Predicting Cell Type-Specific Chromatin States from Genetic Regulatory Networks

Christopher Probert, Anthony Ho

Introduction

The human body consists of hundreds of unique cell types with unique characteristics, but yet all contain the same DNA sequence. This disparity is created by cell type-specific gene regulation, which drives complex patterns of expressing certain gene sets in specific cell types, and different gene sets in others. An important variable in this regulatory program is the physical state of chromatin, a complex of DNA and proteins inside the nucleus of cells.

The packaging of a human genome into chromatin allows different regions of the genome (in particular, enhancer elements) to be made accessible or protected from different gene regulators, which are proteins that bind DNA and activate or repress expression of genes. Understanding the factors affecting enhancer chromatin state may reveal processes that govern cell differentiation.

Here, we seek to predict chromatin states in diverse cell types from regulator gene expression profiles and prior information about regulatory gene physical characteristics. We built a model based on a boosted alternating decision tree (ADT) to predict the chromatin state of genomic enhancer regions, using motif composition and cellular gene expression as input features.

Dataset

Our data is derived from recent large-scale genomics projects by the two sequencing consortia (Epigenomics Roadmap[4] and ENCODE[3]). The primary results are data from various types of sequencing experiments across 270 different cell lines.

Chromatin states: Chromatin states are the labels we wish to predict. DNase-seq experiments quantify chromatin state by degrading DNA at sites not protected by chromatin structure. (DNase-seq data from ENCODE, DNase-seq data from Epigenomics Roadmap)

Regulator expression: We use regulator gene expression as a feature to capture variation across different cell types. ENCODE RNA-seq data measures gene expression for 200 cell types. For the remaining 70 cell types, we used regularized logistic regression models developed by the Kundaje lab at Stanford[6] to impute expression values from other epigenomic state information. (ENCODE RNA-seq data)

Motif presence: We use motif presence as a feature to capture variation between enhancer regions. The input data are a set of systematically discovered human regulator binding motifs[5].

Enhancer regions: We used the human genome for enhancer annotations and sequences. (genome hg19)

Features and Preprocessing

Feature Space

Our enhancer-cell type training examples have both intragenomic and intergenomic variation: they are from both different positions in the genome, and from different cell types. To capture both sources of variation, we use two feature spaces: motif occurrence per enhancer (to capture variance by genome position), and regulator expression per cell type (to capture variance by cell type). Thus, for each enhancer-cell type training example, our model has a chromatin state label and two feature vectors (200 gene expression values and 500 motif presence values).

We believe these features are appropriate because they capture variation on both the genomic enhancer region and cell type axes. For performance, we represent the feature space as three matrixes (Figure 1), where the goal is to predict the enhancer state matrix from the inner product space of the motif occurrence and regulator expression matrices. The model can be represented as $Y_1 \sim F(X_1^T \cdot X_2^T)$, where the goal is to learn the function F.

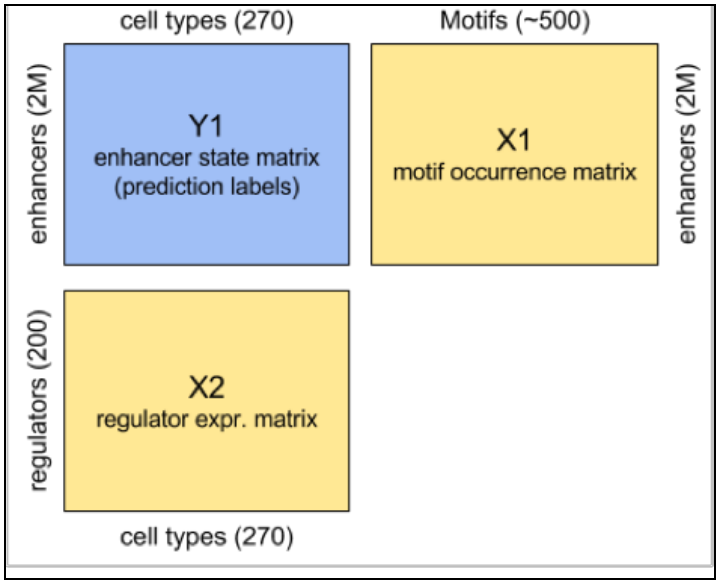

Figure 1: Feature Space Overview. Each example is an enhancer-cell type pair (Y_1). This corresponds to an enhancer motif presence vector (X_1), and a cell type gene expression vector (X_2).

