Computational Network Science

RWTH AACHEN UNIVERSITY

Bachelor Thesis at the Department of Computer Science
RWTH Aachen University

September 4, 2023

Multilayer Network Embeddings for Clustering Spatial Transcriptomics and Genomics Data for Myocardial Infarct Detection

Robert Maximilian Giesler

Computational Network Science Group
Prof. Michael Schaub

Supervisor: Damin Kühn
First Corrector: Prof. Michael Schaub
Second Corrector: Prof. Ivan Costa

Abstract

Clustering is a fundamental technique in data analysis, enabling the identification of inherent structures, patterns, and irregularities within complex datasets. As data becomes increasingly multi-faceted, there is a growing need for methods capable of clustering multilayer networks, which offer rich representations of complex entities with multi-dimensional features. This is a challenging problem with applications across numerous domains.

In this thesis, we introduce *HeartNet*, a modular, domain-agnostic computational pipeline designed for the clustering of multilayer networks based on structural similarities. Utilizing the HeNHoE-2vec algorithm for multilayer network embedding and Gromov-Wasserstein optimal transport for pairwise distance calculations between embeddings, HeartNet provides a robust and generalizable solution for multilayer network clustering. To evaluate its performance and applicability, we focus on a motivating application: the clustering of human heart tissue samples into their respective zones of ischemic heart disease. The single-cell RNA sequencing and spatial transcriptomics data of the heart tissue samples are integrated in multilayer network representations, which are clustered using HeartNet. Experimental results demonstrate HeartNet's capability to accurately and reliably cluster myogenic and ischemic samples, while also identifying outliers that deviate from typical clustering patterns. However, HeartNet faces limitations in discriminating fibrotic samples and distinguishing between finer zones within the class of myogenic samples.

Despite challenges related to computational intensity and hyperparameter tuning, HeartNet's modular design and domain-agnostic approach make it a versatile and promising method for the analysis of complex multilayer networks in various fields.

Contents

1. Introduction

1.1. Motivation

Single-cell RNA sequencing (scRNA-seq) is a breakthrough technology in cellular biology that enables the analysis of gene expression profiles of individual cells. While traditional bulk RNA sequencing methods measure average gene expression levels across thousands or millions of cells in homogenized tissue samples, scRNA-seq provides high-resolution insights into the gene expressions of individual cells. This enables researchers to capture and examine the diversity of cell types, states, and gene expressions in complex tissue samples [1–3].

Since the first experiments with scRNA-seq in 2009 [4], which could only process a few cells and produced noisy and incomplete data, the technology has been the focus of intense research and development to increase throughput, improve accuracy, and reduce costs [5]. As a result, modern scRNA-seq methods can accurately and efficiently measure gene expression in tens of thousands of cells simultaneously, allowing researchers worldwide to gain unprecedented insights into cellular heterogeneity in different species, targeted cancer therapy, disease development, and many other areas [6–8].

While scRNA-seq provides high-resolution genetic information for individual cells, it does not provide any spatial information regarding the location of cells in the tissue sample. Spatial transcriptomics (ST), on the other hand, captures both spatial and genetic information to generate spatially resolved gene expression data. The spatial context of cells in tissue is critical information in many applications. ST enables researchers to leverage this information for the study of tissue architecture and organization, functional relationships between cells and their microenvironment, and spatially defined gene expression patterns [3]. In ST, the target tissue is divided into hundreds or thousands of spots which each contain 1-10 cells. The resolution of cells per spot depends on the cell types residing in the tissue and the ST method applied. For each spot, ST provides the *average* gene expression level of all cells in that spot, resulting in a genetic map of the histological sample. Current state-of-the-art approaches, however, do not consistently reach single-cell resolution in spots, resulting in averaged transcriptomes which may obscure important biological differences between individual cells [3].

In a recent study, Kuppe et al. [8] use both scRNA-seq and spatial transcriptomics along with other modalities to provide a comprehensive spatially resolved characterization of gene regulation of the human heart in homeostasis and after myocardial infarction. To take advantage of both the single-cell resolution gene expression patterns from scRNA-seq and the spatial information provided by ST, the authors perform cellular deconvolution, where the cell composition of each spot is estimated based on the single-cell gene expression profiles. This results in high-resolution gene expression data in the spatial context of the cells, allowing the authors to gain a series of novel insights into cell types, states, and interactions in the disease progression context.

Comprising 31 heart tissue samples from 23 individuals, the study performed by Kuppe et al. [8] delivers the largest cohort of spatially resolved genetic data from heart tissues to date. Given that the study includes samples from multiple different zones of ischemic heart disease (IHD), the data collected in this context promises novel insights into the causes, signs, and effects of myocardial infarction at a genetic level. Motivated by the availability of this data, we aim to demonstrate genetic differences between different zones of IHD by representing the genetic information of each tissue sample as a *multilayer network* and clustering said networks into their distinct zones. In the following, we provide a definition of multilayer networks which we will use throughout this thesis.

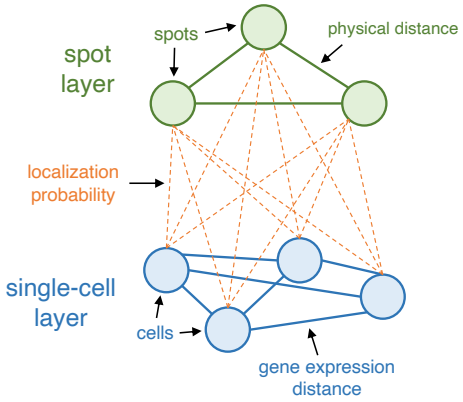


Figure 1.1. | Schematic illustration of the multilayer network representation of the genomics data of a heart tissue sample.

1.2. Multilayer Networks

We define a multilayer network as $\mathcal{M} = (\{G_1, \ldots, G_m\}, E_{IL}, w_{IL})$, where $m \in \mathbb{N}$ is the number of layers, $\{G_1, \ldots, G_m\}$ is a collection of networks, E_{IL} is a collection of *inter-layer* edges, and w_{IL} is the *inter-layer* edge weight function. $G_l = (V_l, E_l, w_l)$ models the lth layer of the network and consists of a set of nodes V_l, the *intra-layer* edges $E_l \subseteq \{(u, v) \mid u, v \in V_l\}$, and the *intra-layer* edge weight function $w_l : E_l \to \mathbb{R}$ that describes the weights of the edges in layer l. The collection of *inter-layer* edges $E_{IL} \subseteq \{(u, v) \mid u \in V_l, v \in V_{l'}, l, l' \in \{1, \ldots, m\}, l \neq l'\}$ and the *inter-layer* edge weight function $w_{IL} : E_{IL} \to \mathbb{R}$ describe the relationships between nodes in different layers[1].

For the main application of multilayer networks throughout this thesis, namely the representation of genomic data from heart tissue samples, we will work with *undirected* multilayer networks. In undirected multilayer networks, edges indicate bidirectional relationships between nodes, i.e., edges can be traversed in both directions. Where appropriate, we will discuss the generalization of the proposed methods and algorithms to *directed* multilayer networks, where edges indicate unidirectional relationships between nodes, which may be more suitable in other applications.

It is important to distinguish between multilayer networks and *multiplex* networks, which are sometimes also referred to as multilayer networks in the literature. A multiplex network is a specific type of multilayer network where all layers share the same set of nodes but have different types of edges. Each layer thereby represents a specific type of interaction between the same set of nodes. This is a simpler, constrained form of the more general *multilayer* network where different layers may have different node sets.

1.3. Objectives and Outline

NovoSpaRc [9, 10] is a cellular deconvolution method that can be applied to the genomics data collected by Kuppe et al. [8] to probabilistically map the single cells to the spots in each sample (novoSpaRc is discussed in further detail in Section 2.1). The resulting mapping can be represented as a multilayer network with two layers — the single-cell layer and the spot layer — where the *inter-layer* edge weights are given by the probabilities of cells appearing in spots, and the *intra-layer* edge weights in the single-cell layer and the spot layer are given by gene expression distance between cells and physical distance between spots, respectively. This is the basic concept of how the genomics data of a heart tissue sample may be represented as a multilayer network, and we provide more detail in Section 3.1.2. See figure 1.1 for a schematic illustration.

The heart tissue samples collected by Kuppe et al. [8] stem from multiple different zones of ischemic

[1]Notation: we write $w(u, v)$ instead of $w((u, v))$ to refer to the weight $w(e)$ of an edge $e = (u, v)$.

heart disease, namely *remote zone*, *border zone*, *fibrotic zone*, *ischemic zone*, and *control*. Applying the steps outlined above provides us with multilayer network representations of the spatialized gene expression profiles of heart tissue samples from different zones of IHD.

The overarching objective of this thesis is to develop a method of *clustering* the multilayer networks in a manner that differentiates tissue samples taken from different zones of IHD. Our proposed multilayer approach offers a novel opportunity to leverage the relationship between scRNA-seq and spatial transcriptomics data for this purpose. If such a clustering can be performed, it suggests that there are distinct genetic markers and/or cell distributions in heart tissue samples, which are indicative of the progression stage of IHD. Further research may reveal novel opportunities for the targeted prediction, prevention, and treatment of ischemic heart disease at the genetic level.

This objective can roughly be broken down into the following work steps:

1. Data preprocessing
2. Embedding the multilayer networks into lower dimensional spaces
3. Defining distances between embeddings using Gromov-Wasserstein optimal transport
4. Clustering the embeddings

We discuss each of these steps in detail throughout this thesis.

Although we develop our methods with a specific application in mind, the clustering of multilayer networks is a task that extends to any scenario where complex entities that can be represented as multilayer networks of measured features need to be clustered. Furthermore, there might be additional modalities of single-cell data in the future which we would like to be able to incorporate into our multilayer networks as additional layers, thereby increasing their expressiveness. Motivated by these considerations, we aim to develop our methods in a way such that they generalize easily to other settings where networks may be directed and/or consist of a different number of layers.

The remainder of this thesis is structured as follows: In Section 2, we briefly summarize related work. In Section 3, we provide a detailed review of background concepts, methods, and algorithms that are essential to the understanding of this work. Section 4 focuses on the process of developing and implementing the methods required to fulfill the objectives of this thesis. We present and evaluate the results of our work in Section 5, and, finally, we conclude in Section 6.

2. Related Work

In the literature, the term "clustering multilayer networks" is often used to refer to the process of clustering the nodes within a single multilayer network. In the context of this thesis, however, we use the term "clustering multilayer networks" to refer to the process of grouping multiple multilayer networks based on structural or functional similarities between them. To the best of our knowledge, there is currently no available work that explores this form of clustering multilayer networks. In this section, we discuss two approaches, NovoSpaRc [9, 10] and SCOT [11], which are both closely related to the present work in the methods they apply and their area of application.

NovoSpaRc [9, 10] is a cellular deconvolution method that probabilistically assigns cells to tissue locations, and is used in the preprocessing stages which lead up to our work to probabilistically map cells to spots. The authors use Gromov-Wasserstein optimal transport (GW-OT) to define distance measures between distributions in different spaces. GW-OT also plays a significant role in our work.

SCOT [11] is another approach from the single-cell biology domain that utilizes GW-OT, in this case, to align single-cell multi-omics datasets that reside in different domains.

2.1. NovoSpaRc

With novoSpaRc, Nitzan et al. [9, 10] introduce a computational framework that aims to recover the spatial organization of cells and genes in their tissue-of-origin based on single-cell data. The authors argue that the physical context of cells is vital for the understanding of biological function at the global collective scale. Furthermore, spatial information is critical at the local level to study cell-cell interactions and individual cellular states [10]. They address the need to decipher spatial information from the vast amounts of single-cell data that already exists and to efficiently integrate scRNA-seq data with complementary high-quality spatial transcriptomic data.

NovoSpaRc is based on the hypothesis that physically neighboring cells share similar transcriptional profiles, so that gene expression, on average, does not change abruptly but in a continuous manner for a substantial subset of genes [10]. In its most basic form, novoSpaRc uses Gromov-Wasserstein optimal transport to reconstruct the spatial organization of cells in their tissue-of-origin based only on the scRNA-seq data. In this process, cells are probabilistically mapped to spots such that the Gromov-Wasserstein discrepancy [12, 13] between pairwise distances of cells in gene expression space and pairwise distances of locations in physical space is minimized. In other words, the mapping of cells to spots is optimized such that cells with similar gene expressions are mapped with high probability to spots that are close together. We explain GW-OT in more detail in Section 3.3.

As an optional input, novoSpaRc can also be provided with a reference atlas that carries information about the expression levels of a subset of genes across the target space. The incorporation of such a reference atlas can improve the reconstruction quality as it encourages the resulting mapping to be consistent with the atlas. The reference atlas may, for example, take the form of spatial transcriptomic data which provides average gene expression levels for each spot, resulting in a meaningful integration of scRNA-seq and ST data.

NovoSpaRc also offers the possibility to incorporate other prior biological knowledge such as marginal distributions of cells in the scRNA-seq data, allowing for nonuniform mappings from cells to spots.

2.2. SCOT

Demetci et al. [11] outline the importance of multi-modal single-cell measurements in understanding cell development and disease. Integrating multiple single-cell measurements is, however, challenging

because it is often impossible to obtain multiple types of measurements from the same individual cell, and because different measured properties are usually defined over different domains. This highlights the need for data integration methods that are capable of combining disparate data types. Demetci et al. [11] therefore propose Single-Cell alignment using Optimal Transport (SCOT), an unsupervised learning algorithm that uses Gromov-Wasserstein optimal transport to align single-cell multi-omics datasets.

Similar to novoSpaRc, SCOT computes a probabilistic mapping from cells in one domain to cells in the other domain, minimizing the Gromov-Wasserstein discrepancy between pairwise distances of cells in the first domain and pairwise distances of cells in the second domain. In other words, the mapping of cells is optimized such that cells that are similar in one domain are mapped with high probability to cells that are likewise similar in the other domain.

3. Background

In this section, we introduce previous work that acts as a basis for this thesis, and we discuss concepts, methods, and algorithms that are essential to the understanding of this work.

3.1. Human Myocardial Infarction Genomic Data

The main motivation for this work and the basis for the evaluation of the methods we propose is the multi-omic data of heart tissue at different stages of IHD provided by Kuppe et al. [8]. This data was preprocessed by our collaborators at the Institute for Computational Genomics of RWTH Aachen University Hospital [14] to produce the multilayer networks that we work with throughout this thesis. In the following, we outline the structure of the original data and give an overview of the preprocessing performed by the Institute for Computational Genomics.

3.1.1. Original Data

Kuppe et al. [8] collected a total of 31 heart tissue samples from 23 individuals, including four non-transplanted donor hearts as controls. From patients with acute myocardial infarction, samples were taken from four different zones relating to different stages of the disease. These zones are the ischemic zone (immediate location of oxygen-deprived tissue), the remote zone (unaffected by ischemia), the border zone (between ischemic and remote zone), and the fibrotic zone (scar tissue formed in the healing process after myocardial infarction), as illustrated in.

From each tissue sample, three sub-samples (which may be viewed as identical) were taken, one of which was analyzed using scRNA-seq, one using spatial transcriptomics, and one using single-nucleus assay for transposase-accessible chromatin sequencing (snATAC-seq). The snATAC-seq data lies beyond the scope of this thesis and is not considered further. The result of this analysis is a dataset containing scRNA-seq and ST data from 31 heart tissue samples at various stages of IHD.

For each tissue sample, scRNA-seq produces a cell-by-gene expression matrix where each entry is the count of RNA molecules per cell [10] (Table 3.1), and ST produces a mapping of the average gene expression in each spot to the spot's physical location on the histological sample. The spots can be clustered by similarity of gene expression and then different clusters can be visualized by color as exemplified in Figure 3.2. Kuppe et al. evaluated that there are an average of four cells per ST spot [8].

3.1.2. Data Preprocessing

Our collaborators from the Institute for Computational Genomics at RWTH Aachen University Hospital [14] preprocessed the data described above to represent each tissue sample as a 2-layer multilayer network.

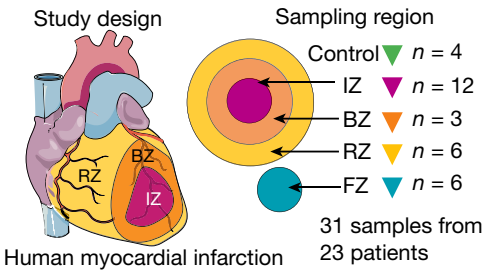


Figure 3.1. | Schematic of the study which collected the heart tissue samples from different zones of IHD [8]. IZ, ischemic zone; BZ, border zone; RZ, remote zone; FZ, fibrotic zone.

