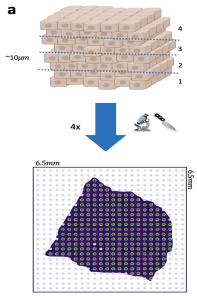
Spatial Transcriptomic Imputation

Supervisor: Dr. Leon Anavy
Student: Guy Attia - 305743437

Project Overview

Genes Analysis Background

Analysis of gene expression profiles in biological samples is a powerful tool used to study various biological systems and phenomena. Traditional assays measure bulk gene expression levels of relatively large samples (i.e. containing millions of cells) yielding robust measurements but hiding cell to cell variability. In the last decade, high throughput single cell RNA-Seq (scRNA-Seq) technologies were developed to capture this variability for thousands of cells simultaneously allowing for in-depth analysis of biological tissues. However, even with single cell resolution, the spatial information over the measured tissue is lost with scRNA-Seq.



Recently, new technologies measure gene expression profiles of biological tissues while maintaining spatial information. Spatial transcriptomics (ST) is a new technology for measuring RNA expression in tissues while preserving spatial information. ST involves placing a thin slice of tissue on an array covered by a grid of barcoded spots and sequencing the mRNAs of cells within the spots.

These Spatial Transcriptomics (ST) technologies allow for studying complex tissues where direct interactions between different cell types affect the biological system. For example, when studying cancerous samples, the effect of the tumor microenvironment is directly associated with the disease stage, treatment decisions and survival rate. Figures 1 and 2 illustrate the type of data included in a ST dataset.

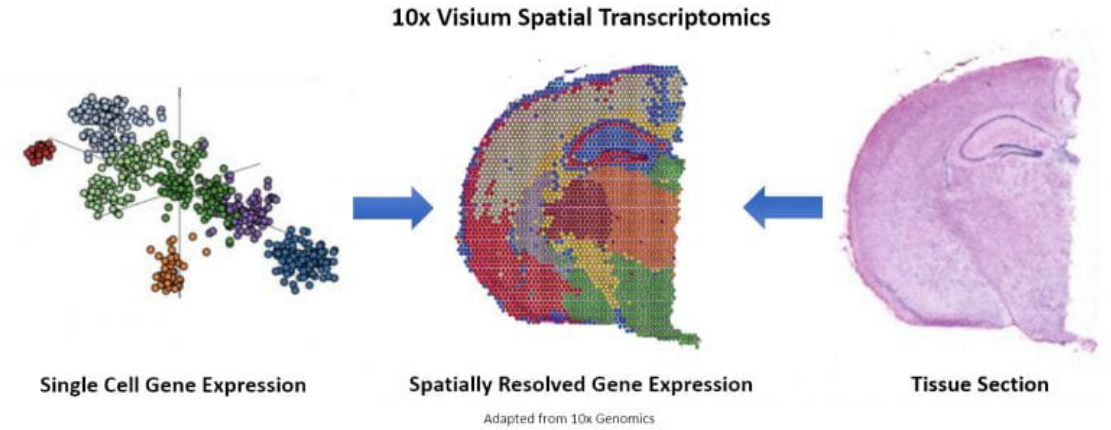


Figure 1: Example of the difference between a single cell gene expression and the spatial transacriptomics data for some tissue (CQB [1]).

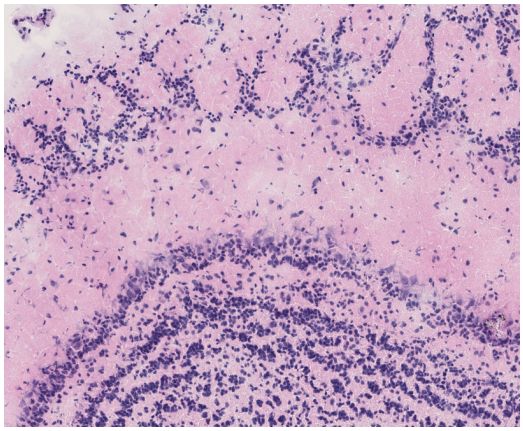


Figure 2: Example of an high resolution H&E image, specifically of a mouse's olfactory bulb (10x Genomics dataset [2]).

ST Limitations

With today's technology, RNA molecules are captured and sequenced in the spots on the spatial genomic array aligned to the locations on the tissue. For example, spatial transcriptomics technology based on 10x Genomics Visium kit reports the number of copies of RNAs by counting unique molecular identifiers (UMIs) in the read-pairs mapped to each gene. There is also the issue of multiple cells per spot which means that the "bulk measurements hides variability" problem still holds. There are still significant technical difficulties. It's capturing has a low RNA capture efficiency. The sample preparation requires highly specific handling of tissue sections. The spots in some tissue regions might entirely fail to fix and permeabilize RNAs due to various possible issues in preparing tissue sections. In scRNA-seq data analysis, the missing gene expressions are called dropout events, which refer to the false quantification of a gene as unexpressed due to the failure in amplifying the transcripts during reverse-transcription. It has been shown in previous studies on scRNA-seq data that normalizations will not address the dropout effects [3, 4].

