Evaluation of Explainability Methods on Single-Cell Classification Tasks Using Graph Neural Networks

A thesis presented
by
Momoka Kobayashi
to
The Department of Computer Science
in partial fulfillment of the requirements
for the degree of Bachelor of Arts in the subject of Computer Science

Brown University
Providence, Rhode Island
May 2023
Evaluation of Explainability Methods on Single-Cell Classification Tasks Using Graph Neural Networks

ABSTRACT

The emergence of single cell sequencing data has led to new developments in using Graph Neural Networks (GNNs) to classify cells. In particular, these methods have been crucial for finding explanations that reveal underlying cellular mechanisms for GNNs, which are known to be “opaque” boxes. However, there is still a lack of research on how reliable the explanations are, mostly due to difficulties in obtaining datasets with known ground truth graphs.

To this end, this thesis evaluates two graph neural network explanation methods (GNNExplainer, PGExplainer) on synthetic datasets with a known ground truth network and varying levels of sparsity produced using SERGIO. A binary graph classifier is then trained using the GraphSage convolution network. After being generated using the aforementioned explanation methods, the explanations are then evaluated using fidelity and sparsity metrics.

The results revealed that GNNExplainer consistently outperformed the other methods. However, the fluctuations in the evaluation metrics due to the use of gene inference methods and sparsity in the datasets indicate that further evaluations are needed to test the reliability of explanation models for high sparsity, complex single cell data.
## Contents

1 Introduction .................................................. 1

2 Methods ....................................................... 4
   2.1 Construction of Synthetic Datasets ...................... 6
   2.2 Graph Neural Network .................................. 7
   2.3 Post-Hoc Explanation Methods ......................... 9

3 Experimental Setup ........................................ 11
   3.1 Datasets ................................................. 11
   3.2 Evaluation Metrics and Benchmarks ..................... 13

4 Results ....................................................... 16
   4.1 Fidelity .................................................. 17
   4.2 Sparsity ................................................ 20

5 Conclusion .................................................. 22

References .................................................... 26
I would like to express my deepest gratitude to Professor Ritambhara Singh for introducing me to the field of computational biology and her mentorship throughout this project. Thank you to Pinar Demetci and Hossam Zaki for their constant enthusiasm and support. I would also like to thank Dr. Erica Larschan for her valuable feedback and guidance. Finally, a big thank you to all of my friends and family for their support throughout my time at Brown.
Graph neural networks (GNN) have been increasingly used to solve more complex tasks. These networks are often used for real-world applications with high-stakes situations and it is important that we can both understand and trust their functionality. Foundational work to generate explanations for predictions made by GNNs has been carried out. Despite these efforts, the field of explainability in GNNs is still an emerging area and lacks standardized evaluation strategies.
GNN explanations are crucial in the classification of cell types from single-cell sequencing datasets.[7] Advancements in single-cell RNA sequencing have allowed for new opportunities in understanding gene regulatory mechanisms.[16] The increase in the availability of single-cell omics has opened up doors for finding new cell types and understanding cellular dynamics for areas such as developmental processes.[10] While traditional bulk methods of sequencing take average values across a mix of millions of cells, single-cell sequencing isolates each cell and performs sequencing on thousands of genes at a time.[14] Therefore, single-cell sequencing leads to higher-resolution data and more insights into the underlying gene regulatory mechanisms of cells.[3]

There has already been much research in the area of discovering new cell types through both clustering and classification-based methods. Clustering using classical ML methods including random forests, support vector machines, and k-nearest neighbors. These methods can compute cell similarities and label cells for small datasets with reasonable feature selection.[16] However, these methods are not scalable for larger datasets without preliminary feature selection.[14] Classification methods using deep learning models have also been studied. The benefit of using a deep learning model is that it does not require domain knowledge, and high-level features can be learned from the data rather than being selected.

