Community Detection in Spatial Omics Data

Kyle McCrocklin, Abdullah Al Fahad

1. INTRODUCTION

PROBLEM STATEMENT

Spatial omics is a relatively new field of technology which allows scientists to investigate the spatial patterns of gene or protein expression in a small tissue sample. In the case of gene expression (spatial transcriptomics), the data generated consists of an expression matrix where each row is one cell or spot (covering multiple cells) and the value in each column is number of RNA transcripts for a specific gene detected in that cell/spot. There are typically 1,000-500,000 individual spots/cells and 1,000-30,000 individual genes. Additionally, there is a matrix of spatial coordinates (X and Y) for each individual cell/spot.

Together this data allows researchers to investigate the workings of biological tissues. It is an advancement over previous sequencing technologies due to the inclusion of spatial coordinates. This information puts the gene/protein expression of an area of tissue into the context of its neighbors' expression.

There is an active area of computational research which seeks to develop analytical frameworks for spatial omics data. One such task is community detection or domain segmentation of the tissue sample. Such algorithms can be applied to a tissue to identify similar cells/spots which may fall into the same cell type or tissue type category.

OVERVIEW OF THE DATA

The data used for this project are 12 samples of human dorsolateral pre-frontal cortex (DLPFC) spatial transcriptomics data generated with the 10x Genomics Visium platform. This data was acquired with the "spatialLIBD" package. This data is spot based, meaning that each row in the data corresponds to a spot which covers multiple cells. This is lower resolution than single-cell spatial transcriptomics but has the benefit that the assay covers the entire human genome rather than just a few thousand selected genes.

For each sample there is a sparse gene expression matrix with about 4,000 rows, one for each spot, and 30,000 columns, one for each gene which could be detected. In each cell is the integer count of how many times that gene was detected in that spot.

There is also a spatial coordinate matrix with the X and Y coordinates of each spot. These specific data also include ground truth labels for each spot according to which cortical layer it belongs to. These are the communities which we wish to detect.

