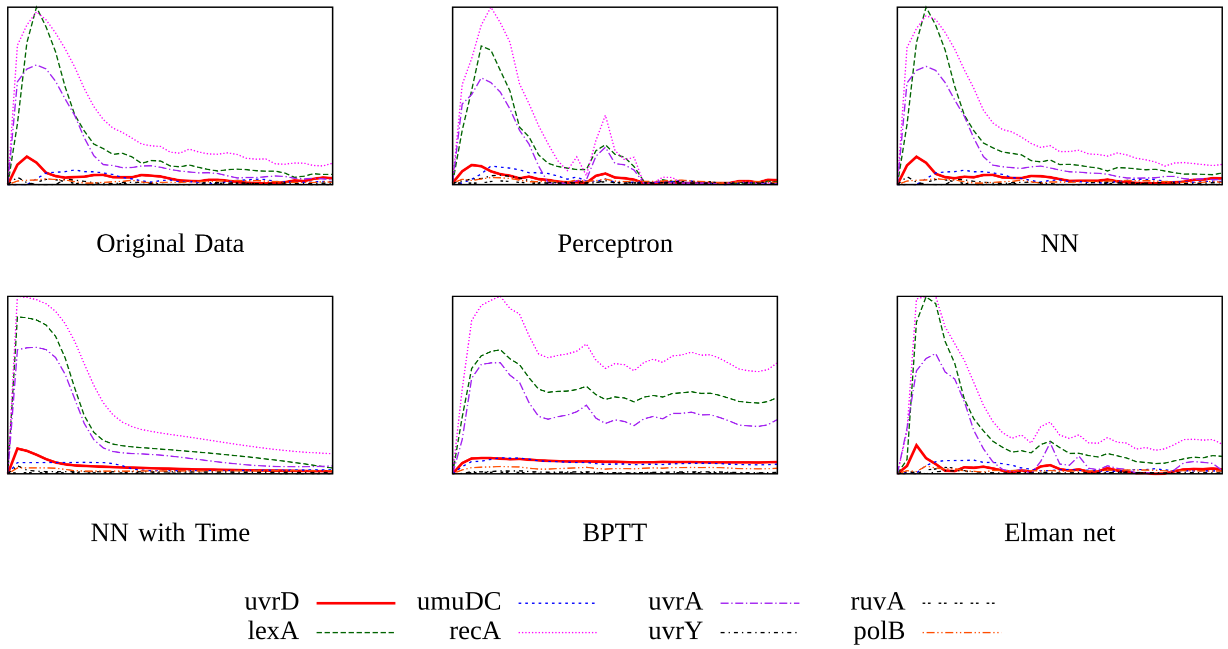
Model	uvrD	lexA	umuD	recA	uvrA	uvrY	ruvA	polB	ave
Perceptron	0.57	0.58	0.59	0.60	0.69	0.87	0.79	0.34	0.63
Percep time	0.38	0.63	0.63	0.47	0.69	0.43	0.89	0.28	0.55
NN	**0.09**	**0.07**	**0.06**	**0.03**	**0.04**	**0.23**	**0.20**	**0.04**	**0.09**
NN time	0.37	0.18	0.27	0.05	0.10	0.35	1.18	0.23	0.34
BPTT	1.25	2.26	1.07	1.50	2.61	0.28	1.29	0.52	1.35
Elman	0.39	0.23	0.16	0.21	0.51	0.35	0.60	0.19	0.33
NFRN	0.17	0.08	0.09	0.10	0.09	**0.16**	0.20	0.08	0.12
EK	0.20	0.10	0.21	0.12	0.14	0.45	0.22	0.31	0.22

TABLE II
Comparison of mean error on the SOS data set

The SOS data set is also used to compare our models with two other models; Expression Kinetics (EK) [11] and Neural Fuzzy Recurrent Networks (NFRN) [10]. The models are evaluated using mean error. Both training and testing use the first two-thirds of the first experiment 1 as was done by Maraziotis et. al. [11]. This will demonstrate how well each model is able to learn the GRN.The mean error values for each model are shown in Table II. With the exception of BPTT, all of the models with a hidden layer have lower mean error than the perceptron models. Overall, the NN model achieves the lowest mean error for every gene (including the NFRN and EK models) except for uvrY. Using the mean error metric, the augmented data set (Percep w/ time and NN w/ time) does not help in predicting the gene expression level, but rather creates more error. It is interesting that the NN with the augmented data set performs similarly to Elman nets in terms of error. The recurrence in the Elman net model is better able to capture the behavior of the data where as the NN with time model smooths the data. BPTT does not perform well having the highest error on every gene except for one.