The field is now transitioning from using only gene expression values to also including gene-gene interactions in Graph Convolutional Networks (GCN) for further analysis. GCNs can incorporate prior knowledge in the form of genetic networks into the model and encode the topology of these regulatory networks.[17] By doing so, they have been more successful compared to GNN models in studying the underlying mechanisms of gene regulation, especially for complex datasets with more classes and cells.[17]
Post-hoc explanation methods such as GNNExplainer have been used in finding explanations for single-cell classification to guide scientists regarding which genes to focus on in further studies.[14] However, the reliability of these explanations in the context of single-cell omics applications has not yet been fully explored.[4] [13] Due to the complex nature of gene-to-gene interactions, high sparsity, and the difficulty in finding ground truth datasets, the evaluation of explanations specifically designed for single-cell classification is pertinent.[19]

This thesis will be evaluating the performance of state-of-the-art explanation methods for GNNs using a realistic synthetic dataset of single-cell sequencing data. These datasets will be produced by SERGIO, a method proposed by Dibaenia et al, which has been shown to produce realistic cell expression datasets.[5] The method takes in a user-provided ground truth gene regulatory network which it uses to create single-cell RNA sequencing data. To capture the sparse nature of single-cell sequencing data, several datasets will then be produced at varying levels of sparsity up to 97%, which is typical for a real-world dataset. These datasets with different levels of sparsity will then be trained using a three-layer graph convolutional network for a binary classification task. Associated explanations of the classification predictions will be produced using GNNExplainer and PGExplainer. A Random Explainer will also be used as a trivial baseline explanation method. These explanation methods will then be compared and evaluated against each other to see how well they can represent the underlying ground truth network to determine the reliability of the methods for use in single-cell sequencing datasets.
Given a tissue or sample composed of heterogeneous cells, single cell sequencing methods function by first separating the mixture into individual cells and then performing sequencing on that isolated cell. The data is often converted into a matrix, known as the expression matrix. The expression matrix, $X \in \mathbb{Z}^{n \times p}$ of $n$ cells and $p$ genes where each element in $X$, denoted as $X_{ij}$ is a count value of the expression level of gene $j$ at cell $i$. 

2

Methods
Gene expression is controlled by transcription factors and has spatiotemporal specificity. For each cell, there is a combination of active TFs, which affect downstream target genes. These regulatory interactions are modeled as a gene regulatory network (GRN) where the nodes in the graph are regulators and the edges are the interactions between two genes.

The input generation is composed of two main components. First, a synthetic dataset in the form of an expression matrix first is generated by feeding a user defined “ground truth” GRN into the SERGIO method. Then, two sets of datasets are created. To generate the first set, the synthetic sequence datasets are converted into another GRN through a gene inference algorithm, grnboost2, in order to be used as input for the GCN network. The other set does not use a gene inference algorithm and instead uses the ground truth GRN inputted into SERGIO as the GRN.

The following sections outline the methods for both input generating components. The structure of the GNN used for classification is also outlined.
2.1 Construction of Synthetic Datasets

Single cell RNA sequence datasets were created using Single-cell Expression of Genes in Silico (SERGIO), a simulation method which models the stochasticity of transcriptions and regulators with stochastic differential equations.[5] The simulation takes in a ground truth gene regulatory network, transcription factors, and a list of master regulators in order to create a realistic single cell RNA sequencing dataset.

SERGIO uses stochastic differential equations (SDEs) to model the stochasticity of transcription factors and regulators in gene expression.[5] It generates an expression matrix, \( X \) in log scale then adds zeros to the matrix by sampling from a Bernoulli probability function.

\[
X_{ij} = I_{ij}X_{ij} \quad \text{where} \quad I_{ij} \sim Bernoulli(\pi_{ij})
\]

\[
\pi_{ij} = \frac{1}{1 + \exp(k(X_{ij} - y_0))} \quad \text{and} \quad y_0 = (100 - s)\% \text{ percentile of } X
\]

We construct datasets consisting of two cell types with a defined ground truth gene regulatory network of 100 genes, transcription factors, and a list of master regulators at various sparsities using the two hyperparameters, \( k \) and \( s \).

Eight datasets with sparsity levels ranging from 20% to 97% were produced through this method. 97% is the sparsity level that is typical for a real world single cell dataset. Each dataset comprises of 2000 cells. 1000 of these cells are of cell type A and the other 1000 of the cells are of cell type B. For each dataset, the cells were then split into training (70%), testing (20%), and validation (10%) splits, each having an equal number of cells of type A and B labels.

In order to model gene-gene interactions in a cell, we generated a gene regulatory
network (GRN) using the grnboost2 algorithm.

The inputs to the grnboost2 algorithm include the single cell expression matrix along with a list of known transcription factors (TFs). The algorithm then uses a tree-based regression model in order to find an expression profile for the 100 genes in the dataset.[12] As a result, the algorithm returns a matrix where the rows represent the transcription factor, a gene that is regulated by the transcription factor, and finally the importance value. The importance value represents the confidence the model has for the transcription factor and the associated gene. From there, the mean and the standard deviation of the importance values were calculated. Only the interactions with an importance values which are one standard deviation above the mean were used to determine the adjacencies between the genes.

