Predicting Cell Type and Extracting Key Genes using Single Cell Multi-Omics Data and Graph Neural Networks

A senior thesis presented
by
Hossam A. Zaki

to
The Department of Computer Science

in partial fulfillment of the requirements
for the degree of
Bachelors of Science
in the subject of
Computer Science

Brown University
Providence, RI
May 2022
Predicting Cell Type and Extracting Key Genes using Single Cell Multi-Omics Data and Graph Neural Networks

Abstract

Single-cell RNA sequencing, along with single-cell chromatin accessibility sequencing, has revolutionized our approach to understanding cellular processes in biological and biomedical research. One important problem in this realm is classifying cell types. With more cell types being discovered, quickly identifying them has become a key priority. Additionally, extracting cell-type-specific genes can prove useful in studies of differentiation, or in developing therapeutic targets. Previous studies have shown that incorporating gene regulatory network information increases performance in cell-type classification. However, extraction of cell-specific genes has not yet been done. In this study, we propose an end-to-end Graph Neural Network (GNN) to use gene interaction networks to predict cell type. Following the training of the model, we run GNN interpretation methods to extract cell-type-specific genes. Additionally, we integrate multi-omics data to improve model performance and integrate more information. Our model performs as well as the baseline in many datasets. The model is also able to extract the marker genes separating the cell-type clusters in the datasets. The integration of chromatin accessibility data as been shown to improve performance and result in a better interpretation.
Contents

0 Introduction 1

1 Methods 5
  1.0.1 Gene Regulatory Network Inference 6
  1.0.2 Input Generation 6
  1.0.3 Graph Neural Networks (GNN) 7
  1.0.4 GNN Interpretation 9

2 Experimental Setup 10
  2.0.1 Datasets 10
  2.0.2 Evaluation Metrics and Benchmarks 12

3 Results 14
  3.0.1 Mouse Brain 14
  3.0.2 Baron Human 16
  3.0.3 Baron Mouse and Muraro 17
  3.0.4 SNARE-Seq performance 18

4 Conclusion 20

Appendix A Supplemental Figures 22

References 27
Acknowledgments

I would like to thank Professor Ritambhara Singh and Pinar Demetci for their support, guidance, and mentorship throughout this project. Would like to thank Chris Chae for helping me build this project out. Thanks to Erica Larschan for introducing me to the wonders of genetics. Last and most certainly not least, would like to thank family and friends for the support during these crazy four years.
Introduction

Each cell type carries out a distinct set of functions in the organism by following cell-type-specific regulatory mechanisms. Gene expression is regulated by a variety of pathways, including histone modification, transcription factor activation/inactivation, and micro-RNAs, among many others. Gene regulation can be modeled as a graph, termed a gene regulatory network (GRN). Simply, two genes are connected if there's an interaction between them. GRNs can include more information in
the edges including up/down-regulation. Dysregulation or misregulation of these mechanisms can result in developmental disorders, cancers or neurodegenerative diseases \(^{10,14}\). It is also possible for cells to shift their gene expression patterns as a result of their microenvironment through a series of epigenetic modifications \(^8\). Next-gen sequencing has given researchers a glimpse into how cells are regulated, and how they are related to human development and other diseases.

The introduction of single cell sequencing (scRNA-seq) has revolutionized the study of gene expression. Traditional methods of genomic profiling report an average molecular readout for a given population of cells, missing the key differences between individual cells. Single-cell sequencing on the other hand can extract the expression for each individual cell, allowing us to get a mechanistic view of biological processes, such as development, gene regulatory networks, or cell-specific mechanisms. The first use of single cell sequencing focused solely on RNA, which although can capture a subset of the regulatory mechanisms in a cell, other ‘omics data is required to fully capture the complexity of gene regulation. Recent advances in single-cell assays has enabled access to a growing number of molecular dimensions, including genetic sequence, transcription, epigenome, proteome, and phenotype, and a high resolution across many thousands of cells. In this study, we used SNARE-seq \(^4\), which extracts both mRNA expression as well as single-nucleus chromatin assembly data. By analyzing multiple aspects in the same cell across different cell types, we will get a mechanistic view of biological processes, such as those mentioned previously. In doing so, we may learn about how and why a cell adopts a particular state, and why it may deviate from it.

Significant amount of work has been done to analyze scRNAseq data, including cell clustering, differential expression analysis, and trajectory analysis. Many cell-types and sub-types have been
identified already along with their transcriptomic profile, so the focus has shifted to learning cell-
type-specific regulatory profiles based on classification-based methods. Classification is a supervised
learning method, which unlike unsupervised methods, can efficiently and accurately identify unla-
beled cells, even if those cells are from different experiments. Upon training a classifier for a cell type,
the model will be able to learn the regulatory signature of each cell-type, which can then be extracted
by interpretation methods, and analyzed for biological significance.

