From SVM to LSTM: Classification of Time-Series Gene Expression Data

Aiyappa Parvangada, Shefali Umrania

Computational Biology Department, Carnegie Mellon University
Pittsburgh, PA

Microarray and RNA-Seq experiments are well-established technologies for the measurement of time series gene expression. The information in labelled instances of these datasets can be used for classification of unlabelled temporal expression data or for identifying hidden biological networks in individual classes. Such classification finds motivation in clinical applications where prediction of a patient's response to drug treatment is crucial for timely alterion of therapeutic strategies. In this project, we compared the temporal SVM model developed by Orsenigo et al. with other approaches to this problem, namely HMM, KNN$_{\mathbf{DTW}}$, CNN and LSTM, for predicting the yeast cell cycle phase. The dataset is comprised of 384 genes categorized into five phases of the mitotic cell cycle and is freely available under the Stanford Yeast Cell Cycle Analysis Project. Our results show that deep learning based approaches of CNN and LSTM achieve higher classification accuracies compared to the other models, despite the small number of training instances.

[0]All code is available on https://github.com/sumrania/compgen

Introduction

Traditional microarray experiments are static, in that they measure the transcriptional profile of a sample at a single time-point. Such static data are insufficient for studies aimed at understanding dynamic biological processes. For example, research on cyclical or developmental processes require analysis of transcriptional activities at different phases. Another domain where static microarrays fall short is the study of perturbation-response kinetics, since here different genes may respond with different rates across the entire response-period (*1*). Better suited to these studies is time-series gene expression experiments which summarize cellular transcriptional activity at different time-points.

The additional temporal dimension makes the data amenable to analyses of dynamic processes such as the mitotic cell cycle which informs disease progression, development and drug responses. Common analyses for time series gene expression data include identification of differentially expressed genes, clustering, classification and elucidation of dynamic regulatory networks. An overview of the computational methods for various analyses of time series is provided in (*1*) (*2*).

The *Sacchromyces cerevisiae* genome was analysed to identify genes which are differentially expressed in each cell cycle phase (*3*) (*4*). Our project was aimed at developing a method to classify unlabelled gene expression trends to the cell cycle phase in which the gene is known to be differentially expressed.

We plan to compare the temporal SVM model developed by Orsenigo et al (*5*) with other approaches to this problem, namely HMM, K-Nearest Neighbors, CNN and LSTM for predicting the yeast cell cycle phase of an unlabelled gene expression trend.