Sample X					
	Cell 1	Cell 2	Cell 3	$\cdots$	Cell n
Gene 1	2	13	3	$\cdots$	0
Gene 2	0	0	18	$\cdots$	7
Gene 3	3	7	6	$\cdots$	14
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Gene k	16	4	0	$\cdots$	0

Table 3.1. | Schematic representation of scRNA-seq output.

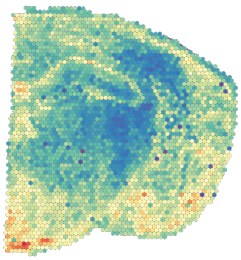


Figure 3.2. | Example visualization of ST data [15].

The scRNA-seq data of the 31 tissue samples were aggregated to form one large $k \times n$ gene expression matrix, where k is the total number of unique genes measured across all samples and n is the total number of cells across all samples. Low-quality data points (cells with too high or too low gene counts) were filtered out, and the data was normalized. Highly variable genes (HVG) between the cells were identified and then dimensionality reduction using principal component analysis (PCA) was performed. After dimensionality reduction, the cells were clustered into 33 distinct cell types and subtypes. To reduce the complexity of the problem and to ease computation, the decision was made to map cell types to spots rather than single cells to spots.

The multilayer network $\mathcal{M} = (\{G_{expr}, G_{spat}\}, E_{IL}, w_{IL})$ for each heart tissue sample consists of two layers, G_{expr} and G_{spat}, which represent the gene expression information and the spatial information of the sample, respectively. The nodes in the gene expression layer represent the distinct cell types present in each tissue sample after the clustering steps outlined above. Note that not each of the 33 cell types is present in each sample, so we have $V_{expr} = \{\text{Type } 1, \text{Type } 2, \ldots, \text{Type } m\}$ for some $1 \leq m \leq 33$ for each sample. The nodes in the spatial layer represent the spots in each tissue sample, so we have $V_{spat} = \{\text{Spot } 1, \text{Spot } 2, \ldots, \text{Spot } n\}$ for a sample with n spots.

In the gene expression layer G_{expr}, the edge weight $w_{expr}(u, v)$ between two nodes $u, v \in V_{expr}$ is proportional to the gene expression distance between the cell types u and v. The gene expression distance between cell types u and v is measured by calculating the Euclidean distance between the total gene counts of all cells in that sample belonging to each type. The gene expression layer G_{expr} is fully connected, i.e., $E_{expr} = \{(u, v) \mid u, v \in V_{expr}\}$.

In the spatial layer G_{spat}, the edge weight $w_{spat}(u, v)$ between two nodes $u, v \in V_{spat}$ is proportional to the physical distance between the spots u and v in the tissue sample. G_{spat} is also fully connected.

For the inter-layer edges, novoSpaRc [9, 10] was applied to the scRNA-seq and ST data to calculate probabilistic mappings of cell types to spots. The scRNA-seq data was leveraged for the calculations based on the hypothesis that physically neighboring cells have similar gene expressions, and the ST data was used as a reference atlas providing prior information about average gene expressions in spots. For each tissue sample, novoSpaRc was also provided with a sample-specific marginal cell-type distribution as not all cell types appear in each sample with equal probability. The inter-layer edges fully connect the two layers of each network, i.e., $E_{IL} = \{(u, v) \mid u \in V_{expr}, v \in V_{spat}\}$. The edge weight $w_{IL}(u, v)$ is proportional to the probability that cell type u appears in spot v, as calculated using novoSpaRc. Figure 3.3 shows a schematic illustration of a multilayer network representing an individual heart tissue sample.

3.2. Network Embeddings

Networks are high-dimensional and complex structures, making them difficult to process in downstream tasks such as prediction or clustering. Classical machine learning methods such as neural networks, support vector machines, or k-means clustering are designed for vector-based data structures. Due to

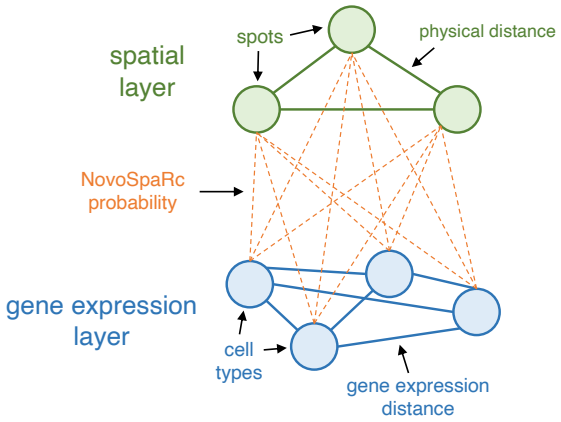


Figure 3.3. | Schematic illustration of the multilayer network representation of the gene expression and spatial data of a heart tissue sample. Note the difference to Figure 1.1: the gene expression layer now consists of cell types rather than single cells, and the inter-layer edge weights are given by novoSpaRc localization probabilities.

the inherent complexity and non-Euclidean nature of graph structures, they cannot be directly applied to graph data. Networks are therefore often encoded as low dimensional vectors $\vec{v} \in \mathbb{R}^d$ which capture the relevant features of the networks. These so-called feature representations or embeddings are more interpretable and much lower dimensional than the original networks, making them easier to process in downstream tasks.

Embeddings can be computed either at the node level (node embeddings) or at the level of the entire network (full-network embeddings). Node embeddings produce a separate embedding for every single node and are commonly used for node or link prediction tasks. Full-network embeddings, on the other hand, produce a single embedding for the entire network and are commonly used for network prediction tasks.

The crucial property of a good network embedding is that it accurately reflects a notion of similarity between nodes or networks. This means that two nodes or two networks that share similar properties or fulfill similar roles should have embeddings that are similar according to some similarity measure defined over the embedding space.

Network embeddings are a general optimization problem that is independent of the downstream tasks [16]. This makes network embeddings a very versatile processing step in a range of different tasks, attracting much research in recent years, especially on the side of node embeddings. In this thesis, we use node embeddings of the multilayer networks representing the heart tissue samples to cluster the samples according to their embeddings. We outline a selection of proposed node embedding methods below.

3.2.1. node2vec

node2vec [16] is a very popular node embedding method proposed by Grover and Leskovec which builds on the idea of DeepWalk [17] to define similarity between nodes using random walks. For a given network $G = (V, E)$, node2vec samples $r \in \mathbb{N}$ random walks of fixed length $l \in \mathbb{N}$ starting from every node $u \in V$, and defines the neighborhood of node u as the nodes which are visited on the random walks originating at u. The intuition is that nodes that appear in neighborhoods together are similar and should therefore have similar embeddings. The notion of similarity between embeddings is defined by the dot product: the embeddings $\vec{z_u}$ and $\vec{z_v}$ of nodes u and v are considered similar if their dot product $\vec{z_u} \cdot \vec{z_v}$ is large. The embeddings are optimized using the Skip-gram model [18, 19].

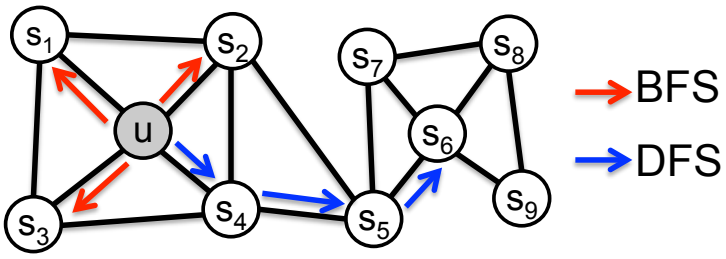


Figure 3.4. | BFS and DFS search strategies from node u ($k = 3$) [16].

Notions of Neighborhoods

Grover and Leskovec [16] aim to design an embedding that preserves neighborhoods of nodes. They note that, in general, there are two forms of similarity between nodes in networks. On the one hand, nodes can be organized based on local communities they belong to (i.e., *homophily*), and on the other hand, they can be organized based on structural roles in the network (i.e., *structural equivalence*). For instance, in Figure 3.4, nodes u and s_1 belong to the same local community of nodes, while nodes u and s_6 share the same structural role of a node hub.

The authors argue that previous work such as DeepWalk [17], which defines node neighborhoods based on fixed-length, unbiased random walks, captures a very constrained notion of node similarity and does not account for different connectivity patterns unique to networks. To overcome this limitation, Grover and Leskovec propose a flexible neighborhood sampling strategy that can capture similarity based on both homophily and structural equivalence.

Generally, there are two extreme sampling strategies to generate a neighborhood set of size k for a source node u:

- **Breadth-first Sampling (BFS)** The neighborhood of node u is restricted to nodes which are immediate neighbors of u. For example, in Figure 3.4 for a neighborhood of size $k = 3$, BFS samples the nodes s_1, s_2, s_3.
- **Depth-first sampling (DFS)** The neighborhood of node u consists of nodes sequentially sampled at increasing distances from u. In Figure 3.4, DFS samples the nodes s_4, s_5, s_6.

Grover and Leskovec maintain that BFS and DFS strategies play a key role in producing node representations that reflect either structural equivalence or homophily. In particular, neighborhoods sampled by BFS lead to embeddings that correspond closely to structural equivalence. Intuitively, it is often sufficient to accurately describe local neighborhoods to capture structural equivalence. For instance, structural equivalence based on network roles such as hubs and bridges can be identified by viewing the immediate neighborhoods of each node [16].

Inversely, DFS can explore much larger parts of the network as it can move further away from the source node. Neighborhoods sampled by DFS therefore reflect a higher-level view of the network which is essential in discovering communities based on homophily [16].

The authors note that real-world networks commonly exhibit a mixture of both homophily and structural equivalence and therefore conclude that it is essential to develop a neighborhood sampling strategy that can obey both principles. This would allow feature learning algorithms to generalize across a wide range of domains and prediction tasks.

In light of these observations, Grover and Leskovec devise a biased random walk procedure that serves as a flexible neighborhood sampling strategy and can simulate both BFS and DFS. We summarize the random walk procedure introduced in the original paper [16] in the following.

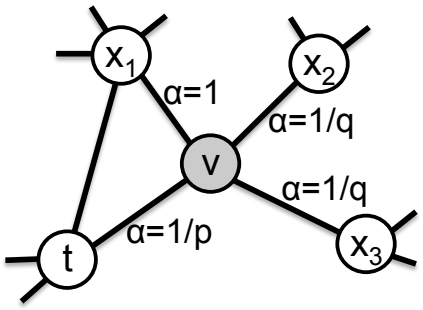


Figure 3.5. | Illustration of the random walk procedure in node2vec. The walk just transitioned from t to v and is now evaluating its next step out of node v. Edge labels indicate search biases α [16].

Biased Random Walks

Let $G = (V, E)$ be a given network. The proposed method applies to any (un)directed, (un)weighted network. Due to the different notions of neighborhoods in networks outlined above, a single neighborhood sample for each node might not suffice to adequately reflect the connectivity patterns of the network. Therefore, $r \in \mathbb{N}$ different neighborhoods are sampled for each node $u \in V$.

Formally, a single neighborhood of a node $u \in V$ is sampled as follows: We simulate a random walk of fixed length $l \in \mathbb{N}$ starting from source node u. Let c_i denote the ith node in the walk, starting with $c_o = u$. Nodes c_i are generated by the following distribution:

$$P(c_i = x \mid c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z_v} & \text{if } (v, x) \in E \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

where π_{vx} is the unnormalized transition probability between nodes v and x. Z_v is the normalizing constant defined as $Z_v = \sum_{x \in C(v)} \pi_{vx}$, where $C(v) = \{x \in V \mid (v, x) \in E\}$ is the set of nodes which v is connected to via an edge.

The authors of node2vec define a 2^{nd} order random walk with two parameters $p \in \mathbb{Q}_{>0}$ and $q \in \mathbb{Q}_{>0}$ which guide the walk: Consider a random walk that just traversed edge (t, v) and now resides at node v (see Figure 3.5). To decide the next step of the walk, the transition probabilities π_{vx} on all edges (v, x) leading from v need to be calculated. Grover and Leskovec define a search bias term α as follows:

$$\alpha_{pq}(t, x) = \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases} \tag{3.2}$$

where d_{tx} denotes the shortest path distance between nodes t and x. Note that d_{tx} is always one of $\{0, 1, 2\}$. The unnormalized transition probability is then defined as

$$\pi_{vx} = \alpha_{pq}(t, x) \cdot w_{vx}, \tag{3.3}$$

where w_{vx} is the weight of edge (v, x) (in the case of unweighted networks, $w_{vx} = 1$).

Intuitively, the parameters p and q control how fast the walk leaves the local neighborhood of the source node and explores the network, allowing the sampling strategy to assume either more of a BFS fashion or more of a DFS fashion.

Return parameter, p The parameter p controls the likelihood of immediately revisiting a node in the walk. A high value of p ($> \max(q, 1)$) reduces the probability of sampling an already visited node in the next two steps, thereby encouraging moderate exploration. A low value of p ($< \min(q, 1)$), on the other hand, encourages the walk to revisit nodes, keeping the walk "local" (close to the source node).

In-out parameter, q The parameter q controls the likelihood of the random walk to explore outward, away from the source node. A high value of q ($> \max(p, 1)$) causes the walk to be biased towards nodes close to the source node, thereby approximating BFS behavior. On the contrary, a low value of q ($< \min(p, 1)$) encourages the walk to visit nodes that are further away from the source node, akin to DFS which encourages outward exploration.

Using random walks as a neighborhood sampling strategy poses advantages in time complexity. As random walks impose graph connectivity in the sample generation process, the effective sampling rate can be conveniently increased by reusing samples across different source nodes [16]. By simulating a random walk of length $l > k$, we can generate samples of length k for $l - k$ nodes at once due to the Markovian nature of the random walk. For example, in Figure 3.5, sampling a random walk $\{u, s_4, s_5, s_6, s_8, s_9\}$ of length $l = 6$ results in the neighborhoods $N(u) = \{s_4, s_5, s_6\}$, $N(s_4) = \{s_5, s_6, s_8\}$, and $N(s_5) = \{s_6, s_8, s_9\}$ of size $k = 3$. While sample reuse may introduce some bias in the overall procedure, Grover and Leskovec found that it greatly improves efficiency.

Skip-gram

During the neighborhood sampling stage of node2vec, $r \in \mathbb{N}$ random walks of length $l \in \mathbb{N}$ are sampled for each node using the biased random walk procedure described above. A neighborhood of node u is defined as the set of nodes that are visited on a random walk starting at u. Grover and Leskovec extend the Skip-gram model [18, 19], a model from the domain of natural language processing used to learn continuous feature representations for words, to networks. Skip-gram uses stochastic gradient descent (SDG) to optimize a neighborhood-preserving likelihood objective which is based on the hypothesis that words that appear in similar contexts tend to have similar meanings and should therefore be embedded close to each other. This notion is extended to networks by viewing nodes as "words" and defining the context of a node as its neighborhood (as sampled by some neighborhood sampling strategy).

Let $G = (V, E)$ be a given network and $f : V \to \mathbb{R}^d$ be the mapping function from nodes to feature representations (embeddings) we aim to learn, where $d \in \mathbb{N}$ is the parameter specifying the dimensionality of the embeddings. For every source node $u \in V$, we define $N_S(u) \subset V$ as a network neighborhood of node u generated through the neighborhood sampling strategy S. Note that Skip-gram optimization can be applied for an arbitrary neighborhood sampling strategy S, although in node2vec the biased random walk strategy outlined in the previous section is employed.

