Continuous Modeling of Cellular Trajectories in The Alzheimer's Brain

Guy Lutsker Supervised by Dr. Naomi Habib

Abstract

Cell states transitions have long been considered to be a continuous process with respect to time. However, in single nuclei RNA-sequencing (sNuc-seq) data it is hard to determine temporal processes such as that since the data presents us with only a single snapshot in time. Fortunately, the diversity of the cell's gene expressions is often enough to infer meaningful inquiries regarding the cell's biological behavior. The main answer to deducing cell states within a certain cell type has been using trajectory inference models, and yet despite the fact that numerous methods for trajectory inference have been created, the majority of them lack a cohesive statistical model and reliable uncertainty quantification. Our data consists of 24 brain samples from aging individuals, healthy or Alzheimer's disease patients, and our hope is that by understanding the biological mechanisms responsible for cell state transitions in human brain cells we would be one step closer in understanding the main drivers of Alzheimer's. In particular we are interested in sNuc-seq data of human Astrocytes in the Medial Frontal Cortex as these cells have already shown evidence of a continuous-like property regarding their cell states. In this paper we will review several classical solutions that do not use traditional trajectory inference methods including language models, diffusion maps, pareto front optimization etc. We will also present TRIVIA (TRee Induced VarIational Autoencoder), a probabilistic AI model which has taken inspiration from both variational autoencoder, and traditional trajectory inference methods such as using minimum spanning trees in order to encourage the data cloud to arrange itself into a trajectory pattern.

Background

Alzheimer's disease (AD) is the most prevalent form of dementia and a progressive neurodegenerative disease that affects millions of individuals worldwide (Butler, 2018). By 2050, some estimates argue that more than 29 million people will be affected by AD, in the US alone (Sims-Robinson, 2010), for which there is now no effective therapy. AD is associated with moderate cognitive impairment in the early stages (Butler, 2018), but it progresses to include declines in cognition, memory, and language, as well as damage to fundamental physiological functions and, eventually, death. The pathology of Alzheimer's disease is characterized by significant cerebral cortical degeneration and extensive neuron loss (Anders M. Fjell, 2014). In addition, we also see that AD patients exhibit an abnormal aggregation of proteins, specifically, amyloid-β (Hartmann, 1999) and neurofibrillary tangles, which are neurotoxic and causes synapse damage. Traditional research of AD focused on the damage and degeneration of neurons in the brain (Anders M. Fjell, 2014), whilst more recent work, from our lab and others, showed multiple cell types changing in AD brains beyond neurons (Habib N. M., 2020). For example, our lab has been able to show that Astrocytes have appeared to change their cellular state along changes with AD. (Habib N. M., 2020) Astrocytes are a characteristic star-shaped glial cells in (Verkhratsky, 2013) the brain and spinal cord. Astrocytes perform many functions, including biochemical support of endothelial cells that form the blood–brain barrier, provision of nutrients to the nervous tissue, maintenance of extracellular ion balance, regulation of cerebral blood flow, and a role in the repair and scarring process of the brain and spinal cord following infection and traumatic injuries (Burda, 2014). The proportion of astrocytes in the brain is not well defined, studies have found that the astrocyte proportion varies by region and ranges from 20% to 40% of all glia (Verkhratsky, 2013). Astrocytes in humans are more than twenty times larger than in rodent brains and contact with more than ten times more the number of synapses (Sloan SA, 2014). Using the novel single nuclei RNA-sequencing (sNuc-seq) technology (Habib N. L.-D., 2016) (Habib N. A.-D., 2017), which enables high-throughput profiling of single cell transcriptomes, we can profile the entire cellular environment of the brain along different stages of disease progression. This novel technology thus enables the characterization of changes along AD progression at the cellular and molecular level, including discovering new molecular pathways and cellular states driving AD. By combining sNuc-seq data and computational tools such as unsupervised machine learning algorithms, we can infer such new insights. The common analysis pipeline of sNuc-seq data (Figure 1), includes reducing the dimension of the data (Van Der Maaten, 2009) (from 20K gene space down to a couple of dozens of dimensions, to reduce noise and data complexity), followed by clustering algorithms (Traag, 2019) (typically based on similarities in K-Nearest Neighbor graphs) to divide the cells into discrete subsets. While this approach is indeed successful in capturing the diversity of cell

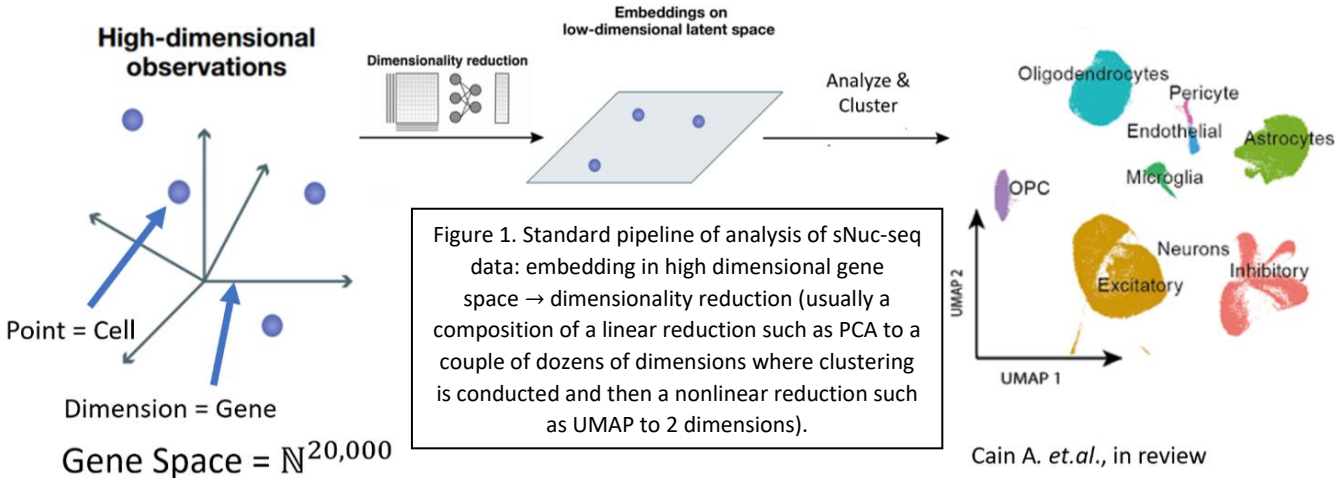


Figure 1. Standard pipeline of analysis of sNuc-seq data: embedding in high dimensional gene space → dimensionality reduction (usually a composition of a linear reduction such as PCA to a couple of dozens of dimensions where clustering is conducted and then a nonlinear reduction such as UMAP to 2 dimensions).

types (Habib N. A.-D., 2017) (Cain, 2020), it often fails to capture the diversity of sub-populations such as different cell states within a cell type. We suspect that the reason for this failure is that cells within a cell type do not divide to discrete sub-populations, but rather on a continuous spectrum between different cell states. For example, states of astrocyte cells in the medial frontal cortex were previously observed to be modeled along a continuous trajectory between a homeostatic and reactive states (Habib N. M., 2020). There are some issues with these sort of discrete clustering algorithms for single nuclei data (Kiselev, 2019), for example, using them for cell state categorization is problematic because transitions between cell states are thought to be a continuous process, and we expect to capture cells changing at any given moment. Furthermore, at any one time, no cell is in a single state. This is the root cause of the failure of the discrete clustering approach, since assigning cells to single label is incorrect.

