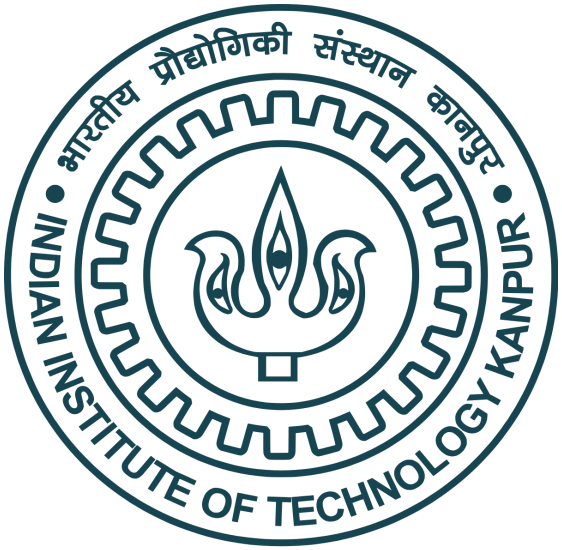
CS 690: Computational Genomics

Indian Institute of Technology, Kanpur



TEAM - 5

Authors:
Mubashshir Uddin[190516]
Suyash Mallik
Divyam Jain

Instructor - Hamim Zafar

PROJECT REPORT

Date -23/11/2023

All of the code for the following report can be found on-
https://github.com/pitabread7022/CS690.git

Objective:

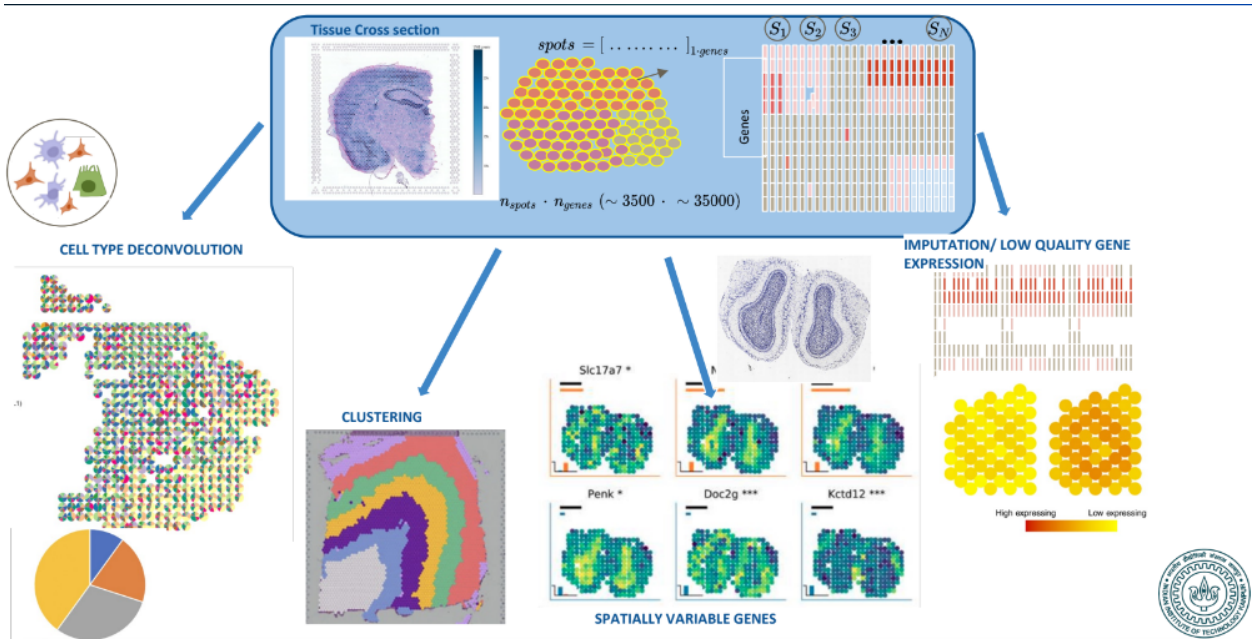
Imputation and denoising of spatial transcriptomic data

Spatially resolved transcriptomics (SRT) provides gene expression close to, or even superior to, single-cell resolution while retaining the physical locations of sequencing and often also providing matched pathology images. However, SRT expression data suffer from high noise levels, due to the shallow coverage in each sequencing unit and the extra experimental steps required to preserve the locations of sequencing.