The training objective of the Skip-gram model is to find node representations that are useful for predicting the neighborhood nodes [19]. More formally, Grover and Leskovec [16] use the Skip-gram model to optimize the following objective function, which maximizes the log-probability of observing a network neighborhood $N_S(u)$ for a node u, given its feature representation $f(u)$:

$$\max_f \quad \sum_{u \in V} \log Pr(N_S(u) \mid f(u)). \tag{3.4}$$

To make the optimization problem tractable, the authors make two standard assumptions:

1. *Conditional independence.* The likelihood of observing a neighborhood node is independent of

observing any other neighborhood node given the embedding of the source node:

$$Pr(N_S(u) \mid f(u)) = \prod_{n_i \in N_S(u)} Pr(n_i \mid f(u)). \tag{3.5}$$

2. *Symmetry in feature space.* A source node and a neighborhood node have a symmetric effect over each other in feature space, i.e. for every $u \in V$ and every $v \in N_S(u)$:

$$Pr(v \mid f(u)) = Pr(u \mid f(v)). \tag{3.6}$$

In accordance with assumption 2, the conditional likelihood of every source-neighborhood node pair is modeled as a softmax unit parametrized by a dot product of their features:

$$Pr(n_i \mid f(u)) = \frac{\exp(f(n_i) \cdot f(u))}{\sum_{v \in V} \exp(f(v) \cdot f(u))}. \tag{3.7}$$

With these assumptions, the objective in Eq. 3.4 simplifies as follows:

$$\begin{aligned}
& \max_f \sum_{u \in V} \log Pr(N_S(u) \mid f(u)) && (3.4)\\
= & \max_f \sum_{u \in V} \log \prod_{n_i \in N_S(u)} Pr(n_i \mid f(u)) && (3.8)\\
= & \max_f \sum_{u \in V} \sum_{n_i \in N_S(u)} \log Pr(n_i \mid f(u)) && (3.9)\\
= & \max_f \sum_{u \in V} \sum_{n_i \in N_S(u)} \log \left(\frac{\exp(f(n_i) \cdot f(u))}{\sum_{v \in V} \exp(f(v) \cdot f(u))} \right). && (3.10)
\end{aligned}$$

Intuitively, we are trying to maximize the probability that, given the embedding of a node $u \in V$, we observe the neighborhood nodes of u, while minimizing the probability that we observe any non-neighborhood nodes.

The equation $Z_u := \sum_{v \in V} \exp(f(v) \cdot f(u))$ is expensive to compute for large networks and is therefore approximated using negative sampling [19]. Intuitively, Z_u (approximately) represents the probability that we observe non-neighborhood nodes given the embedding of node u. We can approximate this by summing over a few selected negative samples, i.e., non-neighborhood nodes, instead of summing over *all* nodes in the network. With negative sampling, the softmax term of Eq. 3.10 is approximated as follows:

$$\log \left(\frac{\exp(f(n_i) \cdot f(u))}{\sum_{v \in V} \exp(f(v) \cdot f(u))} \right) \approx \log(\sigma(f(n_i) \cdot f(u))) - \sum_{j=1}^{k} \log(\sigma(f(n_j) \cdot f(u))), n_j \sim P_V \tag{3.11}$$

where σ is the sigmoid function, $k \in \mathbb{N}$ is the number of negative samples (non-neighborhood nodes) selected for each source node, and P_V is a probability distribution over the network nodes such that each node is sampled with a probability proportional to its degree. Mikolov et al. [19] suggest using a value of k in the range $5 - 20$ in practice.

The Skip-gram model [18,19] is a neural network architecture with a single hidden layer that optimizes

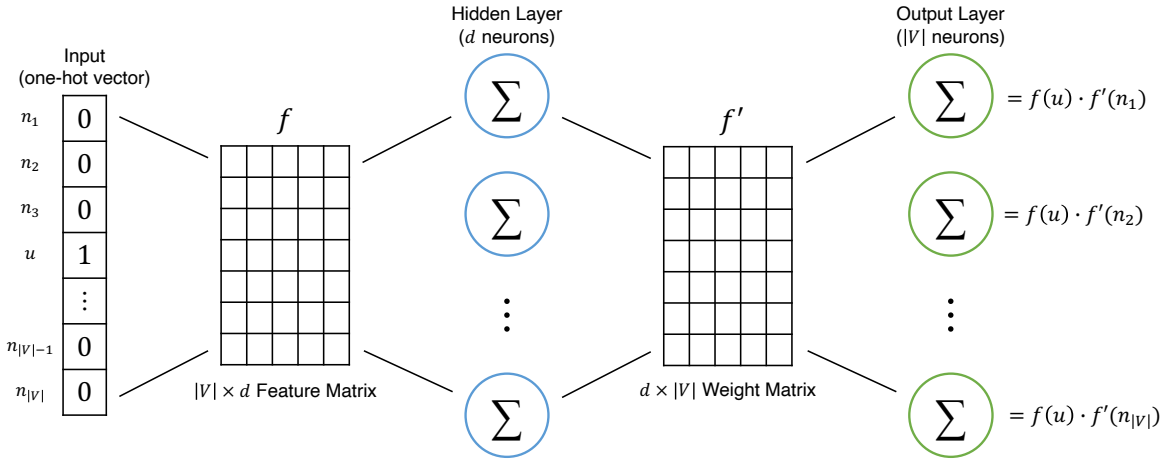


Figure 3.6. | Illustration of the Skip-gram model. The input is the source node for which to predict the neighborhood, encoded as a one-hot vector. The weight matrix f of the hidden layer is the feature representation matrix of nodes that we wish to optimize. The output layer has an output neuron for each network node.

the feature representations (embeddings) of the nodes using stochastic gradient descent (SGD). Figure 3.6 shows a schematic illustration of the Skip-gram model.

Skip-gram takes a node u encoded as a one-hot vector as input and predicts its neighborhood. The mapping function $f : V \to \mathbb{R}^d$ from nodes to feature representations we aim to learn can equivalently be thought of as a matrix of size $|V| \times d$ parameters. This matrix is given by the weight matrix of the hidden layer of the model. The Skip-gram model has an output neuron for each node in the network we wish to embed. Provided node $u \in V$ as an input, the output for node $v \in V$ is given by $f(u) \cdot f'(v)$, where $f(u)$ is the "input" vector representation of u, i.e., the embedding of u, and $f'(v)$ is the "output" vector representation of v.

In accordance with the optimization problem defined in Eq. 3.10 and Eq. 3.11, the loss function of the Skip-gram model is defined as

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_S(u)} -\left(\log(\sigma(f'(v) \cdot f(u))) - \sum_{j=1}^{k} \log(\sigma(f'(n_j) \cdot f(u))) \right), n_j \sim P_V. \tag{3.12}$$

This loss function is optimized using SGD. The training set consists of the random walks sampled using the biased random walk sampling procedure outlined in the previous section. Each random walk has a source node $u \in V$, and the neighborhood $N_S(u)$ of u is defined as the nodes visited on the walk. In each training step, the parameters $f(u)$ of the source node, the parameters $f'(v)$ of all neighborhood nodes $v \in N_S(u)$, and the parameters $f'(n_j)$ of all negative samples $n_j \sim P_V$ for all $1 \leq j \leq k$ are optimized. Although only the embedding of the source node u is optimized in each step, all node embeddings are optimized throughout training because multiple random walks are sampled for every source node $u \in V$.

After training on all sampled random walks for one or more epochs, the embeddings of the nodes are extracted as the feature matrix f. Note that Skip-gram does not exactly solve the optimization problem stated in Eq. 3.10, but rather estimates it. The reason for this is threefold:

1. To reduce computational cost, the softmax term in Eq. 3.10 is approximated using negative sampling, as stated in Eq. 3.11.

2. Stochastic gradient descent is a stochastic approximation of gradient descent.

3. In each training step, we optimize the parameters in f' for the neighborhood nodes and the negative samples as opposed to the embedding parameters in f.

Nonetheless, Grover and Leskovec show that node2vec, building on Skip-gram, achieves state-of-the-art performance in both multi-label classification and link prediction tasks on several real-world networks [16].

3.2.2. HeNHoE-2vec

Although node2vec [16] has established itself as a very popular and effective node embedding method on single-layer networks, it is not applicable to multilayer networks in its native form. Valentini et al. [20] propose *HeNHoE-2vec*, an extension of node2vec to networks with heterogeneous nodes (*HeN*) and homogeneous edges (*HoE*). HeNHoE networks integrate an additional layer of information compared to "basic" networks by imposing that each node belongs to a distinct "type". We note that HeNHoE networks are equivalent to multilayer networks, in which each node belongs to a distinct layer, and we therefore refer to HeNHoE networks as multilayer networks henceforth.

Let $\mathcal{M} = (\{G_1, \dots, G_m\}, E_{IL}, w_{IL})$ be a given (un)directed, (un)weighted multilayer network with $m \in \mathbb{N}$ layers. HeNHoE-2vec adapts the random walk procedure from node2vec, making it layer-aware by introducing a "switching" parameter $s \in \mathbb{Q}_{>0}$ which controls the way in which the random walks can move between layers. As in node2vec, we simulate a random walk of fixed length $k \in \mathbb{N}$ starting from source node $u \in V_l$ for some $l \in \{1, \dots, m\}$. Let c_i denote the ith node in the walk, starting with $c_o = u$. Nodes c_i are generated by the following distribution:

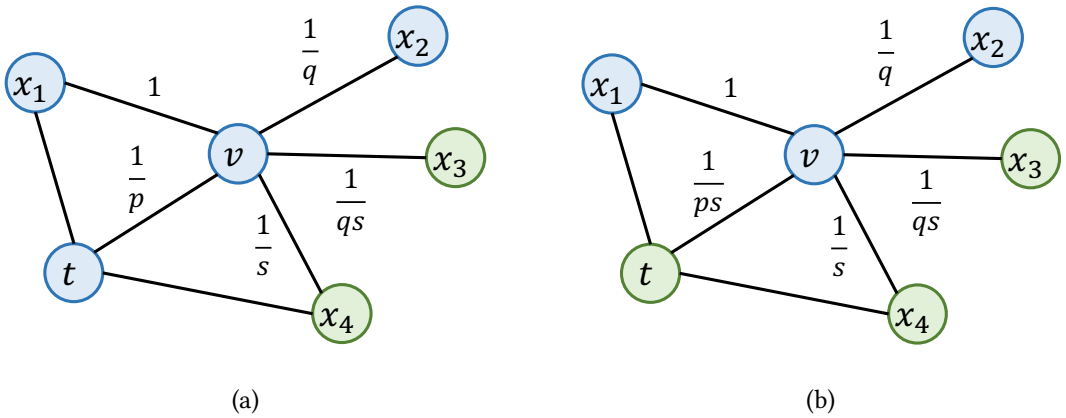
$$P(c_i = x \mid c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z_v} & \text{if } (v,x) \in E_{l_1} \cup E_{IL} \\ 0 & \text{otherwise} \end{cases} \tag{3.13}$$

where π_{vx} is the unnormalized transition probability between nodes $v \in V_{l_1}$ and $x \in V_{l_2}$ (note that if $l_1 = l_2$, we have $E_{l_1} = E_{l_2}$). Z_v is the normalizing constant defined as $Z_v = \sum_{x \in C(v)} \pi_{vx}$, where $C(v) = \{x \in V \mid (v,x) \in E_{l_1} \cup E_{IL}\}$ is the set of nodes which v is connected to via an intra-layer or inter-layer edge.

Let $L : \bigcup_{l \in \{1,\dots,m\}} V_l \to \{1, \dots, m\}$ be the function which maps a node $u \in V_l$ to its layer l. Following node2vec, Valentini et al. [20] define a 2nd order random walk with the return parameter $p \in \mathbb{Q}_{>0}$, the in-out parameter $q \in \mathbb{Q}_{>0}$, and the new switching parameter $s \in \mathbb{Q}_{>0}$ which guide the walk: Consider a random walk that just traversed edge (t, v) and now resides at node $v \in V_l$ (see Figure 3.7). To decide its next step, the walk needs to evaluate the transition probabilities π_{vx} on all edges (v, x) leading from v. Valentini et al. define the search bias term α as follows:

$$\alpha_{pqs}(t,v,x) = \begin{cases} \text{if } L(v) = L(x): & \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases} \\ \text{else}: & \begin{cases} \frac{1}{ps} & \text{if } d_{tx} = 0 \\ \frac{1}{s} & \text{if } d_{tx} = 1 \\ \frac{1}{qs} & \text{if } d_{tx} = 2 \end{cases} \end{cases} \tag{3.14}$$

where d_{tx} denotes the shortest path distance between nodes t and x. The unnormalized transition probability is then defined as $\pi_{vx} = \alpha_{pqs}(t,v,x) \cdot w_{vx}$, where w_{vx} is the weight of edge (v,x) (in the



(a) (b)

Figure 3.7. | Illustration of the random walk procedure in HeNHoE-2vec. (a) A multilayer network with two layers G_1 and G_2, where $V_1 = \{t, v, x_1, x_2\}$ and $V_2 = \{x_3, x_4\}$. (b) A multilayer network with two layers G_1 and G_2, where $V_1 = \{v, x_1, x_2\}$ and $V_2 = \{t, x_3, x_4\}$. In both cases, the walk just transitioned from t to v and is now evaluating its next step out of node v. Edge labels indicate search biases α.

case of unweighted networks, $w_{vx} = 1$).

Intuitively, as in node2vec, the parameters p and q control how fast the walk explores and leaves the local neighborhood of the source node. The switching parameter s controls the likelihood of the walk to switch between layers: if $s < 1$, switching between network layers is encouraged. Conversely, if $s > 1$, switching between layers is discouraged. The introduction of this parameter enables control over the extent to which embeddings of nodes are influenced by their neighboring nodes in other layers.

Valentini et al. [20] further increase the modularity of HeNHoE-2vec by introducing multiple different "switching modes":

Simple switching The switching method outlined above where we define a single switching parameter $s \in \mathbb{Q}_{>0}$ which regulates the probability of the random walks to switch between layers.

Versus specific switching We define a switching parameter $s_{l_1,l_2} \in \mathbb{Q}_{>0}$ for every layer pair $(l_1, l_2) \in \{1, \ldots, m\} \times \{1, \ldots, m\}$ with $l_1 \neq l_2$. The parameter s_{l_1,l_2} regulates the probability of a random walk to switch from layer l_1 to layer l_2. Say that $L(v) = l_1$ and $L(x) = l_2$ with $l_1 \neq l_2$, then the parameter s_{l_1,l_2} replaces the parameter s in the definition of the search bias term α in Eq. 3.14. Note that we may have $s_{l_1,l_2} \neq s_{l_2,l_1}$, i.e., we can define different probabilities to switch from layer l_1 to layer l_2 and to switch from layer l_2 to layer l_1.

Versus specific switching allows us to bias the random walks towards specific layers during neighborhood sampling, thereby enabling us to set a focus on specific layers during the embedding process. We note that Valentini et al. introduce two more switching modes, *multiple switching* and *special node switching*, however, as these are merely special cases of versus specific switching, we do not discuss them further here.

The remaining steps of the embedding process, including the reuse of samples and the optimization of embeddings using the Skip-gram model, are identical to node2vec. HeNHoE-2vec is able to leverage the information carried by the inter-layer connections of the network, and it can be tuned to appropriately model a variety of network settings where different significance levels may be assigned to different layers.

3.3. Optimal Transport

As discussed in Section 1.3, we aim to cluster the embeddings of the multilayer networks representing the heart tissue samples in a manner that differentiates samples taken from different zones of ischemic heart disease (IHD). During clustering, we aim to group the set of samples in a way such that samples that are in the same group (or cluster) are more *similar* to each other than to those in other clusters. This implies that, in order to achieve a meaningful clustering of samples, we must define a meaningful measure of *similarity* (or, equivalently, *distance*) between samples. We may consider a similarity (or distance) measure between samples "meaningful" if it accurately reflects our understanding of the similarity between the physical objects that underpin the samples.

In our case, the set of samples we aim to cluster is the set of embeddings of multilayer networks. Each embedding is underpinned by the physical heart tissue sample whose spatially resolved gene expression data was used to generate the multilayer network. We therefore need to define a similarity (or distance) measure between the embeddings of the multilayer networks such that two embeddings are similar if their underlying heart tissue samples are similar.

The definition of similarity between heart tissue samples can be based on a wide variety of modalities and is highly use-case-dependent. As we aim to investigate differences in gene expression with respect to the stage or zone of IHD, we consider two samples to be similar if they stem from the same zone of IHD.

When node embeddings are generated for multiple networks using the methods introduced in Section 3.2, we obtain a separate embedding space for each network. Even if the dimensionalities of the spaces are the same, the node embeddings between networks are not directly comparable because they were generated independently of each other. Figure 3.8 shows two separate node embeddings of the gene expression layer of sample IZ_P15 into 2-dimensional space using node2vec. It is clear that both embeddings capture the same sense of similarity between nodes in the gene expression layer: nodes that are close together in one embedding are also close together in the other embedding. However, the embeddings appear shifted with respect to one another because they reside in different metric spaces, making it difficult to compare them directly.