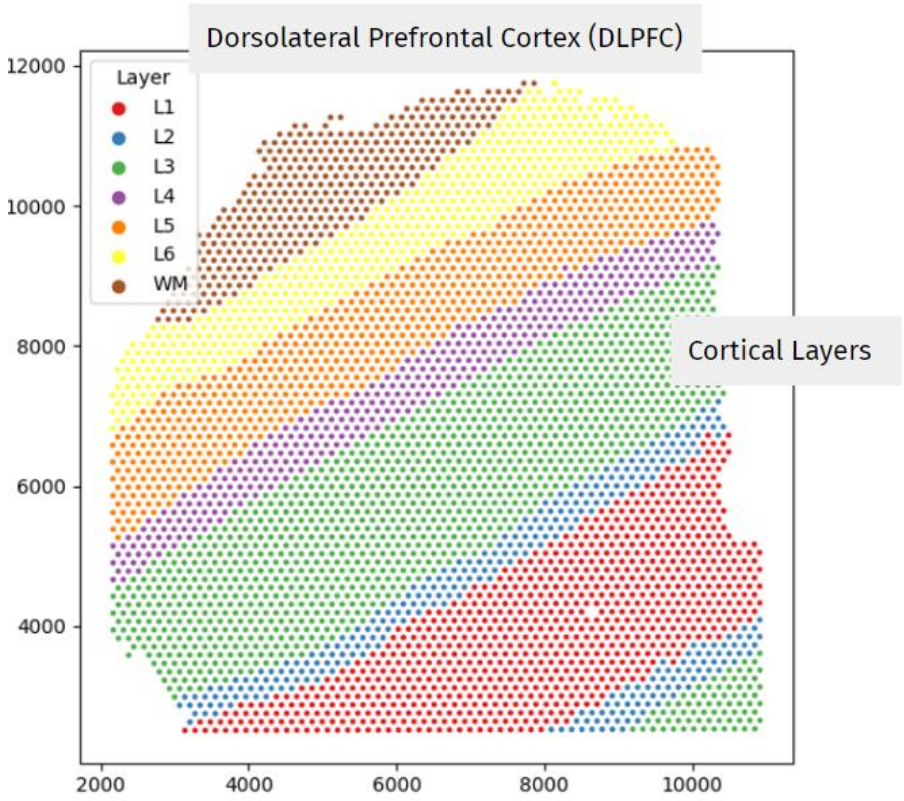

Figure 1: Spatial Plot of Ground Truth Cortical Layers for a Single Sample

CONTRIBUTION

Our contributions were for the most part very equal as we worked on the project in tandem. Fahad did contribute more heavily to preprocessing, finding the best k, and clustering. Kyle contributed more to data collection, neighborhood aggregation, and validation.

2. RELATED WORK

CellCharter, published in 2023 by Marco Varrone et al, is a method for community detection in spatial omics data. They utilize a variable auto encoder, graph construction, neighborhood aggregation, and gaussian mixture model clustering. We have referenced their paper and will attempt similar strategies while incorporating the lessons learned from our Data Mining course.

3. METHODOLOGY

Spatial omics data is comprised of a sparse matrix that indicates the expression of a cell/spot in terms of how many times a gene was detected in that cell/spot. There are approximately 30,000 genes which could be detected, and these are the features to consider. To handle this high-dimensional data, we applied Principal Component Analysis (PCA) to capture 99%, 90%, or 80% of the variance. The spatial coordinates of each spot were used to construct a graph with the spots as nodes and their immediate neighbors connected by edges. Neighborhood features were then aggregated by summing the average gene expression values from neighboring spots weighted by their distance. Finally, we employed K-means clustering using the optimal number of clusters determined by the elbow method. Adjusted Rand Index (ARI) was used as our metric for performance.

List of the libraries we have used are:

1. **Panda** *for dataset reading*
2. **Numpy** *array creation*
3. **Networkx** *to create graph network*
4. **Scipy** *for mathematical calculation*
5. **Torch** *data conversion, and graph aggregation*
6. **Scikit-learn** *clustering*
7. **Matplotlib** *for plotting*
8. **Seaborn** *for plotting*

4. IMPLEMENTATION

DATA PREPROCESSING

For the gene expression data, it is common to standardize the gene expression matrix so that each gene has a mean expression of 0 and a standard deviation of 1. This was the only preprocessing performed on the expression matrix.

For the spatial data, some of the 12 samples were missing spatial coordinate values for one or two spots, making those spots impossible to incorporate in our methodology. These spots were removed from those datasets along with their corresponding gene expression data. Removing one or two spots from a sample has a very small effect on domain detection and is not a concern to us.

EXPLORATORY DATA ANALYSIS

We first analyzed one of the 12 samples chosen at random. In this exploratory analysis we built a rough pipeline and were able to test multiple methods for each step. Here we tried several aggregation strategies such as using the mean of neighbors' expression, standard deviation, and a combination of the two. We tried concatenating all neighborhood features to the spot's own expression which resulted in about 15,000-30,000 features. We also experimented with combining the neighborhood expression features by weighted averaging to result in only 4,000-8,000 features after concatenation.

We also tried multiple clustering algorithms: Agglomerative Clustering, DBSCAN, Gaussian Mixture Model (GMM), and K-means.

This exploratory analysis ended with a working pipeline and a rough idea of what ranges of hyperparameter values might be best. Our next step was to implement performance metrics and perform a hyperparameter search using all 12 samples.

