UNIVERSITY OF CALIFORNIA SAN DIEGO

Achieving Multi-scale Cell Morphology Clustering using Machine Learning

A Thesis submitted in partial satisfaction of the requirements
for the degree Master of Science

in

Biology

by

Je Seung An

Committee in charge:

Professor Eugene Yeo, Chair
Professor Amy Pasquinelli, Co-Chair
Professor Ivonne Gonzalez-Gamboa

2024

The Thesis of Je Seung An is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

TABLE OF CONTENTS

Thesis Approval Page iii

Table of Contents iv

List of Figures v

Acknowledgements vi

Abstract of the Thesis vii

Introduction 1

Materials and Methods 6

Results 29

Discussion 48

References 52

List of Figures

Figure 1: Autoencoder Workflow 5
Figure 2: Datasets Compiled for Initial Autoencoder Training 8
Figure 3: Datasets Used for Comparing Clustering by Gene Expression and Latent Space 10
Figure 4: Extracting Cells from Datasets 15
Figure 5: Preprocessing of Cell Crops for Autoencoder Input 17
Figure 6: Data Offloading and Ingestion of Preprocessed Data using DataLoader 19
Figure 7: Architecture of Autoencoder 21
Figure 8: Optimizing Model Performance during Training 23
Figure 9: Validating Autoencoder Generalization with Test Dataset 27
Figure 10: Utilizing Optimized Latent Space of Cell Clustering 28
Figure 11: Calculating Spatial Features of Hand-crafted Transcriptomics Dataset using Bento . 30
Figure 12: Spatial Feature Estimator Training 33
Figure 13: Training Autoencoder on Combined Dataset 35
Figure 14: Data Compression Analysis: Comparing Latent Space to Original Image Size 37
Figure 15: Xenium Human Pancreas Dataset Clustering by Gene Expression 40
Figure 16: Xenium Human Pancreas Clustering by Latent Space 41
Figure 17: Xenium Mouse Colon Dataset Clustering by Gene Expression 42
Figure 18: Xenium Mouse Colon Dataset Clustering by Latent Space 43
Figure 19: Observing the Impact of Small Gene Expression Variations on Cell Morphology 45
Figure 20: Exploring the Effect of Cell Morphology on Cell Localization 47

Acknowledgements

To begin, I want to first acknowledge my committee for their valuable time and insight. I would like to acknowledge Dr. Eugene Yeo especially and the Yeo Lab for being such a welcoming and supportive community during my past nearly three years at the lab. This lab has not only provided me with the resources necessary to grow but also the scientific mind necessary to make a difference in the industry. I am truly grateful to my mentors Noorsher Ahmed, who has guided me better than I could have asked throughout my time at the lab, and Dylan Lam, who first introduced me to this lab and the field of Bioinformatics. I would also like to acknowledge Clarence Mah, Avery Pong, and the rest of the SpaceForce team for their unwavering support and aid with this project. There were so many questions and obstacles that I had to overcome, and without the support of this team, I would not have had the success possible. Thank you for all the time and resources invested in growing my skills and for making this project possible for me.

I also want to extend my deepest gratitude to my friends and family who have been with me throughout this journey. To my sister and my parents, thank you for your constant support and for being my rock during my school years. Your sacrifices and endless love have been the foundation upon which I've been able to build my academic and professional endeavors. I am eternally grateful and will continue to strive for excellence, learning from my mistakes and pushing forward with new endeavors. Your belief in me has been my greatest motivation, and I hope to make you proud with all that I accomplish in the future.

ABSTRACT OF THE THESIS

Achieving Multi-scale Cell Morphology Clustering using Machine Learning

by

Je Seung An

Master of Science in Biology

University of California San Diego, 2024

Professor Eugene Yeo, Chair
Professor Amy Pasquinelli, Co-Chair

Understanding the heterogeneity of cell types and gene expression in complex tissues is crucial for advancing single cell genomics. Spatial transcriptomics enhances this understanding by adding spatial context, enabling a more comprehensive view of cellular function and organization. Additionally, the morphology of a cell is known to influence gene expression, and gene expression,

in turn, affects cell morphology, highlighting the intricate relationship between a cell's physical structure and its molecular activity.

Despite efforts to apply morphogenomics to single cell spatial data, existing methods face significant challenges in efficiently scaling to entire datasets. To address this limitation, I developed an autoencoder using PyTorch, a Python machine learning package, capable of replicating a 64x64 image of a cell mask. By utilizing the encoded latent layer, which is significantly smaller in dimension, this approach allows for multi-scale clustering of cell morphologies. I demonstrate Xenium datasets, Tissuenet datasets, and a pool of other datasets that were made publicly accessible.

To promote recreation and usability, the autoencoder is designed to integrate seamlessly with Bento, a computational toolkit developed in our lab. By being part of a diverse portfolio of software analyses tools, it maximizes the functionality and accessibility of the tool for further research and analysis.

Introduction

Understanding the heterogeneity of cell types and gene expression in complex biological tissues is fundamental to the field of single-cell genomics. This discipline has significantly advanced our knowledge by enabling researchers to analyze the genetic and phenotypic diversity within a single tissue, identifying unique cell populations and their functions. For instance, single-cell sequencing technologies such as scRNA-seq[1] have been pivotal in profiling individual cell transcriptomes, allowing for detailed insights into cellular differences and heterogeneity[2].

However, while single-cell genomics provides detailed insights into cellular differences, it lacks spatial information, which is crucial for understanding how cells interact within their native environments[3]. Spatial transcriptomics builds on the advancements of single-cell genomics by incorporating spatial context into the analysis. This approach enables the mapping of gene expression within the spatial architecture of tissues, offering a more comprehensive view of cellular function and organization. For example, spatial transcriptomics techniques have been essential in revealing how cells influence each other and their microenvironment, providing valuable insights into tissue functionality and pathology[4]

The morphology of a cell is another critical factor that influences gene expression and vice versa. Cellular morphology and gene expression are interdependent, with changes in one often leading to alterations in the other[5-7]. This dynamic relationship underscores the importance of considering both aspects when studying cellular behavior. Morphogenomics, a term my mentor and I use for integrating morphological and genomic data, aims to make sense of these complex interactions[8]. However, applying morphogenomics to single-cell spatial data presents significant

challenges, particularly in scaling analyses to whole datasets[2]. Recent advancements in spatial transcriptomics methods, such as MERFISH[9], seqFISH+[10], and Ex-Seq[11], have enabled RNA localization measurements close to the transcriptome scale, while methods like Xenium[12] have reached tissue-scale identification of cells, highlighting the scale and complexity of data involved[4].

To address these limitations, I developed an autoencoder with hopes to efficiently reduce the dimensionality of cell image data, thereby facilitating scalable and accurate clustering of cell morphologies. An autoencoder is a type of artificial neural network used to learn efficient encodings of unlabeled data. The purpose of using an autoencoder in this context is to capture the essential features of cell morphology within a compact, encoded latent space, which significantly reduces the dimensionality of the original data. By doing so, the autoencoder can transform high-dimensional cell images into a lower-dimensional representation that retains critical morphological information while discarding redundant details. This compressed representation enables more efficient clustering of cell morphologies, which is crucial for analyzing large-scale single-cell datasets without overwhelming computational resources.

The autoencoder I developed is capable of replicating a 64x64 image of a cell mask. This autoencoder utilizes the encoded latent layer, significantly smaller in dimension, to enable multi-scale clustering of cell morphologies (Figure 1). By reducing the dimensionality of the data, the autoencoder facilitates efficient and scalable clustering, overcoming the challenges faced by current methods of studying morphogenomics[12], which are severely limited by the sheer memory consumption of clustering full image arrays or require handcrafted supervised features detailing different spatial measurements of each cell. The encoded latent layer captures essential features of

cell morphology while discarding redundant information, enhancing the clustering process and improving the accuracy of cell classification. This approach not only allows for the analysis of individual cells within a population but also supports the examination of cell subtypes and their unique morphological characteristics. Additionally, the autoencoder's design enables it to handle large and diverse datasets, making it a robust tool for comprehensive morphogenomic studies. By leveraging advanced machine learning techniques and a well-structured neural network architecture, the autoencoder provides a powerful solution to the inherent scalability issues found in dataset-wide clustering by cell morphologies.

The autoencoder was built using PyTorch[14], a widely-used open-source machine learning library known for its flexibility and dynamic computation graph, which facilitates the development and training of neural networks. This model was trained on several diverse datasets. This extensive training ensures the model's robustness and accuracy, achieving a low mean squared error of 0.008 with an entire spatial dataset. Furthermore, the autoencoder is designed to integrate seamlessly with Bento[15], a toolkit developed in our lab, maximizing the usability and accessibility of the tool for further research and analysis. This integration allows researchers to leverage the autoencoder's capabilities within a comprehensive analytical framework, enhancing the study of cellular morphology and gene expression at scale.

While developing the autoencoder, I came to the realization that the performance of the model was directly influenced by two key factors. The first factor, which is more intuitive, involve changes to the model itself, such as adjusting the number of layers, tweaking the learning rate, or

modifying the architecture. These modifications, while expected to impact the model's performance, revealed only part of the picture.

What was particularly surprising was the critical importance of data preprocessing in achieving optimal performance. It became evident to me that meticulous preprocessing steps were essential for enhancing the accuracy of the model. This included applying various transformations to the input data, such as rotating the cell masks along their major axis to standardize orientation, cropping the images to maximize the utilization of image space to standardize size, and normalizing the blur to ensure consistent image quality. These preprocessing techniques played a crucial role in preparing the data for the autoencoder, allowing it to learn more effectively and produce more accurate and reliable results.

By implementing these preprocessing steps, I was able to significantly improve the performance of the autoencoder, demonstrating that the quality and preparation of input data are just as important as the model architecture itself. This holistic approach to model development allowed me to understand the necessity of combining robust preprocessing with thoughtful model design to achieve high levels of accuracy and efficiency in morphogenomic analyses. The goal of my analysis tool is to open the pathway for cell morphology insights and to be built upon the current widely-used method of single cell analysis, gene expression analysis. The hope of this is to achieve more powerful clustering and separation of cell types, cell states, and perhaps even disease types.

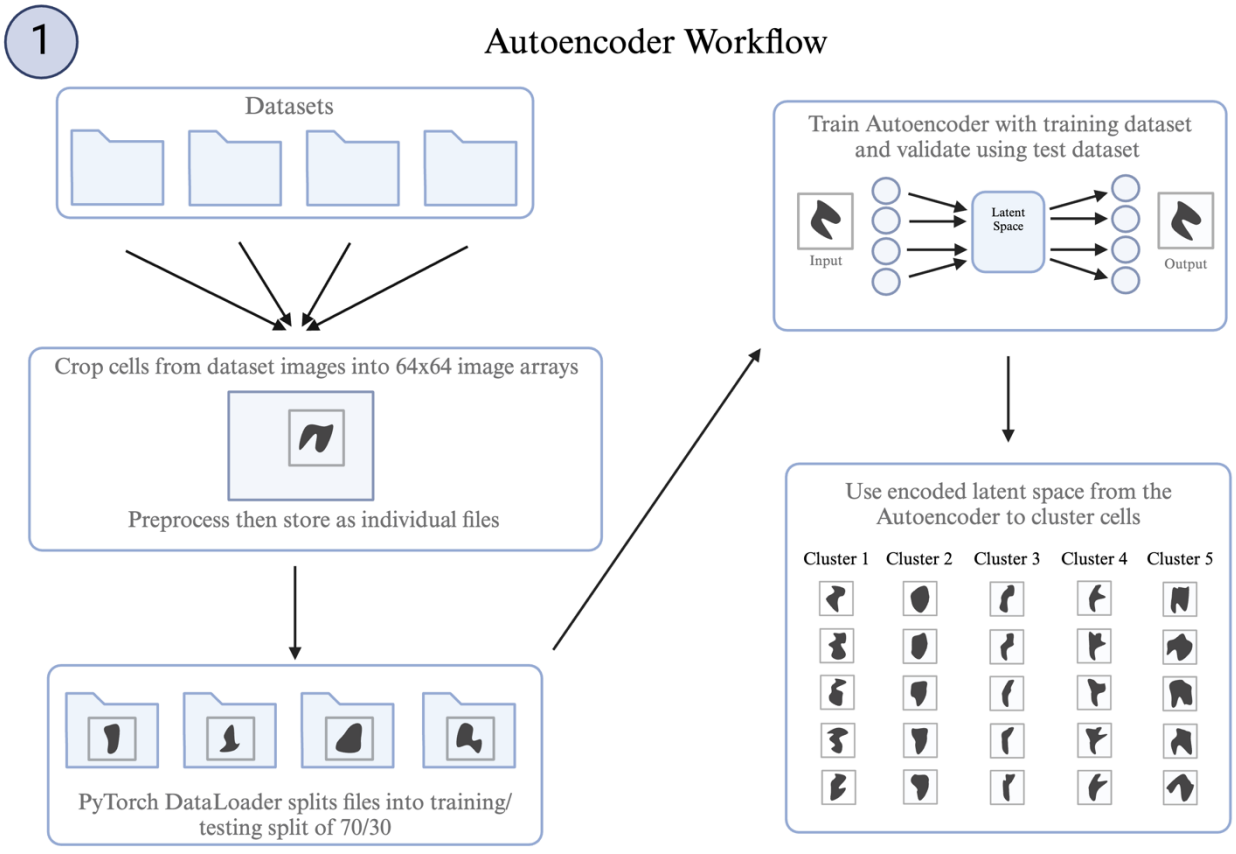

Figure 1: Autoencoder Workflow

Materials and Methods

Datasets Used for Initial Autoencoder Training

To get an initial proof of concept, I ran through a compiled list of four publicly available datasets. My goal was to enable the autoencoder to cluster all the cells in a complete spatial transcriptomics dataset. For this, I used four datasets, achieving a total number of 52,708 cells. This initial data collection and research project were given to me by my mentor to help me learn the practice of ingesting spatial datasets. It also highlighted the challenge of limited availability of spatial datasets with ground truth segmentations.