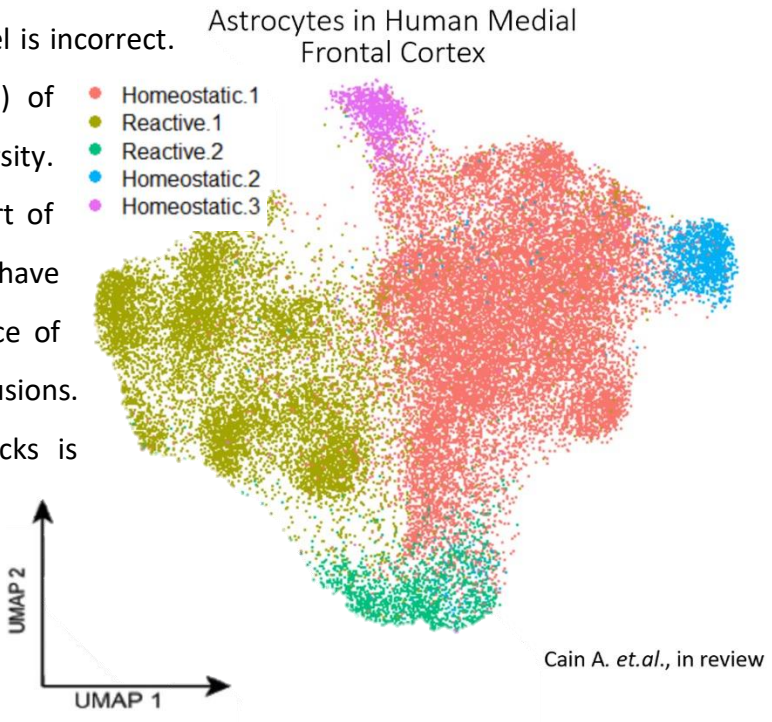


Figure 2 shows an attempt by our lab (Cain, 2020) of assigning cells to discrete groups of astrocytes diversity. Since the data is continuous, the problem in this sort of discrete assignment is that cells along the borders have arbitrary assignment that depend on noise and choice of parameters and could result in misleading conclusions. Another noteworthy feature that this technique lacks is trajectory analysis (Qiu, 2017); this information is crucial since it can teach us a lot about how cell state changes occur in the brain. However, despite multiple efforts and algorithms developed (Kushal K. Dey, 2017) (Laleh Haghverdi, 2015), we are still lacking a robust algorithm that align cells along a trajectory of cell state transitions. We aim to examine current as well as build new techniques for modeling cell states and their continuous

Figure 2. Analysis of MFC Astrocytes by Anael Cain. UMAP representation with annotations of clusters meant to approximate cell states.3D variation

transitions and apply them to sNuc-seq data in order to reveal new mechanisms of Alzheimer's disease. In general, it is theoretically possible to model cell states transitions in all cell types, but as it has already been shown that Medial Frontal Cortex (MFC) Astrocytes exhibit continuous-like properties regarding their cell states, we chose to focus our research on Astrocytes as an interesting and informative starting point.

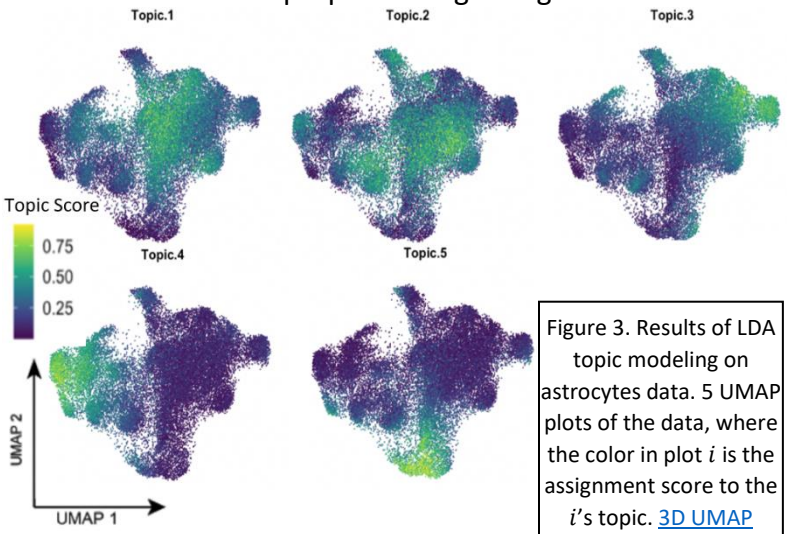


Figure 3. Results of LDA topic modeling on astrocytes data. 5 UMAP plots of the data, where the color in plot i is the assignment score to the i's topic. 3D UMAP 5topicsLDA.html

Results

The data consists of single nucleus RNA sequencing data from 24 post-mortem human brain tissue. The data is from 12 healthy patients, and 12 patients with advanced AD. Formally, the data is a sparse count matrix $M \in R^{\#cells \times \#genes}$, consisting of rows representing cells we captured, and columns representing the approximately 20,000 possible genes that could be expressed in human cells. We use as a reference point an annotation of the cells (Cain, 2020) where Anael Cain in our lab conducted Leiden community clustering of the cells to 5 distinct clusters named Homeostatic 1 through 3 (Exhibit more of marker genes of Homeostatic Astrocytes), and Reactive 1 and 2 (Exhibit more of marker genes of Reactive Astrocytes), which exhibited distinctive gene expression patterns. These cluster annotations will be used as a ground truth for validations in this work.

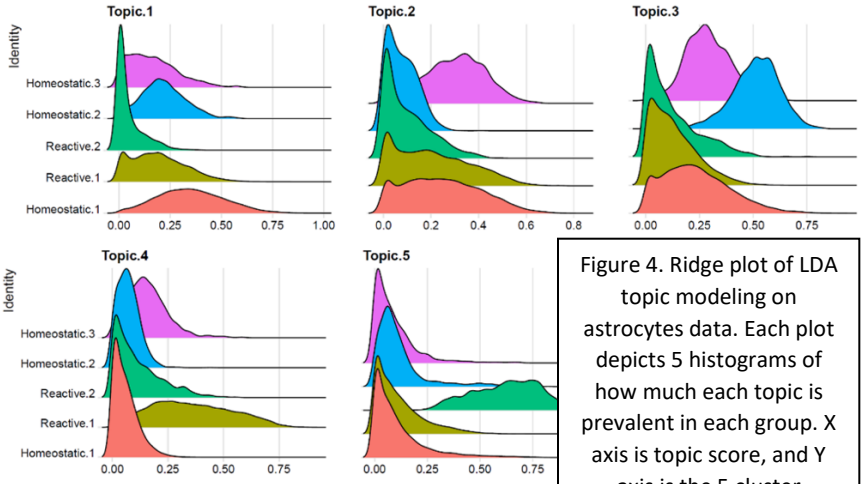


Figure 4. Ridge plot of LDA topic modeling on astrocytes data. Each plot depicts 5 histograms of how much each topic is prevalent in each group. X axis is topic score, and Y axis is the 5 cluster annotations.

Topic Modelling: Topic Modelling (Jelodar, 2019) is a soft assignment algorithm that is based on Latent Dirichlet allocation (LDA), originally used for Natural Language Processing (NLP) (Blei, 2003). In this method topics are found in a set of documents which consists of words. The result is a soft assignment for each document (based on the words in each document) to a set of topics (Kushal K. Dey, 2017). Here we model our data as follows: Each document $\boldsymbol{m}$ represents a **Cell** in our sample, each word in a document $\boldsymbol{w}$ represents a **Gene** that is expressed in a cell, and each **Topic** that is found represents a **Biological Expression Program.** Running Topic Modeling with 5 topics results in distinct expression programs along domains of the 2D UMAP (Becht, 2019) embedding (figure 3). Topic Modeling with 5 topics can indeed capture some of the diversity we observed in the cluster annotations (Figure 4), as Topics 1-3 capture Homeostatic-associated gene expression (although it is hard to tell what the difference between the topics is, since the

genes of these topics are quite similar), while topic 4 has high scores in Reactive 1 cluster cells, and topic 5 has high scores in Reactive 2 cells. Topic Modeling seems to be able to find a signal which produces topic score distributions that aligns with our clustering, as in topics 4,5, while sometimes it could be quite uninterpretable.