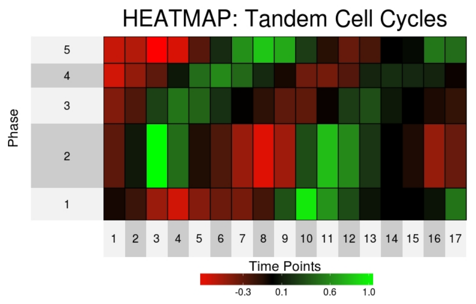

Figure 1: Heatmap showing cyclical nature of the dataset

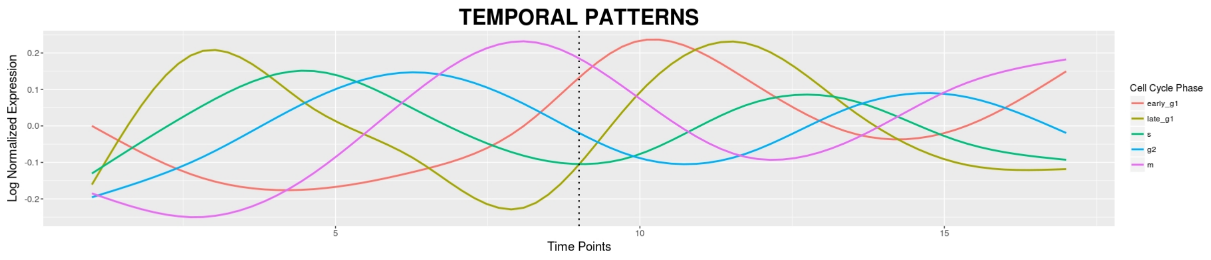

Figure 2: Expression trends of the different classes of expression profiles

Dataset

We obtained the two datasets used by (*4*). The first dataset consists of labeled time-series gene expression for a set of 384 genes of budding yeast. It consists of 17 time ticks with an interval of 10 minutes encompassing two complete mitotic cell cycles. It also includes partition-clustering based labels for the genes; each label corresponds to the cell cycle phase to which the gene belongs(in which the gene was found to be differentially expressed), namely Early G1, Late G1, S, G2 and M. The data was log-transformed and standardized before all downstream analysis. The cyclical nature of the expression data is evident in the smoothed heatmap in Figure 1 with minor inconsistencies in the early part of cycle#1, which is an artifact of the synchronization procedure on the yeast. As shown in Figure 2 cubic splines fit to the expression values of all the genes in a particular class reflect the fact that expression trends between classes are distinct.

The second is the original dataset released by (*6*) which contain unlabeled time-series gene expression profiles of 6000 genes across 17 time-points. As our first dataset used the cdc28(a modified yeast strain) samples, we used the same samples for the unlabeled genes.

Methods

HMM models are particularly well-suited to the task of identifying temporal dependencies (as trasition probabilities) and have already been applied to clinical data (TRAM and GQL) (*7*) (*8*) and finanical time-series data (*9*) (*10*).

K-nearest neighbour methods with novel distance measures have also been applied to this problems in the past (*11*).

Neural networks are a class of machine learning models that are inspired by the biological functioning of neurons. Convolutional Neural Networks (CNN) are a type of feed-forward neural network which establish translational invariance to learn useful features in a heirarchical manner. Translational invariance equips this method to classify similar temporal trends with varying rates(stretched/squeezed) into the same class. CNNs have been shown to outperform traditional machine learning algorithms like SVMs in various learning tasks like classification and regression (*12*) (*13*).

Recurrent Neural Networks (RNN) are a type of neural networks which recognize patterns in sequential data. As opposed to feed-forward neural networks where input and output examples stay distinct, RNNs reuse the output as input to create a feedback loop. Long Short Term Memory Networks (LSTMs) are a type of RNN which are able to use outputs from distant neurons as input for the feedback loop. LSTMs have been shown to successfully predict the next time step for time series data (*14*) (*15*) (*16*).

SVM The paper that served as an inspiration for this project developed an 'L-1 norm Temporal SVM' and applied it to the same yeast dataset. The model results from a modification of the traditional SVM objective function to include a term for the sum of all pairwise DTW distances between genes classified into the same group. This has the effect of maximizing the similarity between all genes classified into the same half-space by the decision boundary. We chose to not recreate this experiment and instead used the accuracy from their model as a benchmark for our project.

HMM Treatment Response Alignment Models (TRAM), a discriminative HMM developed for the task of classifying time-series gene expression profiles of patients was applied on the yeast dataset (*7*). In this model, a hidden state is the conceptual equivalent of a cell cycle phase. The emission probabilites are modeled as multivariate gaussian distributions where the dimension equals the number of genes in a particular class. First, separate HMMs are trained on each class using the traditional generative model. Next, a conditional maximum likelihood estimate of the true classes given the data, the maximum mutual Information estimate, is optimized. This estimate optimizes the discriminative measure between classes. A hyperparameter in this method is the number of hidden states; we set this number at 5 since we know a priori the number of hidden states/classes.

KNN$_{\mathbf{DTW}}$ We chose to apply an implementation of KNN augmented with a dynamic programming method, called dynamic time warping (DTW), to compute the distance metric. as it allows for more accurate classification for time series data as compared to Euclidean distance. (*17*) [https://github.com/markdregan/K-Nearest-Neighbors-with-Dynamic-Time-Warping]. The 1 Nearest Neighbor method in particular using DTW distance has been shown to outperform more complex classifiers (*18*). During calculation of distances between two instances, DTW allows the model to identify similar trends with varying rates of up-regulation or down-regulation. For example, overall expression signatures of patients maybe similar yet unaligned due to different rates of individual responses.