EVICAN

The EVICAN dataset comprises cell microscopy images taken from the open research data repository, Edmond[16]. This dataset contains expert-annotated grayscale images of 30 different cell lines, captured using multiple microscopes, contrasts, and magnifications. The dataset consists of 4,600 images, from which I was able to extract 25,275 cells. These images provided a diverse set of cell types and imaging conditions, crucial for training a robust autoencoder.

MOESM1

The MOESM1 dataset was accessed publicly through the open access download link provided on BMC Bioinformatics[17]. It includes images of five cellular assays in 96-well

microplates, acquired using GE's IN Cell Analyzer[18] systems. This dataset contains various cell lines such as Hela, HEPG2, fibroblasts, and U20S cells. It is an expert-annotated dataset where CellProfiler[19] was initially used to generate segmentation labels, which were then refined by hand. From this dataset, I was able to extract 6,492 cells. This dataset's well-annotated and diverse cell types were valuable for training and validating the autoencoder.

MP6843

The MP6843 dataset was accessed from the open-source cell image database, Cell Image Library[20]. This dataset includes cultured neuroblastoma cells stained with phalloidin and DAPI. The ground truth dataset was manually generated using Adobe Photoshop[21] and stored as grayscale tiff files. Originally created for benchmarking segmentation algorithms, this dataset includes cells imaged on a Zeiss[22] epifluorescence microscope. From this dataset, I extracted 3,108 cells. The high-quality, manually annotated images provided a robust training set for the autoencoder.

Tissuenetv1.1

Tissuenetv1.1 is a comprehensive segmentation dataset featuring six different tissue types: breast tissue, gastrointestinal tissue, immune cells, lung, pancreas, and skin tissue. Images were captured using six different platforms: Codex[23], CyCif[24], IMC[25], Mibi[26], Mxif[27], and Vectra[28]. Given the dataset's variety in tissue types, I believed it would enhance the autoencoder's generalizability. From this dataset, I extracted 17,833 cells. The diversity and comprehensiveness

of this dataset were instrumental in training an autoencoder capable of generalizing across various tissue types and imaging conditions.

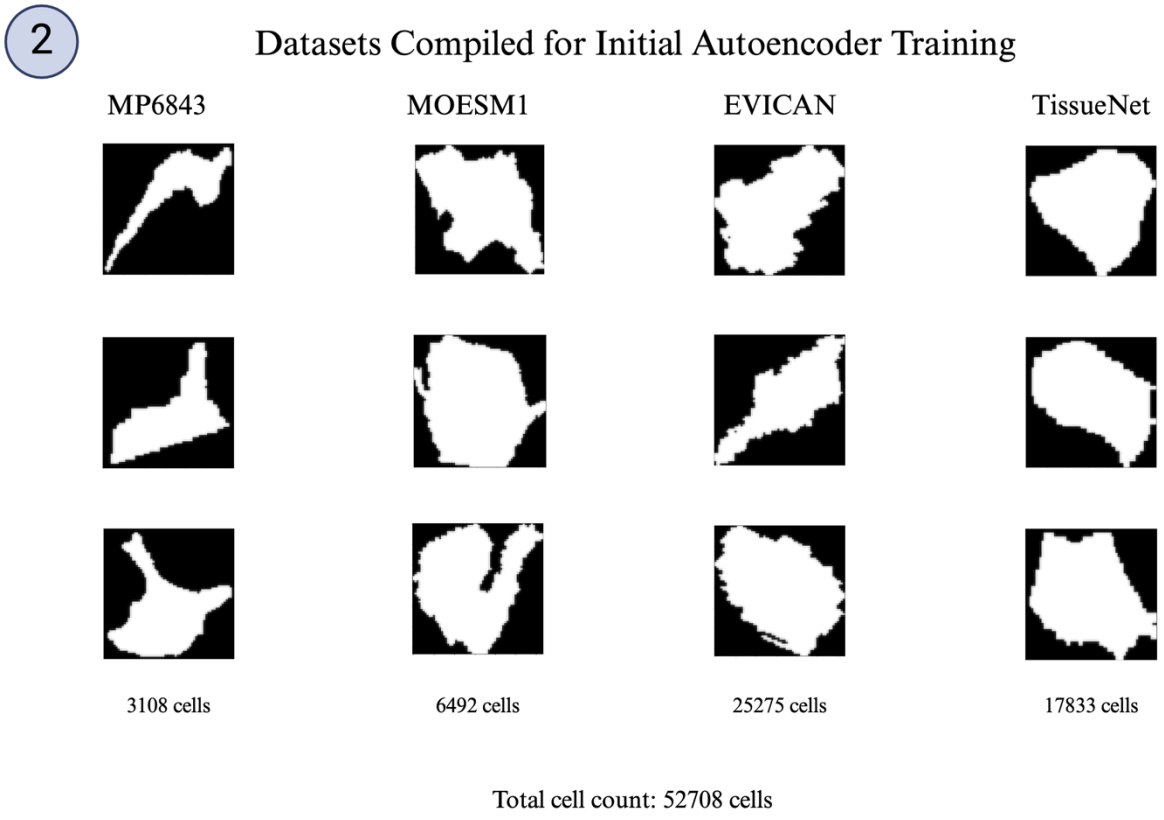

Figure 2: Datasets Compiled for Initial Autoencoder Training

Datasets Used to Compare Clustering by Gene Expression and Latent Space

10X Xenium Human Pancreas

This dataset involves fresh frozen human pancreas tissue that was spatially analyzed using Xenium, a modern spatial transcriptomics platform enabling high throughput imaging of RNA. The complete spatial experiment pipeline provided by Xenium includes gene expression data from a 377-gene panel and cell segmentations created with nuclear expansion from DAPI-based staining. After preprocessing, I extracted 117,315 cells from this dataset. This rich dataset allowed for a robust comparison of clustering based on gene expression versus morphology using latent space.

10X Xenium Mouse Colon

The 10X Xenium Mouse Colon dataset underwent the same fresh frozen tissue experimental pipeline as the Human Pancreas dataset. It includes gene expression data from a 379-gene panel, designed using reference data from CZ Biohub San Francisco[29] and The Tabula Muris Consortium[30]. This dataset has been validated in the whole mouse pup and in nine separate tissues in adults, including the brain, colon, heart, kidney, liver, lung, skin, spleen, and thymus. After preprocessing, I extracted 182,789 cells from this dataset. The extensive validation and diverse tissue representation in this dataset provided a robust framework for comparing clustering techniques.

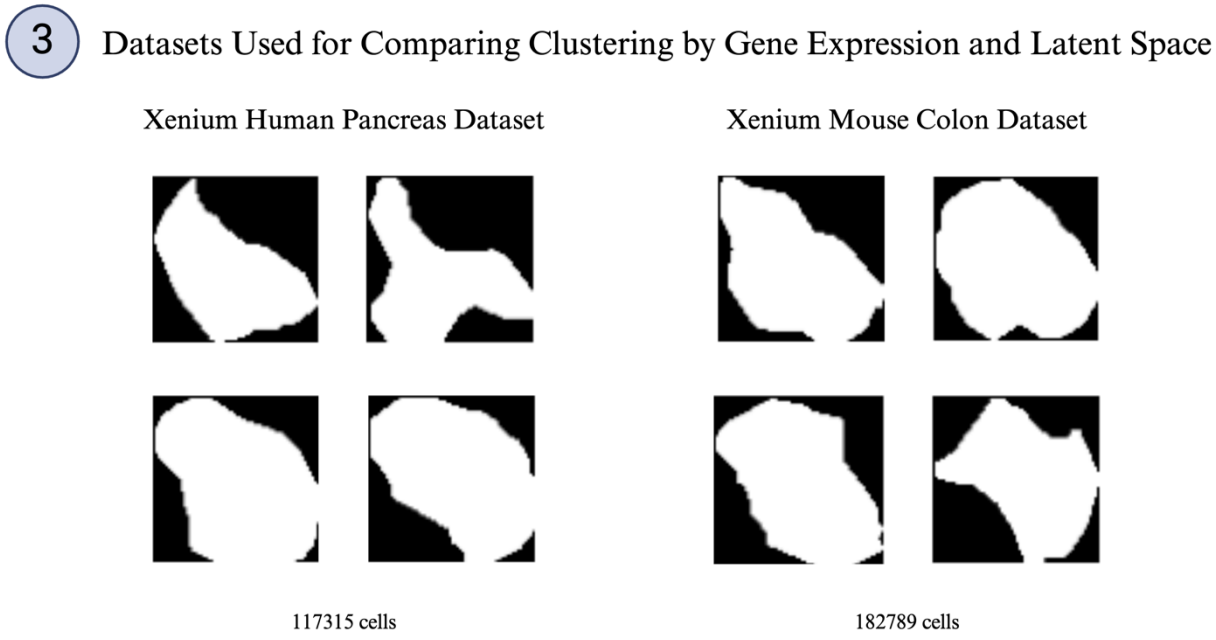

Figure 3: Datasets Used for Comparing Clustering by Gene Expression and Latent Space

These datasets collectively provided a comprehensive training and validation set for the autoencoder, facilitating the exploration of clustering by morphology and gene expression in a variety of biological contexts. The integration of diverse and high-quality datasets was crucial for developing a versatile and effective tool for morphogenomics.

The importance of high-quality and diverse datasets in the generation of machine learning models cannot be overstated. In complex fields like single-cell genomics and spatial transcriptomics, the value of a machine learning model is fundamentally tied to the quality and diversity of the data it is trained on. Well processed datasets ensure that the models are trained on accurate and reliable data, which directly impacts their performance and robustness. Diverse datasets, on the other hand, allow models to generalize better by exposing them to a wide range of scenarios and variations, thus capturing the nuances of different biological contexts. For instance, in the development of my autoencoder, the datasets provided a comprehensive training and validation set, facilitating the exploration of clustering by morphology and gene expression across various biological samples. This diversity was essential for developing a versatile and effective tool for morphogenomics[12, 31].

The EVICAN dataset, with its expert-annotated grayscale images of various cell lines, provided a rich source of data for understanding different cellular shapes and textures under varied imaging conditions. The MOESM1 dataset contributed additional diversity with its images of cellular assays from multiple cell lines and its expert-annotated segmentation masks, ensuring that the autoencoder could learn from accurate and meticulously labeled data. The MP6843 dataset's

manually annotated neuroblastoma cells added another layer of specificity, allowing the model to handle high-quality, manually generated ground truth data.

Tissuenetv1.1's inclusion of multiple tissue types and imaging platforms introduced an even greater level of complexity and variety. This variety was crucial for the autoencoder's ability to generalize across different types of biological tissues and imaging conditions, enhancing its robustness and applicability in real-world scenarios. The diversity of imaging platforms also exposed the autoencoder to different types of imaging artifacts and noise, further improving its ability to handle varied real-world data.

For the comparison of clustering by gene expression and latent space, the 10X Xenium datasets provided comprehensive spatial transcriptomics data, allowing for a detailed analysis of how gene expression patterns correlate with cellular morphology. These datasets, with their extensive gene panels and validated segmentation data, were indispensable for testing and validating the autoencoder's performance in a practical, application-oriented context.

The integration of such varied datasets ensured that the autoencoder was not only well-trained on a broad spectrum of cellular images but also capable of adapting to new and unseen data. This adaptability is a key feature of any effective machine learning model, particularly in the rapidly evolving field of single-cell genomics, where new data types and imaging techniques are continually being developed.

In summary, the availability of high-quality, diverse datasets was critical for the success of the autoencoder. It allowed the model to learn the complex relationships between cellular morphology and gene expression, facilitated robust validation of its clustering capabilities, and ensured its applicability across a range of biological contexts. This foundation of comprehensive data is what makes the autoencoder a powerful tool for advancing our understanding of cellular behavior and function in complex biological tissues, paving the way for future research and discoveries in morphogenomics.

Extracting Cells from Each Dataset (Figure 4)

The dataset files are stored in various formats, including .npz, .jpg, and .tif. To handle these files, I utilized the image reading function from Sci-kit image[32]. This function efficiently reads the files and converts them into image arrays, which are suitable for further processing. The ability to work with multiple file formats was crucial for accommodating the diverse datasets used in this study.

Once the images were read, I needed to assign unique values to the connected regions within each image array. This was achieved using Skimage's measure label function, which identifies connected pixels that are direct neighbors and have the same value. By assigning unique labels to these connected regions, I could effectively distinguish between individual cells within the image.

With the cell labels established in the dataset, the next step was to extract each cell from the image array. Using NumPy's[33] np.where function, I iterated through each unique cell index. This function allowed me to create a new image array where only the pixels corresponding to the current cell index were non-zero. This process of isolating individual cells was crucial for ensuring that each cell could be accurately preprocessed and analyzed independently.

The Xenium datasets presented an additional challenge, as the cell boundaries were stored in a CSV file. This file contained the coordinates of each cell boundary, which needed to be accessed and processed. I used Pandas'[34] CSV reading function to read this file, extracting the necessary boundary coordinates. Once the boundaries were read, the cells were separated into image arrays in a manner similar to the previous method. This approach ensured that each cell's boundary could be accurately isolated and prepared for further analysis.