MODEL TRAINING AND VALIDATION

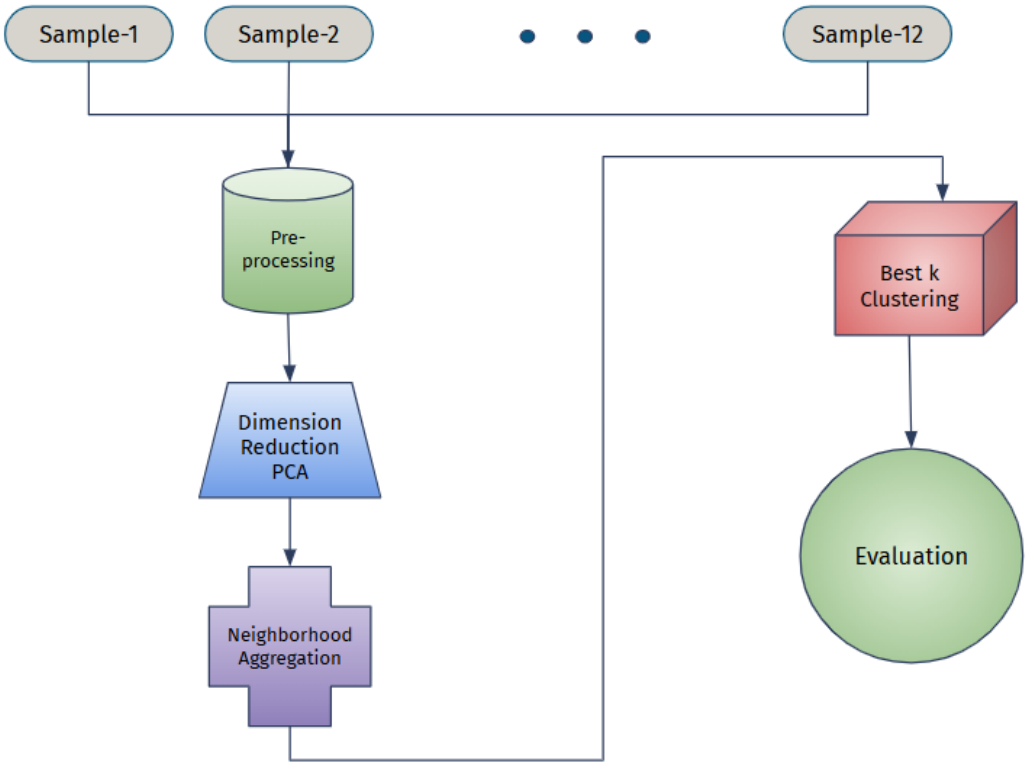

Figure 2: Community Detection Pipeline

The pipeline depicted in Figure 1 was run for all 12 samples using many combinations of hyperparameters and resulting metrics were computed. Table 1 shows the hyperparameter grid.

PCA variance	99, 90, 80
Number of Hops	1, 2, 3, 4, 5
Neighborhood Weights multiplier	.75, 1, 1.25, 1.5

Table 1: Hyperparameter Grid

Neighborhood Aggregation:

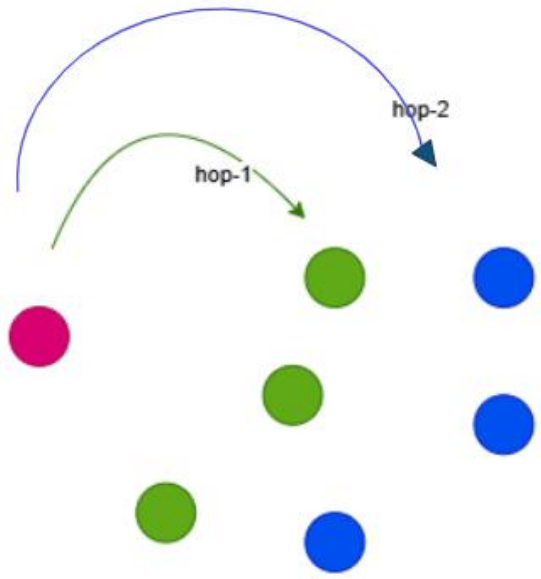

Figure 3: Two Hops of Neighborhood Aggregation

This process involves combining the expression values of a spot with those of its neighboring spots. The steps are as follows:

a. **Aggregate Neighbor Expressions**: Calculate the mean expression values of neighbors at each level (or hop).
b. **Weight and Sum**: Assign weights to the mean expression values from each level of neighbors and sum them up.