These problems and more can be classified into two topics:

- Sparsity or zero inflation: The generated expression data is very sparse, capturing only a partial view of the complete gene expression profile in each spot. Sparse data is a variable in which the cells do not contain actual data within data analysis. It is empty or has a zero value. In our case, the meaning is the that there is a very high percentage of 0's expressions in the ST datasets, which is a big limitation in analyzing it for downstream tasks as described above.
- Cost: Generating an ST dataset is costly both in time and in money and thus the number of available measurements is limited.

These limitations may be solved in the future with newer and smarter technology, but based on recent publications, it might take a long time before the generated expression data will be dense enough.

Project Goal

In the literature, many imputation methods such as Zero-inflated factor analysis (ZIFA) [5], Zero-Inflated Negative Binomial-based Wanted Variation Extraction (ZINB-WaVE) [6] and BISCUIT [4] have been developed to impute scRNA-seq. While these methods are also applicable to impute the spatial gene expressions, they ignore a unique characteristic of data, which is the spatial information among the gene expressions in the spatial array, and do not fully take advantage of the functional relations among genes for more reliable joint imputation.

The goal of this project is to deal with the depth limitation of the ST technology and perform data imputation on ST dataset to reduce the data sparsity. To do so, I will use DL models and techniques for using the spatial information to enrich the ST information.

Data

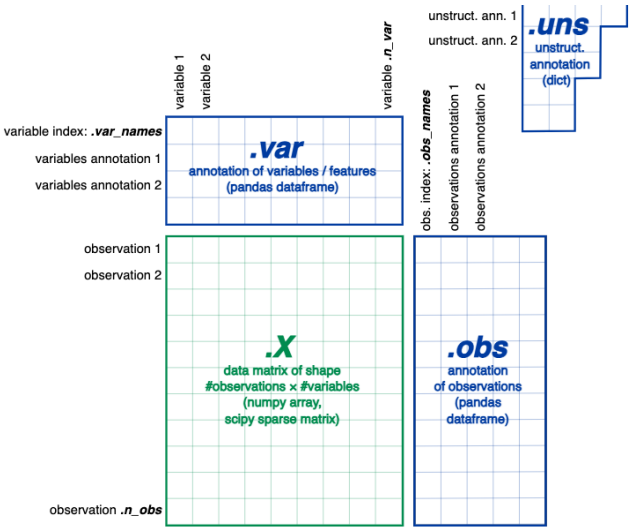
10X Genomics obtained fresh frozen mouse olfactory bulb tissue from BioIVT. The tissue was embedded and cryosectioned. Tissue sections of 10µm were placed on Visium Gene Expression slides, then fixed and stained following Methanol Fixation, H&E Staining & Imaging for Visium Spatial Protocols. More detailes about how the data is created can be found here.
10X Genomics have released free datasets for the purposes of collaborative research.
Out of these datasets, I've chosen to work on the "**Visium Mouse Olfactory Bulb**" dataset.

Dataset Structure

10x Genomics data type is h5. This data object is an object store which contains multiple objects in key-value pairs. Those values can be lists, dictionaries, pandas DataFrames and more. The object store is accompanies with various tissue images describing the tissue and it's coordinates.

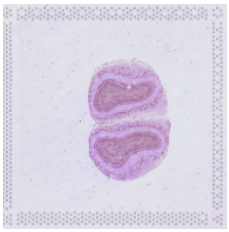
Specifically, this object store is wrapped with an AnnData object and consists 4 key-value pairs:

1. **X** - A #observations × #variables data matrix (Scipy sparse matrix). In our case, the gene-spot expressions
2. **obs** - Key-indexed one-dimensional observations annotation of length #observations. (Pandas DataFrame). obs features: ['in_tissue', 'array_row', 'array_col'] which correspond to the spatial position of the spot in the tissue
3. **var** - Key-indexed one-dimensional variables annotation of length #variables. (Pandas DataFrame). var features: ['gene_ids', 'feature_types', 'genome']
4. **uns** - Key-indexed unstructured annotation. (AnnData.OverloadedDict object)