For this study, we propose a Graph Neural Network (GNN) to predict cell-type and extract key
genes from multi-omic and scRNA-seq data and GRNs. GNNs iteratively aggregate neighbor infor-
mation via message passing schemes to learn a topology-aware latent representation of nodes. The
node representations can be used to predict edge connections, node labels, and in our case, graph
labels. The graph will be represented by a gene regulatory network to model gene-gene interactions.
This will allow the model to encode the information in the topology of the regulatory networks. We
hypothesize that this will perform better than a standard neural network as instead of representing
genes as separate entities, GNNs are able to represent genes based on their interactions with other
genes, which is more biologically significant. Many previous methods have ignored this inherent
aspect of gene regulation. The integration of multi-omics data can also make significant use of this
premise of gene-interactions. In our case, the connection of genes that have open chromatin regions
can provide further insights into chromatin accessibility information for each cell type.

In addition to making our method more biologically significant, GNNs also allow for model in-
terpretability. Most other Deep Learning based models are black-boxes, meaning that the model
may perform well for a task, but the researcher is unable to determine how the model is coming
to its decision. Because GNNs are graph-based, we are able to exactly view which nodes are most important to the decision. The current state-of-the-art method for GNN interpretation is GNNExplainer \(^{18}\). By utilizing GNNExplainer, we will be able to directly extract genes and sub-networks that are specific to each cell type. This will allow biologists to determine potentially which genes to target for a study.

Previous models have utilized GNNs to predict cell-type. sigGCN \(^{17}\) combines a GNN and a fully connected layer to predict cell-type. It utilizes a decoder on the output of the GNN to further train the GNN. However, the drawback of this method is that it prevents the ability to run interpretation. Additionally, it only utilizes scRNA-seq, and not other multi-omic datasets. Our model allows for interpretation, and also utilizes multi-omics data.

The performance of this model was evaluated against 5 single-cell datasets, and 1 SNARE-seq datasets.
Methods

Previous computational methods to predict cell-type from single-cell/multi-omic data neglect to capture significant gene information. To take this into account, we create a gene regulatory network, with nodes being genes, and edges representing gene-gene interactions. The features of the nodes will be single-cell counts and/or chromatin accessibility counts. After training, the model will be able to predict cell type from single-cell/multi-omic data, and also pick out important genes in the
dataset, which allows researchers to deduce significant genes without having to experimentally determine them.

1.0.1 Gene Regulatory Network Inference

In order to model gene-gene interactions in a cell, we generated a gene regulatory network (GRN) using the grnboost2 algorithm. The input to the algorithm is the single-cell expression matrix along with a list of known transcription factors (TFs) for the species. grnboost2 utilizes a tree-based regression model to predict an expression profile for each gene in the dataset. The algorithm then returns a matrix, with each row being a transcription factor, a gene that the transcription factor regulates, and an importance score representing the confidence the model has in the association between the TF and the gene. We calculated the mean and standard deviation of the importance values, and chose the top $H$ interactions with an importance value greater than one standard deviation above the mean. From this, we created a graph connecting TFs and genes as our gene interaction graph and the input to our GNN.

1.0.2 Input Generation

For every cell in the dataset, the same graph constructed as detailed in the previous section was used, but the features for each node would differ based on the cell. For both expression and scATAC-seq data, the values were transformed into the log scale, and then divided by the maximum value in the dataset, as per the preprocessing from sigGCN. Chromatin reads were translated into genes using the Ensembl API using mm38 or hg19 depending on the dataset. Reads were translated to genes...
using the transcription start and end site +/- 20kb. If a chromatin region overlapped with the gene region, we would put the chromatin accessibility data in that gene. If many chromatin regions overlapped, we’d sum the chromatin accessibility data together. The buffer of 20kb was added to take into account promoters, enhancers, and other regulatory elements up and downstream of the gene that could be included in the chromatin accessibility data. The result is a graph with node features being either a matrix with 1 value, or 2 values, excluding/including scATAC-seq data, respectively, for each cell. This is then passed into the GNN.

