CS690: Computational Genomics

Lecturer: Hamim Zafar
Assignment: 1

Group #2
October 4, 2022

1 INTRODUCTION

Spatial transcriptomics is an application of clustering algorithms applied on scRNA seq data. They allow us to determine the locations of different parts of a tissue by virtue of the expression levels of cells in their locations. The expression data obtained consists of expression information from around 10-15 cells collected from a specific spot. Later on, using the clustering, the location of the different parts can be determined by overlapping the clustered spots on the tissue image.

2 TASK 1

For the task one, the tutorial provided by the SpaGCN Repository was followed for cell clustering. Firstly, gene expression and histology are integrated into a Graph, followed by data preprocessing of gene expression and setting up of hyperparameters. SpaGCN then trains and fits a graph convulational network on the data, and plots spatial domains and refined spatial domains.

During the model training and searching of resolution, the code repetitively crashed due to full RAM usage, which on diagnosis, is believed to be caused due to a bug in unmaintained louvain package. Hence, we modified the source code of SpaGCN to allow cell clustering using Leiden algorithm[3], which is a slightly improved version of Louvain algorithm, and its python package is currently being maintained properly. Hence, the modified version of SpaGCN code is being submitted along with the Jupyter Notebook and running scripts.

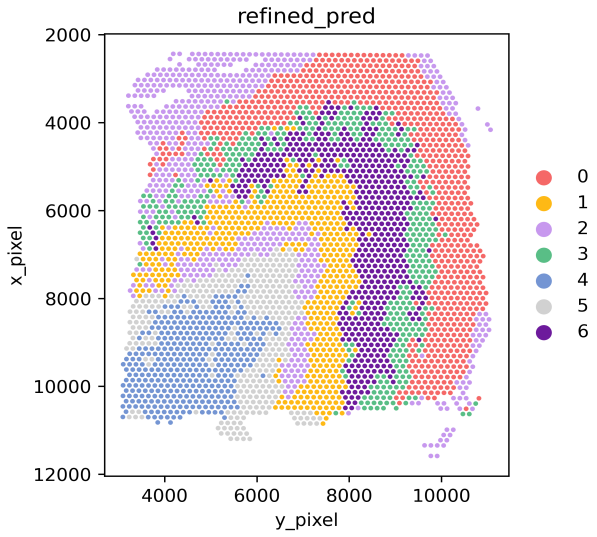

Fig. 1. Refined spatial domain on dataset with SpaGCN

3 TASK 2

For the task two, the following pipeline was adopted:

First, the training and testing datasets were integrated following a tutorial from the scvi tools documentation [1]. This implementation integrates two datasets by the use of variational autoencoders. Latent space of size 30 was used as according to the documentation, this number has been found to work well. This step was necessary, so that both the datasets had their dimensions reduced through the same model, and

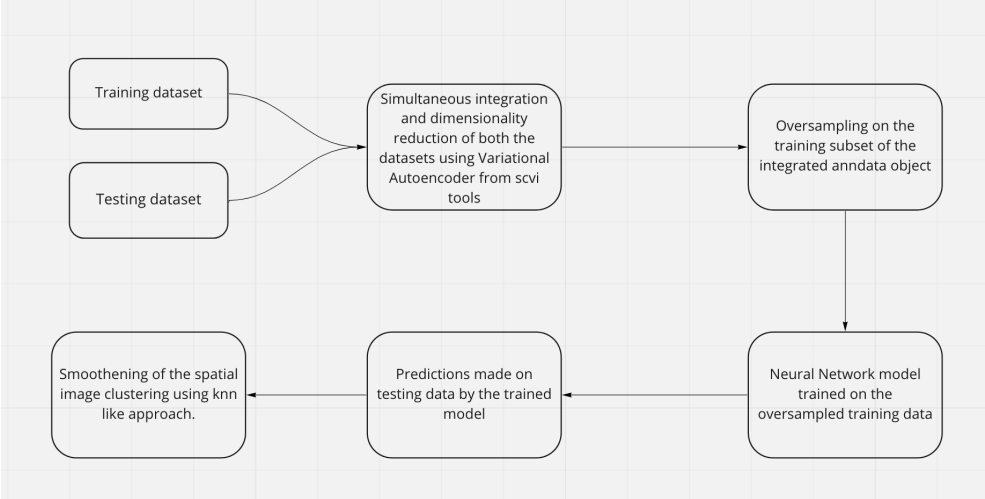

Fig. 2. The Pipeline used

the neural network model later on could make sense out of the testing dataset's latent variables and make correct predictions. If we would have performed dimension reduction on the two datasets (training and testing) separately, then the model for getting their latent space from the high dimensional data would have been different, and the neural network model wouldn't have been able to correctly predict the testing data's behavior.