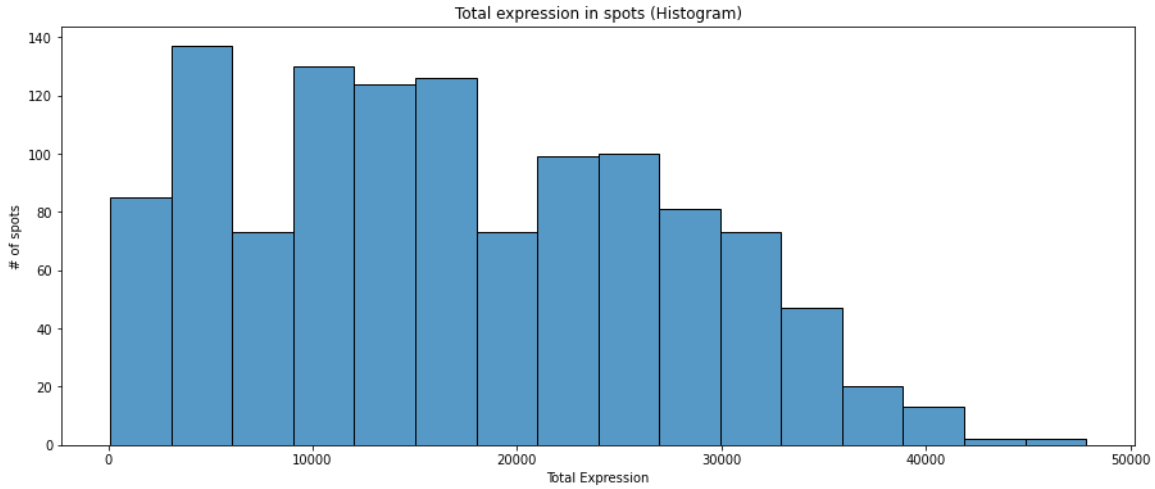
Exploratory Data Analysis

The Visium Mouse Olfactory Bulb" dataset is describing the spatial gene expressions of **32285 genes over 1185 spots.** Where the actual tissue can be seen in the following image:

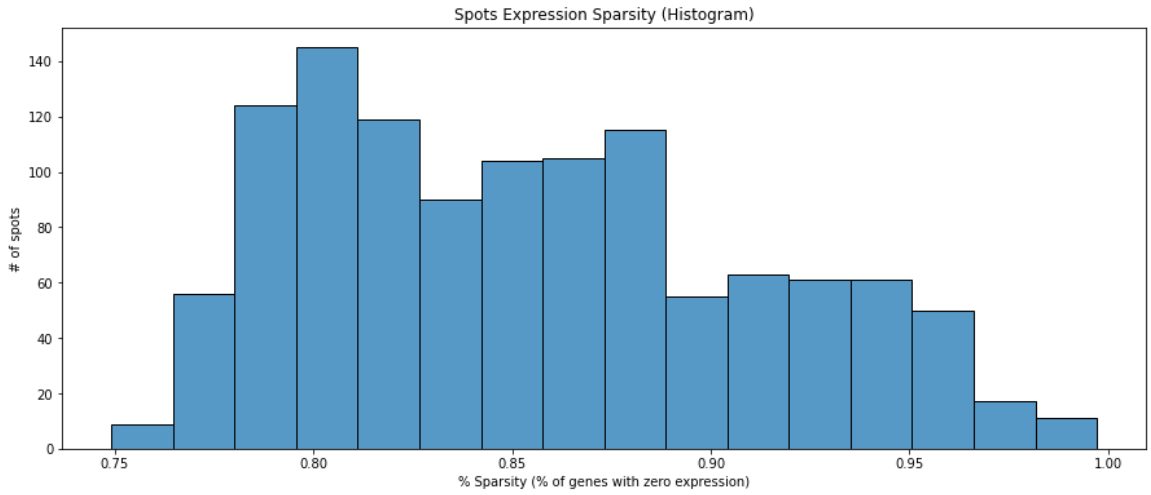


As always, the first part of the project should be a deeper analysis over the data to understand the data distributions. More precisely in this project we would like to understand the data sparsity much better to understand how to prepare and manipulate the data.

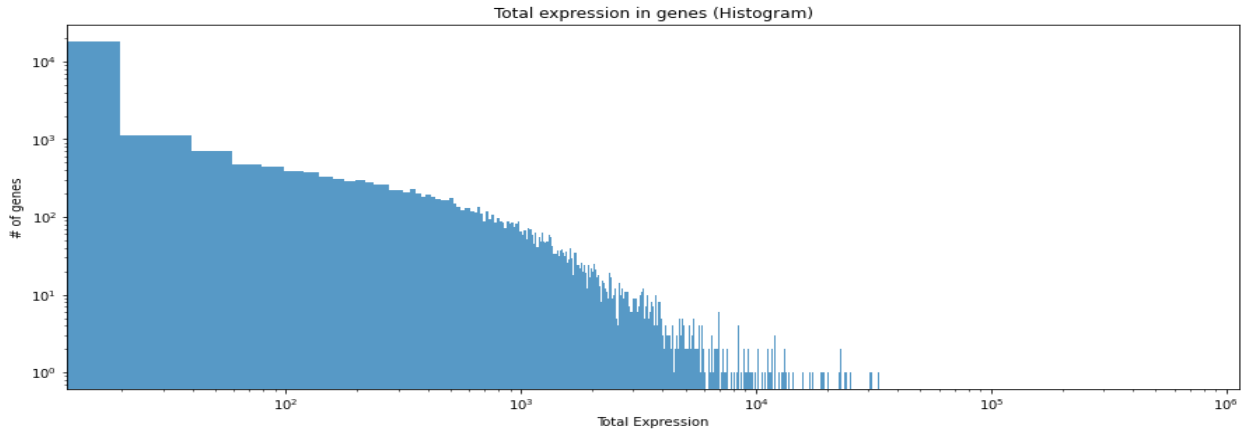
First, to understand the overall spots sparsity (how many spots have total 0 expression), I've showed their total expression histogram. It's clear that most spots have expression in at least some of their genes so their total expression is higher than 0.