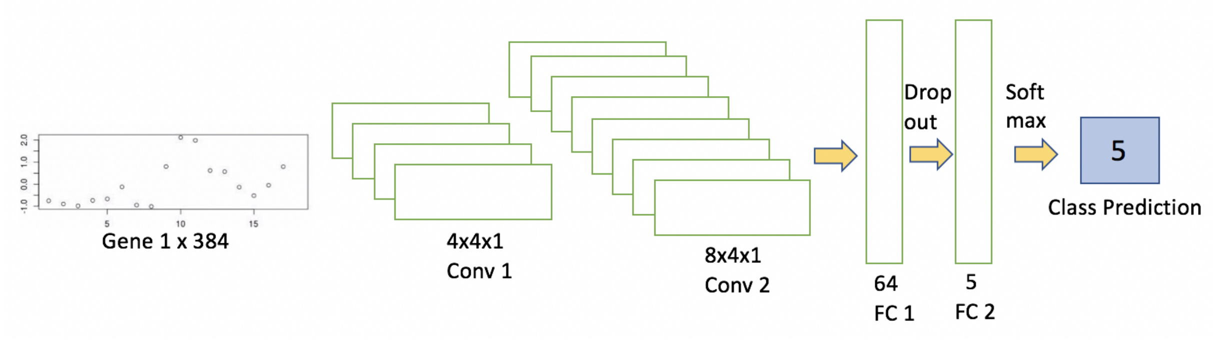

Figure 3: Architecture for the CNN model used for classification

CNN As our dataset consisted of timeseries gene expression values, a 1 Dimensional CNN was used to classify the temporal patterns into a specific class. The architecture shown in Figure 3 was chosen to mitigate the overfitting problems associated with our small dataset.

A Sequential model from Keras was created and the layers were stacked to create the desired topologies.

1. Convolutional layer, 4 filters of length 4 followed by ReLU activation
2. Convolutional layer, 8 filters of length 4 followed by ReLU activation
3. Flatten layer
4. Fully connected layer with 64 units followed by ReLU activation
5. Dropout layer at 50%.
6. Fully connected output layer with 5 units (number of classes) and a softmax activation function.

We used the sparse categorical cross-entropy loss function as this was a multi-class classification task. For learning the gradients, we used ADAM, an algorithm for stochastic gradient optimization which computes a different adaptive learning rate for each parameter by estimating the first and second moments of the gradient. We chose to use this optimizer over SGD as it gave us better accuracy on our validation set. Moreover, theoretically it makes sense that Adam works better than SGD because it uses a different learning rate for each parameter as opposed

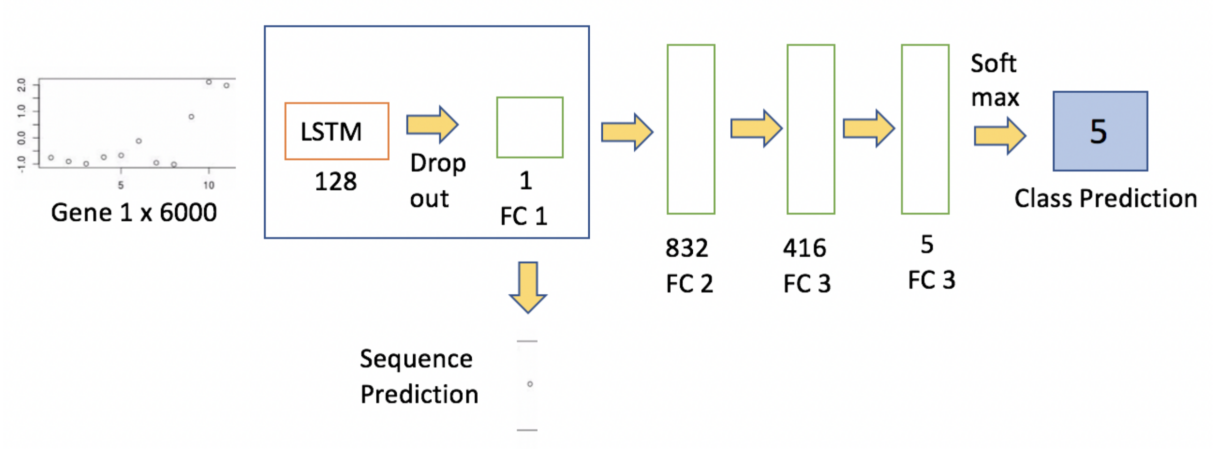

Figure 4: Architecture for the LSTM pre-trained model used for prediction and classification