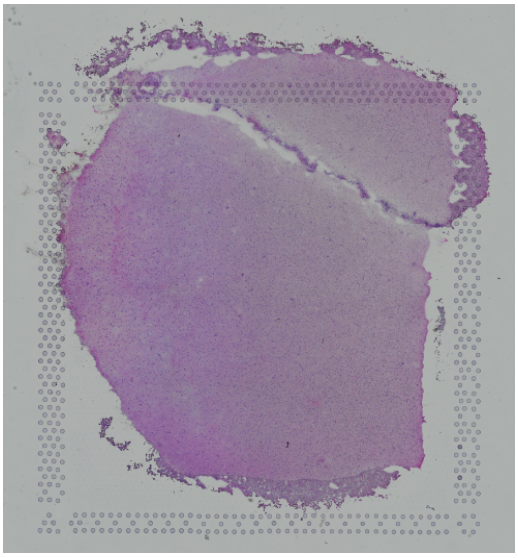
The goal of this project is to develop a deep generative model for spatial transcriptomics data that will utilize the information from the physical locations of sequencing, and the tissue organization reflected in corresponding pathology images to learn a latent representation of the data which can be used for denoising the data. Several graph neural network models are available for data imputation. Some of these models can be extended to incorporate the spatial graph for imputation. Standard datasets used by other imputation methods can be used for evaluation.

Introduction:

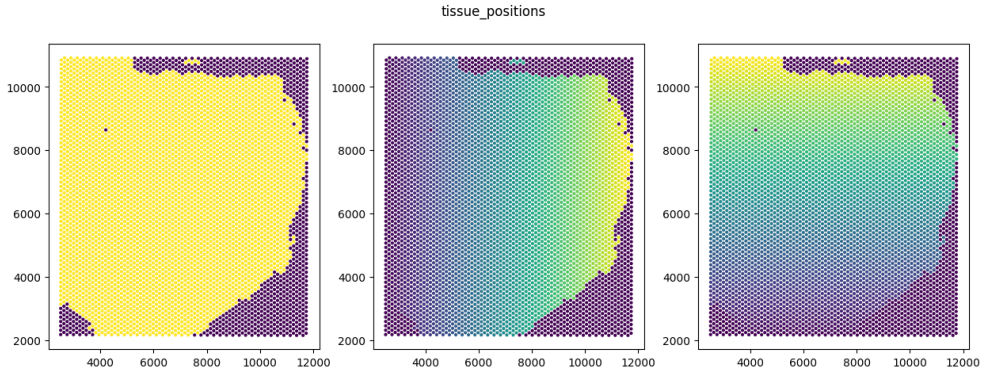
Spatially transcriptomics refers to the mutual compilation of data in the form of microscopic images and in-place gene sequencing:



Modalities:

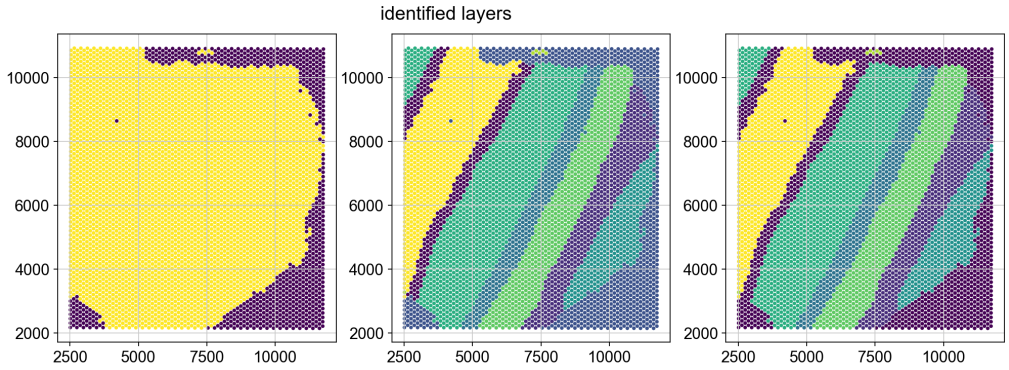


- Image Modality:
  - High-resolution microscopic image of the tissue with a scale factor of 0.15
  - Passing in the image through convolutional VAE, turning into a latent vector
  - Scale-invariant VAE training to best reproduce the image implies that most of the data from the image is retained in the latent space.
  - Passing image in the forms of different scale patches, e.g. if the image is 100 x 100 then we pass 100,00 (1x1), 2500(2x2), 625(4x4), .., 4(50x50),1(100x100). Set of image patches in a hyperparametric combination, letting us control the information sharpness content of the image at different levels.
- Spatial position modality:
  - The spatial positions of the spots that have been mapped are provided -



  - Above is the information on the gradient space in the cells and the presence /absence of the spots.
  - We also have available metadata.tsv file which is downloaded from the 10xvisium database, has information about which layer the given barcode comes from, we have heavily mixed but

nearly complete barcode mapping from the metadata.tsv to the tissue_positions_list.csv.

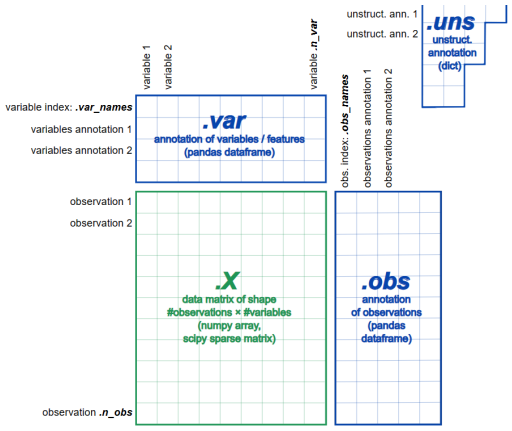


The data presenting tissue positions list - x,y are the coordinates of the barcode

	barcode	present	xgrad	ygrad	x	y	layer_guess	color
0	ACGCCTGACACGCGCT-1	0	0	0	2510	2174	NaN	2
1	TACCGATCCAACACTT-1	0	1	1	2630	2243	NaN	2
2	ATTAAAGCGGACGAGC-1	0	0	2	2511	2312	NaN	2
3	GATAAGGGACGATTAG-1	0	1	3	2631	2381	NaN	2
4	GTGCAAATCACCAATA-1	0	0	4	2511	2450	NaN	2

(4992, 8)

- Single-cell RNA sequencing data:
  - A relatively lower quality sc-RNA seq read is obtained from each one of the labelled cells -




```
ADATA uniqlyr: {'BA9', 'BA24', 'BA46', 'PFC'}

Adata obs:
                                  ground_truth batch             barcode
GAGGTGAAGTGCGTGA-1_4899_BA24-0    Neu-NRGN-II     0  GAGGTGAAGTGCGTGA-1
AGAGCGAAGCACCGTC-1_5554_BA24-0    Neu-NRGN-II     0  AGAGCGAAGCACCGTC-1
CACCAGGGTGTATGGG-1_5841_BA9-0     Neu-NRGN-II     0  CACCAGGGTGTATGGG-1
CAGAGAGTCAGTTTGG-1_5841_BA9-0     Neu-NRGN-II     0  CAGAGAGTCAGTTTGG-1
AGTGAGGTCATACGGT-1_5387_BA9-0     Neu-NRGN-II     0  AGTGAGGTCATACGGT-1
...                                       ...   ...                ...
GCCAAATCACCAGTTA-1_6032_BA24-16       AST-PP    16  GCCAAATCACCAGTTA-1
AAAGATGGTACCGTTA-1_5945_PFC-16        AST-PP    16  AAAGATGGTACCGTTA-1
CCGTACTTCACGATGT-1_5531_BA24-16       AST-PP    16  CCGTACTTCACGATGT-1
CACAGGCTCTGGAGCC-1_5565_BA9-16        AST-PP    16  CACAGGCTCTGGAGCC-1
GACGGCTGTTCCTCCA-1_5939_BA9-16        AST-PP    16  GACGGCTGTTCCTCCA-1
```