As for the set of datasets without grnboost2, the ground truth GRN was used to determine the adjacencies. Lastly, 2000 graphs which connect TFs and genes were created for each set of data. These resulting graphs are the inputs to the GNN.

2.2 Graph Neural Network

We construct a binary graph classifier using a network of Graph Convolutional layers and linear layers to classify the cells in the dataset into two distinct cell types. The network is composed of 3 pairs of convolution and non-linear layers, followed by a pooling layer, a dropout layer, and a classification head.

Most GNNs follow a method of recursive neighborhood aggregation.[15] Each node takes the feature vectors of its neighbors in order to compute a new feature vector. Upon $k$ iterations of aggregation, a node captures the structural information of the node’s $k$-hop neighborhood by its transformed feature vector.[15] Finally, the entire
graph is represented by pooling the representations of each node in the graph.

The GraphSage formulation for aggregation and pooling was used for this network as it learns a function of embeddings through aggregating features from a node’s neighborhood instead of training embeddings for each individual node, allowing it to better capture the structural information in the entire graph.\[8\] For a given graph, $G = (V, E)$ with node feature vectors $X_v$ for $v \in V$, The $k$th layer is given by

$$h_k^v = ReLU(W \cdot MEAN(h_{u}^{(k-1)}), \forall u \in N(v) \cup v)$$

where $h_k^v$ is the feature vector of node $v$ in the $k$-th iteration and $N(v)$ is the set of nodes which are adjacent to $v$.\[8\]

Following each convolutional layer, a non-linear layer in the form of a rectified linear activation function (ReLU) is used.\[15\] The purpose of this layer is to set all negative values to zero. By doing so, the vanishing gradient problem which occurs with sigmoid or $tanh$ activation functions is solved, and training is sped up.

After three pairs of convolution and non-linear layers, the outputs of the network are passed through a global max-pooling layer. The max-pooling layer is used for returning only maximum value in the feature dimensions for each of the nodes present in the graph.

The outputs of the global max-pooling layer are then used as inputs for the dropout layer. In this layer, elements of the input vector are transformed to 0 with a probability of 0.5. Dropout layers are used to prevent overfitting.

The final step of the network is to pass the outputs of the dropout layer into a linear layer for the binary classification task. Logits for the two classes are outputted in this layer. The class prediction the model returns is the maximum probability value
of the logits from this layer. The Cross Entropy Loss function was used as the loss function for training. It is given by the following formula.

\[
\text{Cross Entropy Loss} = - \sum_{i=1}^{n} t_i \log(p_i) \text{ for } n \in \{1, 2\}
\]

The true label of the graph is \(t_i\), the probability for each class is \(p_i\), and \(n\) is the number of classes. A learning rate of 0.001 and an Adam optimizer was used to train the model.

2.3 Post-Hoc Explanation Methods

After the datasets were trained using the network outlined above, explanations for the predictions made were generated. The explainer methods that this thesis focuses on are GNNExplainer, PGExplainer, and Random Explainer as a trivial baseline.

- **Random Explainer**
  
The Random Explainer is used for comparison purposes only. It assigns each edge a random value in the range (0, 1) sampled uniformly.[4]

- **GNNExplainer**
  
GNN Explainer is a model-agnostic explainer method which can be used for graph classification.[18] If \(G_0\) is a graph with node features \(X_0\), the method finds a subset of features which are important for the predictions made by the trained GNN, \(\phi\). GNNExplainer finds the importance of each node and edge using Mutual Information, which is a measure of the change in probability of the prediction made by \(\phi\) when the input graph is limited to the explanation subset graph found.
• **PGExplainer**

PGExplainer was proposed after GNNEexplainer in order to provide a more comprehensive understanding on the predictions made by GNNs.[11] It uses an optimization framework that is similar to GNNEexplainer, however it uses continuous variables in the range (0, 1) for the edge weights. Furthermore, it uses reparameterization to optimize the objective function using gradient-based methods. It makes parameter size independent from the actual graph size by predicting the importance of all of the edges in the graph.
3