Overall, these steps allowed me to efficiently process and prepare the dataset for subsequent analysis. By utilizing a combination of Sci-kit image and NumPy functions primarily, I was able to handle the diverse file formats and accurately isolate individual cells. This meticulous preprocessing was essential for ensuring the quality and reliability of the data used in this study.

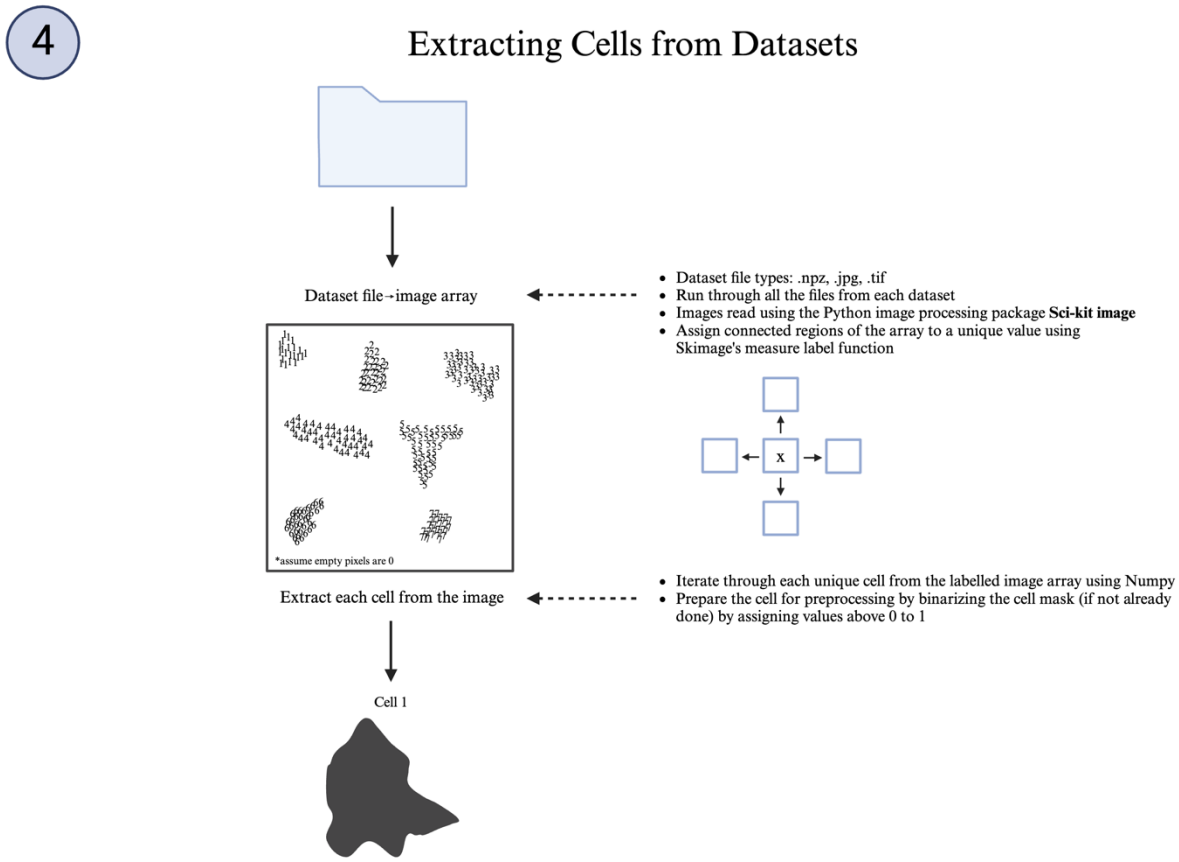

Figure 4: Extracting Cells from Datasets

Preprocessing of Cell Crops for Autoencoder Input (Figure 5)

A major preprocessing step for getting optimal autoencoder performance is binarizing the cell mask. The reason for this is because each dataset has a different sized image holding the cell masks and different imaging techniques which will in turn show a variation in the blur present around each cell mask. I achieved the binarization of the cell mask by using NumPy's np.where function to set any pixels above 0 to 1 in the image array. Now that I have normalized the blur from each dataset, I proceed to a more orientational preprocessing step. Cells are in many different positions varying from laying horizontal or vertical to diagonal, however, the aspect of rotation must not affect the later clustering by cell morphologies. This is because the clustering algorithm will not be clustering effectively since a cell's rotation will heavily influence where it is clustered. An example would be if a cell that is imaged as horizontal may belong to the same cluster as one that is imaged vertically but without normalization these edge cases cannot be caught. By normalizing this, I was able to solely isolate the clustering to focus on the morphology of the cell and its unique, subtle characteristics. After rotating the cell mask, I went on to resize the image to the autoencoder input size of 64x64. The size of 64x64 was chosen as it seemed sufficient to capture enough minute morphological features of a cell.

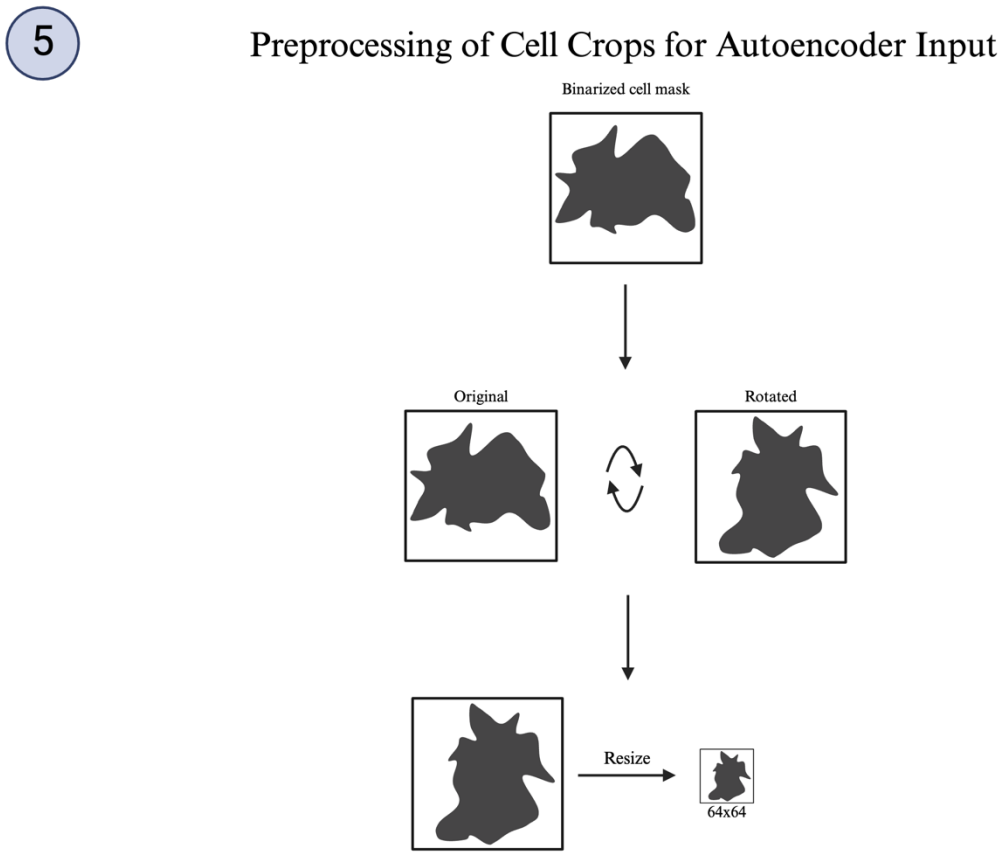

Figure 5: Preprocessing of Cell Crops for Autoencoder Input

Data Offloading and Ingestion of Preprocessed Data Using PyTorch DataLoader (Figure 6)

The preprocessed cells were offloaded into a directory as .tiff files, which were then split into training and test directories, with 70% of the data allocated to training and 30% to testing. This strategy was necessary due to the memory limitations of my workstation, which has 32 GB of RAM, insufficient to hold the entire dataset's images in memory simultaneously. To handle this efficiently, I utilized PyTorch's DataLoader, a tool that facilitates the streamlined processing of images from directories.

The DataLoader in PyTorch allows for efficient batch processing of data, making it ideal for handling large datasets. By setting the batch size, the DataLoader processes multiple images at a time, balancing memory usage and computational speed. Additionally, the DataLoader uses 4 workers to load the data in parallel, significantly reducing the time spent on data loading and thus speeding up the training process. By enabling shuffling, the DataLoader ensures that the data is randomly shuffled at each epoch, preventing the model from learning any unintended patterns in the data order and enhancing its generalization capabilities. This approach not only overcomes the memory limitations of my workstation but also ensures efficient and effective data handling during model training. The use of PyTorch's DataLoader allows for seamless integration of large datasets into the training pipeline, optimizing both memory usage and processing time.

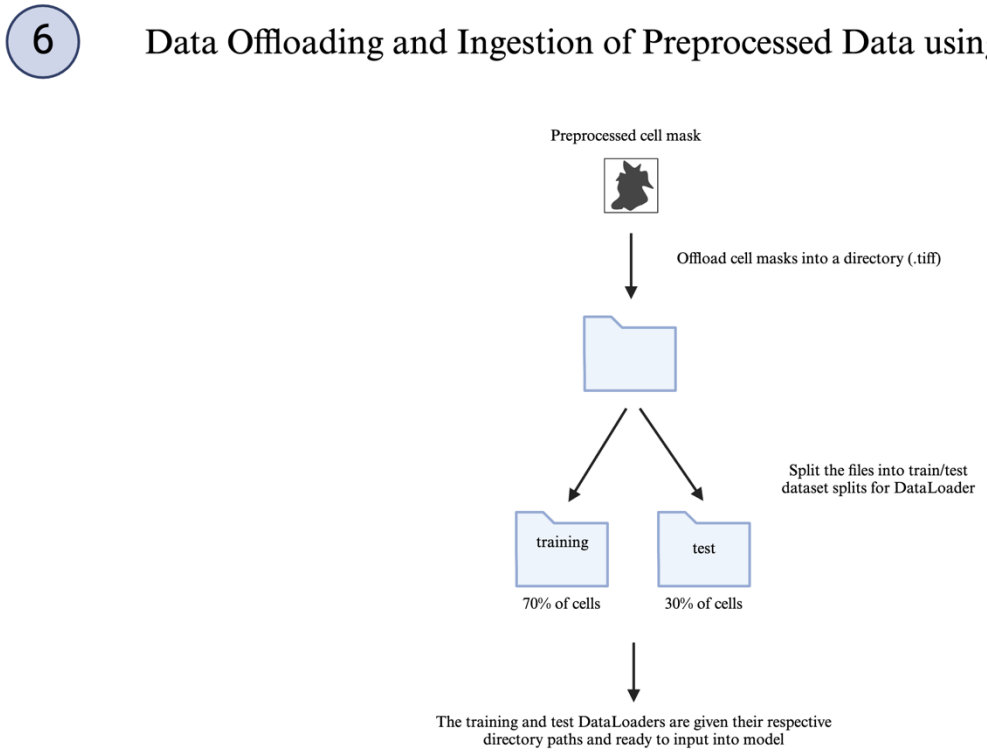

Figure 6: Data Offloading and Ingestion of Preprocessed Data using DataLoader

Autoencoder

The architecture of the autoencoder is described by its layers (Figure 7). The input to the autoencoder is a 64x64 preprocessed cell image, which is passed through the encoder. The encoder consists of four convolutional layers, implemented using PyTorch's nn.Conv2d module. These convolutional layers sequentially reduce the spatial dimensions of the image while increasing the depth, capturing increasingly abstract features at each layer. Each convolutional layer is followed by a Rectified Linear Unit (ReLU) activation function, which introduces non-linearity into the model and helps it learn complex patterns in the data.

After the encoder, the compressed representation of the image, also known as the latent space, is passed to the decoder. The decoder is composed of four transposed convolutional layers, implemented using PyTorch's nn.ConvTranspose2d module. These layers perform the inverse operation of the convolutional layers, gradually reconstructing the spatial dimensions of the image while decreasing the depth. Similar to the encoder, each transposed convolutional layer in the decoder is followed by a ReLU activation function, except for the final layer. The last layer of the decoder uses a Sigmoid activation function to ensure that the outputted image has pixel values constrained between 0 and 1, making it suitable for comparison with the original input image.

Overall, the autoencoder is designed to efficiently compress the input image into a compact representation and then accurately reconstruct it, preserving essential features while reducing

dimensionality. This architecture enables effective learning and generalization, which are critical for various downstream tasks such as clustering and anomaly detection.