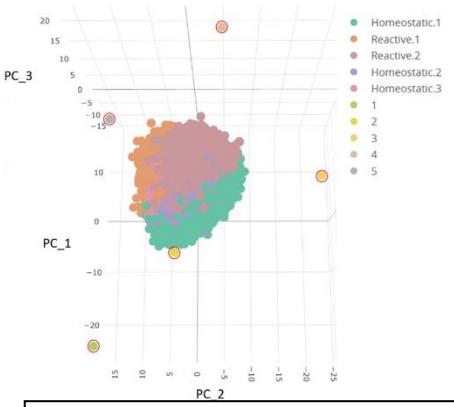


Figure 5. Archetype analysis results. Data cloud is a PCA projection of the data onto 3 dimensions. Points with red circle around them are the nodes of the convex hull – The Archetypes Themselves.3D

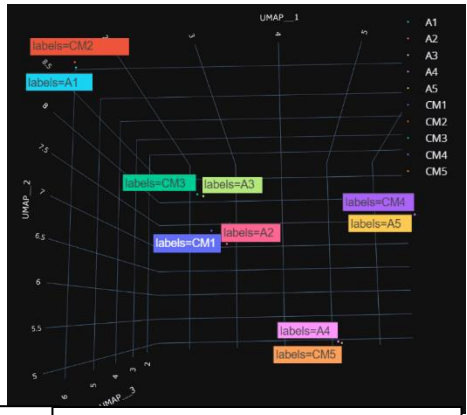


Figure 6. UMAP representation of embedding the data with the archetypes. A1-5 are the archetypes, CM1-5 (cluster mean) are the centroids of the clusters.

Pareto Archetypes: Looking at the problem from another angle, we may observe that a cell that needs to execute several functions at once, must confront a fundamental trade-off: it cannot be optimal at all functions given a set of expressible genes. These trade-offs are attempted to be balanced using the Pareto Archetype approach (O. Shoval, 2012). The optimal trade-offs are weighted averages of archetypes—usually a set of phenotypes in nature, but here we can represent them as the combination of gene expression profiles to amount to the expression programs that the cell executes. This is done using the Pareto front concept from economics and engineering (Ngatchou, 2005). In general, the way the algorithm works is to reduce the dimensionality of the samples using principal component analysis (PCA), and then looking for the best fit convex hull with k nodes. The idea is that the position of these k nodes in the gene space we reveal the k dominating features of the data set and can be interpreted as the expression programs the data set contains. Each cell c then would be characterized by a convex combination of these archetypes, such that $c = \sum_{i=1}^{k} \lambda_i \cdot arch_i$. An example of running Archetype analysis on the data is in figure 5. As we can see from the figure the Pareto algorithm was able to construct the nodes of a convex hull (archetypes). However, when trying to represent each cell as a linear combination of archetypes, we noticed that we get very similar values to the cluster centroids. This was quite interesting, and when we tried to see how the archetypes would embed in the UMAP space, we saw (Figure 6) that they are mapped really close to the mean of each annotated cluster by Cain. This meant that archetypes in this data set are quite similar to centroid and are as such less useful than we thought.

Diffusion Maps (DM): Another approach is to try to embed the data in a continuous latent space that captures the diversity of the data. Diffusion maps (Laleh Haghverdi, 2015) are a non-linear dimensionality reduction technique which reorganizes data based on fundamental geometric factors. A local similarity measure is utilized to assess the data set's connectivity, which is then used to generate a time-dependent diffusion process. As the diffusion progresses, local geometry is integrated, disclosing geometric patterns in

the dataset at various scales. The Euclidean distance between points in a diffusion map approximates the diffusion distance in the original feature space. Figure 7.1 shows the results of this method, and while the results show that embedding the data in a lower dimension using DM might enables capturing axes of diversity – DM_1 seems to somehow capture some of the diversity of Reactive.2 group, and DM_2 the same for Reactive.1. Yet as can be seen in Figure 7.2 most of the data is concentrated in a single point and so calculating trajectories in the data, cannot model distinct expression programs that would unmask pathways underlying Alzheimer's disease. In addition, Diffusion Maps may not scale well with data depicting multiple branches transitions (Herring, 2018).

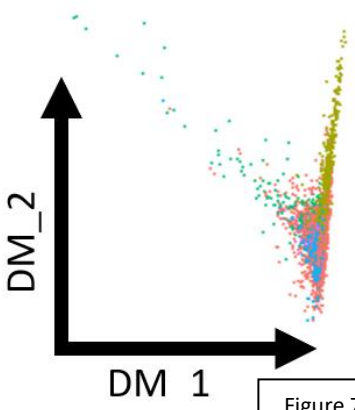


Figure 7.1. Diffusion Map of the data. 3D

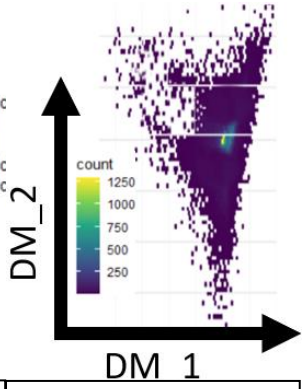


Figure 7.2. Density plot of the Diffusion Map of the

Gaussian UMAP: One might hear of this challenge and think that this is a classical manifold learning problem. The most popular tool, as of writing this, for manifold learning is UMAP (Becht, 2019). Uniform Manifold Approximation and Projection (UMAP) is a dimension reduction technique that can be used for visualization similarly to t-SNE (Linderman, 2019), but also, for general non-linear dimension reduction**.** One of the most recent advancements (Narayan Ashwin, 2020) in UMAP is the ability to choose an output metric for the embedding. Embedding the data using UMAP with the Gaussian output metric results in figure 8. This is the first time so far that we actually seen trajectory-structure in the resulting 2D embeddings. To analyze these two trajectories, one going from Homeostatic.1 to Reactive.1 and the other from Homeostatic.1 to Reactive.2, we marked the centroid of each group, and assign each cell along these two trajectories. Given these cell labels we can consider them as Pseudotime and try to find the underlying biological processes, since it enables us to study the gene expression dynamics of Astrocytes cells. Each trajectory has both a Pseudotime vector and a mean gene expression vector for each gene and cell along this trajectory (figure 8). To identify the genes that are changing coordinately along each trajectory we used continuous mutual information measure (Moddemeijer, 1989) (A. Kraskov, 2004). This measure tells us a score of relationship between a gene and the Pseudotime annotation. Denote genes i's score as MI_i . To calculate the statistical significance for each gene, we used a permutation test to calculate a $p - value$ for each gene with the following formula:

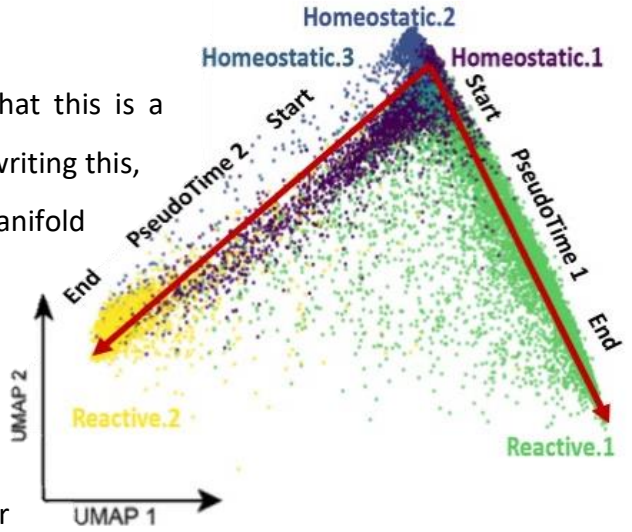


Figure 8.1 Gaussian UMAP plot of the data. Colors are the cluster annotations; red arrows are imposed trajectories. 3D Plot