1.0.3 Graph Neural Networks (GNN)

The proposed model consists of 3 pairs of convolution and non-linear layers, followed by a pooling layer, a dropout layer, and a classification head. The convolutional layers used were of the GraphSAGE\(^6\) formulation. This layer is especially useful for larger graphs that contain rich information. In large graphs, sampling neighbors that are \(N\) hops away, where \(N\) is the number of convolutional layers, may be too large to store in memory, and large enough where the model will not learn anything reasonable. GraphSAGE approaches this issue by sampling neighbors up to the \(N\)-th hop. It starts by sampling a fixed \(k\) number of 1-hop neighbors, then for each of these neighbors, samples another \(k\) neighbors, and continues for \(N\) times. A potential drawback to this method is that it may be less efficient as there could be multiple computations performed on the same node. The GraphSAGE formulation can be summarized by the following:

\[
x_i' = W_1 x_i + W_2 \cdot \text{mean}_{j \in \mathcal{N}(i)} x_j
\]
Following each convolutional layer, a rectified linear activation function (ReLU) will clamp all negative values to 0, which will speed up training, and can prevent the vanishing gradient problem often seen with sigmoid or Tanh activations functions.

Following the 3 pairs of convolution and non-linear layers, a global max-pool layer is run, which will return the maximum value in the feature dimension for every node. It is summarized by the following function.

\[ r_i = \max_{n=1}^{N_i} x_n \]

The output of the max-pooling layer will then be passed through a dropout layer, which randomly zeros elements in the input vector with a probability of .5. In doing so, it will prevent overfitting to the training dataset. The resulting vector is passed into a linear layer, which will output the logits of each class.

The loss function used was Cross Entropy loss, which is summarized by the following

\[ L_{CE} = -\sum_{i=1}^{n} t_i \log(p_i), \text{ for } n \text{ classes} \]

where \( t_i \) is the truth label, and \( p_i \) is the softmax probability for the \( i^{th} \) class. An Adam optimizer with a learning rate of .001 was used to update the weights on the model.

All code was written with Pytorch Geometric\(^5\) to generate graphical datasets, and graphical neural networks, and Pytorch\(^15\) to run the model and update the weights. Models were run using the computational resources and services at the Center for Computation and Visualization, Brown University.
1.0.4 GNN Interpretation

As mentioned previously, one of the hallmarks of using GNNs is their ability to interpret the model and its weights. For the purpose of this project, we opted to use GNNExplainer\textsuperscript{18}. The main idea behind GNNExplainer is to reduce redundant information in a graph, which doesn’t impact the overall decision. This goal can be distilled down to the idea of finding a subgraph in the original graph that minimizes the difference in the prediction scores between the subgraph, and the original graph. GNNExplainer learns a mask for edges and for features. This means that the mask contains the relative importance of the nodes and the features. As such, we are able to determine the most important nodes using the feature mask, as well as the most important gene interactions in the edge mask.

Most important genes were determined by building the node mask, and finding the 20 most important genes for each cell. If the occurrence of a gene was reported, it was added to a counter specific for cell type. The output would be a gene, and the number of times it occurred in the top 20 most important genes.

For the scATAC-seq data, instead of using the node mask, we opted to go with the edge mask, as this would take into account both atac seq and gene expression features, as the node mask would account for each exclusively. Following this, the same procedure would occur.
2.0.1 Datasets

For the scRNA-seq aspect of the project, we utilized 4 different datasets. First, the Mouse Brain dataset consists of 3005 cells from the Mouse cerebral cortex. Cells were sequenced using the STRT method. The dataset consists of 19972 genes from 7 cell populations. Cells were clustered using t-Distributed Stochastic Neighbor Embedding (t-SNE), and marker genes were determined, based on
the differential gene expression of the clusters. Sequence data was obtained from Gene Expression Omnibus, with accession code GSE60361.

The following two datasets, termed Baron Human and Baron Mouse\(^1\) were taken from the pancreas of their respective species. Cells were sequencing using the inDrop protocol. Cells were then clustered using t-SNE, revealing clusters corresponding to cell types. Marker genes were determined by an iterative hierarchical clustering method that restricts genes enriched in one cell type from being used in another cell type. This ultimately resulted in 8,569 cells and 17,499 genes from 14 cell types for the Baron Human dataset, and 1,886 cells and 14,861 genes from 13 cell types for the Baron Mouse dataset. Sequence data was obtained from Gene Expression Omnibus, with accession code GSE84133.

The final scRNA-seq dataset, termed the Muraro dataset\(^1\) was also taken from the mouse pancreas. Cells were sequenced using the CEL-Seq2 protocol, and consists of 2,122 cells, and 18,915 genes belonging to 9 classes. Cell clusters were identified by calculating pairwise cell-to-cell distances using \((1 - \text{Pearson correlation})\). This resulted in well separated cells clusters after using t-SNE. Genes were also determined by looking at differential gene expression between one cluster and another. Sequence data was obtained from the Gene Expression Omnibus, with accession code GSE85241.

