Identifying Transcription Unit Structure from Rend Sequencing Data

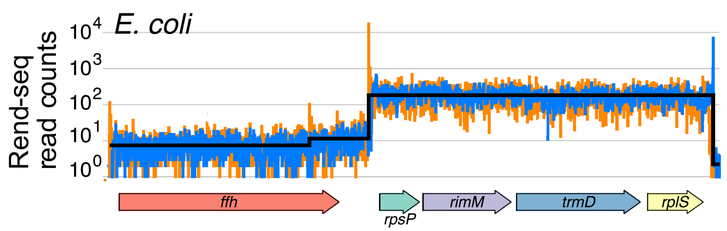
Travis Horst
thorst@stanford.edu
Department of Bioengineering, Stanford University

Summary

The goal of this project is to identify transcription unit initiation and termination sites within a genome through sequencing data to determine which genes are expressed together. Although partially known, identifying all transcription units in an organism can help create more accurate models of biologic behavior by better capturing interactions between coexpressed genes. Unsupervised and supervised methods were used to identify structure from transcript sequencing data. Results show that supervised learning methods performed better at identifying transcription start and stop sites and avoiding false predictions.

Data

Data comes from high throughput Rend sequencing of *E. coli* from Lalanne et al.[1] For every position in the genome, the data contains a read count for the 3' and 5' end of fragmented mRNA. A small subset of the data was labeled by hand with transcription unit initiation and termination sites (152 genome locations).



Example data with genes. Orange and blue are reads, black indicates TUs.[1]

Features

A few feature sets were tested on certain models. In some cases, the raw data for the 3' and 5' reads was used. The raw data follows a Poisson distribution (coming from count data) so a moving average along the position in the genome was used to transform to Gaussian distributions within genes, which also incorporated some positional information in the data. Further, two moving averages to the left and right of a point of interest were taken to account for potential shifts in distributions.

Models

Unsupervised Learning

- Divide data into groups of neighboring genes

DBSCAN:

1. Cluster points based on distance (**ε**).
2. Identify outliers that are part of groups with fewer than a minimum number of points (***min_points***).

Hidden Markov Model:

Number of hidden states dependent on genes in region.

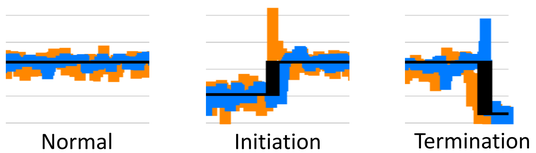
State transition matrix: ($p_{gene} << p_{transition}$)

$$\begin{bmatrix} 1-p_{gene} & p_{gene} & 0 & 0 & \dots & 0 \\ 0 & 1-p_{transition} & p_{transition} & 0 & \dots & 0 \\ 0 & 0 & 1-p_{gene} & p_{gene} & \dots & 0 \\ & & \vdots & & & \\ 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

Supervised Learning

- Sliding window along genome (varied to find optimal window)

- 3 classes:



- Oversample minority classes with SMOTE[2]

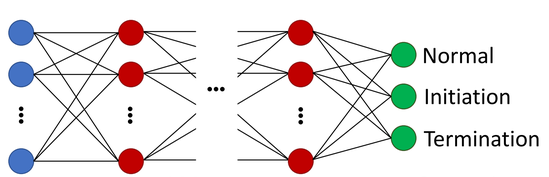
Multinomial Logistic Regression:

Probability for each class (c):

$$P\left(Y^{(i)} = c\right) = \frac{e^{\theta_c X^{(i)}}}{\sum_{k=1}^{3} e^{\theta_k X^{(i)}}}$$

Neural Network:

- Varied model architecture with sigmoid activation function



Input Layer
Nodes: features x window

Hidden Layers
2-4 layers, 5-30 nodes each

Output Layer
Softmax function

Results

	Validation Data		Test Data	
	Sensitivity	Precision	Sensitivity	Precision
DBSCAN	33.1	35.0	14.3	70.0
HMM	23.4	70.7	12.2	42.9
Log Reg	87.5	85.4	47.4	94.7
NN	90.0	81.8	55.3	80.8

Discussion

Unsupervised methods suffered from low precision and sensitivity. At first, supervised methods did not perform as well as expected. After annotating more data for training, performance improved but these methods potentially still suffer from a class imbalance problem due to the low number of spikes in the genome (roughly 1 every 1000 base pairs). Overall, performance for unsupervised methods was surprisingly low and higher sensitivity was expected for supervised methods on the test set. Because of the small sample size, the test and validation data might not be completely representative of the entire genome. The unsupervised methods could be improved by finding a way to encode positional dependence (ie. a sample is likely in the same TU as its neighbors).

Future Work

- Data processing – feature engineering and class imbalance
- Methods – convolutional neural network

References

1. Lalanne JB, et al. (2018). Evolutionary Convergence of Pathway-Specific Enzyme Expression Stoichiometry. *Cell* 173(3) 749-761.
2. Chawla, NV, et al. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *JAIR* 16 321-357.