Inferring Trascription Factors from Gene Expression Data

Nathan Wilkinson

***Abstract*—A better understanding of transcription factors and the genetic network inside human cells would allow synthetic biologists to work with greater confidence and precision. In this paper, I investigate the use of gene expression data to infer transcription factors in human cells. The central hypothesis of this paper is that when a gene is artifically underexpressed/overexpressed, then any genes whose expression levels were significantly impacted by this perturbation could potentially be activated or repressed by the perturbed gene. In addition, I provide open source Python code that can be used to extract information from the gene expression data about potential transcription factors in human cells.**

***Index Terms*—Transcription Factor, Gene Expression**

I. Introduction

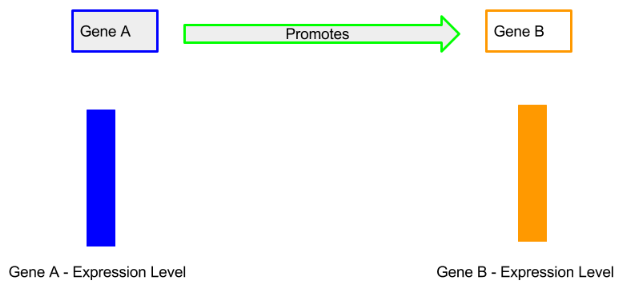
Transcription factors are proteins that control the rate of transcription from DNA to RNA. DNA can be thought of as a blueprint that specifies how human cells should build proteins. RNA acts as a messenger that takes the information contained in the DNA to a cell's ribosomes where proteins are made using the information in the RNA. Therefore, if the rate of transcription from DNA to RNA is altered, this results in a different production level of the gene encoded by the DNA. Proteins are like the building blocks for cells.

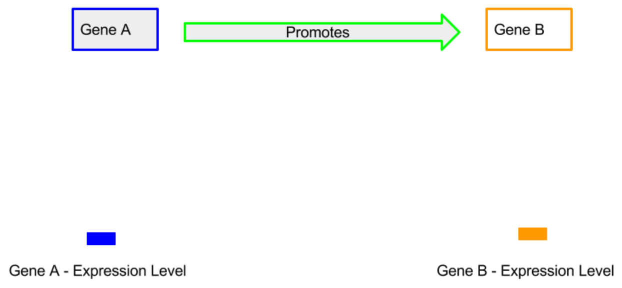
Many human genes act as transcription factors for other genes in human cells. This allows a complex network to be created in which genes regulate other genes. This network is responsible for modifying the production of genes so that the correct amount of each protein is produced. In synthetic biology, these genetic networks can be influenced by artificially modifying the production of genes. In addition, genetic networks could be leveraged to affect desired changes in the human cells by chaining together the effects from various transcription factors.

II. Methods

The Connectivity Map (CMap) at the Broad Institute is creating a genome-scale library of cellular signatures that catalogs transcriptional responses to chemical, genetic, and disease perturbation. To date, the library contains more than 1 Million profiles resulting from perturbations of multiple cell types. I used a subset of the CMap data to identify possible transcription factors. I identified candidate transcription factors based on which genes' expression levels were most changed by perturbations to the cell.

These graphics illustrate the hypothesis of this project. Namely, if by underexpressing gene A, gene B is also underexpressed, then we can infer that gene A activates gene B. This concept is trivially extended to overexpression as well.





First, I explored different methods of getting the CMap data. I investigated the use of the touchstone web app and the CMap API, however they were not able to provide the data necessary. I settled on using a downloadable dataset from https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE70138. I downloaded the GSE70138_Broad_LINCS_Level5_COMPZ_n118050x12328_2017-03-06.gctx.gz data. Then, I used the python package at https://github.com/cmap/cmapPy to interact with the dataset.

It turned out to be a lot of work to get the relevant data out of this dataset. After figuring out the API, the file was too large to be loaded into my laptop's memory, so I had to write some custom parsing code and save interesting information as I parsed the file. I pulled out all the expression signatures in which the perturbation was a knocked down gene using shRNA or an overexpressed gene. I also only used the signatures that were perturbing ”landmark” genes. The landmark genes are those that the CMap team measures directly. The expression levels of genes that are not landmark genes are inferred from the measurements of the landmark genes. Many of the expression level measurements were made on cells which had the same gene perturbed. This means that there were duplicate expression changes which had to be aggregated in some way. I also only wanted to pull out the results that were actually significant instead of giving back too much information to the user. I decided to save the top 100 activators and repressors as measured by expression levels for each signature. To aggregate duplicate expression changes, I averaged the expression levels for the same gene.