As we aim to compare networks based on the embeddings of their nodes, it is crucial that we are able to compute distances between embeddings in different spaces. In the following, we introduce Kantorovich optimal transport followed by its extension to the Gromov-Wasserstein distance which allows us to compute optimal mappings and distances between distributions in different metric spaces.

3.3.1. Kantorovich Optimal Transport

Optimal transport is used to calculate the most cost-effective way to map one distribution to another distribution. Consider, for example, a situation where we have a collection of $n \in \mathbb{N}$ farmers who produce wheat and a collection of n bakeries that use the wheat produced by the farmers to bake bread. Suppose, for this simple example, that the farmers and bakeries form two disjoint subsets F and B of the Euclidean plane $\mathbb{R}^2$, and that we have a *cost function* $c : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}_{\geq 0}$ such that $c(f, b)$ is the cost of transporting one shipment of wheat from farmer f to bakery b. For simplicity, assume that each farmer can only supply one bakery and that each bakery requires exactly one shipment of wheat. Given the above assumptions, a *transport plan* is a bijection $T : F \to B$, i.e., each farmer $f \in F$ supplies exactly one target bakery $T(f) \in B$ and each bakery is supplied by exactly one farmer. We wish to find the *optimal transport plan*, the plan T whose total cost

$$c(T) \coloneqq \sum_{f \in F} c(f, T(f)) \tag{3.15}$$

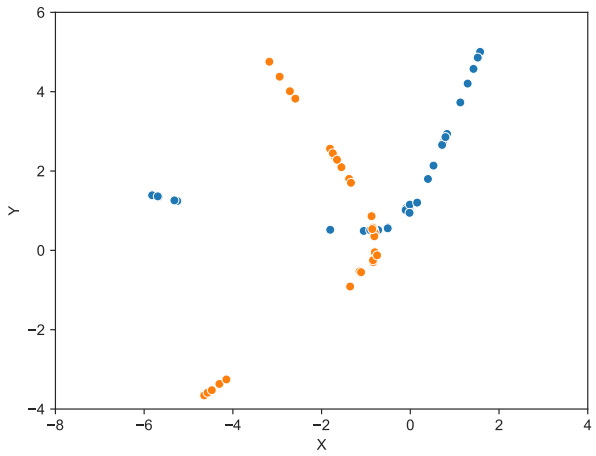


Figure 3.8. | Two separate node embeddings of the gene expression layer of sample IZ_P15 into 2-dimensional space using node2vec. Both embeddings were generated using the same settings. The difference between the embeddings is the result of the pseudo-random nature of the neighborhood sampling strategy of node2vec and the pseudo-random initialization of the Skip-gram model weights.

is the lowest of all possible transport plans from F to B. This simplified example also serves as a useful reference point for the abstract case explained in the following.

Continuous Setting

Kantorovich [21] extends the optimal transport problem to probabilistic mappings, so-called couplings, which represent all possible ways of transporting mass from one distribution to another. Kantorovich optimal transport thereby allows for the splitting of the mass of individual samples, i.e., the mappings need not be 1-to-1. Applied to the farmers/bakeries example, it allows for the possibility that not all farmers or bakeries may be open for business, and it allows farmers to supply more than one bakery and bakeries to accept wheat from more than one farmer.

Formally, given two probability measures μ on a space X and ν on a space Y, and a cost function $c : X \times Y \to \mathbb{R}_{\geq 0}$, the Kantorovich optimal transport problem seeks a coupling γ (a joint measure on $X \times Y$) that attains the infimum

$$\inf \left\{ \int_{X \times Y} c(x, y)\, d\gamma(x, y) \;\middle|\; \gamma \in \Gamma(\mu, \nu) \right\}, \tag{3.16}$$

where $\Gamma(\mu, \nu)$ denotes the collection of all probability measures on $X \times Y$ with marginals μ on X and ν on Y, i.e., $\gamma(A \times Y) = \mu(A)$ and $\gamma(X \times B) = \nu(B)$ for all measurable sets $A \subset X$ and $B \subset Y$.

Intuitively, the cost function $c(x, y)$ indicates how costly it is to transport x to y, and the coupling (or transport plan) $\gamma(x, y)$ gives the amount of mass that is being transported from x to y. The distributions μ on X and ν on Y define the mass distributions in the spaces, and the condition that γ has marginals μ and ν ensures that the mass transported by the coupling γ corresponds to the mass distributions μ and ν. In the farmers/bakeries example, this would ensure that every farmer ships exactly as much wheat as he produces, and every bakery receives exactly as much wheat as they require for baking bread. The integral computes the expected cost of the coupling γ: For each pair of points $(x, y) \in X \times Y$, it multiplies the cost $c(x, y)$ with the amount of mass $\gamma(x, y)$ transported along that pair, and then sums (integrates) over all possible pairs.

When the spaces X and Y are the same metric space with set $\mathcal{M}$ and distance d, the solution to the Kantorovich optimal transport problem gives rise to the Wasserstein distance [22, 23], also known as the

Earth Mover's Distance. When the cost function c corresponds to the distance metric d raised to the p-th power, i.e., $c(x, y) = d(x, y)^p$, the optimal transport distance in Eq. 3.16 is equivalent to the p-th Wasserstein distance:

$$W_p(\mu, \nu) = \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{\mathcal{M}\times\mathcal{M}} d(x, y)^p \, d\gamma(x, y) \right)^{\frac{1}{p}}. \tag{3.17}$$

The Wasserstein distance corresponds to the minimum expected cost of transporting μ to ν under the specific cost function c and thereby quantifies a distance measure between the two distributions. The coupling γ provides the plan of how to achieve the minimum expected cost, while the Wasserstein distance quantifies the magnitude of that cost.

Discrete Setting

In the farmers/bakeries example or in a scenario where we align node embeddings, we are dealing with discrete probability distributions. The discrete optimal transport scenario is more intuitive and computationally tractable. Let $\mu \in [0, 1]^n$ be a discrete probability distribution over a finite set $X = \{x_1, x_2, \ldots, x_n\}$ with associated weights (or probabilities) $(\mu_1, \mu_2, \ldots, \mu_n)$, and $\nu \in [0, 1]^m$ be a discrete probability distribution over a finite set $Y = \{y_1, y_2, \ldots, y_m\}$ with associated weights $(\nu_1, \nu_2, \ldots, \nu_m)$. The weights represent the amount of "mass" at each point in the distributions. As we assume that μ and ν are probability distributions, we have $\sum_{i=1}^n \mu_i = \sum_{j=1}^m \nu_j = 1$. The cost function is then given as a matrix $C \in \mathbb{R}^{n\times m}$, where the entry C_{ij} is the cost of transporting x_i to y_j, and the set of all possible couplings are the matrices

$$\Gamma(\mu, \nu) = \left\{ T \in {\mathbb{R}_{\geq 0}}^{n\times m} \;\middle|\; T\mathbf{1}_m = \mu, T^\top \mathbf{1}_n = \nu \right\}, \tag{3.18}$$

where $\mathbf{1}_n = (1, 1, \ldots, 1) \in \mathbb{R}^n$ is the 1-vector of size $n \in \mathbb{N}$. A discrete coupling matrix T describes how to transport the mass of distribution μ to distribution ν: each row T_i describes how to split the mass μ_i of point x_i onto the points $\{y_1, y_2, \ldots, y_m\}$. The condition $T\mathbf{1}_m = \mu$ demands that the sum of each row T_i is equal to μ_i, the weight (or probability) of sample x_i, and the condition $T^\top \mathbf{1}_n = \nu$ demands that the sum of each column ${T^\top}_j$ is equal to ν_j, the weight (or probability) of sample y_j.

The objective of the discrete optimal transport problem then is to find a coupling that minimizes the cost of transporting the samples:

$$\min_{T\in\Gamma(\mu,\nu)} \langle T, C \rangle_F, \tag{3.19}$$

where $\langle A, B \rangle_F$ is the Frobenius inner product of two real-valued matrices $A, B \in \mathbb{R}^{n\times m}$ defined as $\langle A, B \rangle_F = \sum_{i=1}^n \sum_{j=1}^m A_{ij} B_{ij}$. This problem is usually regularized with entropy for more efficient optimization and empirically better results [24]. Entropy diffuses the optimal coupling, meaning that more masses will be split [9, 11]. With entropic regularization, the optimal transport problem is

$$\min_{T\in\Gamma(\mu,\nu)} \langle T, C \rangle_F - \epsilon H(T), \tag{3.20}$$

where $\epsilon > 0$ is the weight of the regularization and $H(T)$ is the Shannon entropy defined as

$$H(T) = \sum_{i=1}^{n}\sum_{j=1}^{m} T_{ij} \log T_{ij}. \tag{3.21}$$

Higher values of ϵ drive the solution toward a higher-entropy coupling T, while lower values of ϵ result in a more localized coupling T.

The introduction of the entropy term leads to higher numerical stability, avoiding extreme values in the coupling, and it allows Eq. 3.20 to be efficiently solved using the iterative Sinkhorn-Knopp algorithm [25,26]. Especially for large-scale problems, the Sinkhorn-Knopp algorithm typically converges much faster than other methods for solving the original linear program stated in Eq. 3.19 [24].

3.3.2. Gromov-Wasserstein Optimal Transport

Kantorovich optimal transport allows us to compute the Wasserstein distance between two distributions by calculating an optimal coupling which minimizes the expected cost of transporting one distribution to the other. Defining a reasonable cost function across the distributions is crucial in computing a useful mapping and a meaningful distance measure. When dealing with distributions over different metric spaces, however, it is often very difficult to formulate a reasonable cost function across the distributions. Mémoli [27] addresses this issue by generalizing Kantorovich optimal transport to the Gromov-Wasserstein distance which compares distances between pairs of points rather than comparing points directly.

Continuous Setting

Let (X, d_X) and (Y, d_Y) be two metric spaces with probability measures μ and ν respectively. Instead of being concerned with a cost function between points in X and points in Y, the Gromov-Wasserstein distance is concerned with the difference in pairwise distances between points within X and within Y. More specifically, given a cost function $L : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, the Gromov-Wasserstein distance between μ and ν is defined as

$$GW(\mu, \nu) \coloneqq \min_{\gamma \in \Gamma(\mu,\nu)} \int_{X\times Y}\int_{X\times Y} L(d_X(x_1, x_2), d_Y(y_1, y_2))\, d\gamma(x_1, y_1)\, d\gamma(x_2, y_2), \tag{3.22}$$

where $\Gamma(\mu, \nu)$ again denotes the collection of all couplings, i.e., probability measures on $X \times Y$ with marginals μ on X and ν on Y. Intuitively, $L(d_X(x_1, x_2), d_Y(y_1, y_2))$ captures how transporting x_1 to y_1 and x_2 to y_2 would distort the original distances between x_1 and x_2 and between y_1 and y_2 [11]. In the case of $L(a, b) = L_2(a, b) = \frac{1}{2}(a - b)^2$, Gromov-Wasserstein is a distance on the space of metric measure spaces [11,27].

If X and Y share a similar internal structure, i.e., distances between points in X are analogous to distances between points in Y, then the Gromov-Wasserstein distance is small, regardless of where X and Y lie in some larger space. This is a significant difference to the standard Kantorovich optimal transport which is sensitive to the absolute positions of X and Y in the ambient space.

Discrete Setting

Consider two discrete metric spaces (X, d_X) and (Y, d_Y), where $X = \{x_1, x_2, \ldots, x_n\}$ is a finite set of n points and $Y = \{y_1, y_2, \ldots, y_m\}$ is a finite set of m points. We define distance matrices $D^X \in \mathbb{R}^{n\times n}$ and $D^Y \in \mathbb{R}^{m\times m}$, where $D^X_{ij} = d_X(x_i, x_j)$ and $D^Y_{ij} = d_Y(y_i, y_j)$. Let $p \in [0, 1]^n$ and $q \in [0, 1]^m$ be the probability vectors representing the discrete distributions over X and Y, respectively. Given a cost function $L : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, we define the fourth order tensor $\mathbf{L} \in \mathbb{R}^{n\times m\times n\times m}$, where

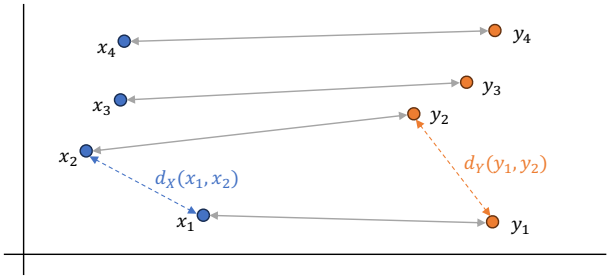


Figure 3.9. | Illustration of a simple example of the discrete Gromov-Wasserstein optimal transport problem. We have two discrete metric space (X, d_X) and (Y, d_Y) with $X = \{x_1, x_2, x_3, x_4\}$ and $Y = \{y_1, y_2, y_3, y_4\}$. We assume a discrete uniform probability distribution in both spaces, so we have $p_i = \frac{1}{4}$ and $q_i = \frac{1}{4}$ for $i = 1, \ldots, 4$. The arrows between the points of the two distributions indicate the optimal (1-to-1) coupling which preserves the relative distances of the two spaces.

$\mathbf{L}_{ijkl} = L(D^X_{ik}, D^Y_{jl})$. $\mathbf{L}_{ijkl}$ thereby captures how transporting x_i to y_j and x_k to y_l would distort the original distances between x_i and x_k and between y_j and y_l. The discrete Gromov-Wasserstein optimal transport problem is then defined as

$$GW(p, q) = \min_{T \in \Gamma(p,q)} \sum_{i,j,k,l} \mathbf{L}_{ijkl} T_{ij} T_{kl}, \tag{3.23}$$

where $\Gamma(p, q)$ is the set of discrete coupling matrices as defined in Eq. 3.18. Intuitively, the goal of the minimization is to find the coupling (or transport plan) T that best aligns the relative distances of the two spaces.

As with Kantorovich optimal transport, the optimal coupling matrix can be efficiently computed using Sinkhorn-Knopp iterations for an entropically regularized optimization problem:

$$GW(p, q) = \min_{T \in \Gamma(p,q)} \sum_{i,j,k,l} (\mathbf{L}_{ijkl} T_{ij} T_{kl}) - \epsilon H(T), \tag{3.24}$$

where $\epsilon > 0$ is the weight of the regularization and $H(T)$ is the Shannon entropy. Again, higher values of ϵ drive the solution toward a higher-entropy coupling T, while lower values of ϵ result in a more localized coupling T.

Figure 3.9 illustrates a simple example of the discrete Gromov-Wasserstein optimal transport problem. Although the two distributions are defined on different metric spaces, they share the same internal structure. The resulting Gromov-Wasserstein distance is very low because it successfully captures the similarity of the structural relationships within both distributions.

Application to IHD Genomic Data

We use the embedding techniques outlined in Section 3.2 to compute node embeddings for each multilayer network representing a heart tissue sample at a specific stage of IHD. As the embeddings of the networks are generated independently of each other, each set of node embeddings is defined over a different metric space. Assuming that the multilayer networks are representative of the genomic data of the heart tissue samples, and assuming that the node embeddings successfully capture the critical information contained in the networks, we hypothesize that the node embeddings of two samples share a similar internal structure and that the degree of similarity depends on the disease stages of the samples. More specifically, we hypothesize that samples at the same disease stage are more similar in gene expression and cell-to-cell interactions than samples from different disease stages and that these similarities (or differences) are reflected in the node embeddings.

We use discrete Gromov-Wasserstein optimal transport to calculate the distances between all pairs of node embeddings, resulting in an $n \times n$ distance matrix, where n is the number of samples. As the Gromov-Wasserstein distance captures similarity in the internal structure of two distributions, we hypothesize that the distance is relatively small for samples from the same zone of IHD, and relatively large for samples from different zones. This would allow us to cluster the samples into their disease stages or zones of IHD based on their pairwise Gromov-Wasserstein distances. We outline a selection of clustering algorithms in the following section.

3.4. Hierarchical Clustering

Clustering is a form of unsupervised machine learning with the goal of grouping (or clustering) similar samples of a dataset together based on a similarity (or distance) measure defined across certain features of the samples. Hierarchical clustering [28] is a clustering method that seeks to build a hierarchy of clusters either by a "bottom-up" approach (agglomerative clustering) or by a "top-down" approach (divisive clustering). We concentrate on agglomerative clustering here, which is the more common of the two approaches.