$\sum_j \frac{\mathbb{1}_{[MI_i < MI(gene_i, \text{pseudotime}_{\text{shuffled}})]}}{\#\text{permutations}}$ with 1000 permutations. We then chose genes with the highest mutual information score that had $p - value_{adj} < 0.01$ (FDR adjusted) as the differential genes along this trajectory (500 genes). To further analyze the expression changes along the trajectory, we clustered the genes based on their expression patterns along the trajectory and discovered distinct clusters of genes, that each had a unique expression pattern. Next, we found enriched pathways per cluster, which uncovered their biological processes (figure 8.2). We found that as the Astrocytes transition from Homeostatic state to a Reactive one, they upregulate immune and stress response and downregulate neuronal structure and development pathways, which matched previous published results. Demonstrating our strategy to transition from trajectories to biological insights. Yet, although this method is capable of producing continuous trajectories, it still has its drawbacks: Gaussian UMAP is quite uninterpretable (meaning we don't know why from all previous methods this one in particular worked), the trajectories were calculated manually by using Cain's annotations and we don't have an automatic method of deriving these trajectories from this kind of embedded data.

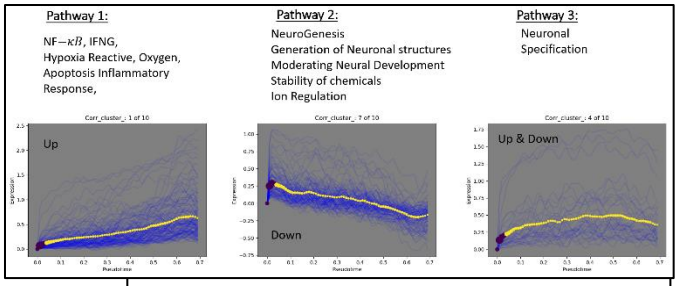


Figure 8.2 3 plots of clustered genes along Pseudotime 1. X axis is Pseudotime, Y axis is gene expression. Blue lines are the genes in this cluster expression along Pseudotime, yellow is mean. Above are names of biological pathways associated with these genes (according to hyper geometric test with low p-value adjusted). Left plot are genes that have high expression as we transition from Homeostatic state to a reactive one (increase of Pseudotime) and is associated with genes that are known pathways of reactive Astrocytes. Middle plot are genes that go down along Pseudotime, and these are genes that are related to Homeostatic Astrocytes. Right plot shows the strength of using mutual information, as it can capture non-linear relationships (namely here, up then down), yet we are unsure the meaning of the pathway. 3D gene-clusters GaussianUMAP)

<u>Our Model</u> - TRIVIA (TRee Induced VarIational Autoencoder): The future of artificial intelligence (AI) has been transformed by deep learning (LeCun, 2015)(DL). It has solved a slew of complicated issues that have plagued the AI community for years. DL models are more complex versions of artificial neural networks (Yegnanarayana, 2009), with both linear and non-linear layers of computation. Because DL models can learn hierarchical features from a variety of datasets (Zhao, 2017), they may be used to solve recognition, regression, semi-supervised, and unsupervised tasks. With these advancements in mind, we sought to use deep learning's newly found potential to create deep learning tools that would embed biological gene expression data in a continuous metric space. The main goal of this method is to provide a more accurate model of cells as a mixture of expression programs and

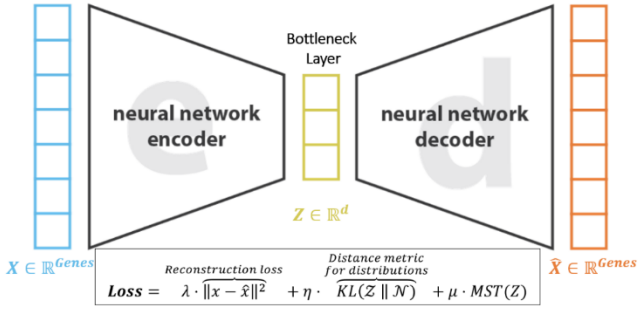
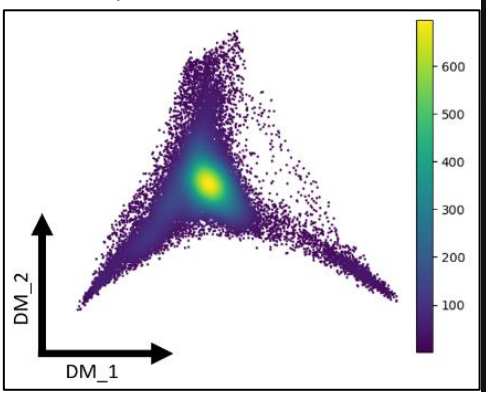


Figure 9. High level overview of the TRIVIA model. overtop in the neural network, and under it is the designed loss function.

infer trajectories of cells between states. Our proposed model which is named TRIVIA (TRee Induced VarIational Autoencoder) is a Variational Auto Encoder that takes in cells in 2000-dimensional gene expression features, and outputs them in an 8-dimensional space using minimum spanning trees (MST) as a "guide" to structure the data cloud in latent space (Outline is in figure 9). Note that using MST as a guide for finding trajectories has been done before by several research endeavors (Ji, 2016) (DeTomaso, 2019). But whilst these methods are usually done in a low 2-dimensional space, we propose using a high dimensional variant (namely in 8 dimensions), to regulate the distribution of the data cloud in this latent space. In a more formal fashion, let our count matrix be $X \in \mathbb{R}^{\#cells \times \#genes}$ $(\#cells \approx 30{,}000 \ , \#genes = 2000)$ and let our network consist of two non-linear transformations: $encode: \mathbb{R}^{\#genes} \to \mathbb{R}^{latent\_dim}$ where in our case we have chosen $latent\_dim = 8$; $decode: \mathbb{R}^{latent\_dim} \to \mathbb{R}^{\#genes}$,Such that the networks function will be denoted by f, and defined by $f := decode \circ encode$. Denote $\hat{X} = f(X)$, and $Z = encode(X)$. To encourage the transformation to preserve the structure of the high dimensional as much as possible, we need to construct a useful loss function. Our loss function consists of three parts: reconstruction loss, which measures how well the network can reconstruct the data from the shrank latent space, namely $\left\|X - \hat{X}\right\|^2$. The second is a measure for the distribution of the data in the latent space. This idea was well described by Diederik P. Kingma& Max Welling in 2013 (Kingma, 2013) and was accomplished by calculating the KL divergence (Kullback & Leibler, 1951) of the latent data against the normal distribution $\mathcal{N}(\mathbf{0}, I)$, namely $D_{KL}(Z \| \mathcal{N}(\mathbf{0}, I))$. The third is a measure of the resulting structure of the data embedded in the latent space. To introduce such an enforcement on the structure of the resulting data cloud we construct a minimal spanning tree (MST) on a set of points derived from Z, and calculate the accumulative distances for each point in Z onto the tree, namely $\sum_{z \in Z} \|z - proj(z)\|^2$ where $proj: \mathbb{R}^{latent\_dim} \to \mathbb{R}^{latent\_dim}$ is the projection operator onto the MST. In addition, a regularization term was added to each part, in order to find the best achievable embedding. In order to visualize the 8-dimensional results we show a Diffusion Map plot in figure 10.1. and as we can see this method was able to produce trajectories in a latent space. Not only do there appear to be trajectories