Experimental Setup

3.1 Datasets

Using the ground truth GRN along with a list of master regulators, the following ground truth GRN was used for creating the synthetic datasets. 10 genes were arbitrarily chosen as the master regulators. This graph with its corresponding nodes and edges will be the ground truth graph for evaluating explanations.
The synthetic datasets produced from this ground truth graph vary in sparsity in order to test for the robustness of the explainer method. The highest sparsity dataset is the most realistic for single cell sequencing as it has a comparable sparsity to it. The datasets and their properties are summarized in the table below.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>20%</th>
<th>25%</th>
<th>50%</th>
<th>75%</th>
<th>80%</th>
<th>90%</th>
<th>95%</th>
<th>97%</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td># edges</td>
<td>113</td>
<td>110</td>
<td>121</td>
<td>111</td>
<td>109</td>
<td>85</td>
<td>69</td>
<td>61</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0.913</td>
<td>0.945</td>
<td>0.928</td>
<td>0.888</td>
<td>0.845</td>
<td>0.785</td>
<td>0.785</td>
<td>0.627</td>
</tr>
<tr>
<td>AUC</td>
<td>0.915</td>
<td>0.944</td>
<td>0.928</td>
<td>0.889</td>
<td>0.848</td>
<td>0.800</td>
<td>0.781</td>
<td>0.628</td>
</tr>
</tbody>
</table>

Table 3.1. Summary of Dataset Produced Using SERGIO and grnboost2

For the set of data using only SERGIO and the ground truth GRN, all graphs produced have 100 nodes and 113 edges representing the gene-gene interactions present in the ground truth graph. The accuracy and AUC values of the datasets for these are also given.
### Table 3.2. Summary of Dataset Produced Using SERGIO and Ground Truth GRN

<table>
<thead>
<tr>
<th>Datasets</th>
<th>20%</th>
<th>25%</th>
<th>50%</th>
<th>75%</th>
<th>80%</th>
<th>90%</th>
<th>95%</th>
<th>97%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.917</td>
<td>0.938</td>
<td>0.915</td>
<td>0.835</td>
<td>0.820</td>
<td>0.750</td>
<td>0.668</td>
<td>0.618</td>
</tr>
<tr>
<td>AUC</td>
<td>0.921</td>
<td>0.934</td>
<td>0.915</td>
<td>0.841</td>
<td>0.819</td>
<td>0.756</td>
<td>0.667</td>
<td>0.624</td>
</tr>
</tbody>
</table>

#### 3.2 Evaluation Metrics and Benchmarks

To evaluate explainability, the following metrics were used.

- **Fidelity**

  The fidelity metric is independent from the existence of ground-truth explanations. It measures the change in the predictive accuracy of the trained GNN when including or excluding the important subgraph determined by the explanation methods. The two types of fidelity metrics are negative fidelity and positive fidelity.

  The negative fidelity score reflects whether the explanation found was sufficient in making the prediction that the model outputs.[13] It does this by only using the subgraph chosen to be important by the explanation method and sums the number of explanations which result in a different prediction from the trained GNN model when only using the subgraph as an input. Therefore, a good and reliable explanation should produce a negative fidelity score close to 0.

  \[
  fid_- = \frac{1}{N} \sum_{i=1}^{N} \left| \mathbb{1}(\hat{y}_i = y_i) - \mathbb{1}(y_i^G = y_i) \right|
  \]

  \(N\) is the number of explanations, \(\hat{y}_i\) is the true label of the graph, \(y_i\) is the predicted label of the graph, and \(y_i^G\) is the prediction produced when only the im-
portant subgraph is used.

The positive fidelity score reflects whether the explanation found was necessary in making the prediction that the model outputs. It does this by using a graph with the important subgraph excluded sums the number of explanations which result in a different prediction from the trained GNN model when only using the subgraph as an input. Therefore, a good and reliable explanation should produce a positive fidelity score closer to 1 to reflect that the subgraph was necessary in making the predicted decision for the GNN.

\[
fid_+ = \frac{1}{N} \sum_{i=1}^{N} \left| \mathbb{1}(\hat{y}_i = y_i) - \mathbb{1}(y_{i}^{G'} = y_i) \right|
\]

\(N\) is the number of explanations, \(\hat{y}_i\) is the true label of the graph, \(y_i\) is the predicted label of the graph, and \(y_{i}^{G'}\) is the prediction produced when only the important subgraph is excluded from the entire graph.