After I had extracted the expression levels of genes, I saved them to avoid having to reparse the large datafile again. Then, I wrote a small Python class that abstracts users from the details of getting gene expression information. The Python class exposes the following functions. Given a gene id like BRCA1, gene info returns the a list of the genes that are activated or repressed by the given gene, or None if there weren't any that made the top 100 cutoff. Each gene returned has the gene id and the score associated with it. If calling gene info with gene A returns gene B with a negative score, then gene A is a repressor for gene B. If calling gene info with gene C returns gene D with a positive score, then gene C is a promoter for gene D. gene promotes and gene represses return the same results as gene info but will only return the promoters or repressors respectively. gene vis promotes and gene vis represses are methods that will plot the promoted or repressed genes for the given gene id. You can pass an optional parameter n if you want to see fewer or more genes in the plot.

III. Walkthrough

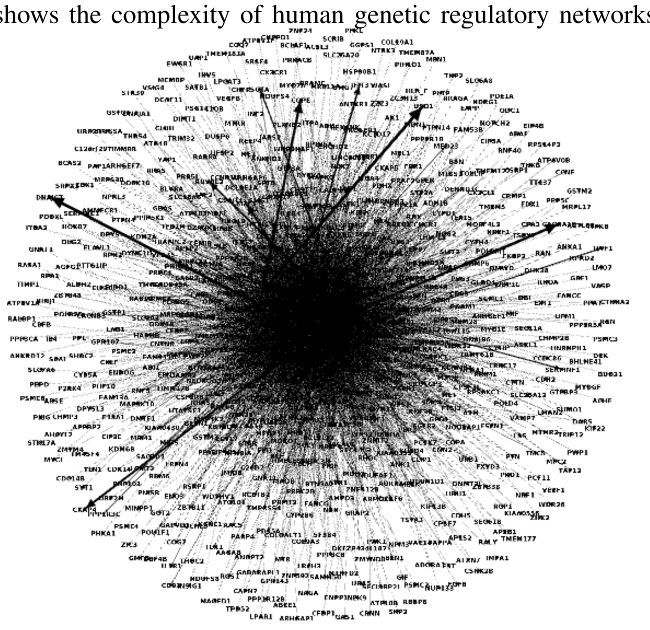
The goal of this project was to provide a framework that could be used to allow biologists to discover novel transcription factors and to enable synthetic biologists to better understand and predict how changes to human cell biology will affect the cell. I will provide a brief walkthrough demo of how the framework I built can be used.

First, clone the git repo https://github.com/nathanwilk7/bio-project. Then cd into the final-project directory. Second, use wget or curl to download the files (these files are big, probably run them in the background concurrently) at https://s3.amazonaws.com/mod-bio-project/averaged_promoters.json and https://s3.amazonaws.com/mod-bio-project/averaged_repressors.json. I will assume that these files have been downloaded to the current working directory. Then, if you have jupyter notebook run the final.ipynb file in jupyter notebook to explore the API (you need to install the matplotlib and networkx packages, you can use pip). If you don't have jupyter notebook, then you can just run the final.py file, but you won't get any fancy visualizations. Feel free to modify the source code and run custom queries. It will probably take a bit to initialize the Query object, but queries should be near instantaneous after the data has been loaded.

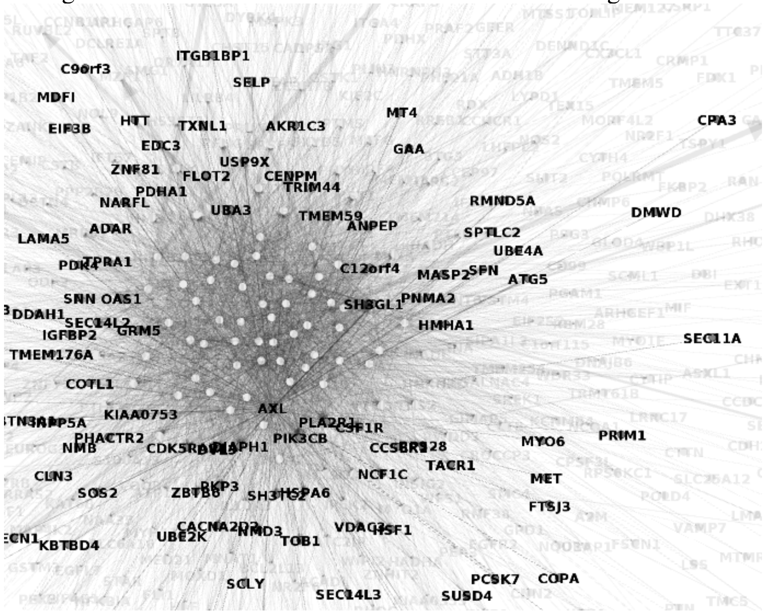
IV. Conclusion

To finish up, I will show some visualizations and output data from the queries. The purpose of this is to show that this framework reaches the goal of enabling biologists to investigate transcription factors.