The multi-omics dataset we used is termed the Cell Mixture dataset\(^4\) from different human cell lines. There is a total of 1,047 cells and 18,666 genes belonging to 4 cell types. This data holds both scRNA-seq data, as well as scATAC-seq data to show chromatin accessibility. Distinct clusters were determined using t-SNE for both sc-RNA expression, and also scATAC-seq expression. Sequence and chromatin assembly data was obtained from the Gene Expression Omnibus, with accession code GSE85241.
Figure 2.1: PCA of data after converting chromatin coordinates in the Cell Mixture Dataset to Genes. Shows that the conversion to genes produces distinct clusters that can be separated using a neural network.

code GSE126074.

Shown in figure 2.1 is the PCA of the Atac-seq data after converting to genes. The relatively distinct clusters show that the translation from chromatin coordinates and genes worked, and a neural network can potentially learn the separations.

Test, train, and validation splits were determined using sigGCNs test and train split as shown in the data they released in the paper.

2.0.2 Evaluation Metrics and Benchmarks

The metrics used to evaluate the model were accuracy, area under the receiver operating characteristic curve (AUROC), and area under the precision recall curve (AUPR). We opted to use these 3 as in single-cell data, there can be massive data imbalances. Accuracy may not capture the performance of the model on the smaller classes, so using AUC and AUPR would allow us to capture these values. AUROC and AUPR were calculated using the sklearn \(^2\) Python package.

Our benchmark for our scRNA-seq models is sigGCN, which as mentioned previously, employs
a GCN along with a fully connected layer to predict cell type, while being aided by a decoder during training.

We evaluated our interpretation by comparing with the marker genes presented in their respective paper, as well as the agreement of the output of the interpretation between all the examples. We also utilized Enrichr to find coexpressed genes and conduct pathway analysis. Specifically for the Mouse Brain dataset, the lab published a website which allows users to query genes, and output expression in different cell types, which we used to further validate our data. This website can be found here.

Furthermore, to continue to validate our interpretation, we ran a Logistic Regression model using the top-10 genes picked out from GNNExplainer for each cell type to classify cells. The performance of this model was then compared against using all genes. If there was not a significant difference between the two performances, then it can be concluded that the genes extracted by GNNExplainer could accurately summarize the separations between cell types.
Results

3.0.1 Mouse Brain

Running our proposed model on the 3005 cells given in the Mouse Brain dataset, we achieved an accuracy of .95, an AUC of .99, and an AUPR of .9, as shown in figure 3.1 and 3.2, respectively. This was the first dataset we experimented with, and we initially had a mean pooling layer, instead of a max pooling layer. This would take the mean of each index in the feature vector instead of the
Figure 3.1: Mouse Brain ROC Curve. The model’s average AUROC on all the examples is .99, meaning that the model can distinguish the positive examples from the negative examples in a 1 vs all scenario. The baseline AUROC score is .5.

Figure 3.2: Mouse Brain PR Curves. The model’s average AUPR was .9, meaning that the model has good precision and recall. A good precision means that when the model predicts a positive class, it is correct. A good recall determines what proportion of actual positives were identified correctly.

maximum. This led to an accuracy of around .83. Thus, for further datasets, we utilized a max-pooling layer.

Based on these results, the model is able to accurately classify cell types with a reasonable accuracy. We were comparable to sigGCN on the AUC and accuracy, but had a slightly less AUPR. What this might mean is that the fully-connected layer of sigGCN allows for more robustness to imbalanced datasets.

Upon running GNNEsolver on this trained model, we were able to extract all marker genes shown in the paper or represented on the website mentioned earlier. The presence of the marker genes is a proof of concept that the model is able to pick out important genes. As such, the other
Figure 3.3: Top 20 genes that occur in the top 20 genes outputted by GNNExplainer for the Interneurons cell type. Our model correctly identifies cell-type marker genes in the Mouse Brain dataset, while also outputting biologically relevant genes.

genes outputted by GNNExplainer also prove useful. For example, there is proof that Ckmt1, a highly occurring gene as shown in figure 3.4, has a very high expression in the mitochondria of neurons\(^1\). Thus, this shows that the output of GNNExplainer, regardless of it being a known marker gene or not, is biologically significant.

3.0.2 Baron Human

Running our proposed model on the 8,569 cells achieved an accuracy of .93, and AUC of .99, and an AUPR of .92, as shown in figure 3.4 and figure 3.5.