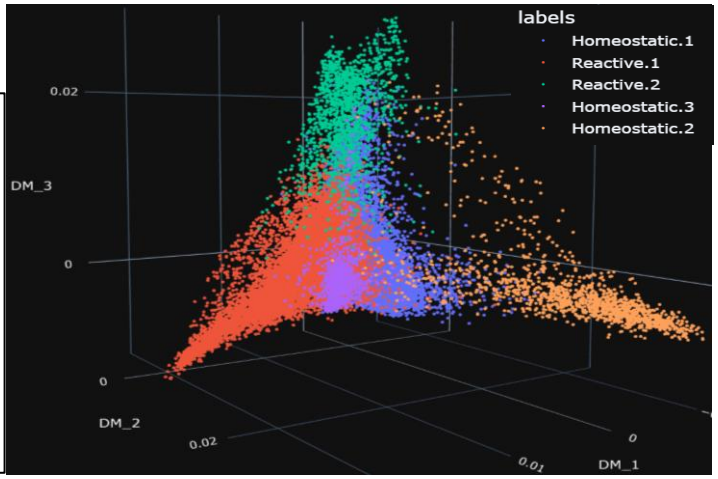


Figure 10.1 Plot of the right is results of TRIVIA model visualized using diffusion maps. Colors are cluster annotations.
Figure 10.2 Plot of the left is a density plot of the big plot. 3D Plot

in this space, but the data cloud does not seem to be as concentrated as it is in diffusion maps (figure 7.2), but rather Cain's annotations are spread throughout each of the 3 trajectories (Figure 10.2) and show a trajectory structure from Homeostatic.1/3 to Reactive.1, Reactive.2 or Homeostatic.2. This indicates this latent space can preserve the cellular diversity of the data, in particular different cell states within Astrocytes. To analyze this method further we can perform pathway analysis similarly to how we did in the Gaussian UMAP section – define as many trajectories as there are edges in the fitted MST on the data cloud in 8 dimensions, and then using these annotations to analyze (using mutual information), and cluster the genes to get biological pathways.

Discussion

In this work we have reviewed several approaches to model cell state transition in human Astrocytes using continuous models. The idea being that cell state transitions are continuous in time and so can be modeled as such, even though the current analysis tools of sNuc-seq data enable accurate modeling mainly on discrete data. We have reviewed language models (Dey, 2017) to conduct soft clustering and assigning a continuous label to each cell for each topic. These models whilst sometimes resulting in quite attractive soft assignments (such as being able to capture the Reactive clusters in a continuous matter), are not always as well-understood (such as topics that seemed to have tried capturing a Homeostatic signal). In addition, these models are uninterpretable, and it is hard to deduce why a certain topic was distributed over the data a certain way or the other and how to determine the optimal number of topics to model the data. Topic modeling lacks the ability to propose a hierarchical structure to the data, which could help with both these issues. We have also tried embedding the data into a continuous latent metric space where the coordinates are supposed to represent the continuous assignment to each cell. The first was Diffusion Maps (Coifman, 2005), which partially captured the diversity and trajectory structure of the data, but we were still unable to assign a meaningful continuous label to the cells. This might be because Diffusion Maps, are sensitive to the distribution of data such that local resolution may be gained or lost (Herring, 2018). Thus, they are largely used for depicting simple topologies that can be derived from the largest variation in the data, with less emphasis on sub-branches, thus might not be well suited to this kind of task. Although it could be that both Topic modeling and diffusion maps capture a similar signal since when running both (link to webpage figure - 3D Diffusion Maps colored in RGB with Topic Modeling with 3 topics, such that each topic takes a single color channel - Diffusion Maps colors by Topic Modeling) we see that similarity in diffusion embedding (special closeness) goes hand in hand with topic scores (similar color palette). The second embedding model we tried was UMAP with a gaussian metric (Narayan Ashwin, 2020), and this was the first model we observed which resulted in trajectory resembling structure. This model led us to the idea that maybe we

should be able to find a lower (and yet not 2d) dimensional representation of the data which will reveal the biological trajectories of the data. We decided to use deep learning and AI as tools toward a new solution to this problem which can be costumed specifically to our purposes. The hypothesis being that the deep model (Zhao, 2017) could result in a new latent embedding space which could be better suited to capture the high-order structure of our data, compared to other methods. This is due to two main reasons: one is the ability of deep models to express a very large space of functions that we could use as our reduction to a latent interpretable space. The second being that we could decide upon any network architecture and any loss function to manipulate the creation of the latent space (Higgins, 2016). This is a very powerful position to be at, since while most dimensionality reduction algorithms (and manifold learning algorithms in general) don't allow the user to manage the interworking of the algorithm (at most some allow the change of some hyper-parameters), engineering a neural network's architecture and designing the loss function, can enable us to use our prior biological knowledge to compose a better suited metric space to embed the data in. As mentioned, our design uses a variation of a variational auto encoder (Higgins, 2016), with a module to consider the structure of the embedded data – it encourages the data to organize into a minimum spanning tree, while trying to enforce a particular distribution upon the embedded data. Recall that our assumption is that the data contains continuous biological trajectories, and so by using MST's we try to encourage the data to embed into these well-defined trajectories (the edges). The embedded representation of the data could be helpful in trying to represent cells as a combination of expression programs and this could lead us to discover pathways and trajectories in the Alzheimer's disease. This algorithm gave promising first results, this is due to the fact that it successfully embedded the cells in the latent space which agreed with the cluster annotations (each cluster is embedded in a distinct space, and they are not mixed randomly). In addition, the cells were arranged in a trajectory like pattern, which could suggest an interpretable underlying expression changes/pattern. Finally, the density of the cells is still quite concentrated in the middle of the graph, but less than what we found in other methods, and so it might reflect the fact that this method was able to capture the diversity of Astrocytes better than previous methods. In conclusion, this algorithm seems to give interesting performance with relation to previous methods, but more analysis is required to determine this with confidence.

Methods

Data Description

As mentioned, the data consists of single nucleus RNA sequencing data from 24 post-mortem human brain tissue. The data is both from healthy patient, as well as patient with AD. The data includes different regions

of the human brain such as white matter, medial frontal cortex (MFC), as well as several cell types (Astrocytes, Oligodendrocytes, etc..), where we have focused in this research at Astrocytes in the MFC. Each sample contains between 5k-8k cells. The raw data is of the form of a sparse count matrix $M \in R^{\#cells \times \#genes}$, consisting of rows representing cells captured, and columns representing the approximately 20,000 possible genes that could be expressed in human cells. In addition, the metadata includes information about the patients such as: sex, age, AD/WT, cognitive decline (4 levels), pathology of disease in brain tissue (4 levels), etc...

Single nuclei analysis by Anael Cain

The data also included various analyses previously done by Anael Cain. The data was analyzed using Seurat's R package. Firstly, it contains the log normalization of the count matrix, and scaling the data. In addition, it contains feature selection of the top $2{,}000$ variable genes out of the possible $\approx 20{,}000$ genes. In my analysis I mostly used the variable scaled data – non sparse matrix $\in \mathbb{R}^{\#cells \times 2000}$. Over this data Cain also preformed principal component analysis (PCA) to 64 dimensions, and performed Leiden community clustering, which after intense inspection and fine tuning of the hyper-parameters, resulted in the 5 annotated clusters I started with. In addition, the visualization in figure 2 of the data in UMAP plot was also done by Cain.

Topic Modelling

As already mentioned, A topic model is a statistical model used in machine learning and natural language processing (NLP) to find the abstract "topics" that appear in a set of texts. Topic modeling is based on Latent Dirichlet allocation (LDA) and was first used for unsupervised topic discovery in NLP applications. Our use models a cell's captured transcriptome as a document. LDA employs sparse Dirichlet prior distributions over document-topic and topic-word distributions to encode the intuition that documents cover a limited number of subjects, and those topics frequently utilize a limited number of words. The algorithm outline is in figure 11. In ours case we have tried many different numbers of topics, but have shown here the run with 5 topics, as we tried to maintain $\#topics = \#clusters$. The topic modeling algorithm we used is by CountClust implementation in R.