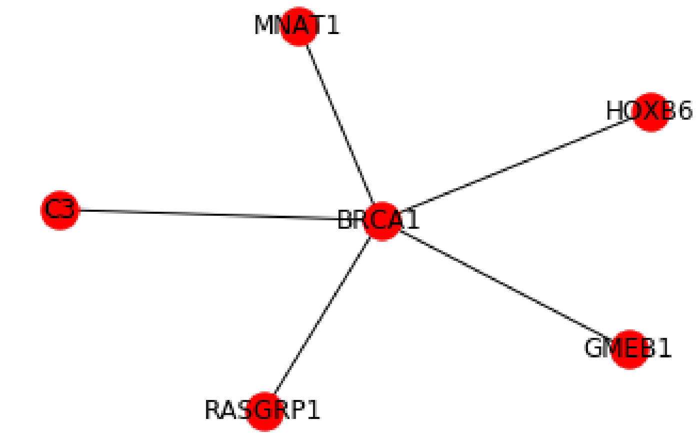
This first graphic shows a subsampling of connections between different genes as edges. Each edge represents either an activation or repression relationship. This graphic shows the complexity of human genetic regulatory networks.



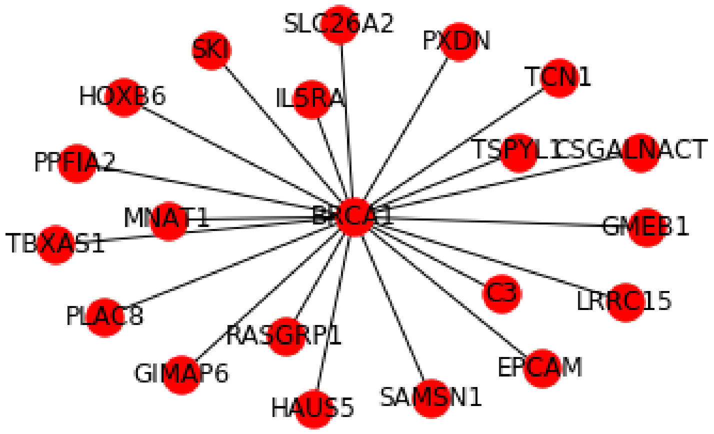
This next graphic shows how some genes have their fingers in a lot of pies, or in other words, some genes affect any other genes. It highlights the gene AXL and its connections to other genes.



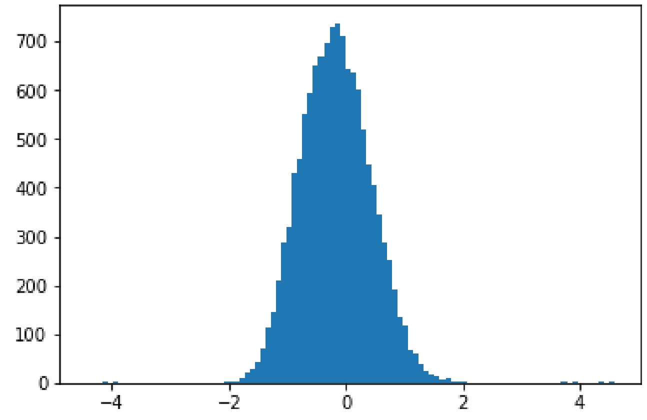
This shows the jupyter notebook visualization available through the API for BRCA1. In this case, there are 5 genes shown that are repressed by BRCA1.



This is the same type of graphic as before, except there are 20 genes shown that are repressed by BRCA1.



This histogram shows the distribution of scores associated with a gene. The tail ends of the distribution are the activators and repressors.



In conclusion, this project provides a framework and dataset that allows biologists to find candidate transcription factors in human cells. For future work, I would like to verify the correctness of the data on an orthogonal dataset and to use more statistically rigourous methods for aggregating signatures.

Acknowledgment

I am very grateful to Professor Myers for teaching the Modeling Biological Networks course and for his assistance and feedback during this project. I also acknowledge the help of the CMap team and their very helpful email collaboration.