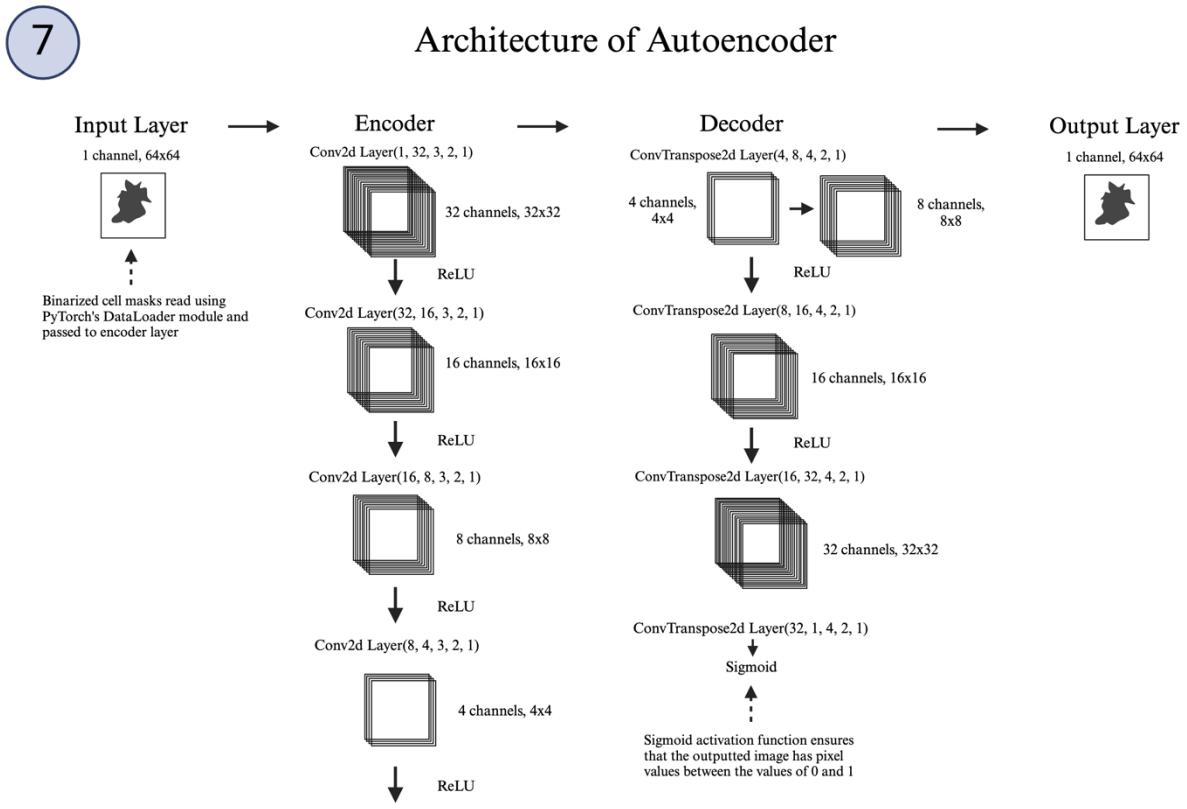

Figure 7: Architecture of Autoencoder

Optimizing Model Performance during Training (Figure 8)

To achieve effective learning and generalization, various tools and techniques are employed to optimize the model for better performance. During the training process, the Mean Squared Error (MSE) loss function was utilized. This choice is driven by its common application in image reconstruction tasks, where it effectively measures the difference between the original and reconstructed images. Additionally, the Adam optimizer was selected due to its popularity and proven performance in training autoencoders. The Adam optimizer efficiently updates the model's weights, facilitating faster convergence and better handling of sparse gradients. The learning rate for the optimizer was set to 2e-4, balancing the speed of convergence with the stability of the training process.

To ensure that the model achieves generalizability, weight decay was incorporated. Weight decay helps prevent overfitting by penalizing large weights, thereby encouraging the model to learn simpler and more general patterns. In this training setup, the weight decay value was kept at 1e-4, forming a balance between regularization and learning capacity. These choices collectively contribute to the robust performance and generalization capability of the autoencoder.

During the training process, I validated that the model was indeed learning to reproduce the images accurately and wasn't learning nonsense patterns. This sanity check was conducted using Matplotlib's[35] Pyplot module, which allows for the visualization of input images alongside their predicted reconstructions. By plotting these images during training, I ensured that the model was learning correctly and making meaningful progress. This method of checking provides crucial

validation during the training session, which is necessary before moving on to actual validation with the test dataset. This step ensures that the model is not only performing well on the training data but is also capable of generalizing to unseen data, confirming its effectiveness and reliability.

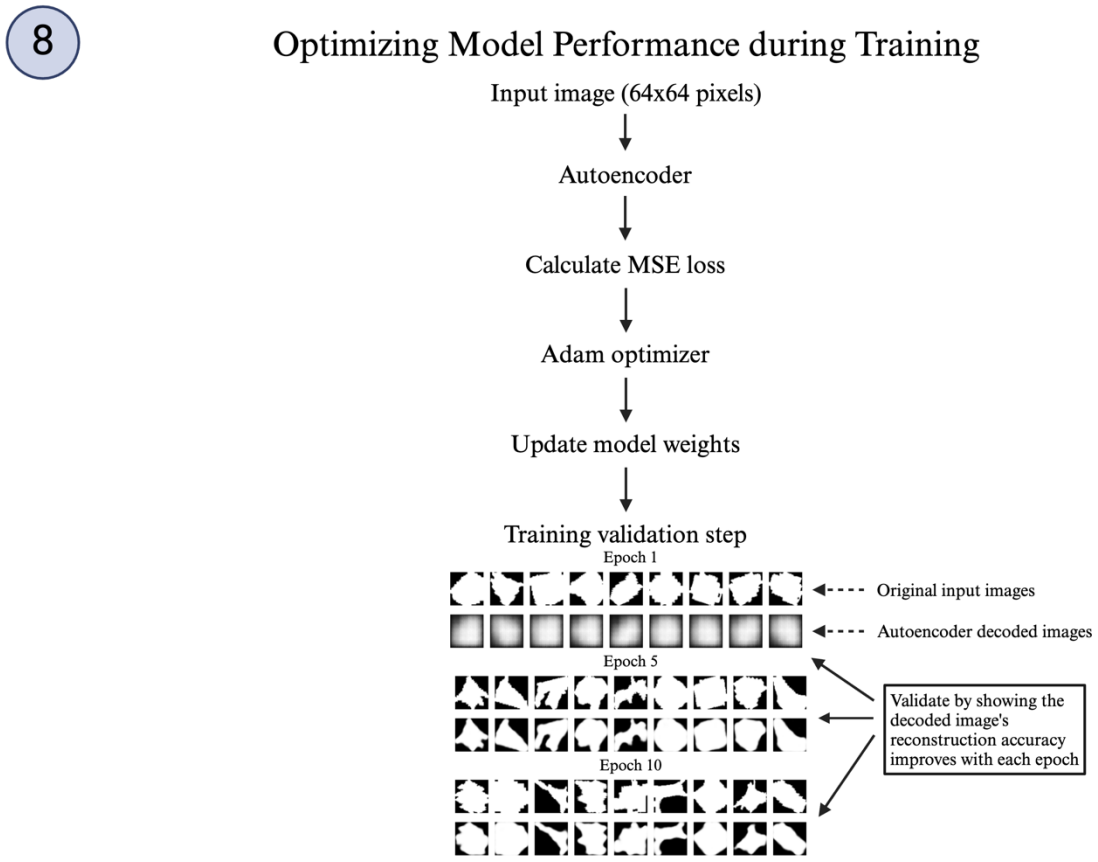

Figure 8: Optimizing Model Performance during Training

Validating Autoencoder Performance with Test Dataset and Clustering

To confirm that the autoencoder is generalizable and not overfitted to the training dataset, I have been validating the model's performance using the test dataset (Figure 9). This validation process replicates the same steps used during training but applies them to the test dataset. By comparing the Mean Squared Error (MSE) values and the outputted images to the original images, I can assess whether the autoencoder is overfitting to the training data or effectively generalizing to new, unseen data.

During this validation, I have been using Matplotlib's Pyplot module to visualize the input images alongside their predicted reconstructions from the test dataset. This comparison helps to confirm that the autoencoder is accurately reconstructing the input images from the test set, thereby demonstrating its ability to generalize beyond the training data. This step is crucial in verifying the robustness and generalizability of the autoencoder, ensuring it performs well on a different set of data and not just on the data it was trained on.

Once this initial validation had been checked, I moved on to utilizing the optimized latent space of the autoencoder for cell clustering (Figure 10). The goal of this clustering step was to validate my concept of cell morphology clustering in reduced dimensions. The clustering algorithm used for this task was k-means due to its simplicity and effectiveness in identifying separations in clusters based on shape features.

K-means clustering works by partitioning the dataset into a predetermined number of clusters (k). It starts by initializing k cluster centroids randomly. Each data point in the latent space is then assigned to the nearest centroid based on the Euclidean distance. After the initial assignment, the centroids are recalculated as the mean of all data points assigned to each cluster. This process of assignment and centroid recalculation is repeated iteratively until the centroids no longer change significantly or the maximum number of iterations is reached.

By applying k-means clustering to the latent space representations of the cell images, I could observe how well the autoencoder's compressed features captured the variations in cell morphology. The clusters formed by k-means provided a way to evaluate whether the latent space contained sufficient information to distinguish between different cell shapes and morphologies. If the clusters were well-defined and distinct, it suggested that the latent space was effectively capturing the necessary features for cell morphology differentiation. Conversely, if the clusters were not well-separated, it indicated that the latent space lacked sufficient information, prompting further adjustments to the autoencoder architecture or training process.

To further validate the clustering results, I visualized the clustered data points in the latent space using UMAP (Uniform Manifold Approximation and Projection). UMAP is particularly effective for visualizing high-dimensional data by preserving both local and global structure in the data. This helps in understanding the structure of the latent space and the separability of the clusters. Additionally, I tried PCA (Principal Component Analysis) and t-SNE (t-Distributed Stochastic Neighbor Embedding) for visualization. PCA helps in understanding the variance

captured by the principal components, while t-SNE is useful for visualizing local similarities in high-dimensional data.

Comparing the visualizations from UMAP, PCA, and t-SNE provided a comprehensive view of how well the latent space representations captured the essential features for clustering. The UMAP latent space representation was by far the best in capturing different relationships as the clusters appeared much more usable than the other two. I also analyzed the clustering effectiveness by visually comparing the intra-cluster distances (a measure of cluster compactness) and inter-cluster distances (a measure of cluster separation). All of these analyses were built using Scanpy[36], a powerful Python package for users to run single cell analyses. Through this analysis and plotting cells per cluster, I was able to determine whether the latent space provided by the autoencoder was optimized for clustering cell morphologies, ensuring that the reduced-dimensionality representations retained the critical features necessary for distinguishing between different cell shapes.

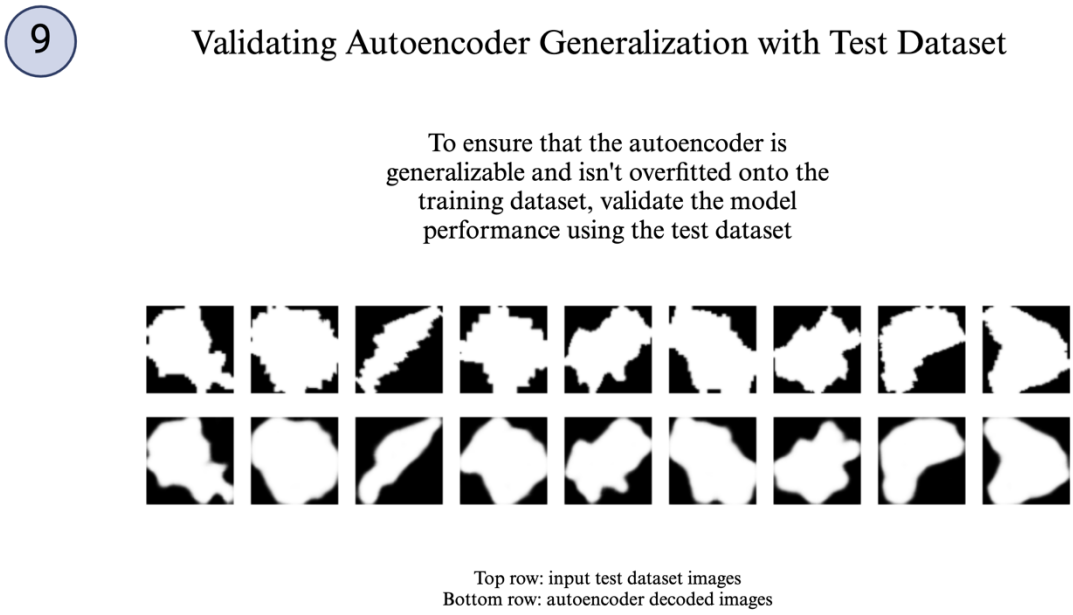

Figure 9: Validating Autoencoder Generalization with Test Dataset

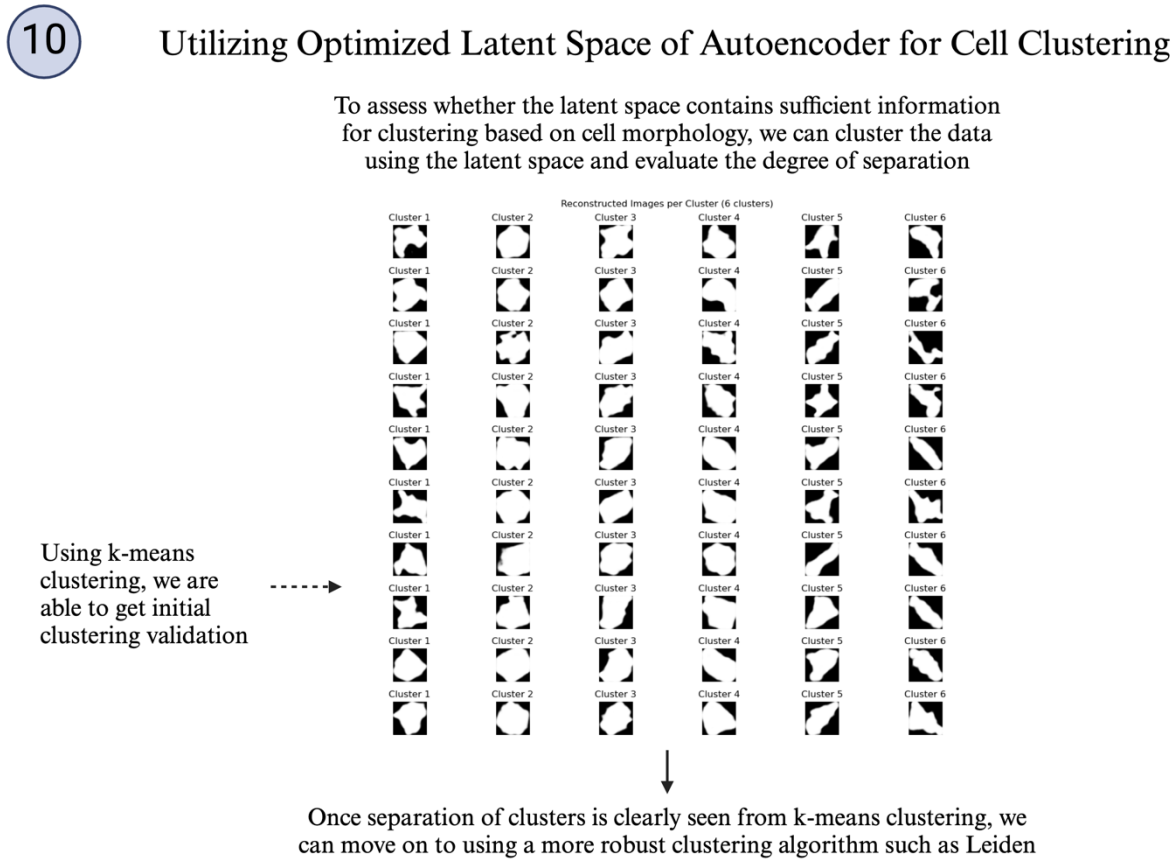

Figure 10: Utilizing Optimized Latent Space of Cell Clustering

Results

Background: Attempting to Calculate Spatial Features of Hand-crafted Transcriptomics Dataset using Bento (Figure 11)