In both metrics, we matched that of sigGCN. However, sigGCNs test split failed to include a couple of cells, hence the AUC curves showing a nan value. Each of those cells had < 10 examples in the dataset.
With respect to the interpretation, our model continued to perform well, retrieving the marker genes of 10/14 of the cells in the dataset. The other 4, as previously mentioned, had very few examples in the dataset, which does indeed make sense.

We evaluated the performance of this dataset using a regular Graph Convolutional Layer. We found that the performance was greatly decreased upon using this layer versus the SAGEConv. We hypothesize that this the case because it aggregates a significant amount of data from the neighbors, overwhelming the model, versus the SAGEConv formulation which samples the neighbors.

3.0.3 Baron Mouse and Muraro

The results of our model on both the Baron Mouse and Muraro datasets are shown in Table 3.1, and the figures are available for viewing in supplemental materials A. Baron Mouse dataset showed
the lowest performance out of all the datasets we tested on. The combination of a very low amount of data alongside with indistinguishable cell clusters made it difficult for the model to learn a proper representation of the data. As such, the interpretation did not work as well as other models, but still was able to pick out a few key genes. See figure A.1 and figure A.2.

The model performed very well on the Muraro dataset as well. There were 2 cell types with very few examples (epsilon and mesenchymal), and the model was unable to pick out the marker genes of those cell types. However, it was able to do so with every other cell type. See figure A.3 and figure A.4.

3.0.4 SNARE-Seq performance

When evaluating SNARE-seq, we ran two separate experiments, one with each node having one feature being gene expression, and another with each node having two features, one being gene expression, and another being chromatin accessibility data translated as genes. In this experiment, we found that the inclusion of ATAC-seq data improved the performance of the model as shown in table 3.2.

This means that inclusion of scATAC-seq data adds another dimension to the problem and additional data that the model deems useful. We also ran the model using only the scATAC-seq gene
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Accuracy</th>
<th>AUC</th>
<th>AUPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excluding Chromatin</td>
<td>.63</td>
<td>.82</td>
<td>.61</td>
</tr>
<tr>
<td>Including Chromatin</td>
<td>.7</td>
<td>.86</td>
<td>.67</td>
</tr>
<tr>
<td>Only Chromatin</td>
<td>.48</td>
<td>.69</td>
<td>.44</td>
</tr>
</tbody>
</table>

Table 3.2: Performance with Atac-seq data. The inclusion of scATAC-seq data improves the performance of the model over just scATAC-seq data or scRNA-seq data.

translations, and found that it performed the worst out of the three, meaning that only Atac-seq data is not enough to classify cells, and that it requires the inclusion of gene expression as well to yield a good result.

Upon running GNNExplainer on each model, we extracted the top 10 genes for each cell type and ran a Logistic Regression model to see if those features can accurately distinguish cell types. The Logistic Regression model using the genes picked out from the gene expression model achieved an accuracy of .93, while the Logistic Regression model using the genes from the combined model achieved an accuracy of .96. This means that the genes extracted from the combined model contain more information to accurately distinguish cells. This potentially means that the explanation in this the combined model is potentially better than in gene expression alone.
In this thesis, we propose a simple GNN model that uses gene regulatory networks and gene expression values to classify cell types, allowing for an interpretable machine learning model. We also found that the inclusion of multi-omics data increases the performance of the model and also improves the overall explanation that the model produces.

We compared the results of our sc-RNA experiments with sigGCN, which uses both a GNN
and a fully-connected layer. sigGCN performed slightly better than our model overall, with a few datasets where our performance was the same. However, our model is much more simple to implement and understand, and is also interpretable.

Our model was able to pick out most, if not all of the marker genes listed in the datasets’ respective papers, demonstrating that our model can pick out important genes in single cell datasets. This yields an important innovation for the field. This can be extended to a variety of fields. Being able to extract marker genes will give us insight into how genes differentiate, which genes are turned on/off in diseases, and/or giving ideas of new genes to target for novel experiments.

Furthermore, with the explosion of multi-omics data, we have shown that integrating both gene expression and multi-omics data can prove valuable when classifying cell types, and also improve the overall explanation outputted from the neural network.

As we continue to work on this project, we hope to experiment with more SNARE-seq datasets, and also other multi-omics datasets. We are also interested in experimenting with sparse attention models to learn a summarized graph for all cell types, and use that as our graph for classification.
Supplemental Figures
Figure A.1: Baron Mouse ROC Curve

Figure A.2: Baron Mouse PR Curves

Figure A.3: Muraro ROC Curve

Figure A.4: Muraro PR Curves
References