- **Sparsity**

Sparsity measures the fraction of the graph that is characterized by important by the explanation method. High faithfulness ensures that the explanation approximates the model behavior well. However, the complete input completely determines the model behavior and therefore explanation sparsity is an important metric for evaluation. It is calculated using the following formula:

\[
\text{Sparsity} = \frac{1}{N} \sum_{i=1}^{N} \left(1 - \frac{|s_i|}{|S_i|_{total}} \right)
\]

\(|S_i|_{total}\) represents the total number of edges in the original graph model and \(|s_i|\) represents the subgraph chosen to be important by the explanation method.
A comprehensive explanation method should produce non-trivial explanations with a high sparsity.

The code for this thesis was written using a combination of PyTorch Geometric and Pytorch to run both the classification GNN and explanation methods. All models were trained using resources and services at the Center for Computation and Visualization, Brown University, Providence, RI.
The explanation methods were evaluated in terms of the metrics outlined in the previous section. The results from the two metrics, fidelity and sparsity, for each of the datasets are shown in the tables below. In nearly all of the datasets and settings, GN-NExplainer had the best performance.
4.1 Fidelity

Both the positive and negative fidelities were calculated for the datasets upon passing the model into the explanation methods. For establishing the necessity and sufficiency of the explanations generated by the methods, a high positive fidelity and a low negative fidelity are indicators of a good explanation class. The results are shown in Tables 4.1 and 4.2. The values in bold in the tables are the best performing fidelity scores for each corresponding dataset.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>20%</th>
<th>25%</th>
<th>50%</th>
<th>75%</th>
<th>80%</th>
<th>90%</th>
<th>95%</th>
<th>97%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fid-</td>
<td>fid+</td>
<td>fid-</td>
<td>fid+</td>
<td>fid-</td>
<td>fid+</td>
<td>fid-</td>
<td>fid+</td>
</tr>
<tr>
<td>Random</td>
<td>0.45</td>
<td>0.335</td>
<td>0.470</td>
<td>0.381</td>
<td>0.179</td>
<td>0.168</td>
<td>0.315</td>
<td>0.341</td>
</tr>
<tr>
<td>GNN</td>
<td>0.08</td>
<td><strong>0.585</strong></td>
<td><strong>0.065</strong></td>
<td><strong>0.425</strong></td>
<td><strong>0.08</strong></td>
<td><strong>0.455</strong></td>
<td><strong>0.13</strong></td>
<td><strong>0.66</strong></td>
</tr>
<tr>
<td>PG</td>
<td>0.485</td>
<td>0.225</td>
<td>0.375</td>
<td>0.205</td>
<td>0.25</td>
<td>0.29</td>
<td>0.385</td>
<td>0.245</td>
</tr>
</tbody>
</table>

**Table 4.1.** Fidelity metrics calculated for datasets produced with SERGIO and grnboost2

<table>
<thead>
<tr>
<th>Datasets</th>
<th>20%</th>
<th>25%</th>
<th>50%</th>
<th>75%</th>
<th>80%</th>
<th>90%</th>
<th>95%</th>
<th>97%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>fid-</td>
<td>fid+</td>
<td>fid-</td>
<td>fid+</td>
<td>fid-</td>
<td>fid+</td>
<td>fid-</td>
<td>fid+</td>
</tr>
<tr>
<td>Random</td>
<td>0.355</td>
<td>0.441</td>
<td>0.38</td>
<td>0.377</td>
<td>0.250</td>
<td>0.243</td>
<td>0.223</td>
<td>0.195</td>
</tr>
<tr>
<td>GNN</td>
<td><strong>0.216</strong></td>
<td><strong>0.43</strong></td>
<td><strong>0.216</strong></td>
<td><strong>0.43</strong></td>
<td><strong>0.04</strong></td>
<td><strong>0.495</strong></td>
<td><strong>0.12</strong></td>
<td><strong>0.426</strong></td>
</tr>
<tr>
<td>PG</td>
<td>0.505</td>
<td>0.235</td>
<td>0.51</td>
<td><strong>0.465</strong></td>
<td>0.48</td>
<td>0.37</td>
<td>0.3</td>
<td>0.11</td>
</tr>
</tbody>
</table>

**Table 4.2.** Fidelity metrics calculated for datasets produced with SERGIO and the ground truth GRN

Comparing the performance of each explanation method across the datasets with varying sparsity, GNNExplainer consistently had the best performance. Both GNNEexplainer and PGExplainer perform above the Random Explainer, which functioned as a trivial baseline, in most datasets.