Before delving into the development of the autoencoder, my focus was on creating a machine learning tool capable of accurately calculating various spatial features from a cell in a transcriptomics dataset, including its nucleus and transcripts, leveraging the capabilities of Bento. Several preparatory steps were necessary before developing the machine learning model.

Initially, I had to generate a simulated dataset of cells due to the limited availability of spatial transcriptomics datasets for public access. This involved creating cell and nucleus masks. To simulate these masks, I generated 10-30 random points within a 64x64 image array using NumPy. The cell mask was then created by encapsulating these points in a Shapely[37] Polygon module object. Similarly, the nucleus masks were generated by applying the same process within the cell boundaries, using only 5-10 random points.

Subsequently, I simulated transcripts for the dataset using simFISH[38]-based functions to create patterns of gene localization. Four different transcript localization patterns were simulated: cell edge, perinuclear, intranuclear, and random. These pattern generation functions and the probability maps required to obtain the patterns were made available to me by mentor. The patterns were generated by inputting points randomly into a probability map that is unique for each pattern. Each cell was generated to contain 20 genes (5 genes per pattern), with each gene represented in

its own image file. This resulted in 20 unique transcript patterns (genes) per cell, each pattern (gene) with a random number of transcripts ranging from 5 to 100.

To calculate the spatial features for the dataset, I utilized Bento's tool package. By having these spatial features calculated prior to training the machine learning model, I established the ground truth necessary for training. The spatial feature vector calculated by Bento consisted of 19 features, normalized to a range between 0 and 1 to facilitate better learning by the model. This comprehensive approach ensured that the machine learning model was provided with accurate and reliable ground truth spatial features, essential for its effective training and subsequent performance.

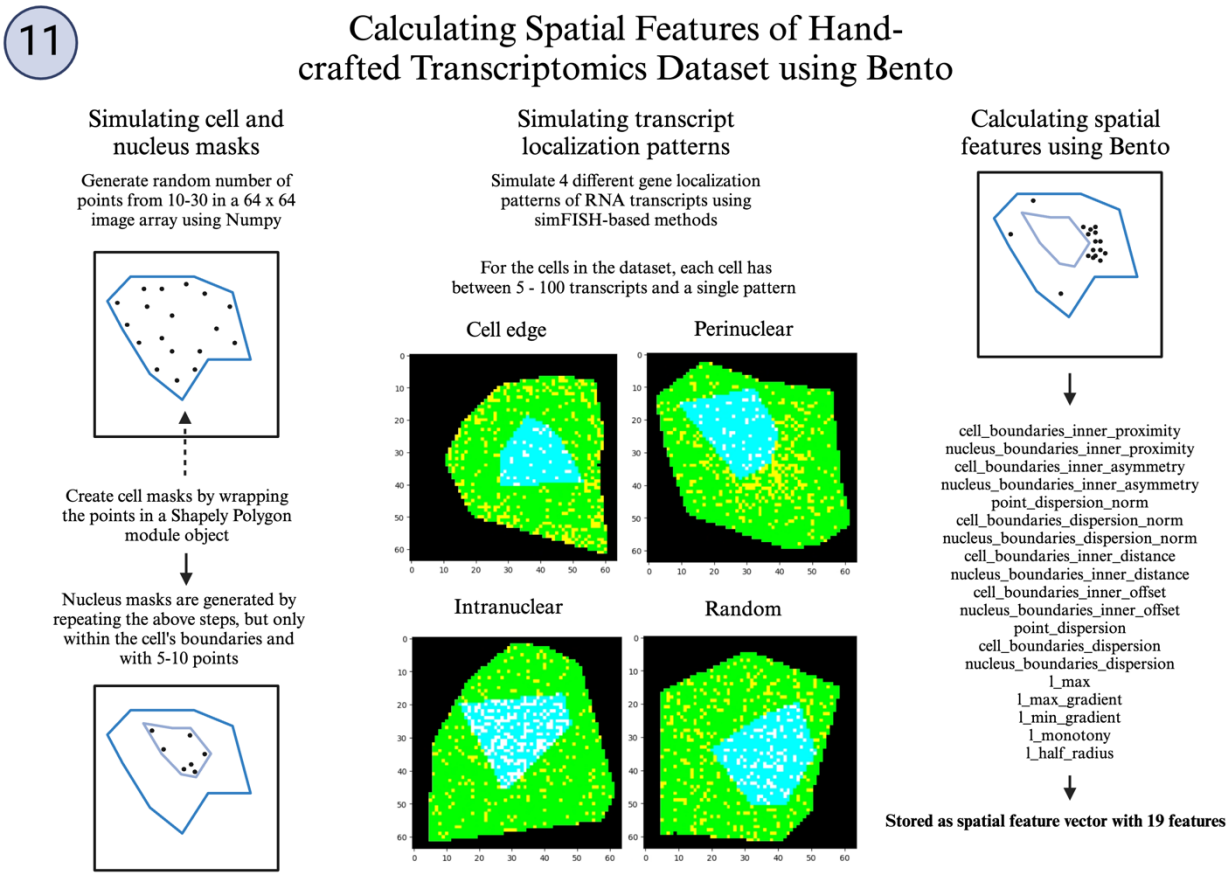

Figure 11: Calculating Spatial Features of Hand-crafted Transcriptomics Dataset using Bento

Spatial Feature Estimator Training

The model I employed for training the spatial feature estimator was the ResNet-50, renowned for its deep convolutional architecture. ResNet[39], short for Residual Network, is a type of convolutional neural network (CNN) that excels in learning intricate patterns from complex data. Its distinctive feature lies in the use of residual connections, or shortcut connections, which facilitate the training of very deep networks by alleviating the vanishing gradient problem. In ResNet, each block contains several convolutional layers followed by a shortcut connection that adds the original input to the output of the convolutional layers. This enables the network to learn residual functions, making it easier to train deeper architectures without suffering from degradation in performance.

Conversely, U-Net[40] is another popular CNN architecture specifically designed for biomedical image segmentation tasks, such as cell and nucleus segmentation. It is characterized by a U-shaped architecture, consisting of a contracting path followed by an expanding path. The contracting path, also known as the encoder, comprises convolutional and pooling layers that progressively downsample the input image to extract high-level features. The expanding path, or decoder, consists of convolutional and upsampling layers that upsample the features and concatenate them with the corresponding features from the contracting path to recover spatial information lost during downsampling. This skip connection mechanism enables U-Net to capture both local and global context, making it highly effective for tasks requiring precise segmentation.

In my experiments, despite the popularity of U-Net for biomedical image segmentation, it did not perform as well as expected when tested for spatial feature estimation with an MSE of 1.532. The ResNet-50, on the other hand, achieved relatively better results with an MSE of 0.529. However, this MSE was influenced by the model's struggle to learn certain specific features, including cell inner offset, cell inner proximity, nucleus inner offset, and nucleus boundaries inner distance. These unlearned features contributed to a skewing of the total MSE, indicating areas where the model encountered challenges in feature extraction.

Despite its limitations in learning some specific features, the ResNet-50 successfully learned 13 spatial features, compared to the 6 it struggled with. Notable learned spatial features included cell dispersion norm, nucleus inner symmetry, cell asymmetry, and point dispersion norm. The analysis of these features provided valuable insights into the model's strengths and weaknesses in capturing spatial characteristics from the input images.

Given the suboptimal performance of the ResNet-50 in fully capturing all spatial features, I explored alternative approaches to developing a novel computational tool for spatial feature estimation. This transition was necessary to address the limitations encountered with the ResNet-50 architecture and to further enhance the accuracy and robustness of spatial feature estimation in biomedical image analysis.

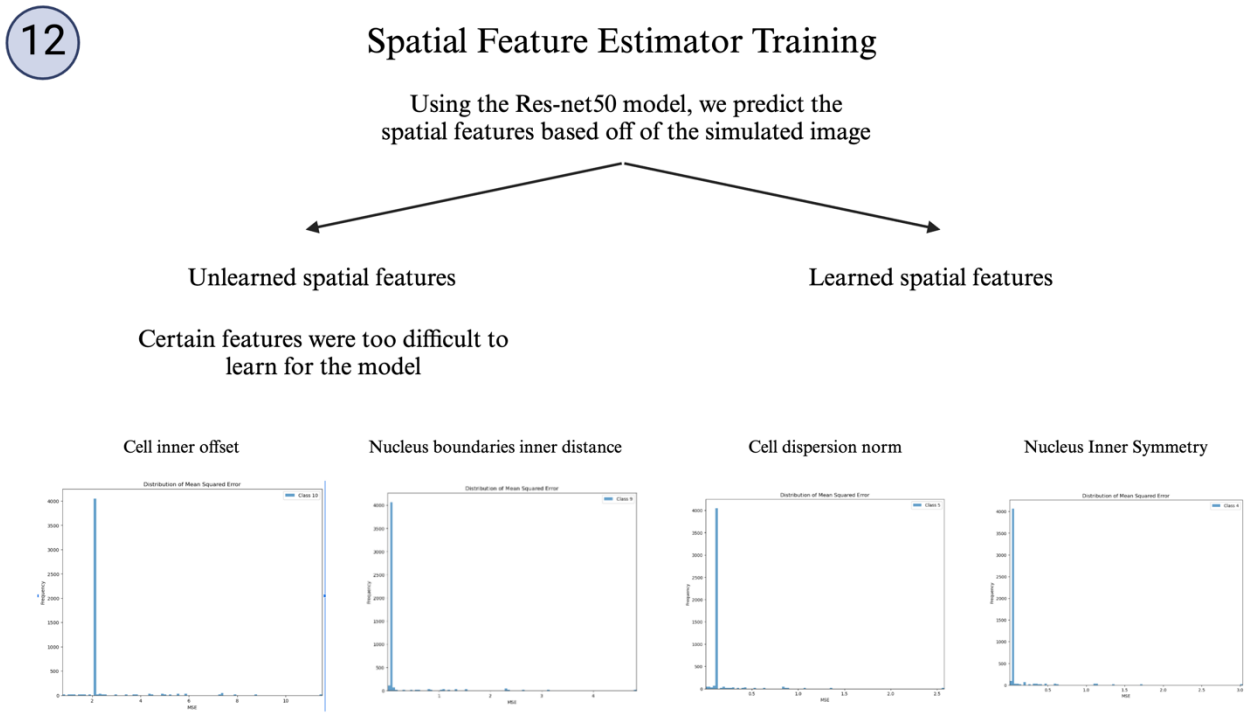

Figure 12: Spatial Feature Estimator Training

Training Autoencoder on Combined Dataset

After successfully implementing the autoencoder, I proceeded to run it through the combined dataset comprising 52,708 cells. During the training process, the Mean Squared Error (MSE) loss per epoch exhibited rapid convergence, indicating that the model was effectively learning the underlying patterns in the data (Figure 13). By the 50th epoch, the MSE value had decreased to 0.012, indicating that the autoencoder was able to reconstruct the input cells with high fidelity.

To visually assess the performance of the autoencoder, I plotted the predictions per epoch using the test dataset. This visualization, generated using Matplotlib's Pyplot library, provided a clear indication of the model's ability to reconstruct the cells accurately over the course of training. Interestingly, the plot revealed that the predictions began to reach a visually accurate level as early as the fourth epoch, suggesting that the autoencoder quickly learned to represent the spatial features of the cells in the latent space.

The rapid convergence of the MSE loss and the early achievement of visually accurate predictions underscore the effectiveness of the autoencoder in capturing and reconstructing the essential features of the cells. This efficient learning process not only demonstrates the capability of the autoencoder but also highlights its potential for use in various applications requiring accurate cell image reconstruction.