Consider a dataset $X = \{x_1, x_2, \dots x_n\}$ of $n \in \mathbb{N}$ samples, and a distance measure $d : X \times X \to \mathbb{R}_{\geq 0}$ which defines the pairwise distances between the samples. Agglomerative clustering starts by treating each sample as a single cluster and then iteratively merges clusters into larger clusters until only a single cluster, containing all samples, remains. More concretely, agglomerative hierarchical clustering operates in the following steps:

1. **Initialization**
   - Initialize n clusters $C_1, C_2, \dots, C_n$, where $C_i = \{x_i\}$, i.e., each sample is treated as a single cluster.
   - Compute the $n \times n$ distance matrix $D \in {\mathbb{R}_{\geq 0}}^{n \times n}$ with $D_{ij} = d(x_i, x_j)$.
2. **Agglomeration**
   - Use the distance matrix D to find the two clusters C_i and C_j which are closest to each other.
   - Merge these two clusters into a new cluster $C_{\text{new}} = C_i \cup C_j$, reducing the total number of clusters by one.
   - Update the distance matrix D to reflect the distance between the new cluster and the original clusters. The distance between two clusters C_i and C_j is defined by a linkage distance $\Delta(C_i, C_j)$.
3. **Iteration**
   - Repeat the agglomeration step until there is only a single cluster left which contains all samples.

The result of this process is a binary tree where each node is a cluster. The leaves of the tree are the individual samples (interpreted as singleton clusters), and the root is the cluster that contains all samples [28].

Linkage Distances

To select the closest pair of clusters at each agglomeration step, we must define a *linkage distance measure* $\Delta : \mathcal{P}(X) \times \mathcal{P}(X) \to \mathbb{R}_{\geq 0}$ which defines the linkage distance $\Delta(C_i, C_j)$ between any two subsets of samples $C_i, C_j \in \mathcal{P}(X)$. When both subsets are singletons $C_i = \{x_i\}$ and $C_j = \{x_j\}$, we define $\Delta(C_i, C_j) = d(x_i, x_j)$. For all other cases, we present four common linkage distance measures:

1. *Single Linkage* [29]. The minimum distance between samples of each cluster:

$$\Delta(C_i, C_j) = \min\{d(x_i, x_j) \mid x_i \in C_i, x_j \in C_j\}. \tag{3.25}$$

2. *Complete Linkage* [30]. The maximum distance between samples of each cluster:

$$\Delta(C_i, C_j) = \max\{d(x_i, x_j) \mid x_i \in C_i, x_j \in C_j\}. \tag{3.26}$$

3. *Average Linkage* [31]. The mean distance between samples of each cluster:

$$\Delta(C_i, C_j) = \frac{1}{|C_i| \cdot |C_j|} \sum_{x_i \in C_i} \sum_{x_j \in C_j} d(x_i, x_j). \tag{3.27}$$

4. *Ward Linkage* [32]. The increase in the sum of the squared error (i.e., the variance) when two clusters C_i and C_j are merged:

$$\Delta(C_i, C_j) = \frac{n_i n_j}{n_i + n_j} \|\mu_i - \mu_j\|^2, \tag{3.28}$$

   where $n_i = |C_i|$, $n_j = |C_j|$, and $\|\mu_i - \mu_j\|$ is the Euclidean distance between the centroids μ_i and μ_j of clusters C_i and C_j, respectively. It is important to note that Ward linkage is only reasonable if the distance measure d between individual samples is proportional to the Euclidean distance.

The Ward linkage method can be defined and implemented recursively by a Lance-Williams formula [33,34]. Suppose that, in a given agglomeration step, clusters C_i and C_j have been selected to be merged next, i.e., D_{ij} is the minimum entry of D. The distance matrix D must now be updated to reflect the linkage distance between the new cluster $C_{\text{new}} = C_i \cup C_j$ and the original clusters. In other words, we add a row D_{new}, compute $D_{\text{new},k} = \Delta(C_{\text{new}}, C_k)$ for all other clusters $C_k \notin \{C_i, C_j\}$, and then remove the rows D_i and D_j. The linkage distance between the new cluster C_{new} and a cluster C_k (where $k \notin \{i, j\}$) is given by the Lance-Williams formula

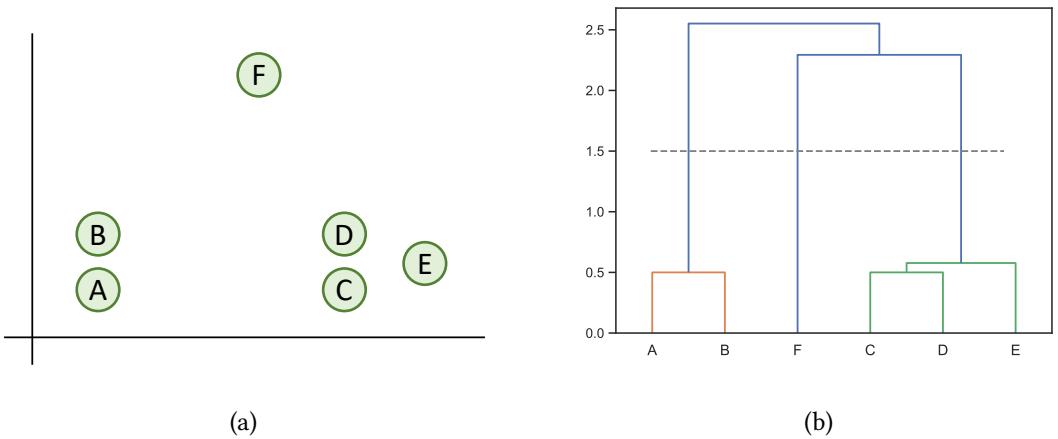
$$\Delta(C_{\text{new}}, C_k) = \frac{n_i + n_k}{S}\Delta(C_i, C_k) + \frac{n_j + n_k}{S}\Delta(C_j, C_k) - \frac{n_k}{S}\Delta(C_i, C_j), \tag{3.29}$$

where $n_i = |C_i|$, $n_j = |C_j|$, $n_k = |C_k|$, and $S = n_i + n_j + n_k$. Note that $\Delta(C_u, C_v)$ is the entry D_{uv} of the previous distance matrix D.

Given that the distance matrix D is initialized using the distance measure d between individual samples, i.e., $D_{ij} = d(x_i, x_j)$, we can always compute the Ward linkage distance using Eq. 3.29, without the need to ever compute Eq. 3.28. The result is an efficient algorithm that only requires the initial distance matrix between samples and does not require the coordinates of the samples themselves to compute the centroids of clusters.

Dendrograms

The resulting binary tree of agglomerative hierarchical clustering is typically presented in a *dendrogram*. Figure 3.10 shows a small example dataset and a dendrogram illustrating the result of the agglomerative hierarchical clustering of the dataset. The dendrogram illustrates how each cluster is composed by drawing U-shaped links between clusters that are merged. The leaves of the dendrogram are the singleton clusters of the individual elements of the dataset. The height of the top of a U-shaped link between two clusters represents the distance between those two clusters as given by the linkage distance measure



(a) (b)

Figure 3.10. | (a) Illustration of a small dataset $X = \{A, B, C, D, E, F\}$ in the Euclidean plane $\mathbb{R}^2$. (b) A dendrogram illustrating the result of the agglomerative hierarchical clustering of X using Ward linkage.

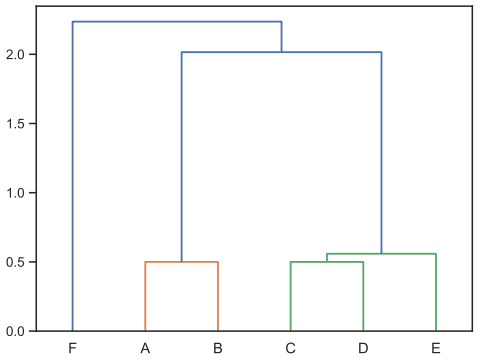


Figure 3.11. | A dendrogram illustrating the result of the agglomerative hierarchical clustering of dataset X (from Figure 3.10) using complete linkage.

and the distance measure between elements. For example, in Figure 3.10, the dendrogram shows that the distance between the singleton clusters $\{A\}$ and $\{B\}$ is 0.5.

The dendrogram can be "cut" at different levels to yield different clusterings of the data. In Figure 3.10, cutting the dendrogram at height 1.5, as indicated by the dashed gray horizontal line, yields the clusters $\{A, B\}, \{C, D, E\}, \{F\}$. Cutting the dendrogram at a higher level will result in a coarser clustering, with a relatively small number of relatively large clusters, while cutting the dendrogram at a lower level will result in a finer clustering, with a relatively large number of relatively small clusters.

It is important to note that the choice of linkage distance measure may significantly impact the result of the clustering. Figure 3.11 shows a dendrogram illustrating the result of clustering dataset X (from Figure 3.10) using complete linkage as opposed to Ward linkage. Cutting this dendrogram such that two clusters are yielded results in the clusters $\{A, B, C, D, E\}, \{F\}$, while cutting the dendrogram in Figure 3.10 such that two clusters are yielded results in the clusters $\{A, B\}, \{C, D, E, F\}$.

Advantages of Hierarchical Clustering

For our use case, hierarchical clustering poses the significant advantage that it does not require the coordinates of the samples in their respective metric spaces in order to cluster them. Instead, hierarchical clustering only requires the distance matrix between samples, which in our case is given by the pairwise Gromov-Wasserstein distances between the node embeddings of the multilayer networks representing the heart tissue samples. This allows us to cluster the heart tissue samples even though they are not embedded in the same metric space.

4. Implementation

In this section, we outline the individual implementation steps that we performed in order to realize the data processing pipeline which, outgoing from the spatial transcriptomics and scRNA-seq data of the heart tissue samples, produces a clustering of these samples into their respective disease stages. Figure 4.1 provides an overview of the stages involved in this process.

In the following subsections, we provide a detailed insight into the implementation of each of the stages that are the subject of this thesis, which in many cases involves the implementation and application of the methods outlined in Section 3 to our specific use case.

4.1. Data Preprocessing

As outlined in Section 3.1.1, the human myocardial infarction data that motivates this thesis and forms the basis of the evaluation of the methods we propose was collected in a study by Kuppe et al. [8]. The authors collected 31 heart tissue samples from 23 individuals, including four non-transplanted donor hearts as controls, and performed scRNA-seq and spatial transcriptomics on each sample.

As explained in detail in Section 3.1.2, this data was preprocessed by our collaborators at the Institute for Computational Genomics at RWTH Aachen University Hospital [14] to represent each tissue sample as a multilayer network. The multilayer network $\mathcal{M} = (\{G_{expr}, G_{spat}\}, E_{IL}, w_{IL})$ of each heart tissue sample consists of two layers, G_{expr} and G_{spat}, which represent the gene expression information and the spatial information of the sample, respectively. The nodes in the gene expression layer represent the distinct cell types present in the sample, while the nodes in the spatial layer represent the spots in the sample. Given that the single cells of all samples were clustered into 33 cell types, the gene expression layer of each multilayer network may contain up to 33 nodes. The spatial layer contains between 1000 and 5000 nodes, depending on the sample. In the gene expression layer G_{expr}, the edge weight $w_{expr}(u, v)$ between two nodes $u, v \in V_{expr}$ is proportional to the gene expression distance between the cell types u and v. In the spatial layer G_{spat}, the edge weight $w_{spat}(u, v)$ between two nodes $u, v \in V_{spat}$ is proportional to the physical distance between the spots u and v in the tissue sample. Both the gene expression layer and the spatial layer are fully connected. The inter-layer edge weight $w_{IL}(u, v)$ for two nodes $u \in V_{expr}$ and $v \in V_{spat}$ is proportional to the probability that cell type u appears in spot v, as calculated using novoSpaRc [9, 10]. The inter-layer edges fully connect the two layers of each network, i.e., $E_{IL} = \{(u, v) \mid u \in V_{expr}, v \in V_{spat}\}$.

For the purpose of this thesis, the Institute for Computational Genomics disregarded 11 samples that were considered outliers or that resulted in low-quality measurements, leaving 22 samples. The tissue samples from diseased hearts are classified into the following zones: the *ischemic zone* (immediate location of oxygen-deprived tissue), the *remote zone* (unaffected by ischemia), the *border zone* (between ischemic and remote zone), and the *fibrotic zone* (scar tissue formed in the healing process after myocardial infarction). The four tissue samples from non-transplanted donor hearts are classified as *control*. The samples may be more broadly classified into *myogenic* (healthy) tissue, *ischemic* (diseased) tissue, and *fibrotic* tissue, where the myogenic class comprises all samples from the remote zone, the border zone, and the control hearts. Henceforth, we use the term *zones* to refer to the finer-grained classification of samples, and we use the term *classes* to refer to the broader classification of samples. A summary of the samples and their respective zone/class labels, as provided by the Institute for Computational Genomics, is given in Table 4.1. For ease of reference, we also assign an index to each sample.

Let $\mathcal{M}^s = (\{G^s_{expr}, G^s_{spat}\}, E^s_{IL}, w^s_{IL})$ be the multilayer network representing sample $1 \leq s \leq 22$, and let n_s and m_s be the number of nodes in the gene expression layer and the spatial layer of sample

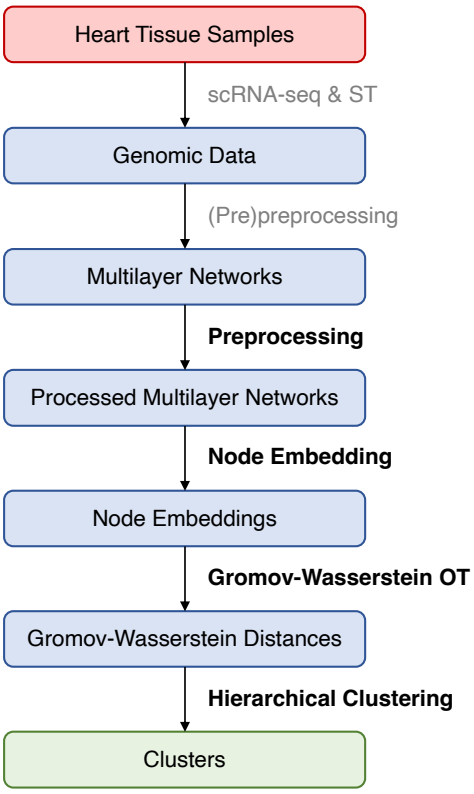


Figure 4.1. | The stages involved in the process of clustering the heart tissue samples into their respective disease stages. The boxes represent objects or entities, and the arrows along with their labels represent the different processing stages. The stages written in light gray were performed by our colleagues at the Institute for Computational Genomics at RWTH Aachen University Hospital, and the stages written in bold are the subject of this thesis.

#	Sample Name	Zone	Class
1	FZ_GT_P19	fibrotic	fibrotic
2	FZ_P14	fibrotic	fibrotic
3	FZ_P18	fibrotic	fibrotic
4	FZ_P20	fibrotic	fibrotic
5	GT_IZ_P9	ischemic	ischemic
6	GT_IZ_P13	ischemic	ischemic
7	GT_IZ_P15	ischemic	ischemic
8	IZ_P10	ischemic	ischemic
9	IZ_P15	ischemic	ischemic
10	RZ_BZ_P2	border	myogenic
11	RZ_BZ_P3	border	myogenic
12	RZ_BZ_P12	border	myogenic
13	RZ_FZ_P5	border	myogenic
14	RZ_GT_P2	remote	myogenic
15	RZ_P3	remote	myogenic
16	RZ_P6	remote	myogenic
17	RZ_P9	remote	myogenic
18	RZ_P11	remote	myogenic
19	control_P1	control	myogenic
20	control_P7	control	myogenic
21	control_P8	control	myogenic
22	control_P17	control	myogenic

Table 4.1. | The heart tissue samples and their respective zones and classes, as provided by the Institute for Computational Genomics at RWTH Aachen University Hospital [14].

s, respectively, i.e., $n_s = |V^s_{expr}|$, $m_s = |V^s_{spat}|$. For every $s \in \{1, \dots, 22\}$, the multilayer network of sample s was provided to us in the form of two adjacency matrices and an inter-layer connection matrix. The two adjacency matrices $A^s_{expr} \in \mathbb{R}^{n_s \times n_s}$ and $A^s_{spat} \in \mathbb{R}^{m_s \times m_s}$ define the gene expression layer and the spatial layer, respectively, and the inter-layer connection matrix $A^s_{IL} \in \mathbb{R}^{m_s \times n_s}$ defines the weights of the inter-layer edges.