As the sparsity of the datasets increased, the negative fidelity decreased until 50%-75% sparsity, at which point it began to rise again reaching a maximum value at 97% sparsity. For positive fidelity, GNNEexplainer performed better than the baseline metric for datasets with sparsities of 20%-80%. As most single cell sequencing data has
sparsities of >95%, the reliability of current state of the art methods on this data re-
mains in question.

Figure 4.1: Negative fidelity vs. dataset sparsity for datasets generated using SERGIO and grn-
boost2

Figure 4.2: Positive fidelity vs. dataset sparsity for datasets generated using SERGIO and grn-
boost2
**Figure 4.3:** Negative fidelity vs. dataset sparsity for datasets generated using SERGIO and the ground truth GRN

**Figure 4.4:** Positive fidelity vs. dataset sparsity for datasets generated using SERGIO and the ground truth GRN
Next, comparing the performances of datasets generated through grnboost2 and datasets generated through using the ground truth GRN, GNNExplainer still performs best in most datasets compared to the other two metrics. At the highest level of dataset sparsity, PGExplainer performed better than GNNExplainer when using the ground truth GRN as opposed to using grnboost2.

Another interesting result across both datasets was found at the highest levels of sparsity (>90%) where all three explanation methods converged in their values for positive and negative fidelity. Random Explainer produces random explanations with no significance to the underlying mechanisms but still achieves similar metrics to the other methods, which highly suggests that these methods need to be further evaluated.

4.2 Sparsity

In order to evaluate the reliability of explanation methods, sparsity must also be considered, as an explanation with 0% sparsity would be an explanation with high positive fidelity and low negative fidelity. The sparsity results are given in the tables below. For Random Explainer, a hard mask edge threshold of 0.4 was used and thus the sparsity levels remained constant at that level.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>20%</th>
<th>25%</th>
<th>50%</th>
<th>75%</th>
<th>80%</th>
<th>90%</th>
<th>95%</th>
<th>97%</th>
</tr>
</thead>
<tbody>
<tr>
<td>GNNExplainer</td>
<td>0.918</td>
<td>0.916</td>
<td>0.917</td>
<td>0.914</td>
<td>0.927</td>
<td>0.927</td>
<td>0.923</td>
<td>0.934</td>
</tr>
<tr>
<td>PG Explainer</td>
<td>0.760</td>
<td>0.519</td>
<td>0.404</td>
<td>0.537</td>
<td>0.714</td>
<td>0.884</td>
<td>0.893</td>
<td>0.508</td>
</tr>
</tbody>
</table>

Table 4.3. Explanation sparsity for datasets generated using SERGIO and grnboost2

GNNExplainer produced more sparse explanations than PGExplainer across most datasets. These difference between sparsity however is more pronounced for datasets
Table 4.4. Explanation sparsity for datasets generated using SERGIO and the ground truth GRN where grnboost2 was used. This difference may be attributed to the significant decrease in density of the graphs produced through grnboost2 as it infers a GRN from a more sparse expression dataset.
Conclusion

In this thesis, current explanation methods for graph neural networks were evaluated on realistic synthetic datasets of single-cell sequencing data generated from SERGIO. Datasets at varying levels of sparsity from 20% to 97% were produced with and without gene inferencing methods (grnboost2). The two main explanation methods, PG-Explainer and GNNExplainer were evaluated using the metrics of fidelity (positive, negative) and sparsity. Finally, a baseline explainer, Random Explainer, with trivial
explanations was used for comparison.

The results showed that GNNExplainer outperformed all methods on most datasets for creating minimal, sufficient, and necessary explanations. Comparing the results across varying levels of sparsity in the datasets, it was noticeable that the performances of all three explanation methods drop at higher sparsities which are typical for real-world single-cell sequencing datasets. Furthermore, when comparing the results from the datasets generated using gene inference methods and those without, there was a noticeable decrease in the sparsity when the ground truth gene regulatory network was used. Overall, these results suggest that further evaluations and explanation methods tailored for single-cell sequencing datasets which have high sparsity and complex mechanisms may be necessary.

As this project continues, we hope to use these explanation methods and the evaluation metrics on real-world datasets with a ground truth GRN. In addition, we anticipate that we will use more explanation methods and perform further analysis on single-cell dataset inference methods and see what role they play in the explanations which are generated.
References