(Assume there are k topics in each document, like clustering) Run for t iterations:
∀ document $m \in$ Documents:
1. Distribute $(\sim\alpha)$ these k topics across document m.
2. ∀ word w in document m, assume its topic is wrong but every other word is assigned the correct topic.
3. Probabilistically $(\sim\beta)$ assign word w a topic.

Figure 11. Overview of the Topic Modeling algorithm.
α is a matrix where each row is a document, and each column represents a topic. A value in row i and column j represents how likely document i contains topic j.
β is a matrix where each row represents a topic, and each column represents a word. A value in row i and column j represents how likely that topic i contains word j.
Usually, each word is distributed initially evenly throughout the topic such that no topic is biased towards certain words.

Diffusion Maps

Coifman and Lafon invented the diffusion mappings algorithm (Coifman, 2005), which computes a series of embeddings of a data set into Euclidean space (typically low-dimensional) whose coordinates may be calculated from the eigenvectors and eigenvalues of a diffusion operator on the data. The "diffusion distance" between probability distributions centered at those locations is equivalent to the Euclidean distance between points in the embedded space. Intuitively, the distance between two points is defined as the probability of going through the nodes using t steps. Diffusion maps, unlike linear dimensionality reduction approaches such as principal component analysis (PCA), are part of a family of nonlinear dimensionality reduction methods that focus on identifying the underlying manifold of the distribution of the data. Diffusion maps provide a global representation of the data set by merging local similarities at various sizes. The diffusion map methodology is resilient to noise disturbance and computationally economical when compared to other methods. A simplified version of diffusion map algorithm is as follows:

1. Compute the matrix $K_{i,j} = \exp\left(\frac{-d(x_i,x_j)^2}{\varepsilon}\right)$, when d is some metric on X and $x_i, x_j \in X$, $\varepsilon \in \mathbb{R}$.
2. Normalizing K, such that $P = \frac{K}{\sum_i K_{i,j}}$
3. Diagonalize P and sort eigenvalues and corresponding left eigenvectors in descending order.
4. The set of n eigenvectors $\psi_1, \dots, \psi_n$ span a space of reduced dimensionality (if n < N) in which the dataset can be efficiently represented.

Diffusion maps algorithm used was in R using the destiny package written by Philipp Angerer which is available on Bioconductor.

Archetypes

A basic trade-off exists in biological systems that execute numerous functions: a single phenotype cannot be optimal at all tasks. The Archetype approach explores how trade-offs influence the diversity of phenotypes observed in nature. The best–trade-off phenotypes are weighted averages of archetypes—phenotypes specialized for single tasks—using the Pareto front idea from economics and engineering. Phenotypes are found on the line connecting the two archetypes for two tasks, which could explain linear trait correlations, allometric relationships, and gene expression patterns. Given a trait (phenotype) vector v,our Goal is: Optimizing a fitness function F across multiple performance tasks: $F(P_1(v), \dots, P_n(v))$. Denote the phenotype who is best at task i : $arch_i$ Redefine our trait vector v as a linear combination of the archetypes: $v = \sum_{i=1}^{n} \lambda_i \cdot arch_i$,where the weights λ_i are calculated as follows: $\lambda_i = \frac{\frac{\partial F}{\partial P_i} \cdot \frac{\partial P_i}{\partial d(v, arch_i)}}{\sum_{j=1}^{n} \frac{\partial F}{\partial P_j} \cdot \frac{\partial P_j}{\partial d(v, arch_j)}}$

In our use case we could think of a cell as a trait vector (where the traits are the gene expressions), and so the resulting archetypes should represent different cell states of astrocytes such that each cell can be represented as a linear combination of these cell states. Archetype algorithm was written in Matlab by Uri Alon from the Weizmann institute.

Gaussian UMAP

By default, UMAP (Becht, 2019) embeds data into Euclidean space, however, there aren't really any major constraints that prevent the algorithm from working with other more interesting embedding spaces. Consider is the space formed by 2d-Gaussian distributions: Let G be a metric space over $\mathbb{R}^d$ with the following metric: $d: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d s.t\, \forall x, y \in \mathbb{R}^d$

$$d(f,g) = -\log(\langle f,g\rangle) = \overbrace{-\log\left(\int_{-\infty}^{\infty} f(x)\cdot g(x)\,dx\right)}^{\text{we measure the distance between two Gaussians by the negative log of the inner product between the PDFs.}}$$

That gives us a metric space to embed into where samples are represented not as points in 2d, but as Gaussian distributions in 2d, encoding some uncertainty in how each sample in the high dimensional space is to be embedded. We can parameterize the covariance in terms of a width, height, and angle, and recover the covariance matrix from these if required. That gives us a total of 5 components to embed into (two for the mean, 3 for parameters describing the covariance). We can do this since the appropriate metric is well-defined already, and then run the UMAP algorithm as usual. Gaussian UMAP was run in python 3.9 by the UMAP package.

Variational Autoencoder

Autoencoders are an unsupervised learning technique in which we leverage neural networks (Yegnanarayana, 2009) for the task of representation learning. Specifically, the neural network architecture includes a bottleneck layer which imposes a compressed representation of the original input. The assumption is that although the data is high dimensional, an intrinsic structure exists in the data (for example correlations between input features), this structure can be learned and consequently leveraged when forcing the input through the network's bottleneck. This type of network can be trained by minimizing the reconstruction error, which measures the differences between our original input and the consequent reconstruction. A bottleneck constrains the amount of information that can traverse the full network, forcing a learned compression of the input data. A variational autoencoder (VAE) is the artificial neural network architecture introduced by Diederik P Kingma and Max Welling (Kingma, 2013), belonging to the families of probabilistic graphical models and variational Bayesian methods. It could be viewed as an

extension of the autoencoder model because of its architectural affinity, but there are significant differences both in the goal and in the mathematical formulation. Variational autoencoders are meant to compress the input information into a constrained multivariate latent distribution (encoding) to reconstruct it as accurately as possible (decoding). Just as a standard autoencoder, a variational autoencoder is an architecture composed of both an encoder and a decoder and that is trained to minimize the reconstruction error between the encoded-decoded data and the initial data. However, instead of encoding an input as a single point, we encode it as a distribution over the latent space. This is what makes possible to express very naturally the latent space regularization - the distributions returned by the encoder are enforced to be close to a standard normal distribution. The loss function is defined as a combination of the reconstruction loss as well as a regularization term which is expressed as the Kulback-Leibler divergence (D_{KL}) between the returned distribution and a standard Gaussian distribution. More formally, denote our input dataset X characterized by an unknown probability function $P(X)$ and a multivariate latent encoding z, the objective is to model the data as a distribution $P_\theta(X)$, with θ being the set of the network parameters. Using the chain rule with total probability over $P_\theta(z)$ we can formulate $P_\theta(X) = \int_z P_\theta(X|z) \cdot P_\theta(z)dz$, where we can impose the prior $P_\theta(z)$ to be a "simple" function (in our case to be a Gaussian), and $P_\theta(z|X)$ is hard to calculate and so we shall introduce a further function to approximate the posterior distribution as $Q_\psi(z|X) \approx P_\theta(z|X)$. Thus, the overall problem can be easily translated into the autoencoder domain, in which the conditional likelihood distribution $P_\theta(z|X)$ is carried by the probabilistic decoder, while the approximated posterior distribution $Q_\psi(z|X)$ is computed by the probabilistic encoder. We would like to calculate the likelihood of the data according to P_θ:

$$\log P_\theta(X) = \mathbb{E}_{Q_\psi(z|X)}[\log P_\theta(X)] = \mathbb{E}_{Q_\psi(z|X)}\left[\log \frac{P_\theta(z,X)}{P_\theta(z|X)}\right] = \mathbb{E}_{Q_\psi(z|X)}\left[\log \frac{P_\theta(z,X)}{P_\theta(z|X)} \cdot \frac{Q_\psi(z|x)}{Q_\psi(z|x)}\right] =$$

$$\overbrace{\mathbb{E}_{Q_\psi(z|X)}\left[\log \frac{P_\theta(z,X)}{Q_\psi(z|x)}\right]}^{\mathcal{L}_{\theta,\psi}(X)=\text{Evidence Lower Bound (ELBO)}} + \overbrace{\mathbb{E}_{Q_\psi(z|X)}\left[\log \frac{Q_\psi(z|x)}{P_\theta(z|X)}\right]}^{D_{KL}(Q_\psi(z|x)\|P_\theta(X|z))}$$

We can rearrange the terms and get that the loss function is: $\mathcal{L}_{\theta,\psi}(X) = D_{KL}(Q_\psi(z|x)\|P_\theta(z|X) - \log P_\theta(X)$.