IV. Discussion

Using NNs with a hidden layer to model GRNs is very promising. We found that NNs with a hidden layer are better able to predict the gene expression level for a GRN than NNs without a hidden layer. This suggests that higher-order


Original Data

Perceptron

NN

NN with Time

BPTT

Elman net

Fig. 1. Predictions from each model for the SOS data set. Individual genes have the same color and line style in each graph for comparison. The graph that is most similar to the ”Original Data” is the most accurate. The x-axis represents time and the y-axis represents the gene expression level for each gene.

correlations do exist in GRNs and it is important that the models used to model GRNs have the capability of capturing these higher-order correlations. The disadvantage of using a hidden layer is that the model is less interpretable, meaning that current techniques for inferring the GRN from the weights of the NN cannot be used. However, a predictive model can generate additional training points that can then be supplied to a simpler model that is able to infer a GRN. The addition of more data points should produce a better model of the data.

We also proposed to combine the data points from time series data to create more data points with different time step values. Training the perceptron model with augmented data set resulted in worse predictions. However, using the augmented training set to train a NN with a hidden layer, the predictions were smoothed and followed the overall trend in behavior for the genes. Due to the noise in the measurement data, the temporal component could be seen as filtering the noise from the expression levels whereas the NN model could be seen as overfitting the data.

The recurrent NNs used in this study do not out perform the non-recurrent NN models. In fact training NNs with the augmented data set had similar error and correlation results with those of the Elman net model. The Elman net model was able to detect fluctuations in the gene expression levels while the NN with time produced a smoothed gene expression level predicting the overall trend of the gene. Both BPTT and Elman nets appeared to be filtering noise by producing smoothed gene expression levels. This is beneficial as the models could simply be used as a noise filtering technique.

References

[1] T. Akutsu, S. Miyano, and S. Kuhara, “Identification of genetic networks from a small number of gene expression patterns under the boolean network model,” in *Pac Symp on Biocomput*, 1999, pp. 17–28.

[2] M. J. L. de Hoon, S. Imoto, K. Kobayashi, N. Ogasawara, and S. Miyano, “Inferring gene regulatory networks from time-ordered gene expression data of bacillus subtilis using differential equations,” in *Pac Symp on Biocomput*, 2003, pp. 17–28.

[3] N. Friedman, M. Linial, I. Nachman, and D. Pe’er, “Using bayesian networks to analyze expression data,” in *RECOMB*, 2000, pp. 127–135.

[4] P. D’haeseleer, X. Wen, S. Fuhrman, and R. Somogyi, “Linear modeling of mrna expression levels during cns development and injury,” in *Pac Symp on Biocomput*, 1999, pp. 41–52.

[5] X. Hu, A. Maglia, and D. Wunsch, “A general recurrent neural network approach to model genetic regulatory networks.” *Conf Proc IEEE Eng Med Biol Soc*, vol. 5, 2005.

[6] H.-Q. Wang, H.-S. Wong, H. Zhu, and T. T. C. Yip, “A neural network-based biomarker association information extraction approach for cancer classification,” *Journal of Biomedical Informatics*, vol. 42, no. 4, pp. 654–666, 2009.

[7] J. L. Elman, “Finding structure in time,” *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[8] P. J. Werbos, “Backpropagation through time: what it does and how to do it,” *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 2002.

[9] D. Marbach, T. Schaffter, C. Mattiussi, and D. Floreano, “Generating Realistic In Silico Gene Networks for Performance Assessment of Reverse Engineering Methods,” *Journal of Computational Biology*, vol. 16, pp. 229–239, 2009.

[10] M. Ronen, R. Rosenberg, B. I. Shraiman, and U. Alon, “Assigning numbers to the arrows: parameterizing a gene regulation network by using accurate expression kinetics,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 16, pp. 10 555–10 560, 2002.

[11] I. A. Maraziotis, A. Dragomir, and A. Bezerianos, “Gene networks reconstruction and time-series prediction from microarray data using recurrent neural fuzzy networks.” *IET Syst Biol*, vol. 1, no. 1, pp. 41–50, January 2007.