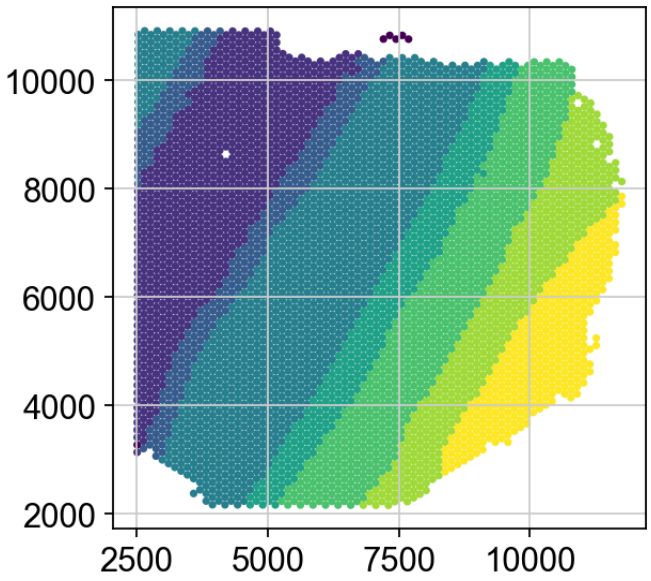
Adata Obs: [*20904 rows x 5 columns*]

Ground truth layers (Obs):
['Oligodendrocytes', 'Endothelial', 'IN-PV', 'Neu-mat', 'IN-VIP', 'OPC', 'AST-FB', 'L5/6-CC', 'L2/3', 'L4', 'AST-PP', 'Neu-NRGN-I', 'L5/6', 'Neu-NRGN-II', 'Microglia', 'IN-SV2C', 'IN-SST']

Adata vars(gene names){33538 *rows x 0 columns*}: [DDX11L1, WASH7P, MIR6859-3, RP11-34P13.3, MIR1302-9, FAM138A, OR4G4P, OR4G11P, Etc.,...]

```
AnnData object with n_obs × n_vars = 4226 × 33538
    obs: 'in_tissue', 'array_row', 'array_col', 'label', 'present', 'x', 'y', 'xgrad', 'ygrad', 'layer_guess'
    var: 'gene_ids', 'feature_types', 'genome'
    uns: 'spatial', 'neighbors', 'umap', 'layer_guess_colors'
    obsm: 'spatial', 'ConGI', 'X_umap'
    obsp: 'distances', 'connectivities'
```

Layer guesses: (as presented in the adata.obs['layer_guess'])

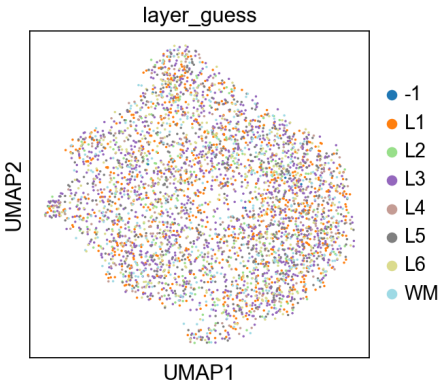


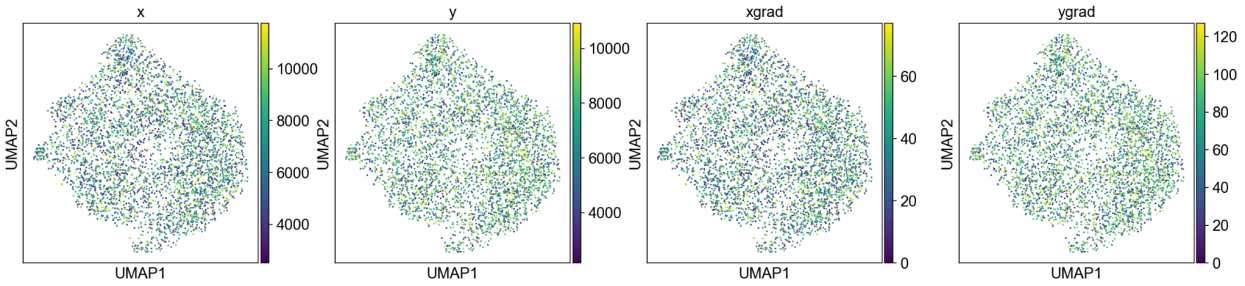
Denoising algorithms proposed:

Learn the best predictions using the latent. then check the variance from the best predictor, if the variance is greater than a threshold then impute.

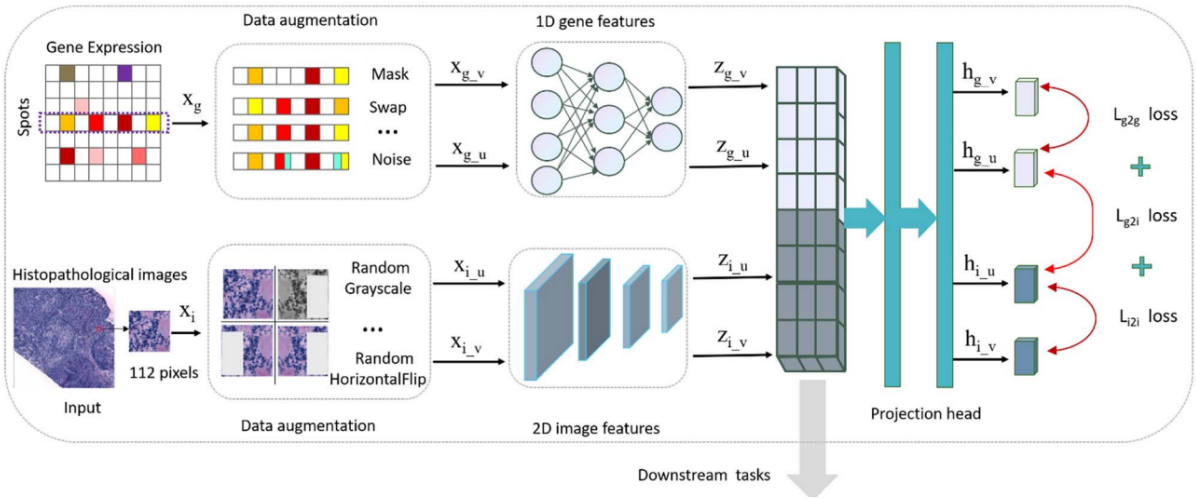
```
(4226, 33538)
                 gene_ids     feature_types  genome
MIR1302-2HG  ENSG00000243485  Gene Expression  GRCh38
FAM138A      ENSG00000237613  Gene Expression  GRCh38
OR4F5        ENSG00000186092  Gene Expression  GRCh38
AL627309.1   ENSG00000238009  Gene Expression  GRCh38
AL627309.3   ENSG00000239945  Gene Expression  GRCh38
```

	in_tissue	array_row	array_col	label	present	x	y	xgrad	ygrad	layer_guess
AAACAACGAATAGTTC-1	1	0	16	0	1	2513	3138	0	14	L1
AAACAAGTATCTCCCA-1	1	50	102	7	1	2633	3207	1	15	L2
AAACAATCTACTAGCA-1	1	3	43	6	1	2514	3276	0	16	-1
AAACACCAATAACTGC-1	1	59	19	0	1	2634	3344	1	17	L1
AAACAGCTTTCAGAAG-1	1	43	9	0	1	2514	3414	0	18	L1





CONGI:



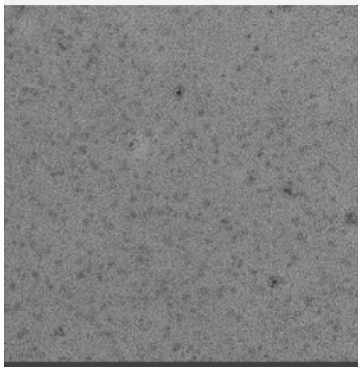
ConGI is a deep learning model that finds out a combined latent representation of the ST data, in the form of embedding. These embeddings can then be downstream clustered to identify which cell barcodes belong to which cluster. The generated embeddings are in the form of a form of **4226 x 64** matrix, i.e. each observation (each labelled spot barcode) is from a 64-dimensional vector:

```
(4226, 64)

array([[-0.55242294,  1.0182328 ,  3.1839    ,  1.2573069 , -0.3348279 ],
       [-0.19708535, -0.15139353,  0.02747403,  0.2082556 , -0.01718359],
       [ 1.117155  ,  0.08230081,  1.6215928 ,  0.54727983,  1.6493006 ],
       [-0.6783914 ,  0.2167799 ,  1.5858941 ,  1.466289  ,  0.86016184],
       [ 0.01491459, -0.29879335,  0.45403385, -0.20938306,  0.496131  ],
       [ 0.01595331,  0.21615848,  0.12688774, -0.22155748,  0.6266215 ],
       [-0.27455813,  0.62711424,  0.33154806,  0.7267024 ,  0.658925  ],
       [-0.45410928, -0.2204201 , -0.17317201,  0.73684335, -0.28500366],
       [ 0.11525718, -0.29261753,  0.40823317,  0.03635505, -0.06608421],
       [-0.4349867 , -0.02552745,  0.1797252 ,  0.47001764,  0.0373777 ]],
      dtype=float32)
```

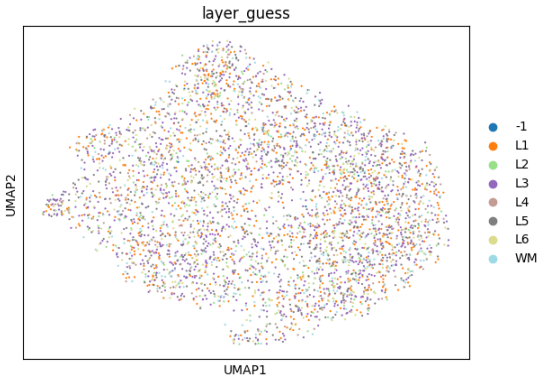
These embeddings can be used to cluster the cells into a set of 9 categories:
Why are we choosing 9 categories - because the initial labelled data has 9 - layers:

```
    set(df['layer_guess'])
✓ 0.0s

{'-1', 'L1', 'L2', 'L3', 'L4', 'L5', 'L6', 'WM', nan}
```

It makes it possible that the cells are lumped into similar layers that are present in the observations.

 <- following is an image patch as is passed into the convolution net

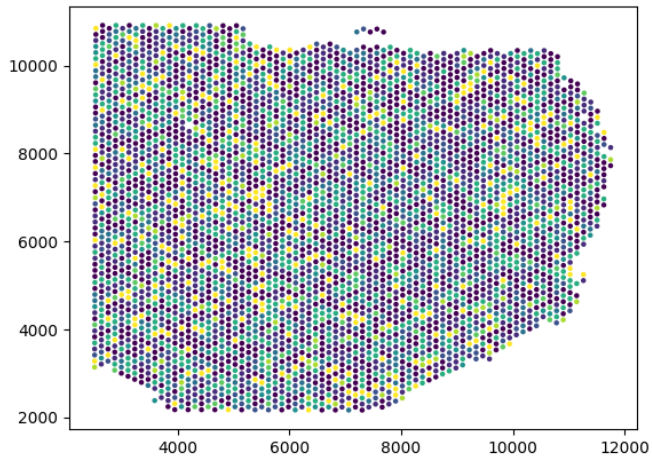
What do we observe:



The congi algorithm produces the nearest neighbour graph such that, we are able to produce a UMAP plot, the map plot is then labelled according to the:

- Cluster predictor from ConGI embedding
- Layers_guess already present in the adata