to treating all parameters equally. Furthermore by using momentum, Adam also achieves faster convergence if the gradient of the previous iteration is in the same direction as the current iteration. We held back 20% of our dataset for testing and used the remaining for training our model. As there are 5 classes in our dataset, any higher validation split would have caused very few testing samples for any given class. Classification accuracy was calculated for the predicted values of the validation set against the known labels.

LSTM As we had access to the 6000 unlabeled time-gene expression profiles, our hypothesis was that if we use LSTM to train a model to predict the next time step for these 6000 genes, that same model should have learned enough features to classify the 384 genes correctly. As shown in Figure 4, we used an LSTM architecture to predict the next time step and followed it with fully connected layers to classify this data.

A Sequential model from Keras (Tensorflow backend) was created and the layers were stacked to create the desired topologies.

1. LSTM layer, 128 units with a window size of 3
2. Dropout layer at 50%.
3. Fully connected output layer with 1 unit as prediction of next time step

We used the mean squared error loss function with RMSprop optimizer as this was a regression task. This trained model was saved at a loss function value of 0.037. Activation values corresponding to the 384 genes from the test data were extracted from the trained model and used as input to the fully connected layers.

1. Fully connected layer with 832 units followed by ReLU activation
2. Fully connected layer with 416 units followed by ReLU activation
3. Fully connected output layer with 5 units (number of classes) and a softmax activation function.

As this is a classification task, we used a softmax activation function. The loss function used was sparse categorical cross-entropy paired with ADAM optimizer. Accuracy was measured against the already known labels of these genes.

Results and Discussion

L1-TSVM	Gen-HMM	Disc-HMM	k-NN$_{\text{DTW}}$	CNN	LSTM
73.9%	68%	54%	65.26%	86.28%	87.5%

Table 1: Classification accuracies obtained using the different methods

Comparison Metric To compare the different methods, we used classification accuracy as the metric for each of the models. The accuracy of the methods have been summarized in Table 1. The benchmark accuracy established by the L1-norm temporal SVM was 73.9%. An accuracy of 68% using the generative model and 54% using the discriminative model was achieved on a hold-out subset of 50 genes (10 from each class). This was unexpected as the discriminative HMM is known to outperform the generative model. After carefully analysing individual expression trends, we concluded that this is because of similar trends between different classes for some instances. For KNN-DTW, tuning of the hyperparameters 'k' and warping window resulted in k=11 and window size of 2 achieving the highest accuracy of 65.26% under a 20-fold cross-validation. The CNN model achieved the highest accuracy of 87.5% on the validation set. A major limitation of this is that the training set is only 80% of 384, it may be over fitting our sample. Interestingly, the pre-trained LSTM model showed a slightly lower accuracy of 86.28%. The reason for this could be that because the data is small, both neural net methods learn similar set of features and hit an upper limit with regards to increasing accuracy. We anticipate varying performances with larger training data.

Evaluation To ensure our prediction results could be trusted, we ran GO enrichment analyses using the Gene Onotology Consortium tools and the Gorilla, gene ontology and enrichment analysis tool on the 6000 unlabeled genes. However, even though we were able to find high level cellular processes for these genes, we were unable to find cell cycle specific details on them. This drives home the point that a pipeline for classifying time series gene expression data to a specific cell cycle phase is a current limitation and our project helps in bridging that gap.