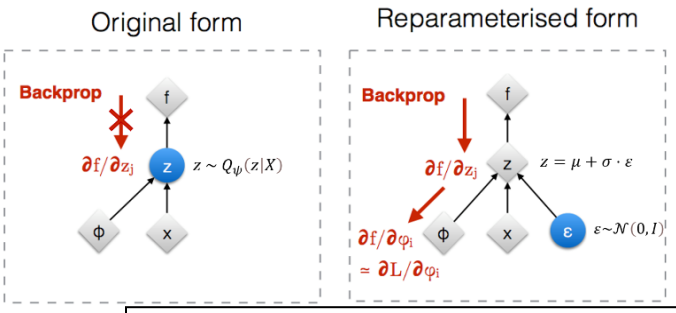


Figure 12. Diagram depicting the reparameterization which allows for backpropagation to work through probabilistic events. Blue nodes are random nodes, whereas grey nodes are standard nodes.

A problem arises when we try to formulize a backpropagation step on this formulation and observe that this it relies on propagating though a probabilistic random node Z (since we rely on sampling from a random node Z which is approximated by the parametric model $Q_\psi(z \mid X)$ of the true posterior). Introducing a new parameter ε allows us to reparametrize z in a way that allows backprop to flow through the deterministic nodes (Figure 12). The

main assumption about the latent space is that it can be considered as a set of multivariate Gaussian distributions, and thus can be described as $z \sim Q_\psi(z|X) = \mathcal{N}(\mu, \sigma^2)$, and given $\varepsilon \sim \mathcal{N}(0, I)$ we can reparametrize it as $z = \mu + \sigma \cdot \varepsilon$. This step makes the gradient descent possible despite the random sampling that occurs halfway of the architecture.

TRIVIA (*TR*ee *I*nduced *V*ar*I*ational *A*utoencoder)

Recall that $X \in \mathbb{R}^{\#cells \times genes}$, is our dataset, $f: \mathbb{R}^{\#genes} \to \mathbb{R}^{\#genes}$ is the composition of the $encode, decode$ functions. We will also denote $Z = \{encode(x) | x \in X\}$. In our model (a minimum spanning tree induced variational autoencoder), we employ a variant of the variational autoencoder, namely, modifying the loss function to impose a tree structure on the latent embedding. Given a representation of the data as a weighted graph, an MST is the tree structure that connects all the nodes in a way that has the minimum total edge weight (Euclidean distance in latent space). The MST trajectory inference method is based on the idea that nodes (cells/clusters of cells) and their connections can represent a spectrum of change – in our case, a change in cellular states. In order to calculates an MST on the data cloud, we need to define nodes through which we will construct the MST. The nodes are constructed though high-resolution clustering, namely using K-means. The idea being that closely related neighbors in latent space should have a high similarity, and so can be represented by a single node – the cluster centroid. The weights between nodes are calculated by Euclidean distance in latent space, and the tree structure will be determined by K nearest neighbors (KNN) with $K = 10$, such that the resulting tree will be connected (a single connected component). The Prim algorithm (Prim, 1957) will be computed to obtain the MST. Finally, all the cells are projected to the nearest point on the trajectory (edge), creating the final ordering. Formally let us define a function $MST: \mathbb{R}^{\#cells \times latent\_dim} \to \mathcal{G} = (V, E)$ as the process described above (where $V = \{C_i\}_{i=1}^K \subset \mathbb{R}^{latent\_dim}$ are the cluster centroids mentioned, and $E \subset V \times V$ are the MST edges). We will also denote $MST_{points} = \{x | x \text{ can be represented as a convex combination of 2 points in } V\}$ as the set of feasible points in the embedded graph $\mathcal{G}$ in $\mathbb{R}^{latent\_dim}$. The projection operator $proj: \mathbb{R}^{latent\_dim} \to \mathbb{R}^{latent\_dim}$ is calculated thusly: $proj(z) = argmin_{z' \in MST_{points}} \|z - z'\|^2$ meaning the projection is the closest point in the embedded MST graph. We will also denote the $MSE_{loss}(Z) = \sum_{z \in Z} \|z - proj(z)\|^2$, which is the cumulative projection loss (cumulative projection distances). In our model the data X in propagated through the variational autoencoder, and the backpropagation step involves calculating the loss function thusly:

TRIVIA model outline

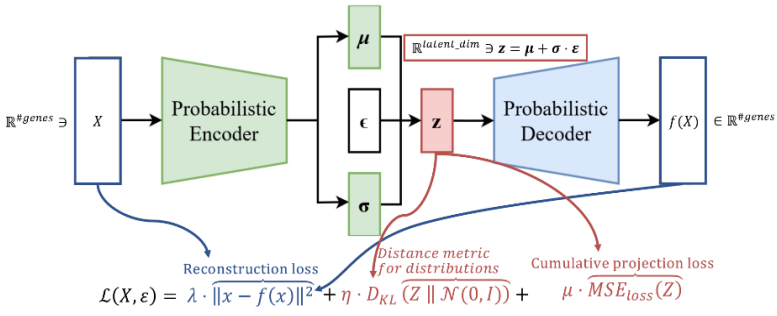


Figure 13. Detailed view of the TRIVIA network design, as a combination of the VAE architecture as well as the addition of the MST imposition.

$$\mathcal{L}(X,\varepsilon) = \overbrace{\|x - f(x)\|^2}^{\text{Reconstruction loss}} + \eta \cdot \overbrace{D_{KL}(Z \,\|\, \mathcal{N}(0,I))}^{\text{Distance metric for distributions}} + \overbrace{\mu \cdot MSE_{loss}(Z)}^{\text{Cumulative projection loss}}$$

Where η, μ are regularization terms, which were decided during the hyper-parameter search.

Hyper-Parameter Optimization

Batch size	η	μ	V_{layers}	lr	K_{kmeans}	K_{KNN}
5020	0.1	0.1	[2000, 512, 64, 8]	0.01	**10**	**10**
15046	**1**	**1**	**[2000, 512, 8]**	**0.001**	100	100
	100	100	[2000, 128, 8]			
	10000	10000	[2000, 1024, 128, 8]			

Table 2. Potential Values for hyper parameter optimization. Optimal values as per grid search are bold.