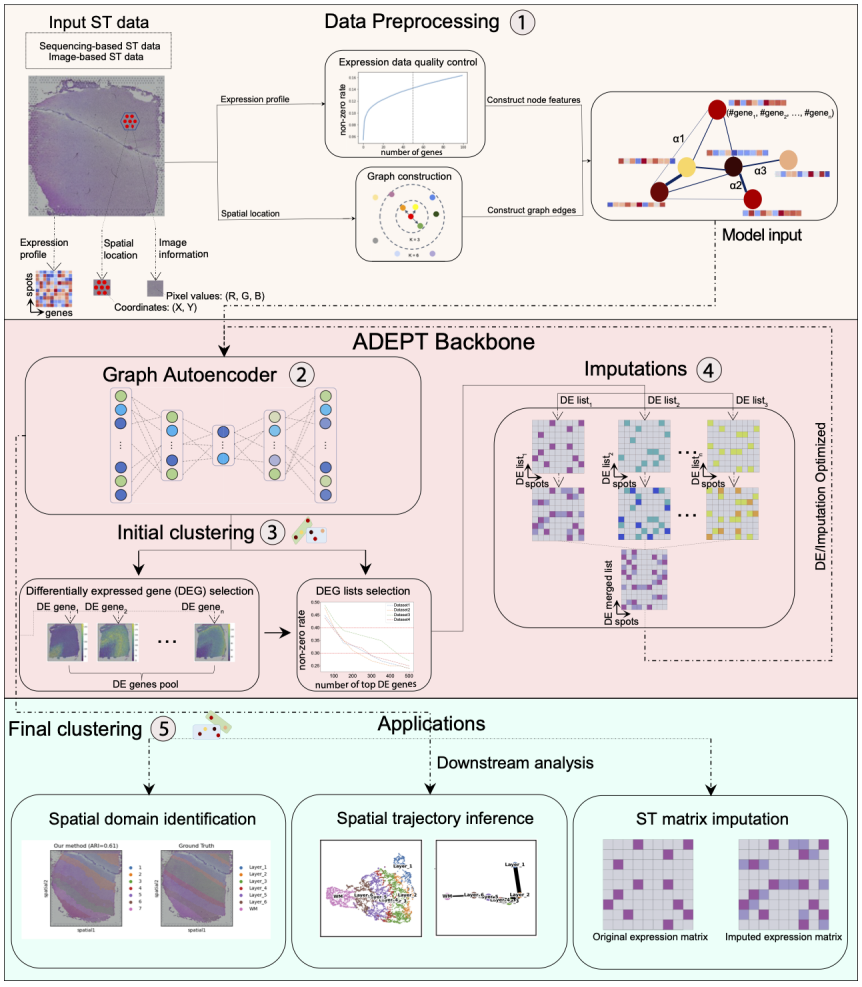
To use the congi embedding as a cluster predictor we first use the Leiden algorithm to assign the clusters.