Preprocessing

Regulator Expression: We normalized RNA-seq data for each cell type using standard techniques (FPKM). We used a list of 200 ENCODE transcription factors[3] as our list of regulator genes, and extracted expression values for these genes in each cell type as a matrix. We used a 0.5 FPKM threshold to binarize expression values in the regulator expression matrix (X_2).

Motif occurrence: We searched genomic enhancer regions using an HMM model for each motif from the set of human regulatory motifs[5]. We binarized motif presence in each genomic enhancer region to obtain the motif occurrence matrix (X_1).

Enhancer state: We used pre-processed DNase-seq binarized peak calls to obtain genomic chromatin state. We intersected this binarized chromatin state data with genomic enhancer annotations to obtain binarized chromatin state for each enhancer (Y_1).

Model Overview

We wish to predict the chromatin state of a given enhancer-cell type pair from information about the motif structure and cell type regulator expression. The input for the model is a regulator expression vector and a motif composition vector for a previously unseen enhancer-cell type pair. The output of the model is an enhancer chromatin state, which is a binarized value. The model is trained on labeled chromatin state-cell type examples.

Response Variable Function

Let Y be the chromatin state response variable, C be the cell type expression vector, and M be the motif occurrence vector (where Y, C, R are binarized). Then, for one (M, C) pair, our model is $Y \sim F(M, C)$. Extending this to the feature space described in Figure 1, this can be written using the the inner product space $X_1^T \cdot X_2^T$ as:

$$Y_1 \sim F(X_1^T \cdot X_2^T)$$

Boosted Alternating Decision Tree

Alternating decision trees are an extension of decision trees, where condition nodes R_i have true/false outputs, and prediction nodes carry real-valued weights a_i. The hypothesis $h(x)$ is generated by summing over all condition/prediction node pairs in an ADT A:

$$h(x) = \sum_i^A R_i(x) \cdot a_i$$

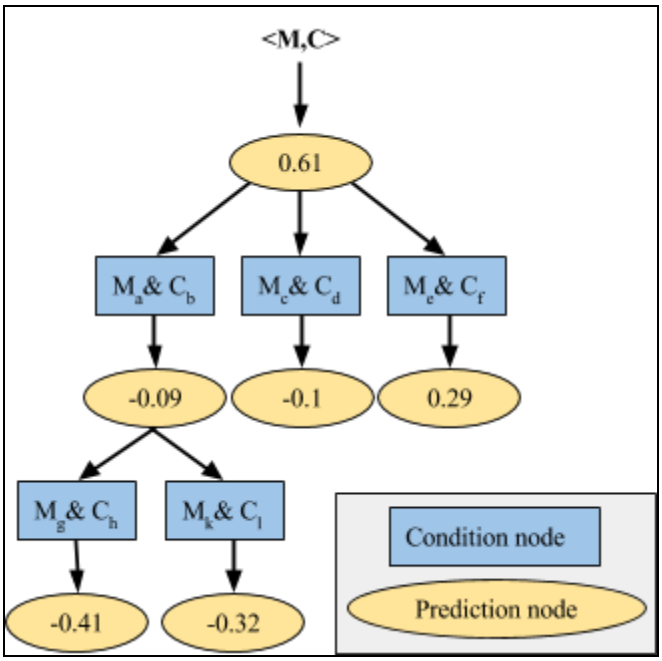

Figure 2: ADT Architecture Overview ADTs have prediction nodes a_i (yellow) as their roots and leaves, which carry real-valued weights. The condition nodes R_i (blue) use a rule based on one M feature and one C feature.