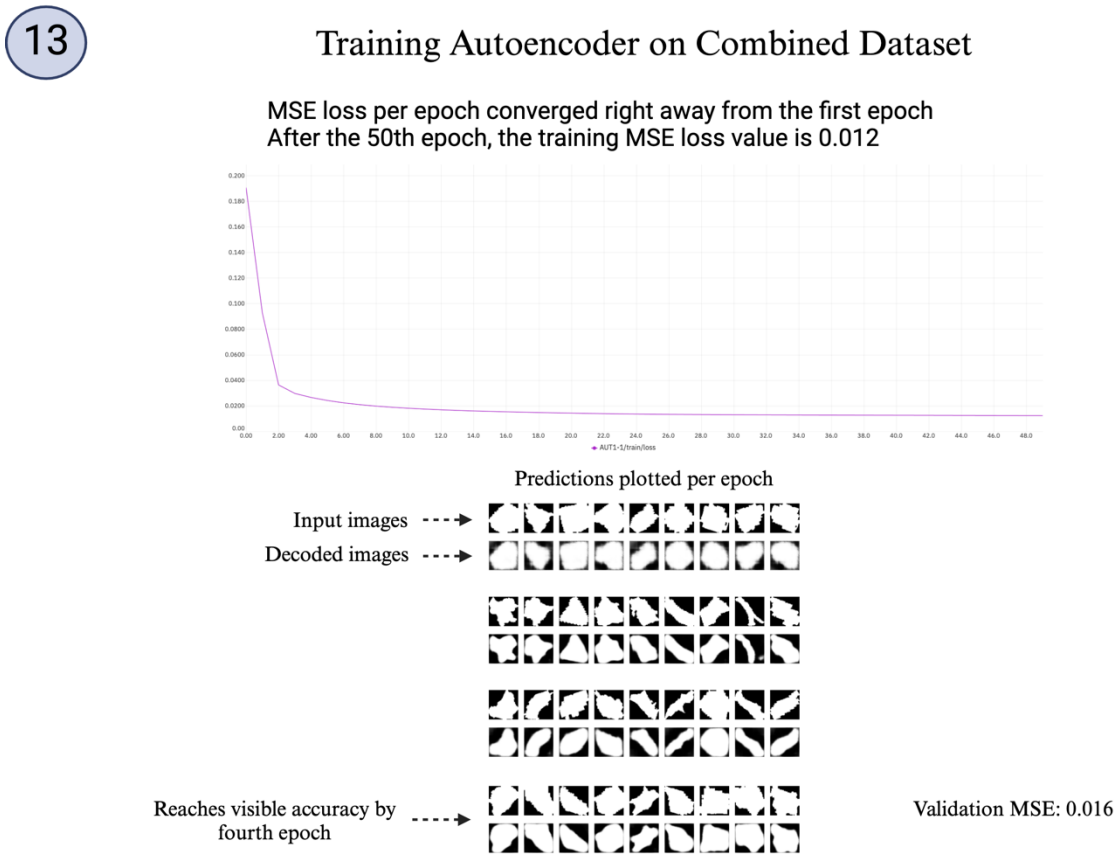

Figure 13: Training Autoencoder on Combined Dataset

Comparison of Latent Space Size to Original Image Size (Figure 14)

To assess the compression achieved by the autoencoder, we can analyze the output of its encoder layers and compare it to the original 64x64-pixel image (4096 total pixel values). The first convolutional layer produces 32 channels with a size of 32x32, the second layer outputs 16 channels with a size of 16x16, the third layer outputs 8 channels with a size of 8x8, and the final layer outputs 4 channels with a size of 4x4 (64 total pixel values).

By comparing the original pixel count of 4096 to the reduced count of 64, we can determine the degree of compression achieved. The compression ratio is calculated by dividing the original image size by the encoded space size. Through this calculation, I found that the reduction factor of my autoencoder is 64x, indicating a significant compression of the input image while retaining essential information for reconstruction.

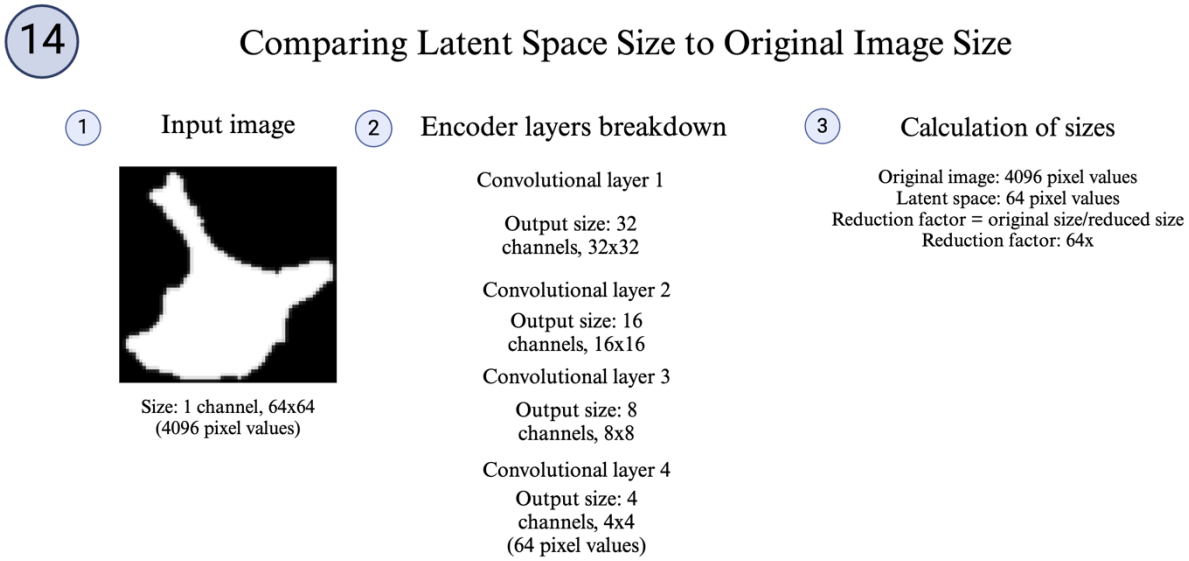

Figure 14: Data Compression Analysis: Comparing Latent Space to Original Image Size

Morphology Clustering Compared to Gene Expression Clustering (Figures 15-18)

Transitioning to a more practical implementation of my autoencoder framework, I applied it to two distinct 10X Xenium datasets: the human pancreas and mouse colon datasets. These datasets served as valuable benchmarks for my model due to their substantial size, comprising hundreds of thousands of cells, and providing essential spatial information necessary for gene expression analyses.

Utilizing the Scanpy package, I processed the data as Anndata[41], a modern data format commonly used to store information such as the cell x gene matrix. After ingesting the gene expression data provided by the Xenium dataset into an Anndata object, I preprocessed the data by removing cells with low gene counts. Subsequently, I used Scanpy's UMAP function to visually represent the data for clustering purposes. The data was then clustered using Scanpy's Leiden clustering function at different resolutions.

The Leiden clustering algorithm is an advanced community detection method in large networks, known for its robustness and accuracy. It works by iteratively refining clusters to optimize the modularity or quality of the partitioning of the network. The algorithm consists of several steps: first, it assigns nodes to clusters in a way that maximizes local modularity, then refines these clusters by reassigning nodes to neighboring clusters to further improve modularity. This process is repeated until no further improvements can be made. One of the significant benefits of the Leiden algorithm is its ability to identify well-separated, high-quality clusters efficiently, even in large and complex datasets.

Adjusting the resolution value in the Leiden algorithm directly influences the number of clusters identified, allowing for the display of unique clustering results and the adjustment of the level of detail captured by each cluster. Higher resolution values yield more granular clusters, whereas lower values produce fewer, larger clusters. This flexibility is particularly advantageous in biological data analysis, where the appropriate level of clustering can vary depending on the specific research question and the nature of the data.

Upon observing the clustering results based on morphology, I recognized the need for improvement in the autoencoder framework. It appears that the current framework may not capture enough information or may not retain the information effectively for optimal performance. This inference is supported by the lack of separation in the clusters and the resulting dilution, where clusters merge and overlap with multiple other clusters.

Remarkably, the clustering of latent vectors for both datasets, each containing over 100,000 cells, only took 4 minutes to run on my computer. This efficiency underscores the feasibility of clustering cell shapes within spatial datasets. I believe that the balance between performance and efficiency is influenced by the quality of the encoded space provided by the autoencoder framework to the clustering algorithms. This aspect is something I intend to investigate further, as I believe it holds the potential to be a powerful and widely applicable metric for clustering in various spatial datasets.