4.1.1. Sparsification

We aim to use node2vec [16] based methods to generate embeddings of the multilayer networks representing the heart tissue samples. For its 2nd order random walks, node2vec precomputes the transition probabilities for the interconnections of the neighbors of each node, i.e., for every node $v \in V$, node2vec computes $\alpha_{pq}(t, x)$ (Eq. 3.2) for all nodes $t, x \in V$ with $(t, v), (v, x) \in E$. Computing and storing these transition probabilities incurs a time and space complexity of $\mathcal{O}(a^2|V|)$, where a is the average node degree in the network [16].

A fully connected network with $|V| = n$ nodes has an average degree of $a = n - 1$. In this case, the term $\mathcal{O}(a^2|V|)$ is equivalent to $\mathcal{O}(n^3)$. This is especially problematic for the spatial layers of our networks, which contain between 1000 and 5000 nodes, resulting in $1{,}000{,}000{,}000 \leq n^3 \leq 125{,}000{,}000{,}000$. To alleviate the effect of the cubic growth in time and space complexity, we sparsify the intra-layer and inter-layer connections of our networks, i.e., we remove a large proportion of the edges.

In addition to improving the time and space requirements of the node embedding process, sparsification also reduces the amount of redundant and insignificant information in the networks. In the spatial layer, the vast majority of edges are not needed to provide a meaningful representation of the spatial arrangement of spots. Instead of fully connecting the nodes, connecting each node to a few of its "closest" neighbors is likely sufficient to capture the relevant spatial information and allows the random walks to concentrate on spatially localized patterns. Similarly, in the gene expression layer, it may be sufficient (or even beneficial) to only connect each cell-type node to a few of the "closest" other cell-type nodes, thereby allowing the random walks to concentrate on the most prominent connections.

The intra-layer edge weights in the gene expression layer are calculated based on the gene counts from the scRNA-seq data, and the inter-layer edge weights are calculated based on the scRNA-seq and spatial transcriptomics (ST) data using novoSpaRc [9, 10]. These measurements are prone to inaccuracies and noise and may therefore introduce connections into the networks that don't represent any useful information. Sparsifying the networks by removing the "weakest" connections is likely to reduce the level of noise introduced hereby.

It must also be noted, however, that sparsification may lead to the removal of edges that express more subtle and nuanced connections between nodes, which nonetheless represent essential information regarding the development of IHD in tissue. We discuss the effects of varying degrees of sparsification in Section 5. In the following, we outline the methods of sparsification which we implemented and applied.

k-Nearest Neighbors

We sparsify the gene expression layer and the spatial layer of each multilayer network using the k-nearest neighbors (KNN) approach. For a given $k \in \mathbb{N}$, we leave each node connected to its k nearest intra-layer neighbors and remove all other edges.

Let $A = A^s_{expr} \in \mathbb{R}^{n_s \times n_s}$ be the adjacency matrix of the gene expression layer G^s_{expr} of sample s. As the weights of the intra-layer edges represent *distances* between cell types, the k nearest neighbors of node $u \in V^s_{expr}$ are given by the nodes $v_1, \dots, v_k \in V^s_{expr}$ that attain the *minimum*

$$\min \left\{ \sum_{i=1}^{k} A_{uv_i} \;\middle|\; v_1, \dots, v_k \in V^s_{expr} \backslash \{u\} \right\}. \tag{4.30}$$

If $k \geq n_s$, the layer remains fully connected. Note that "removing" an edge $(u, v) \in E^s_{expr}$ is equivalent to setting $A_{uv} = 0$. The spatial layer G^s_{spat} is sparsified analogously.

It is important to note that the k-nearest neighbor relationship is not symmetrical. If $u \in V$ is a k-nearest neighbor of $v \in V$, that does not necessarily imply that v is also a k-nearest neighbor of u. As we connect each node to its k nearest intra-layer neighbors, each node is guaranteed to have at least k intra-layer neighbors, although some might have more. For instance, u will be connected to its k nearest neighbors, and it will additionally be connected to v because u is one of v's k nearest neighbors. For our experiments, we typically choose $k = 10$ for the spatial layer and $k = 5$ for the gene expression layer.

We also implemented a k-nearest neighbors type algorithm for the inter-layer connections. Here, we leave each node $u \in V_{expr}$ in the gene expression layer G_{expr} connected to its k nearest inter-layer neighbors in the spatial layer G_{spat} and remove all other inter-layer edges.

Let $A = A^s_{IL} \in \mathbb{R}^{m_s \times n_s}$ be the inter-layer connection matrix of sample s. The weights of the edges between a node $u \in V^s_{expr}$ and all nodes in V^s_{spat} are given by the column ${A^\top}_u$ (cell types are columns and spots are rows). The inter-layer edge weight $w^s_{IL}(u, v)$ for two nodes $u \in V^s_{expr}$ and $v \in V^s_{spat}$ is proportional to the probability that cell type u appears in spot v, i.e., it is a *similarity* measure. Therefore, the k nearest neighbors of node $u \in V^s_{expr}$ are given by the nodes $v_1, \ldots, v_k \in V^s_{spat}$ that attain the *maximum*

$$\max \left\{ \sum_{i=1}^{k} {A^\top}_{uv_i} \;\middle|\; v_1, \ldots, v_k \in V^s_{spat} \right\}. \tag{4.31}$$

Thresholding

In some heart tissue samples, it may be the case that certain cell types are very prominent and have high localization probabilities in many spots, while other cell types are less prominent and only have very low localization probabilities or only appear in a few spots. In such cases, it may be inappropriate to sparsify the inter-layer edges in a k-nearest neighbors fashion, as this would restrict every cell-type node to have exactly k connections to the spatial layer, which may not be reflective of the true distribution of cell types.

To alleviate this issue, we introduce thresholding as an alternative method to sparsify the inter-layer edges. Let $A = A^s_{IL} \in \mathbb{R}^{m_s \times n_s}$ be the inter-layer connection matrix of sample s. We define a threshold value $t \in \mathbb{R}$ and sparsify the inter-layer edges by setting

$$A_{uv} = \begin{cases} 0 & \text{if } A_{uv} < t \\ A_{uv} & \text{else} \end{cases} \tag{4.32}$$

for all $u \in V^s_{expr}$ and $v \in V^s_{spat}$. This method allows cell-type nodes to maintain their strong connections to the spatial layer, while weaker connections are removed.

During our experiments, we typically define

$$t = \mu(A) + \sigma(A), \tag{4.33}$$

$$\text{where} \quad \mu(A) = \sum_{i=1}^{m_s} \sum_{j=1}^{n_s} \frac{A_{ij}}{m_s n_s}, \tag{4.34}$$

$$\text{and} \quad \sigma(A) = \sqrt{\frac{1}{m_s n_s - 1} \sum_{i=1}^{m_s} \sum_{j=1}^{n_s} (A_{ij} - \mu(A))^2}. \tag{4.35}$$

$\mu(A)$ is the mean of all inter-layer edge weights and $\sigma(A)$ is the standard deviation. Assuming that the

inter-layer edge weights follow a Gaussian distribution, defining the threshold as the mean plus one standard deviation results in the removal of roughly the weakest 84% of inter-layer connections.

4.1.2. Similarity and Distance

node2vec [16] uses the weights of the network edges to weight the transition probabilities of its random walks (see Eq. 3.3). The idea is that nodes that are connected by an edge with a high weight have a strong relationship, and therefore the random walks that capture the neighborhoods of nodes should traverse these edges with a high probability. This concept requires the edge weights to represent a *similarity* measure between nodes.

In our multilayer networks representing the heart tissue samples, the intra-layer edge weights in the gene expression layer and in the spatial layer both represent *distance* measures between nodes. The inter-layer edge weights, on the other hand, represent a similarity measure. We must therefore invert the intra-layer edge weights such that they also represent a similarity measure. For every edge $(u, v) \in E_{expr} \cup E_{spat}$ we set

$$w(u, v) = \frac{1}{w(u, v)}, \tag{4.36}$$

where $w = w_{expr}$ if $(u, v) \in E_{expr}$ and $w = w_{spat}$ if $(u, v) \in E_{spat}$.

4.1.3. Normalization

Given that the intra-layer edge weights and the inter-layer edge weights are each defined by different distance measures, the weights can span a wide range of values. Without normalization, it may be the case that the inter-layer edge weights far outweigh the intra-layer edge weights, and vice versa. This would cause the node2vec random walks to be biased towards either the inter-layer or the intra-layer edges.

To alleviate this issue, we (independently) normalize the intra-layer and the inter-layer weights to the range $[0, 1]$ using min-max normalization. Let $A = A^s_{expr} \in \mathbb{R}^{n_s \times n_s}$ be the adjacency matrix of the gene expression layer G^s_{expr} of sample s. We normalize the weights across the whole adjacency matrix by updating the values of A as follows for all nodes $u, v \in V^s_{expr}$:

$$A_{uv} = \frac{A_{uv} - \min(A)}{\max(A) - \min(A)}, \tag{4.37}$$

where $\min(A)$ is the minimum and $\max(A)$ is the maximum entry in A. The weights of the intra-layer edges in the spatial layer and the weights of the inter-layer edges are normalized analogously. The effect of the normalization is that all edge weights are in the range $[0, 1]$, and we attain

$$\min_{u,v \in V^s_{expr}} w_{expr}(u, v) = \min_{u,v \in V^s_{spat}} w_{spat}(u, v) = \min_{u \in V^s_{expr}, v \in V^s_{spat}} w_{IL}(u, v) = 0, \tag{4.38}$$

$$\text{and} \quad \max_{u,v \in V^s_{expr}} w_{expr}(u, v) = \max_{u,v \in V^s_{spat}} w_{spat}(u, v) = \max_{u \in V^s_{expr}, v \in V^s_{spat}} w_{IL}(u, v) = 1. \tag{4.39}$$

4.1.4. Multilayer Edge Lists

The multilayer network of each heart tissue sample was provided by the Institute for Computational Genomics [14] as a set of three matrices: one adjacency matrix per layer and an inter-layer connection matrix. We define *multilayer edge lists* to represent multilayer networks in a contained data structure.

source	source layer	target	target layer	weight
Adipo	celltype	Lymphatic_Endo	celltype	0.82824
Adipo	celltype	Mast	celltype	0.92786
⋮	⋮	⋮	⋮	⋮
19	spot	1908	spot	0.60979
20	spot	83	spot	0.33334
⋮	⋮	⋮	⋮	⋮
881	spot	NK_T	celltype	0.00276
882	spot	Endocardial_Endo	celltype	0.00009
⋮	⋮	⋮	⋮	⋮

Table 4.2. | Selected rows of the multilayer edge list representing a sparsified and normalized instance of the multilayer network of sample control_P17 in tabular form.

Let $\mathcal{M} = (\{G_1, \dots, G_m\}, E_{IL}, w_{IL})$ be a multilayer network and let $E = (\bigcup_{i=1}^{m} E_i) \cup E_{IL}$ be the set of all intra-layer and inter-layer edges. We define the multilayer edge list $\mathcal{E}(\mathcal{M})$ of $\mathcal{M}$ as the set of 5-tuples

$$\mathcal{E}(\mathcal{M}) = \{(u, l, v, l', w) \mid (u, v) \in E\}, \tag{4.40}$$

where $u \in V_l, v \in V_{l'}$, and $w = w_l(u, v)$ if $l = l'$ and $w = w_{IL}(u, v)$ if $l \neq l'$. In other words, for every edge, the multilayer edge list contains a 5-tuple which defines the source node, source layer, target node, target layer, and weight of the edge.

After sparsification, normalization, and converting distance measures to similarity measures, we convert the matrices that define each multilayer network into a multilayer edge list. A multilayer edge list may be presented as a table with five columns, where each row defines an edge. Table 4.2 shows an example of a multilayer edge list representing one of the heart tissue samples.

4.2. HeNHoE-2vec

As of the point of writing, the authors of HeNHoE-2vec [20] do not provide an implementation of their embedding algorithm. We therefore implemented HeNHoE-2vec as a Python package and made the code publically available at [35].

Our implementation of HeNHoE-2vec takes a multilayer network in the shape of a tabular multilayer edge list in CSV format as input and produces a node embedding of the network using the HeNHoE-2vec algorithm outlined in Section 3.2.2. Internally, the multilayer network is stored and handled using the NetworkX package [36]. Our implementation allows for the definition of the standard node2vec parameters (return parameter p, in-out parameter q, walk length l, number of walks m, embedding dimensionality d, etc.), and it allows the user to define a switching parameter s_{ij} for every (directed) layer pair (i, j). For convenience, when embedding networks with many layers, we also allow for the definition of a default s switching parameter which is used for every (directed) layer pair (i, j) for which no specific switching parameter s_{ij} is defined. Our implementation works for both directed and undirected networks. Furthermore, we allow the user to specify whether the edge weights of the network represent distance or similarity between nodes. The node embeddings are learned from the sampled node neighborhoods using the Gensim word2vec (Skip-gram) implementation [37].

For an n-dimensional embedding of a network with m nodes, our implementation outputs a CSV file with m rows and $n + 1$ columns. The i-th row stores the embedding of the i-th node, the first column

stores the name and layer of each node as a tuple, and the $j+1$-th column stores the coordinates in the j-th dimension of the embedding space for all $1 \leq j \leq n$.

The three phases of HeNHoE-2vec, i.e., precomputing transition probabilities, simulating random walks, and learning embeddings using Skip-gram are executed sequentially. The Gensim Skip-gram implementation [37] allows for the utilization of multiprocessing to learn the node embeddings, significantly reducing the computation time for large networks. The pre-computation and random walk phases of HeNHoE-2vec are also parallelizable, however, as of the time of writing, our implementation does not support multiprocessing in these phases because they were found to have a negligible impact on overall computation time.

Our HeNHoE-2vec implementation may be used as a Python script or as a package from within other projects. The code is well documented and rigorously tested using pytest [38]. A full list of parameters along with documentation can be found at [35].

4.3. Optimal Transport and Clustering

We calculate the Gromov-Wasserstein distances between the embeddings of the samples using the Python Optimal Transport (POT) library [39]. POT provides functions to solve both the unregularized (Eq. 3.23) and the regularized (Eq. 3.24) Gromov-Wasserstein optimal transport optimization problems. The unregularized optimization problem is solved using the Frank-Wolfe algorithm [40] while the regularized optimization problem is solved using Sinkhorn-Knopp iterations [25].

When computing the Gromov-Wasserstein distance between two network embeddings, we must define distance measures between node embeddings within each embedding space (distance measures d_X and d_Y in Eq. 3.23 and Eq. 3.24). For our experiments, we define the distance between two nodes in the same embedding space as the Euclidean distance between their embedding vectors.

In each multilayer network, the number of nodes in the spatial layer far outweighs the number of nodes in the gene expression layer. Thus, if the discrete probability distributions in the source space and the target space of the Gromov-Wasserstein optimal transport problem are defined to be uniform (i.e., the same amount of "mass" is assigned to each node), the spatial layer will have a much greater influence on the solution of the optimal transport problem than the gene expression layer. We therefore introduce a hyperparameter $\alpha \in [0, 1]$ which allows for the "balancing" of the discrete probability distributions. The value of α determines the fraction of the total probability that is uniformly distributed amongst the cell-type nodes in the gene expression layer. For example, if $\alpha = 0.8$, the total probability (or mass) of all cell-type nodes will sum to 0.8 while the total probability (or mass) of all spot nodes will sum to 0.2. Setting $\alpha = 0.5$ ensures that the total probability is split evenly between the two layers, resulting in both layers having an equal influence on the solution of the optimal transport problem. Given that the gene expression layer comprises a significant portion of the information content of the underlying tissue samples, we hypothesize that balancing the probability distributions will lead to an improvement in clustering results.

Computing the Gromov-Wasserstein distance between two node embeddings is a computationally intensive task. To calculate all pairwise Gromov-Wasserstein distances between the node embeddings of the 22 multilayer networks, a total of $\binom{22}{2} = 231$ Gromov-Wasserstein optimal transport optimization problems must be solved. To reduce computation time, we parallelize the computation of the individual Gromov-Wasserstein optimal transport optimization problems.

We use the SciPy library's [41] implementation of agglomerative hierarchical clustering to compute the clustering of the samples based on their pairwise Gromov-Wasserstein distances. SciPy offers support for the four linkage methods we discussed in Section 3.4, and it provides functions to plot the resulting dendrograms and to cut the dendrograms to produce clusters. We also use the Seaborn library [42] and

Matplotlib [43] to visualize dendrograms.