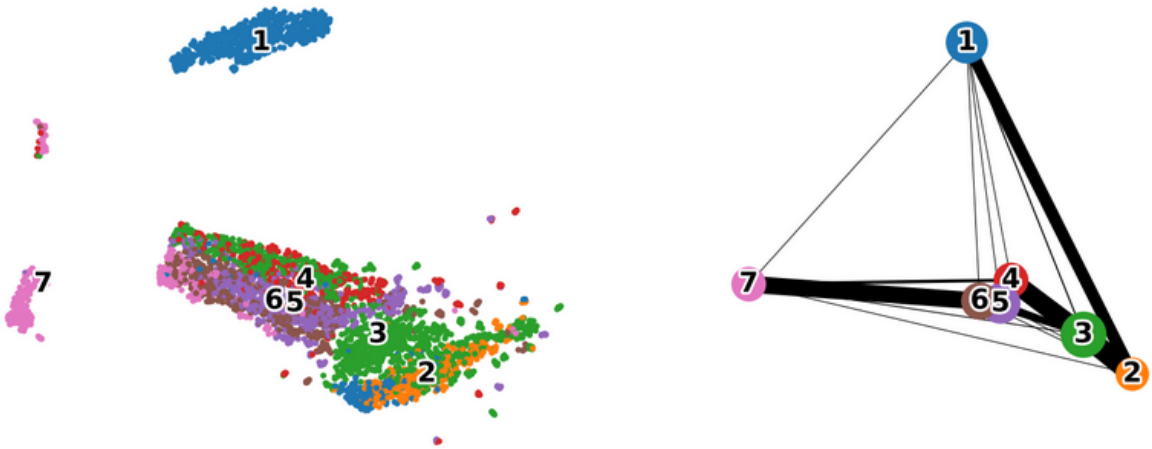


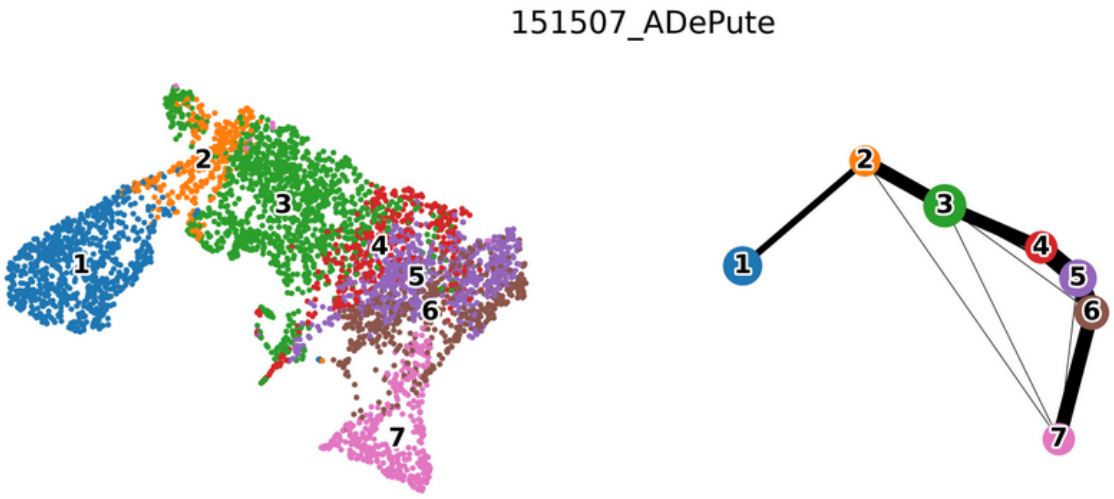
As we see the congi embeddings in themself do not do a nice job of finding the best predictor of imputation, the possible reason for this:

- The ConGI embeddings might not have been generated appropriately and do not effectively capture the cellular variations in the image and the gene matrix, there might be a barcode collapse during the congi process which is entirely separate from the imputations
- The Leiden algorithm clusters might not correspond to the layer guess clusters, there are many things about the cell that one might predict from the gene expression and tissue image, and the cell layers might not effectively be one of them. Also, there might be a weak correlation between the layer's guess and the ConGI predictions. In which case we might amplify the correlation and then impute using CONGI

Other methods:







How we plan to make the predictions:

We can spot-wise impute using the congi algorithm, and then use an adept imputation. For both of these methods, we will use ADEPT's approach of choosing Differentially Expressed genes (chosen by z-score), and then impute the cluster mean on them.
If the magnitude of the imputation difference between CONGI and ADEPT is larger than a certain threshold then we can impute the value of the gene expression of the cell.

THRESHOLD:

- If the CONGI algorithm labels the spot in a different cluster than the layer prediction
- If the ADEPT algorithm proposes a different cluster than the layer prediction
- If both CONGI and ADEPT label most of the other spots in the same layer according to the adata['layer_guess'](other than our subject spot) into the same label basket.
- ONLY if we have confirmation from both - congi and Adept, that the spot is to be labelled as a different layer and the imputation is accepted.

Pre-processing and cleaning of the data:

In the current situation, we've been utilizing the configuration embedding from the prior assignment, and it did not involve the preprocessing step. As a result, the count of spots is 4226, which is higher compared to when preprocessing is applied early in the process. While it's entirely feasible to implement preprocessing at an earlier stage, it's essential to note that only newer embeddings need to be generated in the training data.

Contribution
Mubashshir Uddin - ConGI; Suyash Mallik - ADEPT; Divyam Jain - Data preprocessing