c. **Concatenate with Own Expression**: Combine (concatenate) the aggregated neighbor expressions with the spot's original expression values.
d. **Weight the Neighborhood vs Own Expression:** A multiplier was used to increase or decrease the magnitude of the concatenated neighborhood features relative to the spot's own expression. This allowed us to roughly tune how important the neighborhood information was compared to the individual spot.

This neighborhood aggregation approach effectively doubles the number of features in the data matrix.

Neighborhood aggregation is typically done using neural networks which learn the correct transformation and aggregation strategies based on training data. However, with spatial omics data, there is rarely enough data to train a neural network. Instead, rule-based approaches such as the one just discussed are used.

Weight Generation:

Weights were generated for the weighted average of neighborhood aggreation. The goal was to give more weight to close spots and less to distant spots. To achieve this, we applied an exponential decay function based on the number of hops. The steps are as follows:

1. **Generate Weights**: Compute weights for each hop using the exponential decay function, where closer neighbors (fewer hops) receive higher weights.
$$e^{-\left(\frac{1}{2}\cdot k\right)}$$
2. **Normalize Weights**: Normalize the generated weights so that their sum equals 1.

Clustering:

We have evaluated various clustering methods, including **Agglomerative Clustering, DBSCAN, Gaussian Mixture Model (GMM), and K-means,** and found that K-means performed the best. Agglomerative Clustering failed to detect spatial clusters, DBSCAN was impractical due to the proximity of spots, and GMM was too slow because of the dataset's high dimensionality.

We then ran all our samples with k-means with k-means++ initialization.

Finding the best K:

We used the elbow method to determine the optimal value of k for K-means clustering. In this approach:

1. We performed K-means clustering for different k values, ranging from 5 to 15.
2. For each k, we calculated the cluster inertia, which is the sum of squared distances of samples to their nearest cluster center.
3. By plotting the inertia values against k, we identified the "elbow point," where the decrease in inertia begins to level off, indicating the best k.

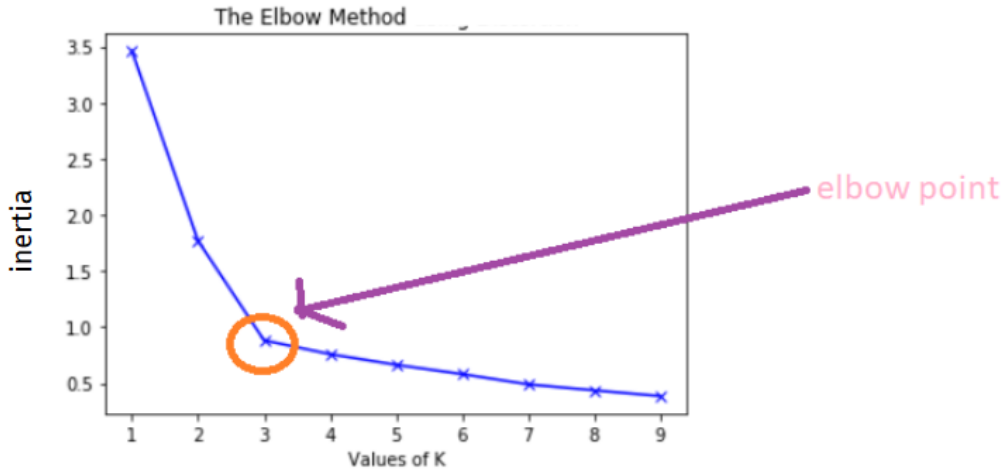

Figure 4: Elbow method for finding best k

Validation

To validate our results, we compared the clusters determined by our method to the ground truth cortical layer labels. The metric we used was the Adjusted Rand Index (ARI).