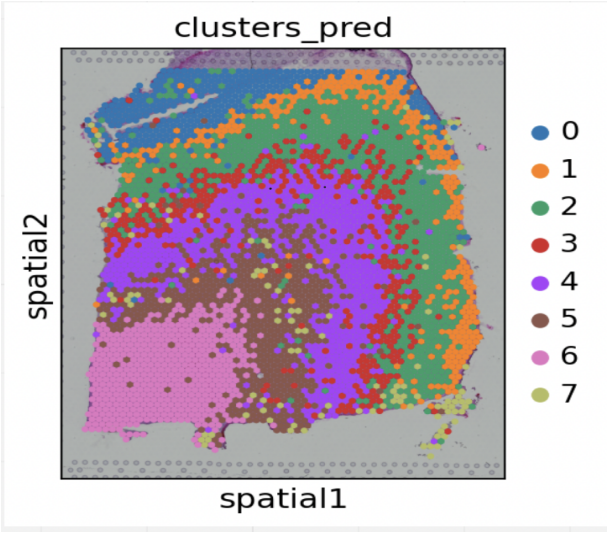

Fig. 3. Clustering on test data set straight after model

After integration, a single anndata object was obtained which had the information about both the datasets. adata.obs was split into two parts and oversampling was performed on the training part of the split. Oversampling involved making the number of datapoints in the dataset for all clusters same. Thus, if cluster 0 had 900 datapoints, as it covered a larger area of the sample, and cluster 2 had only 200, then by means of duplication of datapoints, both clusters' datapoint number was made same. Infact, to even better the model in learning the behavior and compensate for the less number of datapoints, every cluster's number of datapoints was made 5 times the size of the largest cluster initially.

After oversampling, an 80-20 train test split was performed on the training data itself, so as to ensure that no overfitting took place while training. A maximum testing accuracy (on the 20% part of the training data set) of 95% was obtained. The corresponding **ARI was 0.9**. After this, the predictions were made on the test data set using this trained model.

The initial predictions were decent, but required some post processing.

The post processing consisted of a few levels of smoothening, which was done in a method similar to k nearest neighbours. The first smoothening step was picking out spots, which were surrounded by all same coloured, but dissimilar to itself. Such spots were made the colours which their neighbours had.

Next, for any given spot, the predictions by the model for the spots in 2 layers of its vicinity for all the possible clusters were summed up. After summing up, the cluster which had the maximum value for the spot, was made the colour for that spot.

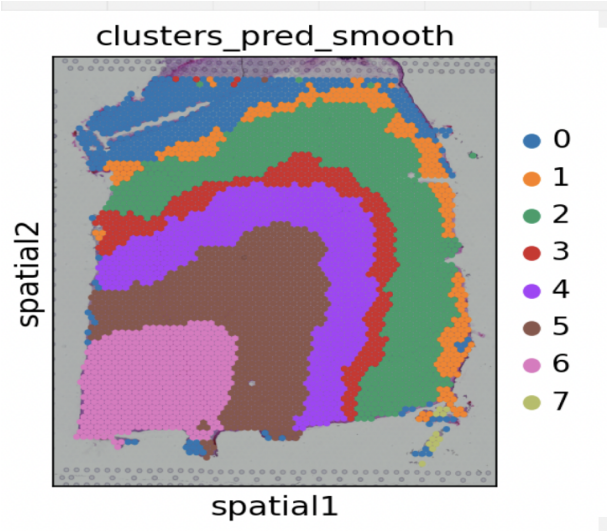

Fig. 4. Clustering on test data set after post processing

After this step, a final slight smoothening was performed, which consisted of making spots, which were surrounded directly by 5 spots of same colour, change into that colour. This particular step was performed iteratively 3 times, to clean up all spots which were in a non native environment. After all these steps, such an image was obtained:

4 TEAM

The names and roll numbers of the team members along with their contribution is listed below:

Name	Roll No.	Contribution
Shashank Katiyar	190794	Pipeline Development and Implementation of Task 2, and Report
Gurbaaz Singh Nandra	190349	Task 1 and SpaGCN Code implementation, Task2 testing and Report
Antreev Singh Brar	190163	Algorithm Development, Literature Review and Report

5 REFERENCES

(1) Atlas-level integration of lung data: https://docs.scvi-tools.org/en/stable/tutorials/notebooks/harmonization.html. Accessed: 2022-10-04.
(2) jianhuupenn/SpaGCN: SpaGCN: Integrating gene expression, spatial location and histology to identify spatial domains and spatially variable genes by graph convolutional network: https://github.com/jianhuupenn/SpaGCN. Accessed: 2022-10-05.
(3) https://github.com/vtraag/leidenalg: Leiden algorithm in C++