As in any AI optimization problem, we had to figure out which hyper parameters would lead our model TRIVIA to its best performance. In addition to the regularization parameters η, μ which related to the weight of distribution loss, and structure loss respectively, we also needed to find standard hyper parameters for the network itself – number of layers V_{layers}, learning rate lr, batch size N_{batch_size}. In addition, we also had parameters regarding the creation of the MST on the latent space – namely number of clusters for K-means K_{kmeans}, and number of neighbors for KNN K_{KNN}. In a standard neural network, the next step would be to see which combination would result in a minimal loss overall, however in this problem we have regularization parameters which will affect the final loss score. This a well-known problem in generative models, and there have been proposed solutions (for example The inception score (Salimans, 2016)), unfortunately these methods would not work on tabular data such as sNuc-seq data. Recalling that a big hallmark of a well-executed dimensionality reduction is the retention of closest neighbors, our idea was running a grid search over a certain range each of the parameters, and checking which combination resulted in the best neighbor retention. We have chosen to evaluate this score in the 8-dimensional latent space against the neighbors in the 64-dimensional PCA projection, since we believe this is a rather unbiased analysis which should represent a good measure of the neighbor makeup of the data. The range of parameters is detailed in table 1 and has totaled in training $2 \times 4 \times 4 \times 4 \times 2 \times 2 \times 2 = 1024$ networks for 1000 epochs. Observe that we also have the cluster annotations which we consider approximating the ground truth, and we would like to be able to use then as a validation step in this optimization step as well. The problem is that these are discrete annotations thus any validation metric using these values would be a noisy (since the cluster boundaries are definite while they should be fuzzy, and so the boundary is quite arbitrary) approximation. Nevertheless, we wanted to use some metric to give each hyper parameter combination a score based on these annotations. One such well known metric is The Silhouette Coefficient or silhouette score. The silhouette score is a metric used to calculate the goodness of a clustering technique. Its value ranges from -1 to 1. Formally let C_i $1 \le i \le 5$ be the clusters, and for $j \in C_i$ let $a(j) = \frac{1}{|C_i|-1}\sum_{k \in C_i, j \neq k} d(j,k)$, for C_t $b(j) = \min_{t \neq i} \frac{1}{|C_t|}\sum_{k \in C_t} d(j,k)$, where $d(i,j)$ is the Euclidean distance. Then the silhouette score is $s(j) = \frac{b(j)-a(j)}{\max\{a(j),b(j)\}}$. Interestingly enough both the silhouette score

and the neighbors score agreed on the best hyper parameter combination mentioned in table 2. The network with the best design according to these tests is the one mentioned in the results section.

References

A. Kraskov, H. S. (2004). Estimating mutual information. *Phys. Rev*.

Anders M. Fjell, L. M. (2014). What is normal in normal aging? Effects of aging, amyloid and Alzheimer's disease on the cerebral cortex and the hippocampus,. *Progress in Neurobiology,*, 20-40.

Becht, E. M. (2019). Dimensionality reduction for visualizing single-cell data using UMAP. *Nat Biotechnol* , 38–44.

Blei, D. M. (2003). Latent dirichlet allocation. *the Journal of machine Learning research*, 993-1022.

Burda, J. E. (2014). Reactive gliosis and the multicellular response to CNS damage and disease. *Neuron*, 229-248.

Butler, A. H. (2018). Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat. Biotechnol*, 411–420.

Cain, A. T. (2020). Multi-cellular communities are perturbed in the aging human brain and with Alzheimer's disease. *bioRxiv*.

Coifman, R. R. (2005). Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the national academy of sciences*, 10.

DeTomaso, D. J. (2019). Functional interpretation of single cell similarity maps. *Nat Commun*, 10.

Dey, K. K. (2017). Visualizing the structure of RNA-seq expression data using grade of membership models. *PLoS genetics*.

Ding, J. A. (2020). Systematic comparison of single-cell and single-nucleus RNA-sequencing methods. *Nat Biotechnol* , 737–746.

Habib, N. A.-D. (2017). Massively parallel single-nucleus RNA-seq with DroNc-seq. *Nature methods*, 955-958.

Habib, N. L.-D. (2016). Div-Seq: Single-nucleus RNA-Seq reveals dynamics of rare adult newborn neurons. *Science*, 925-928.

Habib, N. M. (2020). Disease-associated astrocytes in Alzheimer’s disease and aging. *Nat Neurosci* , 701–706 .

Hartmann, T. (1999). ntracellular biology of Alzheimer’s disease amyloid beta peptide. *European archives of psychiatry and clinical neuroscience*, 291-298.

Herring, C. A. (2018). Unsupervised trajectory analysis of single-cell RNA-seq and imaging data reveals alternative tuft cell origins in the gut. *Cell systems,*.

Higgins, I. M. (2016). beta-vae: Learning basic visual concepts with a constrained variational framework.. *openreview*.

Jelodar, H. W. (2019). Latent Dirichlet allocation (LDA) and topic modeling: models, applications, a survey. . *Multimedia Tools and Applications*, 15169-15211.

Ji, Z. &. (2016). TSCAN: Pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis. *Nucleic acids research*, 117.

Kingma, D. P. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv*.

Kiselev, V. A. (2019). Challenges in unsupervised clustering of single-cell RNA-seq data. *Nat Rev Genet*, 273–282.

Kullback, S., & Leibler, R. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, 79–86.

Kushal K. Dey, C. J. (2017). Visualizing the structure of RNA-seq expression data using grade of membership models. *PLoS genetics*.

Laleh Haghverdi, F. B. (2015). Diffusion maps for high-dimensional single-cell analysis of differentiation data. *Bioinformatics*, 2989-2998.

LeCun, Y. B. (2015). Deep learning. *Nature* , 436–444 .

Linderman, G. R. (2019). Fast interpolation-based t-SNE for improved visualization of single-cell RNA-seq data. *Nat Methods*, 243–245.

Macaulay, I. C. (2016). Single-cell RNA-sequencing reveals a continuous spectrum of differentiation in hematopoietic cells. *Cell reports*, 966-977.

Moddemeijer, R. (1989). On estimation of entropy and mutual information of continuous distributions. *Signal processing*, 233-248.

Narayan Ashwin, B. B. (2020). Density-Preserving Data Visualization Unveils Dynamic Patterns of Single-Cell Transcriptomic Variability. *bioRxiv*.

Ngatchou, P. Z.-S. (2005). Pareto multi objective optimization. *In Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems IEEE.*, 84-91.

O. Shoval, H. S. (2012). Evolutionary Trade-Offs, Pareto Optimality, and the Geometry of Phenotype Space. *Science*, 1157 .

Peters, G. C. (2013). Soft clustering–fuzzy and rough approaches and their extensions and derivatives. *International Journal of Approximate Reasoning*, 307-322.

Peters, G. C. (2013). Soft clustering–fuzzy and rough approaches and their extensions and derivatives. . *International Journal of Approximate Reasoning*, 307-322.

Prim, R. C. (1957). Shortest connection networks and some generalizations. *The Bell System Technical Journal*, 1389-1401.

Qiu, X. M. (2017). Reversed graph embedding resolves complex single-cell trajectories. *Nature methods*, 979-982.

Salimans, T. G. (2016). Improved techniques for training gans. . *Advances in neural information processing systems*, 2234-2242..

Sims-Robinson, C. K. (2010). How does diabetes accelerate Alzheimer disease pathology. *Nat. Rev. Neurol*, 551–559.

Sloan SA, B. B. (2014). Mechanisms of astrocyte development and their contributions to neurodevelopmental disorders. *Current Opinion in Neurobiology*, 75–81.

Tang, X. H. (2019). The single-cell sequencing: new developments and medical applications. *Cell & bioscience*, 1-9.

Traag, V. A. (2019). From Louvain to Leiden: guaranteeing well-connected communities.. *Scientific reports*, 1-12.

Trapnell, C. (2015). Defining cell types and states with single-cell genomics. *Genome research*, 1491-1498.

Van Der Maaten, L. P. (2009). Dimensionality reduction: a comparative.. *J Mach Learn Res*, 66-71.

Verkhratsky, A. &. (2013). *Glial physiology and pathophysiology.* John Wiley & Sons..

Yegnanarayana, B. (2009). *Artificial neural networks.* PHI Learning Pvt. Ltd..

Zhao, S. S. (2017). Learning hierarchical features from deep generative models. *In International Conference on Machine Learning* , 4091-4099.