Model Modifications

We made two modifications to the base algorithm[1,2]: 1) only using positive conditional logic in condition nodes, and 2) using a different Z objective function that is faster to calculate and only includes weights for examples passing the condition node.

1) Positive conditional logic makes the model more representative of a biological system, because we are trying to model the effects of (motif, regulator) pairs on cell type decisions. The absence of a motif or regulator does likely not confer as strong of an effect as presence. Reproducibility and sensitivity issues with sequencing data also introduce noise that may create false negatives, and a model without weights on negative conditions is likely more robust to these errors. A regular ADT[1,2] has prediction nodes for both true and false outcomes of a condition node, whereas our model has only prediction nodes for true outcomes (e.g. $\forall i : R_i(false) = 0$).

2) The algorithm seeks to minimize the Z (loss) objective function. The regular loss function[1,2] is: $z = 2 \cdot \left(\sqrt{W_+(p \wedge c) W_-(p \wedge c)} + \sqrt{W_+(p \wedge \neg c) W_-(p \wedge \neg c)} \right) + W(\neg p \vee \neg c)$, where $W(x)$ is the sum of weights for examples meeting condition x, and $W_{+/-}$ are sums for positive or negative examples only. We used the function $z = 2 \cdot \sqrt{W_+(p \wedge c) W_-(p \wedge c)} + W(\neg p \vee \neg c)$ instead, as we are only interested in positive examples, and this requires fewer searches of a large matrix (2.3M X 270).

Implementation

We implemented the boosted ADT model in Python, using Numpy sparse matrixes to reduce memory footprint, and the Gnumpy[7] GPU matrix algebra library to speed up implicit inner product steps on the large feature matrixes.

Results

We used two different methods to train/test our model: 1) random holdout of training examples, and 2) Leave One Out Cross-Validation (LOOCV) by cell type. These methods present two possible different use cases for the model, and have different results. As our dataset is unbalanced, we used Balanced Error Rate (BER) for error analysis.