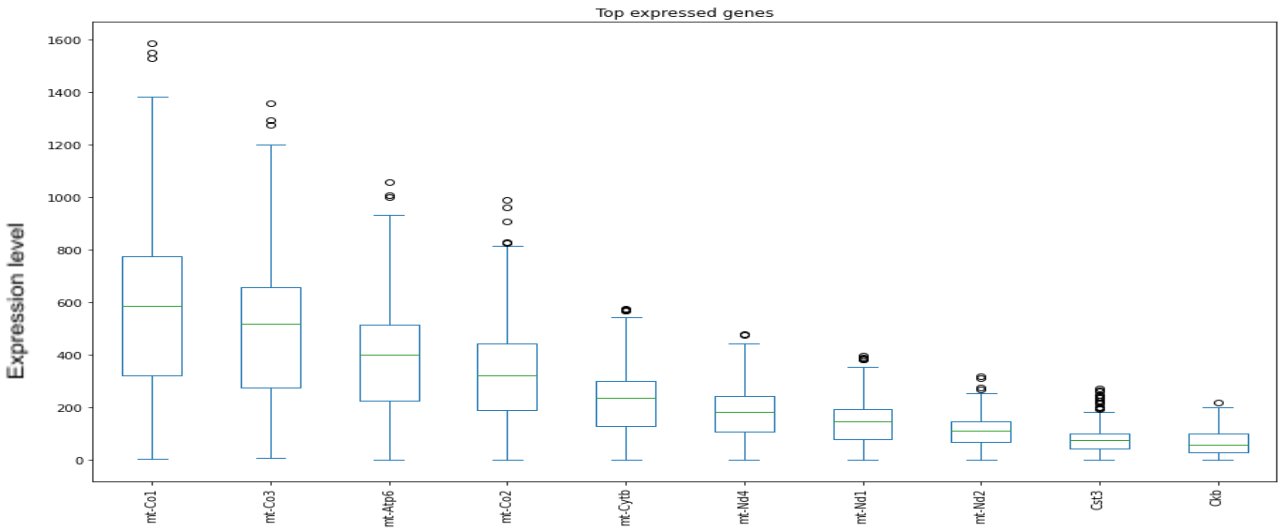


Then, To verify the sparsity assumption known in the literature, I actually care about each spot gene sparsity (in each spot, how many genes have 0 expression). For that I've calculated the % of genes with 0 expression for each spot and shown their histogram.
It's possible to see that in accordance to the literature, most spots have 0 expression in at least 80% of their genes.



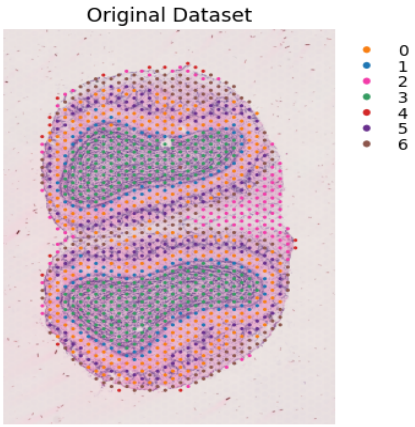
In addition, to understand the distribution of expressions by genes over the spots, I've showed the genes total expression and a box-plot diagram to see the top expressed genes distribution over spots.
It's possible to see that many genes have a small amount of total expressions and specifically there are only few genes who expressed well over many spots. That is to be expected, even in bulk RNA samples, only a small subset of the genes is relevant for a given tissue in a given time.





Using this information we understood that it is possible to filter many genes who are too sparse and for this research project, deal only with minimum expressed genes.

Downstream Task - Spots Clustering



A common way to explore ST data is to apply clustering on the spots and get a nice areas separation on the tissue image by their cluster.
Using the Python library *stLearn*[7] I found a fast and already configured way to apply the Kmeans clustering (K=7 due to the benchmark analysis made by 10xGenomics) algorithm on the expression matrix. Due to the large number of genes and overall data, I also used the *stLearn* PCA functionality to lower the genes dimension before the clustering into 50 PCs (used as a default without optimizing it).