Conclusion

Our results show that even though we were able to successfully classify using a pretrained LSTM with satisfactory results, our approach of using CNN performs better than other established methods. This goes to show that deep learning models can be successfully used for classification of genomic data even if enough samples are not available by applying regularization techniques like Dropout. In the future, we would like to try other methods of semi-supervised learning like co-training and self learning to see if we can obtain better accuracies as compared to the CNN model. It would also be useful to find more samples of labeled data to effectively inform in the clinical setting. Finally, we would like to validate the top-ranking predictions of our CNN on the remaining yeast genome as actually belonging to a cell cycle phase (genes identified post-1998 in literature as belonging to a specific cell-cycle phase).

References

1. Z. Bar-Joseph, A. Gitter, and I. Simon, “Studying and modelling dynamic biological processes using time-series gene expression data,” *Nature Reviews Genetics*, vol. 13, no. 8, pp. 552–564, 2012.

2. I. Androulakis, E. Yang, and R. Almon, “Analysis of time-series gene expression data: methods, challenges, and opportunities,” *Annu. Rev. Biomed. Eng.*, vol. 9, pp. 205–228, 2007.

3. R. J. Cho, M. J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrielian, D. Landsman, D. J. Lockhart *et al.*, “A genome-wide transcriptional analysis of the mitotic cell cycle,” *Molecular cell*, vol. 2, no. 1, pp. 65–73, 1998.

4. K. Y. Yeung and W. L. Ruzzo, “Principal component analysis for clustering gene expression data,” *Bioinformatics*, vol. 17, no. 9, pp. 763–774, 2001.

5. C. Orsenigo and C. Vercellis, “Time series gene expression data classification via l 1-norm temporal svm,” in *IAPR International Conference on Pattern Recognition in Bioinformatics*. Springer, 2010, pp. 264–274.

6. P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher, “Comprehensive identification of cell cycle–regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization,” *Molecular biology of the cell*, vol. 9, no. 12, pp. 3273–3297, 1998.

7. T.-h. Lin, N. Kaminski, and Z. Bar-Joseph, “Alignment and classification of time series gene expression in clinical studies,” *Bioinformatics*, vol. 24, no. 13, pp. i147–i155, 2008.

8. A. Schliep, A. Schönhuth, and C. Steinhoff, “Using hidden markov models to analyze gene expression time course data,” *Bioinformatics*, vol. 19, no. suppl 1, pp. i255–i263, 2003.

9. B. Knab, A. Schliep, B. Steckemetz, and B. Wichern, “Model-based clustering with hidden markov models and its application to financial time-series data,” in *Between Data Science and Applied Data Analysis*. Springer, 2003, pp. 561–569.

10. M. Schader, “Between data science and applied data analysis.”

11. K. Yang and C. Shahabi, “An efficient k nearest neighbor search for multivariate time series,” *Information and Computation*, vol. 205, no. 1, pp. 65–98, 2007.

12. Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, “Time series classification using multi-channels deep convolutional neural networks,” in *International Conference on Web-Age Information Management*. Springer, 2014, pp. 298–310.

13. Z. Cui, W. Chen, and Y. Chen, “Multi-scale convolutional neural networks for time series classification,” *arXiv preprint arXiv:1603.06995*, 2016.

14. Z. C. Lipton, D. C. Kale, and R. C. Wetzell, “Phenotyping of clinical time series with lstm recurrent neural networks,” *arXiv preprint arXiv:1510.07641*, 2015.

15. Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, and G. Cottrell, “A dual-stage attention-based recurrent neural network for time series prediction,” *arXiv preprint arXiv:1704.02971*, 2017.

16. A. Gupta, H. Wang, and M. Ganapathiraju, “Learning structure in gene expression data using deep architectures, with an application to gene clustering,” in *Bioinformatics and Biomedicine (BIBM), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1328–1335.

17. F. Petitjean, G. Forestier, G. I. Webb, A. E. Nicholson, Y. Chen, and E. Keogh, “Dynamic time warping averaging of time series allows faster and more accurate classification,” in *Data Mining (ICDM), 2014 IEEE International Conference on*. IEEE, 2014, pp. 470–479.

18. X. Xi, E. Keogh, C. Shelton, L. Wei, and C. A. Ratanamahatana, “Fast time series classification using numerosity reduction,” in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 1033–1040.