4.4. HeartNet

We aggregate the individual processing steps outlined in the previous subsections into a single modular, highly configurable processing pipeline which we call *HeartNet*. Thus, outgoing from a set of multilayer networks in the "three matrices" form described in Section 4.1, HeartNet performs the following steps:

1. Sparsification
2. Normalization
3. Conversion to multilayer edge lists
4. Node embedding using HeNHoE-2vec
5. Calculation of pairwise Gromov-Wasserstein distances
6. Agglomerative hierarchical clustering

HeartNet is modular in the sense that the individual processing steps are independent and can be easily modified or interchanged. For instance, other methods may be implemented for sparsification or normalization without affecting the rest of the pipeline. Equally, the algorithms used for the node embeddings, pairwise distance calculations, and clustering may be switched out, as long as the input and output formats are maintained.

The individual steps of HeartNet are highly configurable, and the configuration for the whole pipeline is agglomerated in a single TOML configuration file. An overview of the configuration parameters of HeartNet is provided in Table 1 in Appendix A. The code for HeartNet is submitted along with this thesis.

5. Evaluation

In this section, we present and evaluate the results obtained by running the HeartNet pipeline on the dataset of human heart tissue samples provided by the Institute for Computational Genomics at RWTH Aachen University Hospital [14]. First, we outline the metrics that we use to quantitatively evaluate the performance of our proposed methods. We then present the results of our experiments and evaluate them using the previously introduced metrics. Beyond the evaluation using quantitative metrics, we perform a deeper analysis and interpretation of intermediary results at different stages of the pipeline in an attempt to understand and explain the final results. We also compare our results to those obtained when clustering the samples based only on gene expression data (excluding spatial information). Finally, we summarize our key findings, and we discuss the limitations of our methods, their generalizability to other settings, and their intended use cases.

5.1. Metrics

The main objective of this thesis is to develop methods of clustering multilayer networks based on structural or functional similarity. The basis of evaluation for our methods is the dataset of multilayer networks representing heart tissue samples taken from different zones of ischemic heart disease, as summarized in Table 4.1. The main goal of our evaluation therefore is to evaluate the similarity between the predicted clustering and the ground truth clustering of the samples. To this end, we employ the adjusted Rand index, which we introduce in the following.

5.1.1. Adjusted Rand Index (ARI)

The Rand index [44] is a statistic used to evaluate the similarity between two clusterings of the same dataset. It may be used to evaluate the performance of a clustering algorithm by comparing the predicted clustering to a ground truth clustering. Intuitively, the Rand index computes the fraction of correct decisions made by the clustering algorithm.

Let $S = \{s_1, \dots, s_n\}$ be a set of $n \in \mathbb{N}$ elements, and $A = \{A_1, \dots, A_l\}$, a partition of S into l clusters, and $B = \{B_1, \dots, B_m\}$, a partition of S into m clusters, be the two clusterings to compare. We count the following:

- a: the number of pairs of elements that are in the same cluster in A and in the same cluster in B.
- b: the number of pairs of elements that are in different clusters in A and in different clusters in B.
- c: the number of pairs of elements that are in the same cluster in A but in different clusters in B.
- d: the number of pairs of elements that are in different clusters in A but in the same cluster in B.

The Rand index is then given by the following formula:

$$RI = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}} \tag{5.41}$$

Intuitively, $a + b$ can be regarded as the number of agreements between A and B, while $c + d$ is the number of disagreements. The denominator is equal to the total number of possible pairs of elements. The Rand index thereby represents the frequency of agreements over all pairs or the probability that two clusterings agree on a randomly chosen pair of elements. A limitation of the Rand index is that it does not account for the probability of agreement between random clusterings.

The adjusted Rand index (ARI) [45] alleviates this issue by adjusting the Rand index for the chance grouping of elements, resulting in a more robust and interpretable metric[1]. The ARI is defined as

$$ARI = \frac{RI - E}{M - E}, \tag{5.42}$$

where RI is the Rand index of the clusterings, E is the expected Rand index if the elements were clustered randomly, and $M = 1$ is the maximum possible Rand index. Unlike the Rand index, which ranges from 0 to 1, the adjusted Rand index has a range of -1 to 1, where

- $ARI = 1$ indicates that the two clusterings are identical.
- $ARI = 0$ indicates that the clustering is no better than random.
- $ARI = -1$ indicates that the clustering is completely incorrect.

Although the adjusted Rand index serves as a good objective indicator of the quality of a clustering, it also has limitations. Notably, the ARI is sensitive to cluster sizes. If the sizes of the clusters are imbalanced, the ARI may provide a quality measure that is not in line with our intuitive understanding of a good clustering. Furthermore, the ARI measures the quality of a clustering solely based on cluster membership in the predicted clustering and the ground truth clustering. It does not provide any information about the structure or shape of the clusters, and it cannot provide insights into relationships between dataset elements that go beyond cluster membership.

With these considerations in mind, we carefully use the ARI as an objective, quantitative indicator to compare the quality of different clustering results. Additionally, we manually analyze distances between elements and predicted clusters to gain a deeper insight into the structure of clusters and the relationship between elements.

5.1.2. Computation Time

Computation time is a critical metric, especially when focusing on medical applications that might eventually be deployed in clinical settings where results must be produced reliably and quickly. We therefore investigate the computational time requirements of HeartNet and the individual stages within. We also discuss how individual parameters influence computation time.

All of our experiments are run on the high-performance computing cluster of the Institute for Computational Genomics at RWTH Aachen University Hospital [14]. Every HeartNet run was allocated 20 CPU cores and 3 GB of RAM per CPU core.

5.2. Results

As presented in Table 1, the behavior of HeartNet is tunable through a series of hyperparameters. While the large number of hyperparameters allows for high flexibility in tuning HeartNet for different use cases, it also leads to a vast hyperparameter space, making it challenging to find the optimal configuration for any given use case. At this scale, performing a grid search becomes computationally infeasible: assume that we have 10 hyperparameters, for each of which we wish to consider four different values. This results in $4^{10} = 1{,}048{,}576$ possible different combinations. Assuming a moderate computation time of one hour per configuration, this leads to $1{,}048{,}576$ hours ≈ 120 years of total computation time.

We therefore manually tuned the hyperparameters of HeartNet for our specific use case. Outgoing from an initial configuration with moderate hyperparameter values, we systematically altered the

[1] Note that, mathematically, the ARI is not a metric because it does not satisfy the triangle inequality. We use the term "metric" here not in a mathematical sense, but to refer to measures that aid us in quantifying the quality of our results.

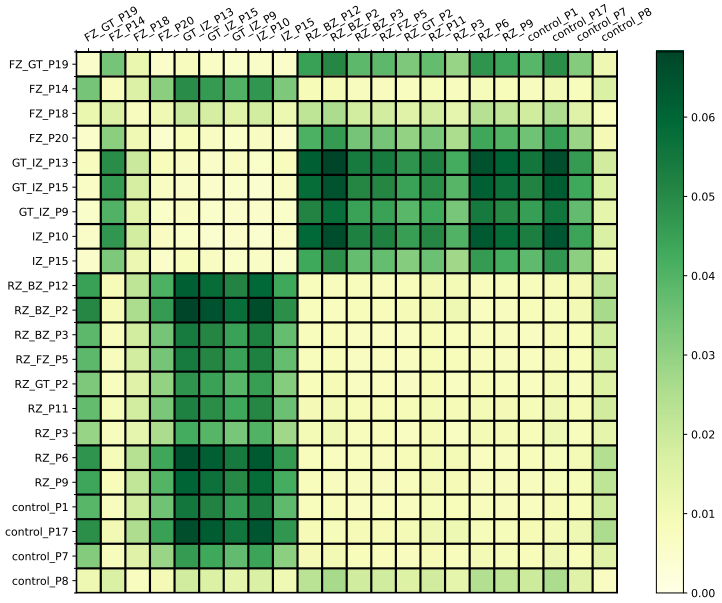


Figure 5.1. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with Configuration 51. Tick labels indicate tissue sample names.

values of one or multiple hyperparameters to estimate reasonable configurations. In total, more than 60 different HeartNet configurations were tested during our experiments. A table summarizing the hyperparameter values of all configurations is presented in Appendix B. Here, we present and analyze the results attained with two promising HeartNet configurations – configurations 51 and 52. These two configurations differ in the values of the s parameter and the number of random walks sampled per node in HeNHoE-2vec. For Configuration 51, we have $s = 10$ and 20 walks per node, and for Configuration 52 we have $s = 3$ and 10 walks per node. Note that we use the results obtained with Configuration 51 to perform a general evaluation of the results achieved with HeartNet. Much of the analysis also applies to the results obtained with other good configurations. We use the results obtained with Configuration 52, on the other hand, to show how results change with varying hyperparameter values and to highlight shortcomings in the metrics used. We further discuss the effects of altering various hyperparameters on clustering quality and computation time in Section 5.3.

Configuration 51

For a summary of the hyperparameter values of Configuration 51, view Table 2 in Appendix B. Figure 5.1 shows the entropically regularized Gromov-Wasserstein distance matrix that is produced when running HeartNet with Configuration 51. Each entry of the distance matrix represents the entropically regularized Gromov-Wasserstein distance between the embeddings of two samples, as computed by solving Eq. 3.24.

We hypothesized that two tissue samples from the same zone of IHD are more similar regarding their internal structure than two tissue samples from different zones of IHD. Based on this hypothesis, we expect the Gromov-Wasserstein distance between the node embeddings of the multilayer networks representing two samples from the *same* zone of IHD to be relatively *low*, and we expect the Gromov-Wasserstein distance between the node embeddings of the multilayer networks representing two samples from *different* zones of IHD to be relatively *high*.

Figure 5.1 shows clear patterns that agree with this prediction. There are two prominent lighter regions in the plot, one between samples GT_IZ_P13 to IZ_P15, and one between samples RZ_BZ_P12 to control_P7. These two regions correspond to the ischemic (diseased) and myogenic (healthy) classes,

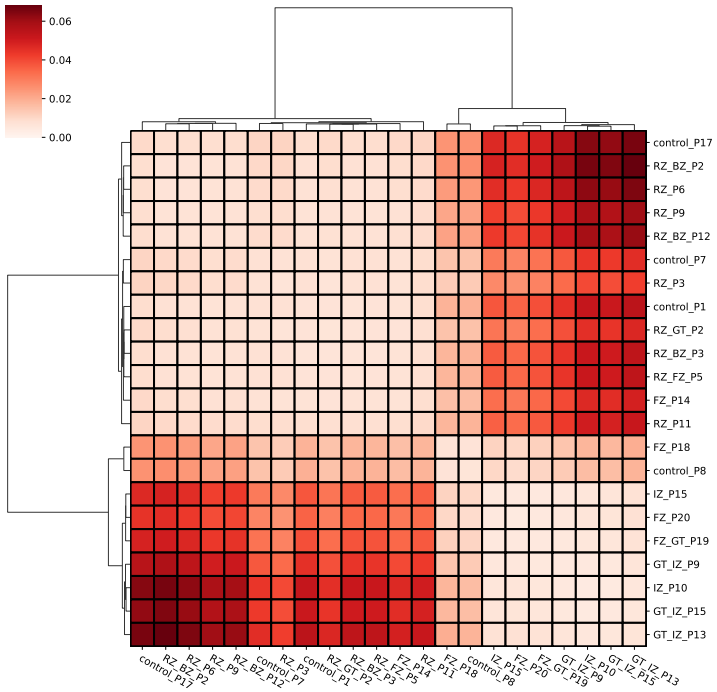


Figure 5.2. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the Gromov-Wasserstein distance calculated using HeartNet with Configuration 51. The matrix is equal to the Gromov-Wasserstein distance matrix presented in Figure 5.1, but the axes are reordered such that the distance between successive samples is minimal. The (identical) dendrograms to the left and on top of the matrix visualize the result of the hierarchical clustering.

respectively (refer to Table 4.1 for the classification of samples into zones and classes). They indicate that the Gromov-Wasserstein distance between the embeddings of two samples from the same class is in fact relatively low. The darker regions of the plot show that the Gromov-Wasserstein distances between embeddings of healthy and diseased samples, on the other hand, are relatively high.

However, Figure 5.1 shows no clear pattern for samples of the fibrotic class (samples FZ_GT_P19 to FZ_P20). Some fibrotic samples, namely FZ_GT_P19, FZ_P18, and FZ_P20, show higher similarity to the ischemic class, while other fibrotic samples, namely FZ_P14, show higher similarity to the myogenic class. We hypothesize that this is the result of different fibrotic samples being at different stages of healing, but further investigation into the individual samples is required to definitively determine the cause. Such investigations, however, lie beyond the scope of this thesis.

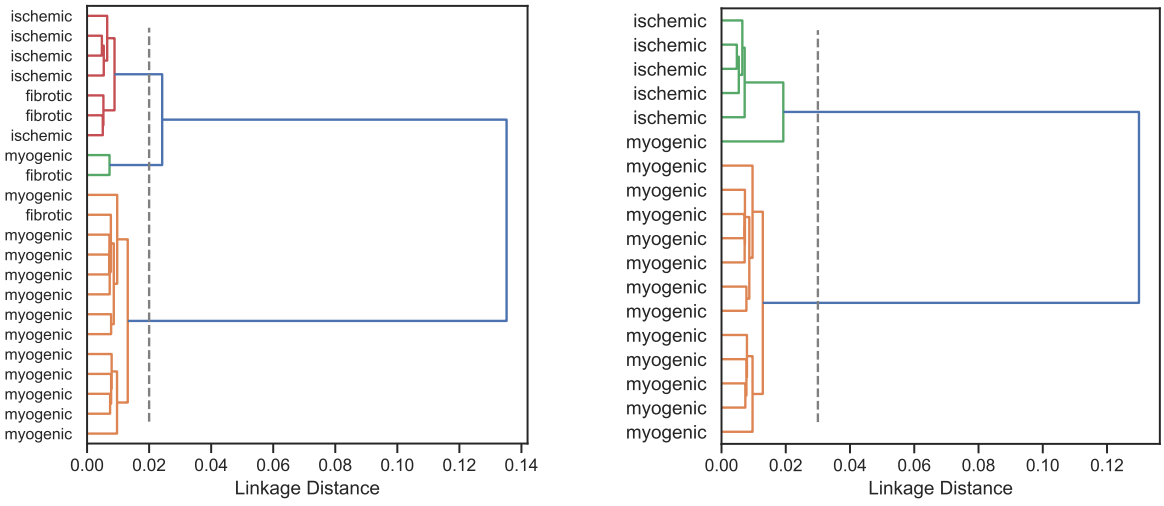
It is also evident from Figure 5.1 that the sample control_P8 is an outlier within the class of myogenic samples. While all other myogenic samples exhibit a low Gromov-Wasserstein distance to other myogenic samples and a high Gromov-Wasserstein distance to ischemic samples, control_P8 is much closer to the ischemic samples.

Performing agglomerative hierarchical clustering with Ward linkage on the samples based on the Gromov-Wasserstein distance matrix results in the clustermap presented in Figure 5.2. To cluster the samples into the three classes *myogenic*, *ischemic*, and *fibrotic*, we cut the dendrogram of the hierarchical clustering such that we attain three clusters. Table 5.1 shows the resulting predicted clusters compared to the ground truth clusters. The resulting adjusted Rand index is 0.64481. Figure 5.3(a) shows another visualization of the same dendrogram, where the sample names have been replaced by their ground truth class labels, along with the cut which results in the clustering presented in Table 5.1.

It is clear from the predicted clustering that our method is not able to group the fibrotic samples. If we remove the fibrotic samples and perform agglomerative hierarchical clustering on only the myogenic

	Predicted Cluster		
Sample	$n = 3$	$n = 2$	**Ground Truth**
FZ_GT_P19	0	–	fibrotic
FZ_P14	1	–	fibrotic
FZ_P18	2	–	fibrotic
FZ_P20	0	–	fibrotic
GT_IZ_P13	0	0	ischemic
GT_IZ_P15	0	0	ischemic
GT_IZ_P9	0	0	ischemic
IZ_P10	0	0	ischemic
IZ_P15	0	0	ischemic
RZ_BZ_P12	1	1	myogenic
RZ_BZ_P2	1	1	myogenic
RZ_BZ_P3	1	1	myogenic
RZ_FZ_P5	1	1	myogenic
RZ_GT_P2	1	1	myogenic
RZ_P11	1	1	myogenic
RZ_P3	1	1	myogenic
RZ_P6	1	1	myogenic
RZ_P9	1	1	myogenic
control_P1	1	1	myogenic
control_P17	1	1	myogenic
control_P7	1	1	myogenic
control_P8	2	0	myogenic
ARI	0.64481	0.77580	

Table 5.1. | Predicted clustering of the samples with Configuration 51. The second column ($n = 3$) shows the clustering attained when hierarchical clustering with Ward linkage is applied to all samples and the dendrogram is cut such that three clusters are yielded. The third column ($n = 2$) shows the clustering attained when hierarchical clustering with Ward linkage is applied to only the myogenic and ischemic samples and the dendrogram is cut such that two clusters are yielded. The last row indicates the adjusted Rand index of the respective clusterings.