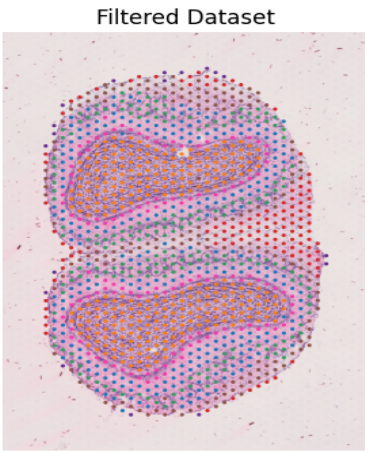
Solution

As mentioned earlier, the goal of this project is to impute the gene expressions using the available spatial information in the ST data.
There are different known ways to apply data imputation:

- Imputation using mean / median / mode values
- Imputation using KNN algorithm (based on features similarity)
- Extrapolation and interpolation methods
- More …

In this project I've turn into a DL technique to impute the missing values. The idea is to embed the gene expression dataset into a latent space and then reconstruct it to the original shape with the predicted values [8].

Data Preperation



As mentioned above in the EDA, I've decided to filter a lot of the genes who were too sparse from the dataset. I've filtered based on the following two criteria:

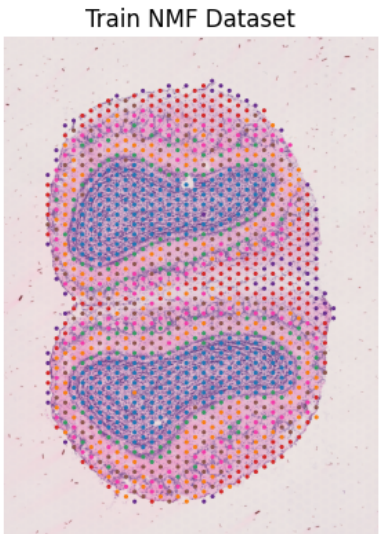
1. Keep genes with at least 15% non zero spots
2. Keep genes with total expression of at least 500 over all spots

The filtered dataset holds 6279 genes out of the 32285 originals and still we can see a nice area separation after the clustering.

Reconstruction Evaluation

The evaluation mechanism is very dependant on the model type and reconstruction technique, while the chosen methodology is the same. Due to the fact that in this project we don't actually have several datasets of the same tissue expression, we need to manually generate the validation and test datasets while trying to avoid data leakage and overfitting.
The spot-gene expression matrix holds all the combinations of spot-gene-expresion, while most of them are 0's. I've decided to evaluate the reconstruced expressions matrix only on X% of these combinations (both in validation and test set). Therefore, I've replaced a X% non zero expressions with the actual 0 expression, and in the evaluation process I've used only these combinations to generate the model score.
A demonstration for the replacement process:

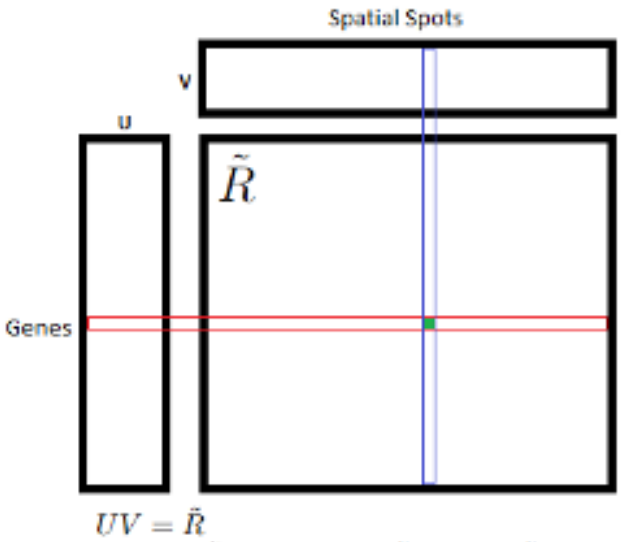
To decide on the X% of zeros in each dataset I had to make sure that the training data is less applicable for the same area separation after the clustering, otherwise we won't see any different in the downstream task. At the end X was defined as 20%.



As common in similar tasks of matrix reconstruction, it's a good way to calculate the evaluation score with the RMSE metric.

Models & Techniques

The original proposed solution was to use the matrix factorization technique, while using the spatial information as a regularization term to better reconstruct the expression data, and require the reconstruction algorithm to generate "spatially smooth" predictions.
Formally, let R be the original m×n real matrix with m genes and n spatial spots covering the slide where $R_{i,j} \in R$ is the expression level of gene i in spot j. Let U and V be the latent representations of the genes and the spots respectively and let $UV = \tilde{R}$ be the reconstructed expression matrix, while the idea is to find U and V such that the mean squared difference (error) between R and $\tilde{R}$ is minimized.



My research methodology is to first set a baseline with the most simple model and than escalate to the more advanced models and adjustments available.
In this project the baseline model was a plain vanilla Neural Matrix Factorization (NMF) model without additional data processing or loss adjustments.
Then, I've performed the 4 additional trials on NMF and also on Auto-Encoder model (AE).