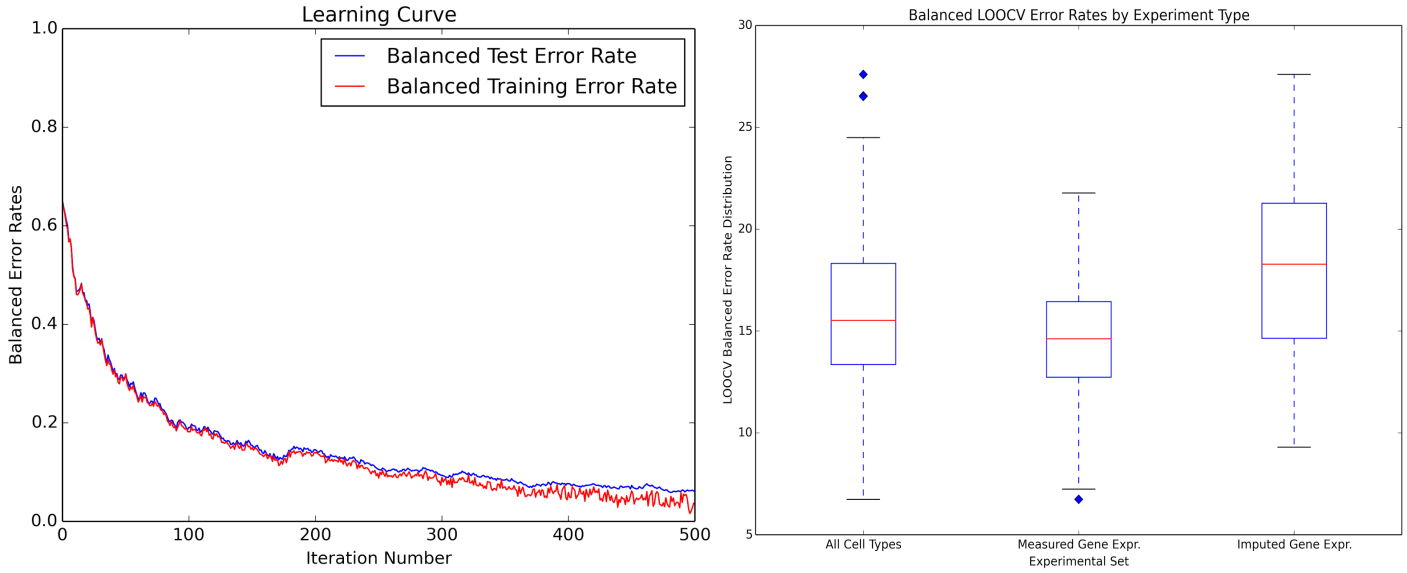

Figure 3: Error Rate Analysis of Two Scenarios. The learning curve (left) shows Balanced Error Rate (BER) for 5% random holdout of training examples. Our criteria for convergence was met at ~300 iterations, however we continued training to observe overfitting. The BER distribution (right) shows BERs for cell type LOOCV at convergence. The middle and right distributions are split by measured (middle) and imputed (right) gene expression.

Discussion

In the random holdout case (Figure 3, left), the model achieves a 0.05-0.08 BER on the test set, with minimal overfitting (0.035 training BER). Since the ADT algorithm adds one condition node and one prediction node at each iteration, iteration number is a direct measure of model complexity. As expected, overfitting (the separation of training and test error) increases with model complexity: at 250 condition nodes training and test BER are within 0.02, but at 500 condition nodes training balanced error is 0.06 less than test balanced error.

In previously unseen cell types, the model achieves a 0.15 average LOOCV test BER. In this setting, the model exhibits much stronger overfitting (0.034 average training BER at convergence, .116 higher than training error). The balanced error rate is higher in cell types with imputed (0.185) expression than with measured expression (0.149) (Figure 3, right), suggesting that using imputed expression features decreases accuracy of chromatin state predictions.

Conclusion

We have shown that boosted ADTs can predict chromatin states using two methods:

Random holdout: Our model performs well (5-8% test BER at convergence) for chromatin state prediction when data points are randomly held out of training. This method is useful for several applications, such as denoising chromatin state data, identifying irregular chromatin state points, and learning about how specific motif/regulator pairs confer chromatin state and cell type specificity.

Previously unseen cell types: Intuitively, predicting chromatin states of unseen cell types is a significantly harder problem than random data holdout, because the new cell types may have different regulatory dynamics than the cell types trained on. The model assumes that the response variable $Y \sim F(M, C)$ is from the same distribution (Y, M, C) in the training set as in the test set, and this may not be true if the unseen cell type has different regulatory dynamics.

Future

Given more time, we would like to improve performance on the LOOCV case by identifying patterns in errors and noising/ bootstrapping the training set. We would also like to try expanding the feature space to include some notion of local genomic context, such as nearby transcriptional activity. Moreover, regulators such as transcription factors are known to function in conjugates (called dimers or complexes), and by comparing accuracy of instantiations of our model with different factorizations of regulators or groups of regulators, we could elucidate combinatorial effects of regulatory gene activity.

Acknowledgements

We thank Prof. Anshul Kundaje and Kundaje Lab members for advice on this project.

References

[1] Y. Freund and L. Mason. The Alternating Decision Tree Algorithm. ICML, 1999.
[2] B. Pfahringer, G. Holmes and R. Kirkby. Optimizing the Induction of Alternating Decision Trees. KDD, 2009.
[3] ENCODE Consortium. An integrated encyclopedia of DNA elements in the human genome. Nature, 2012.
[4] Roadmap Consortium. The NIH Roadmap Epigenomics Mapping Consortium. Nature Biotechnology, 2010.
[5] Xie et al. Systematic discovery of regulatory motifs in human promoters. Nature, 2005.
[6] S. Kyriazopoulou-Panagiotopoulou, A. Kundaje. Unpublished work, 2014.
[7] T. Tieleman. Gnumpy: An Easy Way to use GPU Boards in Python. TNML 2010.