Figure 15: Xenium Human Pancreas Dataset Clustering by Gene Expression

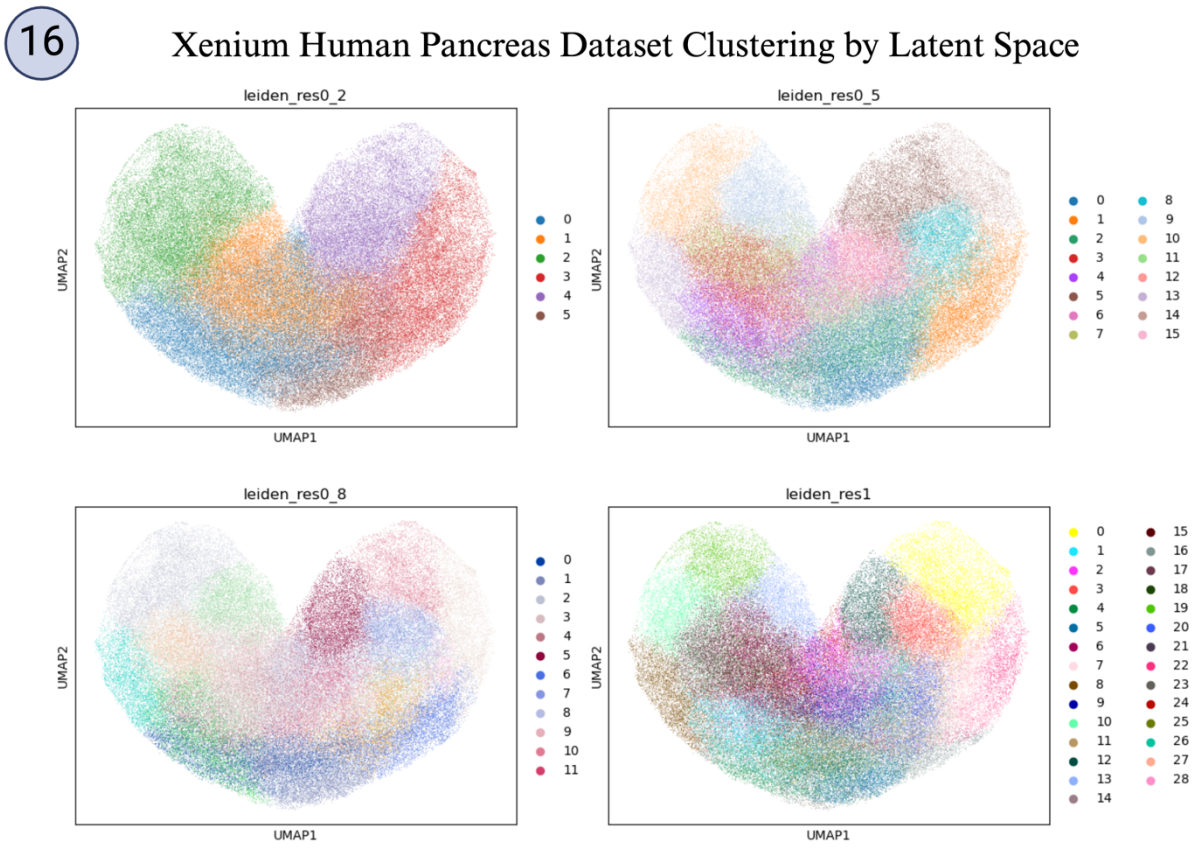

Figure 16: Xenium Human Pancreas Clustering by Latent Space

Figure 17: Xenium Mouse Colon Dataset Clustering by Gene Expression

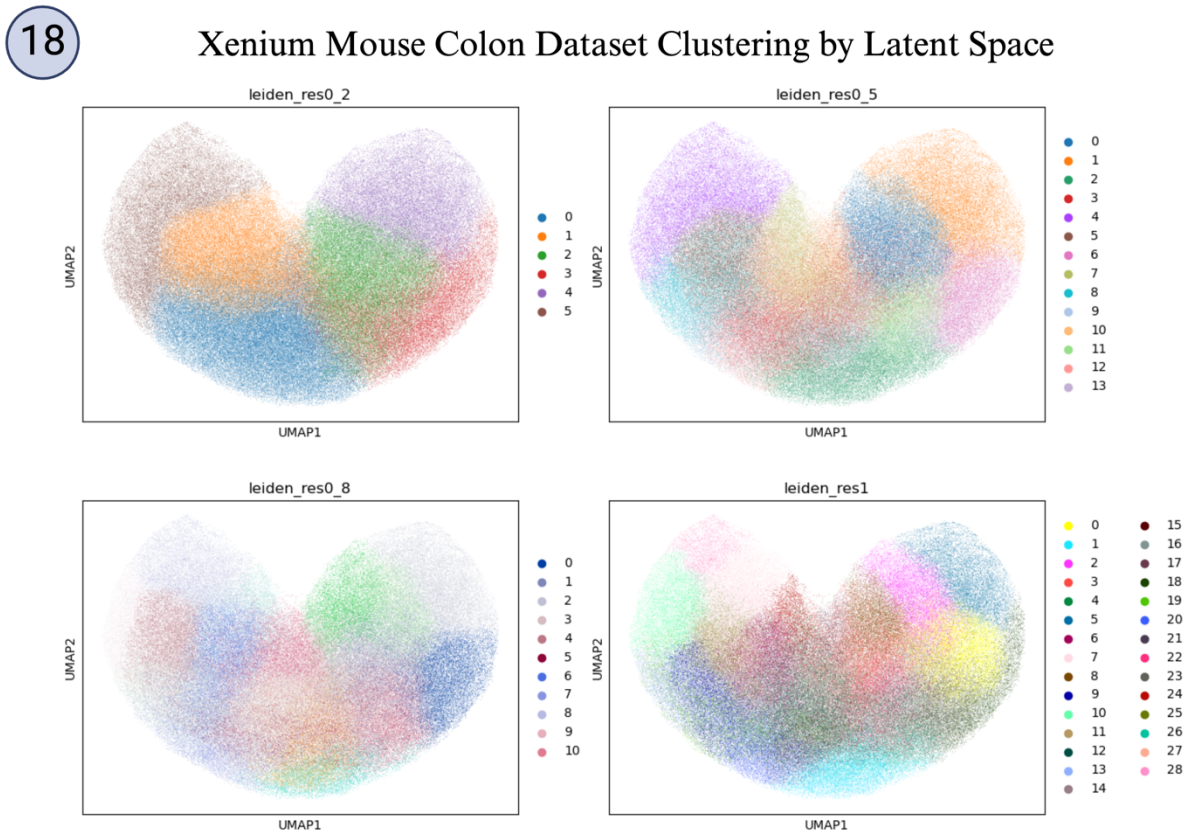

Figure 18: Xenium Mouse Colon Dataset Clustering by Latent Space

Analysis of Latent Space Clustering Disparities in Identical Cell Types

With the addition of morphological analysis using autoencoders, I aimed to uncover unique relationships that might highlight the effects of small gene expression changes on cell morphology (Figure 19). To achieve this, I performed a comprehensive comparison between clustering results from gene expression data and latent space data. Using the Leiden clustering algorithm implemented using Scanpy, I identified clusters within both clustering outputs. For both the gene expression and latent space clustering, clusters were determined by the default resolution of 1 in Scanpy's Leiden clustering function. To detect any discrepancies, I compared the clusters from both outputs and identified cells that were clustered differently, which I will refer to as discrepant cells. These cells were flagged in both datasets for further analysis.

To visualize these discrepancies, I plotted UMAP representations for both datasets, highlighting the clusters and discrepant cells. Separate plots were generated to show the clustering results based on latent vectors and gene expression data. Furthermore, I selected four example cells from the discrepant set to display their morphological reconstructions for the figure. For each cell, I identified its index in the latent vectors, converted the latent vector to a PyTorch tensor, reshaped it, and moved it to the GPU. The latent vector was then decoded to reconstruct the cell's image. These reconstructed images were displayed with annotations indicating the gene expression cluster and latent vector cluster.

This technique of comparing clustering data to observe how minute changes in gene expression affect cell shape can be a useful tool for many researchers. With morphological

clustering, I aim to identify and analyze more complex relationships, similar to those revealed by gene expression clustering. However, the current limitation lies in the quality of the morphological clustering. The clustering shows bottlenecking in the separation of clusters and tends to clump multiple clusters together, resulting in less distinct boundaries between clusters. This effect is particularly pronounced in morphology clustering, leading to a dilution of cluster purity as similar clusters are not well-separated. This highlights the need for further refinement and optimization of the morphological clustering technique to achieve clearer and more distinct cluster separations.

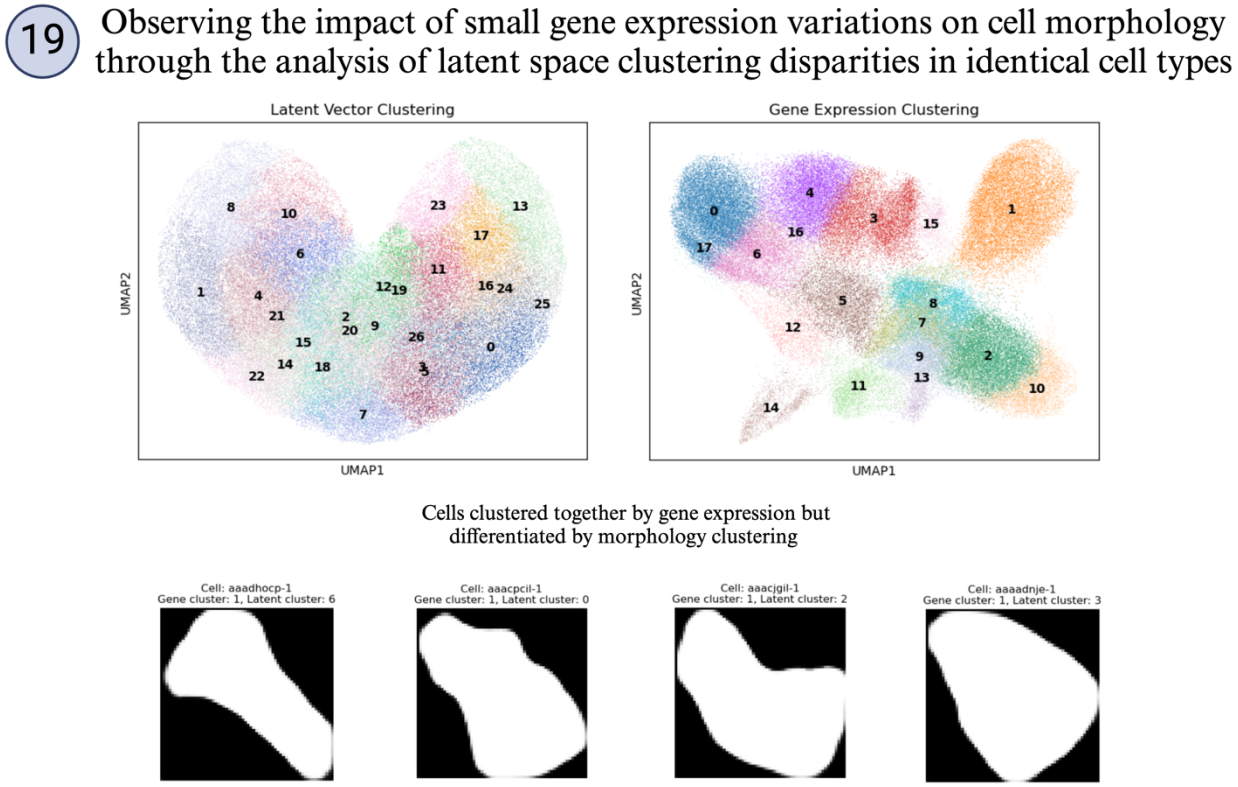

Figure 19: Observing the Impact of Small Gene Expression Variations on Cell Morphology

In addition to identifying clusters and discrepant cells, morphological clustering can be further leveraged to visualize these clusters within the tissue of origin. By mapping the latent space

clusters back to their spatial coordinates in the original tissue images, we can explore whether certain "morpho types" are enriched in specific regions of the tissue. This spatial visualization can reveal critical insights into how cell morphology correlates with tissue architecture and functional zones.

For instance, in Figure 20, the latent space clusters are visualized in Xenium's original human pancreas tissue image. By overlaying the clusters onto the tissue map, we can observe the spatial distribution of different morphological clusters. This helps in identifying regions where certain morphological features are predominant, potentially linking these features to localized tissue functions or pathological changes. Furthermore, by highlighting discrepant cells within the tissue context, we can investigate regions that exhibit significant morphological changes due to small variations in gene expression. These regions of high discrepancy may indicate areas where gene expression changes have a pronounced impact on cell shape and function, providing valuable clues for further biological investigation.

The spatial visualization of morphological clusters, therefore, offers a powerful approach to integrate morphological and spatial transcriptomics data. It allows researchers to explore the spatial heterogeneity of cell morphologies and their relationship with gene expression patterns. This integrated approach not only enhances our understanding of tissue organization and cellular behavior but also supports the identification of potential biomarkers and therapeutic targets based on spatial and morphological characteristics.

In conclusion, the combination of autoencoder-based morphological clustering with spatial visualization techniques opens new avenues for morphogenomic studies. It provides a scalable and efficient method to analyze large-scale single-cell datasets, uncovering intricate relationships between cell morphology, gene expression, and tissue architecture. By continuing to refine these techniques, we can further enhance their utility and accuracy, paving the way for more comprehensive and insightful morphogenomic analyses.

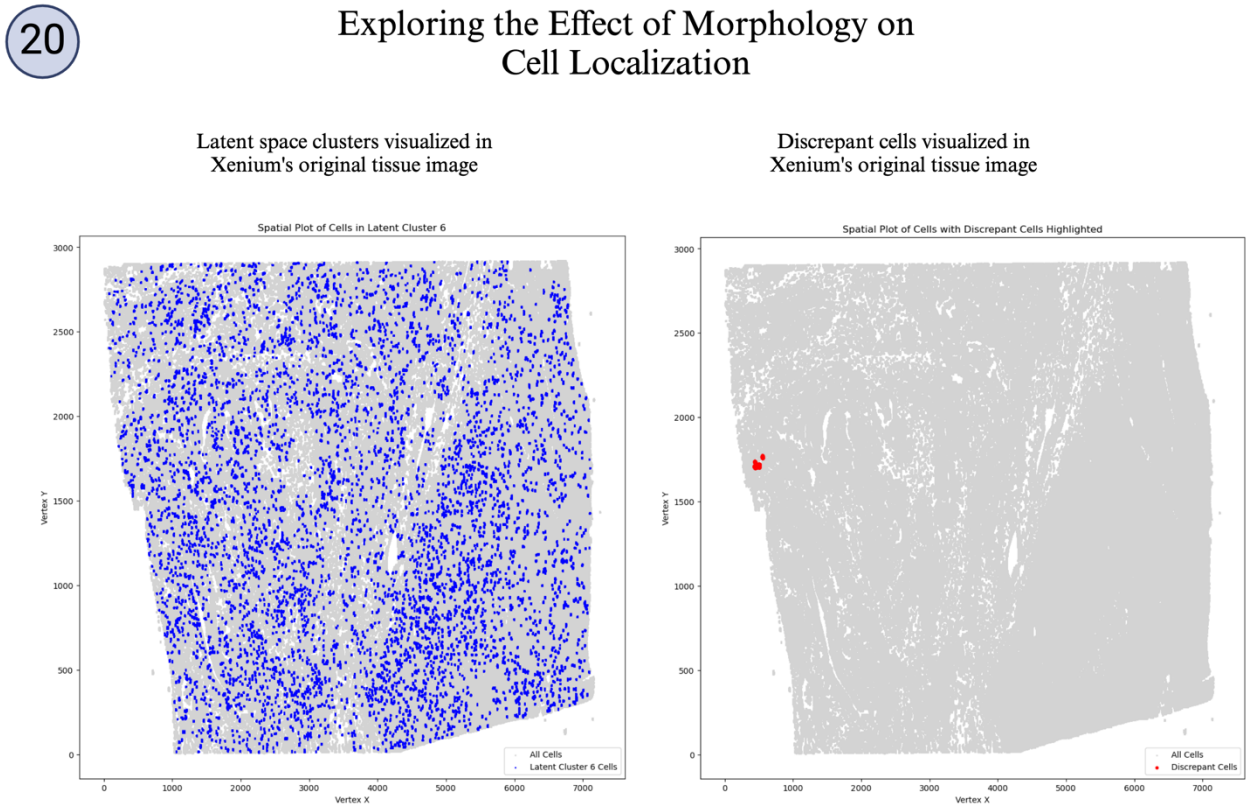

Figure 20: Exploring the Effect of Cell Morphology on Cell Localization

Discussion

The integration of morphology with gene expression data represents a fascinating advancement in the field of single-cell genomics, offering a more comprehensive understanding of cellular behavior within tissues. While single-cell genomics has provided invaluable insights into cellular diversity and function, its lack of spatial information has been a significant limitation. Spatial transcriptomics has emerged as a solution, providing a spatial context for gene expression analysis and enabling researchers to map gene expression within tissue architecture. However, the relationship between cellular morphology and gene expression has been largely unexplored in single-cell spatial data until now. The morphology of a cell is intricately linked to its gene expression profile, with changes in one often influencing the other. Morphogenomics, the integration of morphological and genomic data, has the potential to unravel these complex relationships. Yet, applying morphogenomics to single-cell spatial data has been challenging, particularly in scaling analyses to whole datasets.

To address these challenges, I developed an autoencoder capable of replicating cell masks, bridging the gap between morphology and gene expression in single-cell spatial data. The encoded latent layer, which captures essential features of cell morphology, allows for multi-scale clustering of cell morphologies, enabling me to explore complex cellular interactions and functions. This autoencoder not only addresses the scalability issues faced by traditional methods but also enhances the accuracy and efficiency of morphogenomic analyses. By reducing the dimensionality

of the data, the autoencoder facilitates more efficient clustering, providing a clearer and more detailed view of cellular morphologies and their gene expression profiles.

However, the application of this autoencoder revealed certain limitations that must be addressed for future research. One key challenge is the difficulty in achieving proper separation of clusters and cluster purity in morphology clustering. The morphology clusters often showed less separation and tended to clump multiple clusters together, leading to dilution in similar clusterings. This limitation underscores the need for further refinement and optimization of morphological clustering techniques to achieve clearer and more distinct cluster separations. Addressing this challenge is crucial for ensuring that morphology clustering is truly useful in providing insights into cellular behavior and function. Another challenge lies in balancing the size of the encoded space with efficiency and performance. While a smaller encoded space may improve efficiency, it may also lead to loss of important morphological information. On the other hand, a larger encoded space may capture more morphological details but could result in increased computational complexity. Finding the right balance between these factors is essential for maximizing the utility of the autoencoder in morphogenomic analyses.