NMF is a different way to apply the normal matrix factorization (MF) approach by using a neural network architecture (Figure 3).

- Input Layer binarise a sparse vector for a spot and gene identification where:
    - Spot (i): 1 means the gene u has interacted with gene(i)
    - Gene (u): To identify the gene

  When feeding the network, there is a need to supply it with:
    a. One-hot vector of spot ID
    b. One-hot vector of gene ID
    c. Their expression value
- Embedding layer is a fully connected layer that projects the sparse representation to a dense vector. The obtained spot/gene embeddings are the latent spot/gene vectors. P, and Q are both matrix of weights between the input and embedding layers, holding the actual latent values. Their dimensions are $P \in R^{MxK}$ and $Q \in R^{NxK}$ respectively where K is the embedding size. The trainable model parameters in this case are only between the input layer and the embedding layer.
- The element-wise layer map the latent vectors to prediction scores using a dot product linear transformation.
- The final output layer returns the predicted expression by minimizing the loss.
- $\hat{y}_{ui} = f(P^T v_u^U, Q^T v_i^I | P, Q, \Theta_f)$ where P and Q are the latent factor matrix for genes and spots, and Θ_f is the model's parameters.
- The loss function is: $L = \sum_{(u,i) \in y} (y_{ui} - \hat{y}_{ui})^2$

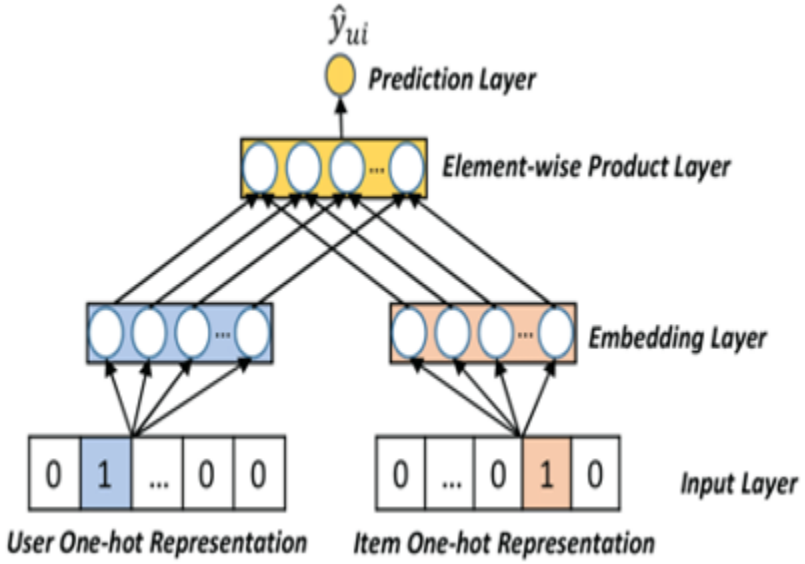


Figure 3: Basic architecture of NMF (Neural Matrix Factorization) for User-Item collaborative filtering.

Trial 1: NMF model + Log transform

To reduce the skewness of the expression data I've applied log transformation on it:
$X = log(X + 1)$ where log denotes the natural logarithm.

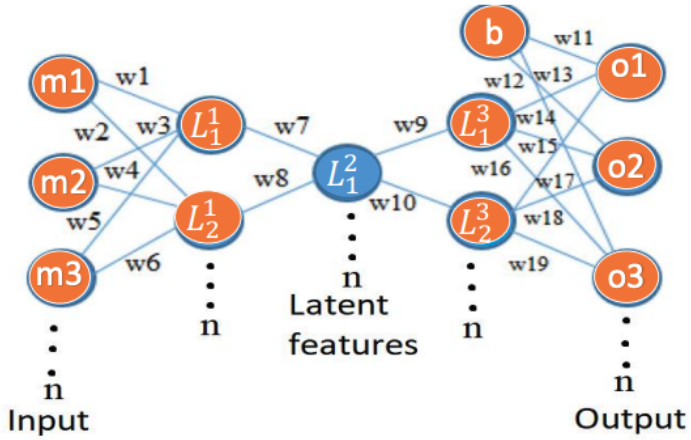
Trial 2: NMF model + Log transform + Non-zero RMSE loss

To reduce the many 0 expressions effect on the overall loss function, I've change the loss function to be calculated only on non zero expressions (of course only in the training and not in validation). In this way I try to tune the model weights to reconstruct non zeros, so I am pushing the expressions away from 0 since there is no penalty on overshooting 0's.

- The new loss function is: $L = \sum_{(u,i);y_{ui}>0} (y_{ui} - \hat{y}_{ui})^2$

Trial 3: Auto-Encoder model + Log transform + Non-zero RMSE loss

Another common way in the recommendation systems field to perform collaborative filtering is using an Auto-Encoder paradigm [8].
In this project expression-based collaborative filtering, we have m genes, n spots, and a partially observed gene-spot expression matrix. Each gene / spot can be represented by a partially observed vector. My aim was to design a gene-based autoencoder which can take as input each partially observed spots vector and project it into a low-dimensional latent (hidden) space, and then reconstruct it in the output space to predict missing expressions.