Besides evaluating the ARI value, we have also compared our cluster plots with the ground truth cluster plots to visually assess the performance of clustering. Other metrics, such as the Silhouette Coefficient, have also been explored; however, our report focuses solely on the analysis based on the ARI score.

ARI was selected as the metric because it is a measure of clustering similarity, which accounts for chance. Visual comparisons also further validated the alignment of our clusters with the ground truth data.

5. EXPERIMENTAL RESULTS AND DISCUSSION

After the hyperparameter search, our best performing community detection pipeline had an **average ARI of 0.233** over all 12 samples with a **standard deviation of 0.038**. This was using **90% PCA**, aggregation over **4 hops**, and a **1.5X neighborhood aggregation multiplier**. The results for a few samples are shown in Figure 5.

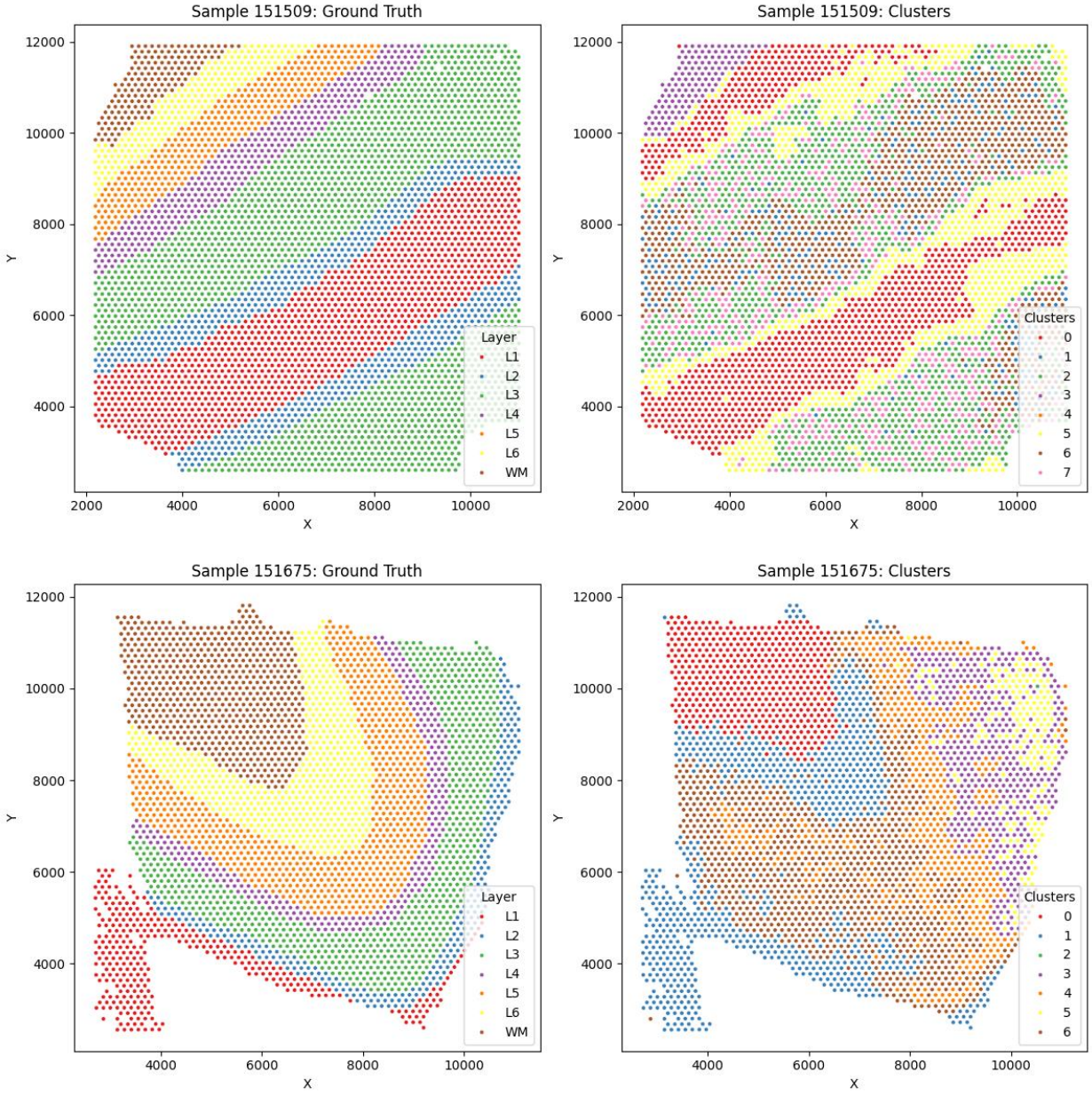

Figure 5: Clusters vs Ground Truth for Two Samples

An ARI of .233 is not very impressive, but this is a challenging task which even more sophisticated published methods struggle with. The CellCharter method claims an ARI of

0.62 on this same data but the method UTAG only achieved and ARI of 0.36 (Figure 6). Our method is considerably simpler, for instance we use PCA for dimension reduction rather than a Variable Autoencoder which was specifically trained on omics data.

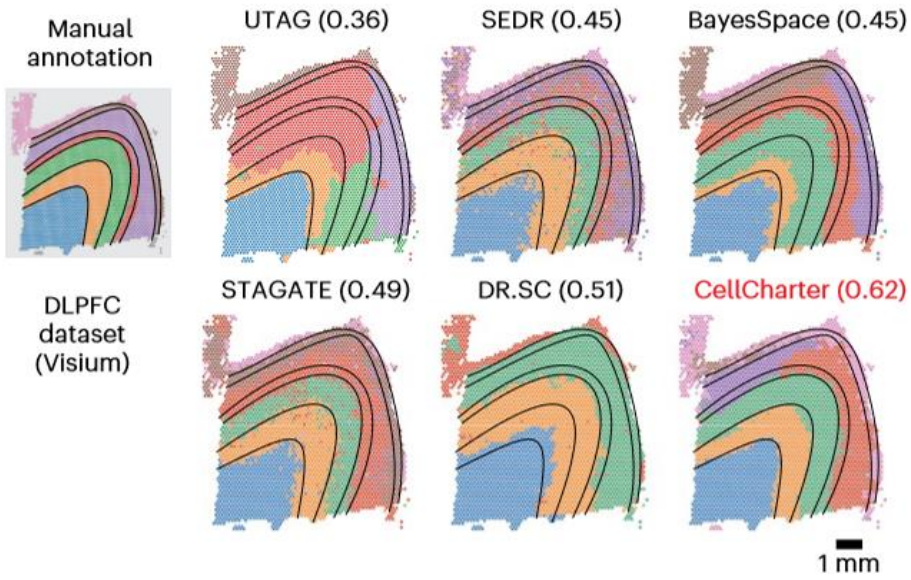

Figure 6: Performance of Published Methods

Additionally, we believe that the neighborhood aggregation strategy is the most influential part of accurate community detection. This can be seen from the results of our algorithm without the neighborhood aggregation step. This only achieved ARI values of about 0.15 (Figure 7).