In moving forward, the next steps for improving this morphogenomic tool include several key areas of focus. Firstly, enhancing the quality of morphological clustering is paramount. This could involve experimenting with different clustering algorithms, adjusting parameters, or integrating advanced techniques like graph-based clustering to achieve more distinct and

meaningful separations. Additionally, optimizing the autoencoder architecture to better capture critical morphological features without unnecessary complexity will be essential.

The integration of this autoencoder with existing spatial transcriptomics techniques creates new avenues for biological insights. By mapping the clusters identified in the latent space back to their spatial coordinates within the tissue, researchers can investigate whether certain morphological types are enriched in specific tissue regions. This approach can reveal how cell morphology correlates with tissue architecture and function, potentially identifying new functional zones or regions impacted by specific gene expression changes. Further, the ability to visualize morphologically discrepant cells in the same cell type provides a powerful tool for uncovering the effects of gene regulation on cell shape. This can lead to the identification of biomarkers or therapeutic targets based on spatial and morphological characteristics, offering new directions for disease research and treatment.

Looking ahead, the potential applications of this tool are vast. It can be used to explore developmental biology questions, such as how cells differentiate and organize into tissues, or to study disease progression by identifying morphological anomalies linked to gene expression changes. Moreover, the integration of multi-scale clustering with cell morphology analysis can facilitate the discovery of previously unknown cellular interactions and functions, providing deeper insights into the complex networks that govern cellular behavior. The development of this autoencoder represents a significant advancement in the field of single-cell genomics, offering a

powerful tool for studying cellular morphology in conjunction with gene expression. By addressing current limitations and identifying areas for future improvement, this work lays the foundation for further research in morphogenomics, with the potential to greatly enhance our understanding of cellular behavior and function in complex biological tissues. Moving forward, efforts to enhance the quality of morphological clustering and to integrate the autoencoder with existing spatial transcriptomics techniques will be key in unlocking deeper insights into the complexities of cellular biology. The ability to combine morphological and genomic data through advanced machine learning approaches heralds a new era of biological discovery, with profound implications for both basic and applied research.

References

1. Tang, F., Barbacioru, C., Wang, Y., Nordman, E., Lee, C., Xu, N., Wang, X., Bodeau, J., Tuch, B.B., Siddiqui, A., Lao, K., Surani, M.A., 2009. mRNA-Seq whole-transcriptome analysis of a single cell. Nature Methods, 6(5), 377-382.

2. Zheng, G.X.Y., Terry, J.M., Belgrader, P., Ryvkin, P., Bent, Z.W., Wilson, R., Ziraldo, S.B., Wheeler, T.D., McDermott, G.P., Zhu, J., Gregory, M.T., Shuga, J., Montesclaros, L., Underwood, J.G., Masquelier, D.A., Nishimura, S.Y., Schnall-Levin, M., Wyatt, P.W., Hindson, C.M., Bharadwaj, R., Wong, A., Ness, K.D., Beppu, L.W., Deeg, H.J., McFarland, C., Loeb, K.R., Valente, W.J., Ericson, N.G., Stevens, E.A., Radich, J.P., Mikkelsen, T.S., Hindson, B.J., Bielas, J.H., 2017. Massively parallel digital transcriptional profiling of single cells. Nature Communications, 8, 14049.

3. Ståhl, P.L., Salmén, F., Vickovic, S., Lundmark, A., Navarro, J.F., Magnusson, J., Giacomello, S., Asp, M., Westholm, J.O., Huss, M., Mollbrink, A., Linnarsson, S., Codeluppi, S., Borg, Å., Pontén, F., Costea, P.I., Sahlén, P., Mulder, J., Bergmann, O., Lundeberg, J., Frisén, J., 2016. Visualization and analysis of gene expression in tissue sections by spatial transcriptomics. Science, 353(6294), 78-82.

4. Moncada, R., Barkley, D., Wagner, F., Chiodin, M., Devlin, J.C., Baron, M., Hajdu, C.H., Simeone, D.M., Yanai, I., 2020. Integrating single-cell RNA-sequencing with spatial transcriptomics in pancreatic ductal adenocarcinoma using multimodal intersection analysis. Nature Biotechnology, 38(3), 333-342.

5. Cutiongco, M. F. A., Jensen, B. S., Reynolds, P. M., & Gadegaard, N. (2020). Predicting gene expression using morphological cell responses to nanotopography. *Nature Communications*, 11, Article 1514. https://doi.org/10.1038/s41467-020-15114-1

6. Giesen, C., Wang, H.A., Schapiro, D., Zivanovic, N., Jacobs, A., Hattendorf, B., Schüffler, P.J., Grolimund, D., Buhmann, J.M., Brandt, S., Varga, Z., Wild, P.J., Günther, D., Bodenmiller, B., 2014. Highly multiplexed imaging of tumor tissues with subcellular resolution by mass cytometry. Nature Methods, 11(4), 417-422.

7. Tegtmeyer, M., Arora, J., Asgari, S., Cimini, B. A., Nadig, A., Peirent, E., Liyanage, D., Way, G. P., Weisbart, E., Nathan, A., Amariuta, T., Eggan, K., Haghighi, M., McCarroll, S. A., O'Connor, L., Carpenter, A. E., Singh, S., Nehme, R., & Raychaudhuri, S. (2024). High-dimensional phenotyping to define the genetic basis of cellular morphology. *Nature Communications*, 15, 347. DOI: 10.1038/s41467-023-44045-w.

8. Tan, X., Su, A., Tran, M., & Nguyen, Q. (2020). SpaCell: Integrating tissue morphology and spatial gene expression to predict disease cells. Bioinformatics, 36(7), 2293-2294. DOI: 10.1093/bioinformatics/btz914.

9. Chen, K.H., Boettiger, A.N., Moffitt, J.R., Wang, S., Zhuang, X., 2015. RNA imaging. Spatially resolved, highly multiplexed RNA profiling in single cells. Science,348(6233), aaa6090.

10. Shah, S., Lubeck, E., Zhou, W., Cai, L., 2017. seqFISH accurately detects transcripts in single cells and reveals robust spatial organization in the hippocampus. Neuron, 94(4), 752-758.e1.

11. Alon, S., Goodwin, D.R., Sinha, A., Wassie, A.T., Chen, F., Daugharthy, E.R., Bando, Y., Kajita, A., Xue, A.G., Marrett, K., Wang, G., Church, G.M., 2021. Expansion Sequencing: Spatially Precise In Situ Transcriptomics in Intact Biological Systems. Science, 371(6528), eaax2656.

12. Hickey, J.W., Hsu, C.T., Li, J., Griffin, M.E., Prieskorn, Z.R., Co, J.Y., Kosoy, R.E., 2021. Xenium: A Scalable Platform for Spatially Resolved Gene Expression Profiling in Tissues. Nature Methods, 18(7), 803-813.

13. Li, J., Sun, L., Liu, L., & Song, H. (2022). Morphogenomics: Integrating morphological and genomic data to elucidate cell behavior. Cell Systems, 6(3), 205-217. DOI: 10.1016/j.cels.2022.05.003

14. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems, 32, 8024-8035.

15. Mah, C. K., Ahmed, N., Lam, D., & Yeo, G. (2024). Bento: A computational toolkit for spatial analysis. Genome Biology

16. Max Planck Society, 2024. Edmond Open Research Data Repository. Retrieved from https://edmond.mpdl.mpg.de/imeji/collection/3zIZ2AoiTBSq56Ko

17. Williams, E., Conway, J.R., Clark, R., Huynh, T., Furtado, L.V., Luu, N., Katz, M., Russell, M.R., Liu, Y., Khuong, A., 2019. The Image Data Resource: A bioimage data integration and publication platform. BMC Bioinformatics.

18. GE Healthcare, 2021. IN Cell Analyzer: Advanced high-content analysis. Available at: https://www.gehealthcare.com/products/in-cell-analyzer

19. Kamentsky, L., Jones, T.R., Fraser, A., Bray, M.A., Logan, D.J., Madden, K.L., Ljosa, V., Rueden, C., Eliceiri, K.W., Carpenter, A.E., 2011. Improved structure, function and compatibility for CellProfiler: modular high-throughput image analysis software. Bioinformatics, 27(8), pp.1179-1180.

20. Cell Image Library, 2024. Open-source cell image database. Retrieved from http://cellimagelibrary.org

21. Adobe Systems, 2021. Adobe Photoshop: Image editing software. Available at: https://www.adobe.com/products/photoshop.html
22. Zeiss, 2021. Zeiss Epifluorescence Microscopes. Available at: https://www.zeiss.com/microscopy/us/products/light-microscopes/epifluorescence.html

23. Goltsev, Y., Samusik, N., Kennedy-Darling, J., Bhate, S., Hale, M., Vazquez, G., Black, S., Nolan, G.P., 2018. Deep profiling of mouse splenic architecture with CODEX multiplexed imaging. Cell, 174(4), 968-981.

24. Lin, J.R., Izar, B., Wang, S., Yapp, C., Mei, S., Shah, P.M., Santagata, S., Sorger, P.K., 2018. Highly multiplexed immunofluorescence imaging of human tissues and tumors using t-CyCIF and conventional optical microscopes. eLife, 7, e31657.

25. Giesen, C., Wang, H.A.O., Schapiro, D., Zivanovic, N., Jacobs, A., Hattendorf, B., Schüffler, P.J., Grolimund, D., Buhmann, J.M., Brandt, S., Varga, Z., Wild, P.J., Günther, D., Bodenmiller, B., 2014. Highly multiplexed imaging of tumor tissues with subcellular resolution by mass cytometry. Nature Methods, 11(4), 417-422.

26. Angelo, M., Bendall, S.C., Finck, R., Hale, M.B., Hitzman, C., Borowsky, A.D., Levenson, R.M., Lowe, J.B., Liu, S.D., Zhao, S., Natkunam, Y., Nolan, G.P., 2014. Multiplexed ion beam imaging of human breast tumors. Nature Medicine, 20(4), 436-442.

27. Gerdes, M.J., Sevinsky, C.J., Sood, A., Adak, S., Bello, M.O., Bordwell, A., Can, A., Corwin, A., Dinn, S., Filkins, R.J., Hollman, D., Kamath, V., Kaanumalle, S., Karunakaran, D., Koepke, I., Krueger, P., Lamoreaux, C., Lazare, M., Li, C., Lowes, C., McCaffrey, J., McDonough, E., Montalto, M.C., Neumann, J., Pang, Z., Rittscher, J., Saliga, R., Preckel, T., and Suter, M., 2013. Highly multiplexed single-cell analysis of formalin-fixed, paraffin-embedded cancer tissue. Proceedings of the National Academy of Sciences, 110(29), 11982-11987.

28. PerkinElmer, Inc. (2021). Vectra Polaris Automated Quantitative Pathology Imaging System. PerkinElmer. Retrieved from https://www.perkinelmer.com/product/vectra-polaris-automated-quantitative-pathology-imaging-system

29. CZ Biohub, 2021. Reference data for gene panels. Available at: https://www.czbiohub.org

30. Tabula Muris Consortium, 2018. Single-cell transcriptomic characterization of 20 organs and tissues from mouse at different stages of life. Cell, 173(5), 1307-1322.

31. Reddi, S., Coleman, C., Kanter, D. (2021). Neural network trained using a diverse dataset outperforms conventionally trained algorithms. Harvard SEAS.

32. van der Walt, S., Schönberger, J.L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., & Yu, T. (2014). scikit-image: Image processing in Python. PeerJ, 2, e453. DOI: 10.7717/peerj.453.

33. Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Oliphant, T.E. (2020). Array programming with NumPy. Nature, 585, 357–362. DOI: 10.1038/s41586-020-2649-2.

34. McKinney, W. (2010). Data Structures for Statistical Computing in Python. Proceedings of the 9th Python in Science Conference, 51-56. DOI: 10.25080/Majora-92bf1922-00a.

35. Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95. DOI: 10.1109/MCSE.2007.55.

36. Wolf, F.A., Angerer, P., & Theis, F.J. (2018). SCANPY: large-scale single-cell gene expression data analysis. Genome Biology, 19(1), 15. DOI: 10.1186/s13059-017-1382-0.

37. Gillies, S., Fogel, E., Dobos, G., Wilson, J., Cochran, D., Gaudet, B., Warner, J., Czaja, J., Krauss, M., Brittain, J., Walker, A., Richard, A., Lamouroux, J., Davies, D., Fawcett, P., Padfield, P., Quinlan, R., Morley, A., & de Hoon, M. (2023). Shapely: manipulation and analysis of geometric objects. Available at: https://shapely.readthedocs.io

38. Imbert, A., Ouyang, W., Safieddine, A., Coleno, E., Zimmer, C., Bertrand, E., Walter, T., & Mueller, F. (2022). FISH-quant v2: a scalable and modular tool for smFISH image analysis. RNA, 28(6), 786-795. DOI: 10.1261/rna.079073.121.

39. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770-778. DOI: 10.1109/CVPR.2016.90.

40. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. Medical Image Computing and Computer-Assisted Intervention (MICCAI), Springer, LNCS, Vol.9351: 234-241. DOI: 10.1007/978-3-319-24574-4_28.

41. Virshup, I., Rybakov, S., Theis, F.J., Angerer, P., & Wolf, F.A. (2021). AnnData: Annotated data matrices for scalable and extensible single-cell analysis. bioRxiv. DOI: 10.1101/2021.12.16.473007.