More specifically, I'm looking for an encoder E that gets as input a spot vector (expression of all genes in this spot) and output a vector in the latent space. This latent space vector is the input of a decoder D who's output is supposed to be the most similar spot vector we started with.

The loss function for this model is the same as before: $L = \sum_{(u,i)\in y \setminus y=0} (y_{ui} - \hat{y}_{ui})^2$

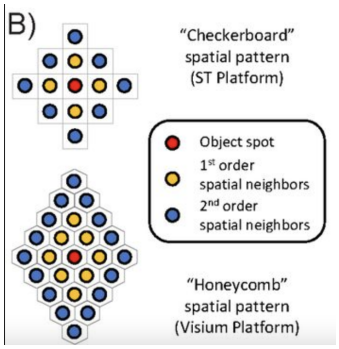
Trial 4: Auto-Encoder model + Log transform + Non-zero **Spatial** RMSE loss

The main goal of this project was to find a way to incorporate the spatial information in a way to optimize the model for a better reconstruction.
In contrast of the original proposed solution I've used the spatial information inside the loss function calculation as a spatial contribution to each spot.

Description

To create the spatial loss function, there is a need in first define the spots relationships and what is the level of contribution each spot should have from its neighbors.

- **Spatial neighborhood** - Based on [9] a group of closest spots to a specific spot spatially.
  By definition, a neighborhood of a spot is defined by the ST type



  In my case, the data is a Visium dataset so I've used the "Honeycomb" pattern.
- **Spatial Contribution -** The contribution will be the weighted spatial expression's difference of the spot from its neighbors (sum over all genes). The weights will be by the spatial neighbors order (1 or 2).

  $$L = |\widehat{R} - R|^2 + 2|\varphi_1(\widehat{R})|^2 + 1|\varphi_2(\widehat{R})|^2$$

  Where

  $$\varphi_1(\widehat{R}) = \sum_i^n \sum_j^{i_{s1}} |\widehat{R_i} - \widehat{R_j}|$$

  $$i_{s1} = first\ order\ neighbors\ (8\ neighbors)$$

  $$\varphi_2(\widehat{R}) = \sum_i^n \sum_j^{i_{s2}} |\widehat{R_i} - \widehat{R_j}|$$

  $$i_{s2} = second\ order\ neighbors\ (16\ neighbors)$$

Preprocessing

In order to build a spatial map for finding the neighbors of each spot, I've used the "array_col" and "array_row" information inside the data. As we can see in Figure 4 there is a correlation between these columns and the image location columns ("imagecol", "imagerow").

After encoding the spots names, I've created a matrix holding the 1st and 2nd order neighbors for every spot.

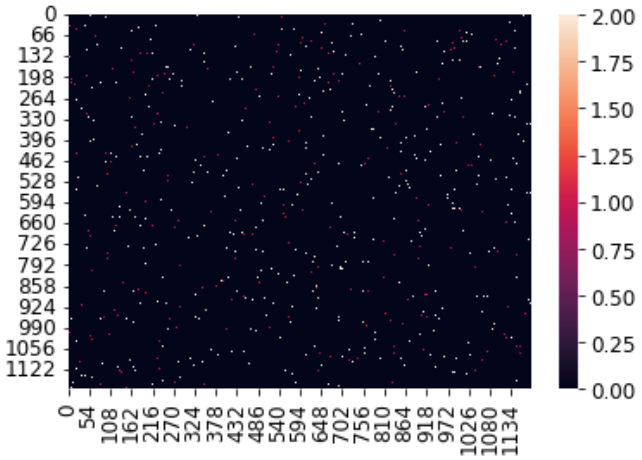


Figure 4: Heatmap spotting the neighbors from 1st and 2nd level degrees for each spot. Red points denotes a 1st level neighbor and white dot denotes a 2nd level neighbor.

Modeling

Due to the data structure for neural matrix factorization which is tuples of (gene,spot,expression), the spatial loss calculation time is too long. Also, because I'm optimizing with batches, it's quite rare to find a neighbor for a spot of the same gene in a specific batch, which makes the spatial loss useless.
As a result, I decided to use the AutoEncoder architecture to leverage the vectorized structure of the data.