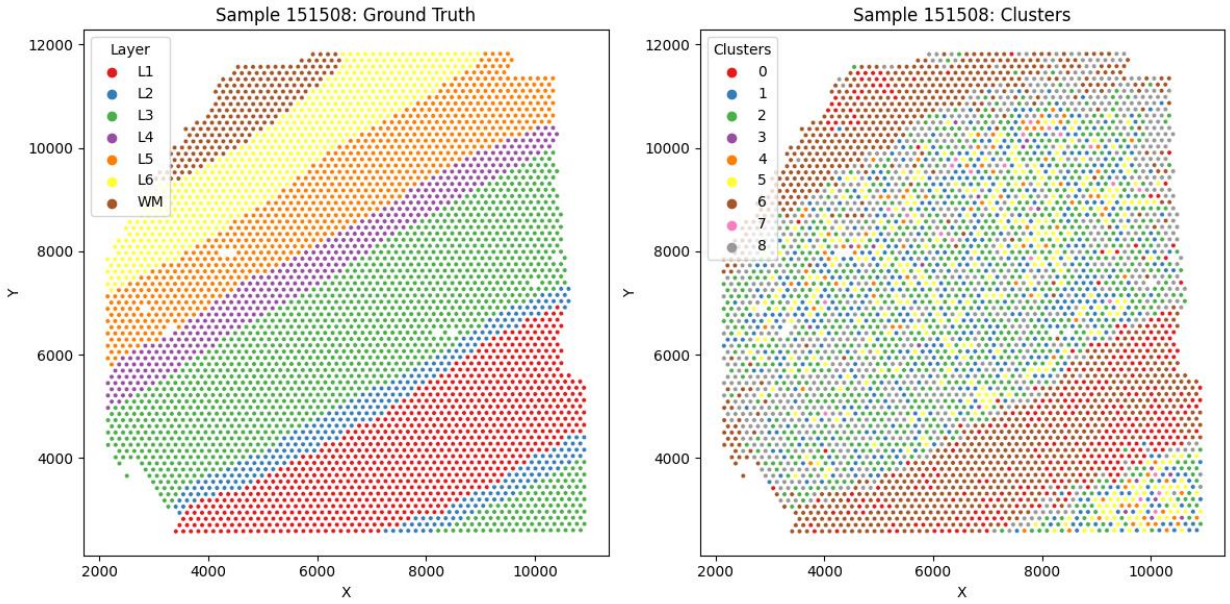

Figure 7: Clusters vs Ground Truth Without Neighborhood Aggregation

The importance of neighborhood aggregation, and the primary challenge of this task, that there is not enough data for a machine learning model to learn an aggregation strategy, is worth discussing. Our opinion is that with enough tweaking, such a rule-based community detection algorithm could be made to perform very well on the specific dataset of interest. It would not likely generalize well to unseen data though. If we tried many aggregations strategies, we may have achieved much better performance. The aggregation strategy we used was based on simple averaging of neighborhood expression, which seemed biologically sound.

6. CONCLUSION

Computational analysis methods for spatial omics data, such as community detection, is an important area of research and one that is developing rapidly. Our exploration of a simple community detection method using the topics covered in the Data Mining course provided useful insights into the field.

Our method clearly detected communities which closely resembled the true cortical layer structure of the 12 samples. We are happy with the performance even though the actual ARI values are low because more complex published methods also struggle with this task.

Our implementation of a simple rule-based neighborhood aggregation strategy significantly improves the clustering performance and provided insight into the importance and inherently challenging nature of this step when there is not enough training data for a neural network aggregation model.

Future work in this direction would likely go towards improving the neighborhood aggregation strategy. Consulting with a domain expert to understand the biological processes governing cell expression and cell-cell communication could yield more biologically sound strategies. Additionally, expanding the analysis to multiple tissue types would be an important study of generalizability.

REFERENCES

Varrone, M., Tavernari, D., Santamaria-Martínez, A. et al. CellCharter reveals spatial cell niches associated with tissue remodeling and cell plasticity. *Nat Genet* 56, 74–84 (2024). https://doi.org/10.1038/s41588-023-01588-4

Pardo, B., Spangler, A., Weber, L. M., Hicks, S. C., Jaffe, A. E., Martinowich, K., ... Collado-Torres, L. (2022). spatialLIBD: an R/Bioconductor package to visualize spatially-resolved transcriptomics data. BMC Genomics. doi:10.1186/s12864-022-08601-w