Results

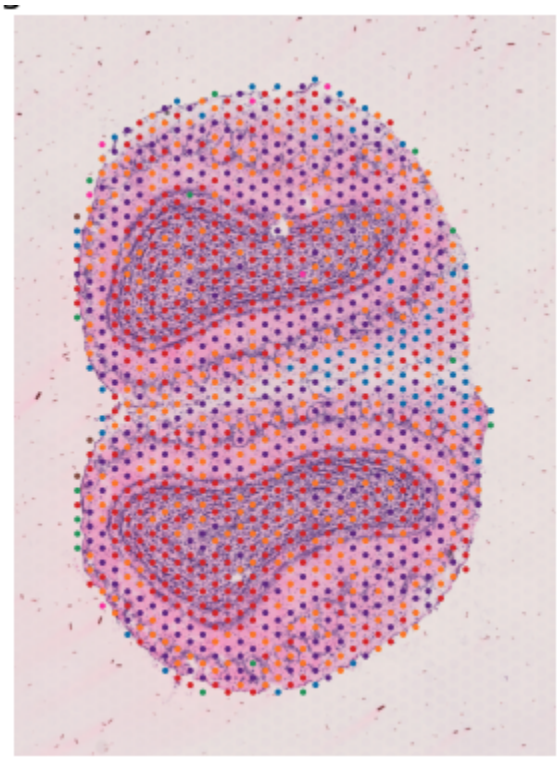
As described above, I've split the data into train, validation and test sets and calculated the RMSE score for each one.
The scores for the validation and test sets include only the replaced expressions and not the entire matrix.
Summary of the project's trials results:

Trial	Train	Validation	Test
NMF (benchmark)	4.0600	6.3200	6.4200
NMF + Log transform	4.0600	6.3800	6.5000
NMF + Log transform + Non-zero RMSE	1.9200	2.3800	2.3700
Auto-Encoder + Log Transform + Non-zero RMSE	**2.3869**	**2.0544**	**2.0751**
Auto-Encoder + Log Transform + Non-zero **Spatial** RMSE	3.3201 (Spatial)	2.1342 (RMSE) 3.1267 (Spatial)	2.1382 (RMSE)

Results Analysis

- There was a major improvement after I've changed the loss function to not include zero expressions during the training.
- NMF models tend to suffer from overfitting, on contrary, it seems like the AE models are underfitting and I can increase their complexity for example by adding layers or increase the latent dimention.
- The spatial loss function didn't help and got worst performance than the regular Non-Zero RMSE loss.
- After reconstruction trail 3 predictions, it is nice to see that the spots area separation is quite clear and not far from the original filtered data. Especially when comparing to trail 4 results.

Trial 3 reconstruction clustering



Trial 4 reconstruction clustering

Future Directions

There are various directions to continue and research this project.
While some other students worked on different ideas on how to incorporate the spatial information, there is still more work to progress in with this idea.

Possible directions:

- Hyperparameters tuning - Optimizing the results wasn't part of this project scope, so no automatic hyperparameter tuning has been done. All the trials was executed with the same hyperparamerers. There is no doubt that it can improve the results much better.
- Use published and well known gene's attributes as part of the input data of the model.
- Add edge ditection to build the neighborhood in an automatic way.
- Keep all available genes and deal with the problematic sparsity data.

References

[1] - CQB (center for qualitative biology) - https://sites.dartmouth.edu/cqb/2020/11/24/new-services-sample-multiplexing-spatial-transcriptomics-and-multiomics/
[2] - 10xGenomics datasets - https://www.10xgenomics.com/resources/datasets/adult-mouse-olfactory-bulb-1-standard-1
[3] - Vallejos CA, Risso D, Scialdone A, Dudoit S, Marioni JC. Normalizing single-cell RNA sequencing data: challenges and opportunities. Nature methods. 2017;14(6):565. Pmid:28504683
[4] - Prabhakaran S, Azizi E, Carr A, et al. Dirichlet process mixture model for correcting technical variation in single-cell gene expression data. In: International Conference on Machine Learning; 2016. p. 1070–1079.
[5] - Pierson E, Yau C. ZIFA: Dimensionality reduction for zero-inflated single-cell gene expression analysis. Genome biology. 2015;16(1):241. pmid:26527291
[6] - Risso D, Perraudeau F, Gribkova S, et al. A general and flexible method for signal extraction from single-cell RNA-seq data. Nature communications. 2018;9(1):284. Pmid:29348443
[7] - stLearn - a downstream analysis toolkit for Spatial Transcriptomics data - https://stlearn.readthedocs.io/en/latest/
[8] - Frigyesi A, Höglund M. Non-negative matrix factorization for the analysis of complex gene expression data: identification of clinically relevant tumor subtypes. Cancer Inform. 2008;6:275-92. doi: 10.4137/cin.s606. Epub 2008 May 29. PMID: 19259414; PMCID: PMC2623306.
[8] - Sedhain, Suvash, et al. "Autorec: Autoencoders meet collaborative filtering." Proceedings of the 24th international conference on World Wide Web. 2015.
[9] - Yusong Liu, Tongxin Wang, Ben Duggan, Michael Sharpnack, Kun Huang, Jie Zhang, Xiufen Ye, Travis S Johnson, SPCS: a spatial and pattern combined smoothing method for spatial transcriptomic expression, Briefings in Bioinformatics, Volume 23, Issue 3, May 2022, bbac116
[10] - My project GitHub repository - here