(a) (b)

Figure 5.3. | (a) The dendrogram from Figure 5.2, but the sample names have been replaced by their ground truth class labels to clarify the clustering result. (b) The dendrogram resulting from hierarchically clustering only the myogenic and ischemic samples (excluding fibrotic samples) with Ward linkage, where distances between samples are given by the Gromov-Wasserstein distance calculated using HeartNet with Configuration 51. The gray dotted line in each plot indicates the cut that yields the clusterings presented in Table 5.1.

	Computation Time		
Stage	Config. 51	Config. 52	Config. 64 (excl. spatial)
Preprocessing	≈ 6.5 minutes	≈ 6.5 minutes	≈ 8 seconds
HeNHoE-2vec	≈ 10.5 hours	≈ 3.5 hours	≈ 3.5 minutes
Gromov-Wasserstein	≈ 2 hours	≈ 1.75 hours	≈ 45 seconds
Clustering	< 1 millisecond	< 1 millisecond	< 1 millisecond
Total	≈ 12.5 hours	≈ 5.5 hours	≈ 4.5 minutes

Table 5.2. | Summary of the average computation times required for computing all stages of HeartNet with configurations 51, 52, and 64 (excluding spatial information).

and ischemic samples, we attain the clustering visualized by the dendrogram in Figure 5.3(b). Cutting the dendrogram to yield two clusters produces the clusters presented in Table 5.1. The results show that only a single sample, namely control_P8, is placed in the wrong cluster. All other samples are correctly clustered into the myogenic and ischemic groups. The resulting adjusted Rand index is 0.77580. The clustering is robust against the choice of linkage method, i.e., single, complete, average, and ward linkage produce the same clusters. Note, however, that this is not generally the case. See Appendix D for the dendrograms visualizing the results of clustering with single, complete, and average linkage.

Due to the pseudo-random nature of the neighborhood sampling strategy of HeNHoE-2vec, some variation between multiple node embeddings of the same multilayer network is almost certain. A good configuration of HeNHoE-2vec ensures that each node embedding accurately and fully reflects the structure of the underlying multilayer network. Multiple HeNHoE-2vec embeddings of the same two networks should therefore all result in very similar Gromov-Wasserstein distances between pairs of embeddings. Six separate runs of HeartNet with Configuration 51 were executed, and each run resulted in the same clustering of the myogenic and ischemic samples into two clusters. This demonstrates the robustness and reliability of HeartNet when run with well-tuned configurations.

Closer inspection of the clustermap in Figure 5.2 reveals that, with this configuration, HeartNet is not able to differentiate between the finer zones within the class of myogenic samples. There are no patterns to suggest that control samples are, on average, closer to other control samples than to remote zone or border zone samples, or vice versa. This result meets expectations because samples from control hearts, samples from the border zone, and samples from the remote zone are all considered healthy tissue. Although border zone tissue samples are closer to the ischemic zone and could therefore be expected to be structurally more similar to ischemic samples, our results suggest that border zone samples are, on average, closer to remote zone samples than to ischemic zone samples.

Computation Time

The computation time of HeartNet with Configuration 51 on the high-performance cluster with 20 CPU cores and 3 GB of RAM per CPU core is summarized in Table 5.2. Note that factors such as resource sharing, load balancing, and communication overhead, which are associated with parallelized computations in high-performance environments, result in slight variations in computation time between different runs of the same experiment. The values presented in Table 5.2 are averaged values over the six HeartNet runs with Configuration 51 that were performed.

Configuration 52

Configuration 52 varies from Configuration 51 in the value of the s parameter and the number of random walks sampled per node in HeNHoE-2vec. For Configuration 52, we have $s = 3$ and 10 walks per node, while we have $s = 10$ and 20 walks per node for Configuration 51. Figure 1 in Appendix C shows the entropically regularized Gromov-Wasserstein distance matrix that is produced when running HeartNet

with Configuration 52. Using this distance matrix to hierarchically cluster the myogenic and ischemic samples into two clusters with Ward linkage results in a perfect clustering with $ARI = 1$. Figure 26 in Appendix D shows the corresponding dendrogram.

Although the adjusted Rand index indicates that the clustering achieved with Configuration 52 is better than the clustering achieved with Configuration 51, the Gromov-Wasserstein distance matrix of Configuration 52 exhibits significantly less structure than that of Configuration 51. The distance matrix of Configuration 51 more clearly shows the disparity between the myogenic samples and the ischemic samples, and it more clearly visualizes the sample control_P8 as an outlier. The available ground truth labels are not sufficiently informative to determine whether control_P8 is indeed an outlier or whether HeartNet fails to capture the similarity of control_P8 to the other myogenic samples. However, numerous HeartNet experiments with various configurations indicate that control_P8 is more similar to ischemic samples than to myogenic samples, suggesting that the tissue of sample control_P8 is structurally more similar to ischemic tissue. Further investigation into the composition of the tissue would be required to confirm this hypothesis.

The fact that the results from Configuration 51 capture this information better than the results from Configuration 52, despite (or rather therefore) resulting in a clustering with a lower adjusted Rand index, highlights that the ARI is limited in its expressiveness as an indicator of clustering quality. The ARI is merely as informative as the underlying ground truth labels. In our use case, it is clear that the classification into five zones (or three classes) cannot fully capture the complexity of the relationships between human heart tissue samples. It is possible that a clustering that does not fully agree with the ground truth clustering more accurately reflects the structural information of the samples than a clustering that perfectly agrees with the ground truth. Therefore, while the ARI may be used as an objective indicator of clustering quality, it is crucial to manually analyze the distance matrices and dendrograms in order to produce a reliable interpretation of the results.

It should also be noted that the results from Configuration 52 are not as robust as those from Configuration 51. While Configuration 51 produced the same clustering results over six separate HeartNet runs, Configuration 52 produced different clusterings over three separate HeartNet runs, with the ARI ranging from 0.35540 to 1.0. Within the same HeartNet run, different linkage methods for hierarchical clustering also resulted in substantially different clusterings (see Appendix D). This indicates that HeartNet with Configuration 52 captures the structural similarities between different heart tissue samples less accurately and less reliably.

Computation Time

The computation times of the individual stages of HeartNet with Configuration 52 are given in Table 5.2. As with Configuration 51, the values are averages over all executed HeartNet runs with this configuration. Most notably, the computation time of the HeNHoE-2vec embedding with Configuration 52 is significantly shorter than with Configuration 51. We attribute this to the fact that `num walks` $= 10$ for Configuration 52, and `num walks` $= 20$ for Configuration 51, i.e., we sample only half as many random walks with Configuration 52. This leads to quicker sampling of random walks and quicker learning of embeddings using Skip-gram as there are only half as many training samples.

Excluding Spatial Information

A key reason for the choice to represent the heart tissue samples as multilayer networks is the possibility offered hereby to integrate the scRNA-seq and the spatial information of the samples. Ideally, our methods leverage the spatial context in these rich representations to produce more accurate clusterings of the samples based on their internal structures.

To investigate the effect of the addition of spatial information on the quality of the clustering results,

we remove the spatial layer from the multilayer networks of the samples and run HeartNet only on the gene expression layers. The gene expression layer of each network is treated as a single-layer network, and without multiple layers or inter-layer edges, HeNHoE-2vec behaves like standard node2vec. The remaining stages of HeartNet remain unaffected.

We manually tuned the HeartNet hyperparameters in an attempt to achieve the best possible clustering results on only the gene expression layers of the networks. Here, we present the results of running HeartNet with Configuration 64. These are the most promising results we achieved while excluding the spatial information. Refer to Table 2 in Appendix B for an overview of the hyperparameter values used in Configuration 64. Note that all hyperparameters that pertain to the spatial layer or the inter-layer edges are irrelevant to this analysis.

Figure 21 in Appendix C shows the entropically regularized Gromov-Wasserstein distance matrix that is produced when running HeartNet on only the gene expression layers of all samples with Configuration 64. The distance matrix exhibits some structures that are similar to those in Figure 5.1: There is a large lighter region which indicates similarity between myogenic samples, there are darker regions between myogenic and ischemic samples which indicate dissimilarity, and there is no clear pattern between fibrotic samples. However, compared to Figure 5.1, the distance matrix shows less consistency in the dissimilarity between myogenic and ischemic samples, and there is no lighter region which indicates strong similarity between ischemic samples. Furthermore, in the results of all HeartNet experiments on only the gene expression layer, sample IZ_P15 stands out as an outlier, exhibiting relatively strong dissimilarity to all other samples. Interestingly, sample control_P8 does not appear as an outlier when excluding the spatial information. This highlights the fact the integration of gene expression and spatial information is crucial, enabling HeartNet to unveil significant differences between samples, which are not apparent otherwise.

Figure 22 in Appendix C shows the clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the distance matrix in Figure 21 in Appendix C. The clustermap clearly shows that HeartNet is again unable to discriminate the fibrotic samples. Excluding the fibrotic samples, performing hierarchical clustering with Ward linkage, and cutting the dendrogram to yield two clusters results in a perfect clustering of the myogenic and ischemic samples with $ARI = 1.0$. Figure 30 in Appendix D shows the corresponding dendrogram.

However, the clusterings produced by running HeartNet on only the gene expression layer are very inconsistent. HeartNet was executed four times with Configuration 64, and the clusterings of the myogenic and ischemic samples into two clusters with Ward linkage resulted in average Rand indices ranging from 0.17105 to 1.0. The choice of linkage method also significantly impacts the result of the clustering, with single, complete, and average linkage all performing significantly worse than Ward linkage. The inconsistency of the results is visualized in Figure 31 of Appendix E.

Computation Time

The computation times of the individual stages of HeartNet with Configuration 64 on only the gene expression layer are presented in Table 5.2. The gene expression layer of each network contains up to 33 nodes while the spatial layer of each network contains between 1000 and 5000 nodes. Removing the spatial layer results in a drastic reduction in the number of nodes and edges of each network. Given that the runtimes of all stages of HeartNet are proportional to the number of nodes and/or edges per network, excluding the spatial layer from the computations results in a significant decrease in overall computation time, as shown by the values in Table 5.2.

5.3. Hyperparameters

The performance of HeartNet, both with respect to clustering quality and with respect to computation time, is significantly influenced by the choice of hyperparameters. Table 1 in Appendix A summarizes the hyperparameters that regulate the behavior of HeartNet, and Table 2 in Appendix B provides an overview of the configurations we tested during the manual hyperparameter search. Outgoing from an initial configuration with moderate values (Configuration 0), we systematically altered the values of one or multiple hyperparameters between configurations in order to estimate the impact of individual parameters. Figure 32 in Appendix E shows an overview of the ARI scores achieved with different configurations. In the following, we discuss the effects of selected hyperparameters on the behavior of HeartNet and, more specifically, on the resulting quality of clustering achieved in our specific application. We refer to Appendix C throughout, where we display the distance matrices and clustermaps resulting from selected configurations.

5.3.1. Sparsification

As discussed in Section 4.1.1, sparsification of the intra-layer edges and the inter-layer edges reduces the complexity of the networks and may remove noise or redundant information. It may, however, also lead to the removal of more nuanced, yet significant, connections between nodes. Smaller values of k in KNN sparsification lead to sparser networks, resulting in better computation time but also a greater loss of information. As the degree of sparsification impacts the number of edges in each network, computation time is affected in the preprocessing stage of HeartNet and in the pre-computation of transition probabilities in HeNHoE-2vec.

In the gene expression layer, the choice of k does not significantly impact computation time. As the gene expression layer contains at most 33 nodes, a fully connected layer would result in a maximum of $33 \times 32 = 1{,}056$ edges, which is insignificant in comparison to the number of edges in the spatial layer. During our experiments, we observed that increasing the value of k for the gene expression layer leads to a significant improvement in clustering results while having a negligible effect on computation time. Our best results were achieved without sparsifying the gene expression layer at all.

For the spatial layer, which contains between 1000 and 5000 nodes, the choice of k has a more significant impact on computation time. Assuming that the spatial layer contains 4000 nodes, increasing k to $k+1$ results in an increase in the number of edges of 4000. With all other parameters left unchanged, increasing $k = 10$ to $k = 100$ for the spatial layer increases the computation time to precompute the transition probabilities of a single multilayer network from roughly 1 second to roughly 2 minutes. We found that increasing the value of k beyond 10 does not positively impact the clustering results. In fact, setting $k = 100$ greatly diminished the quality of the clustering (see Figures 7 and 8 in Appendix C). Our best results were achieved with a value of $k = 10$ for the spatial layer.

For the inter-layer edges, we performed experiments both with KNN sparsification and with threshold sparsification. In Section 4.1.1, we hypothesized that threshold sparsification would allow the multilayer networks to more accurately reflect the distribution of cell types in the tissue samples and that this may lead to better clustering results. However, our experiments showed that threshold sparsification consistently led to *lower* clustering performance than KNN sparsification (see Figures 9 and 10 in Appendix C). We believe that this is not because threshold sparsification leads to poorer representations of the underlying tissue samples, but rather because it changes the structure of the multilayer networks such that other hyperparameters require re-tuning in order to achieve good results. For instance, the effect of the HeNHoE-2vec switching parameter s will change after threshold sparsification, as some cell-type nodes have few connections to the spatial layer while others have many. We were unable to re-tune the other hyperparameters such that good clustering results were achieved with threshold sparsification. Our best results were achieved by applying KNN sparsification with a value of $k = 10$.

5.3.2. HeNHoE-2vec

The HeNHoE-2vec hyperparameters regulate the accuracy of the node embeddings in capturing the underlying multilayer network structures, thereby significantly influencing the outcome of the final clustering.

For the return parameter p, the in-out parameter q, and the embedding dimensionality d, we adopt the default values suggested by the authors of node2vec [16] ($p = 1$, $q = 0.5$, $d = 128$). Our experiments showed no significant improvements in clustering quality (or computation time) resulting from the variation of these parameters.

The switching parameter s controls the extent to which embeddings of nodes are influenced by their neighboring nodes in other layers. As the node embeddings are intended to capture the relationships between cell types and spatial localization, s should be defined such that the random walks occasionally traverse between layers. However, it may be necessary to limit the probability of switching to ensure that the node embeddings also reflect the distinctions between different layers. Our best results were achieved with a value of $s = 10$ for both directions of switching. Interestingly, a value of $s = 10^6$, i.e., the probability of switching between layers is almost zero, also resulted in a Gromov-Wasserstein distance matrix which very clearly distinguishes between myogenic and ischemic samples. The corresponding distance matrix and clustermap are shown in Figure 11 and Figure 12 of Appendix C, respectively.

Sampling a higher number of random walks per node generally increases the robustness of the captured node neighborhoods. However, as shown in Table 5.2, increasing the number of walks per node also leads to a substantial increase in computation time (Configuration 51 samples twice as many walks per node as Configuration 52). We observed that increasing the number of walks per node from 10 to 20 improved clustering outcomes, while further increases showed no significant improvements.

While the walk length determines the length of the random walks sampled by HeNHoE-2vec, the window size determines the size of the neighborhoods considered by the Skip-gram model during training. Choosing a window size that is smaller than the walk length increases the effective sampling rate because each random walk can be used to infer neighborhoods for multiple nodes. Note that the walk length must be greater than or equal to the window size. Choosing a window size that is too small may result in constrained node neighborhoods that fail to capture the critical structural properties of the networks. Choosing a window size that is too large, however, may result in very similar neighborhoods sampled for all nodes, thus reducing the information content of the node embeddings (see Figures 13 and 14 in Appendix C). The relationship between walk length and window size also influences computation time. An increase in walk length will lead to a slight increase in computation time during neighborhood sampling as longer walks must be sampled. Reducing the window size effectively increases the number of training samples for the Skip-gram model and therefore leads to an increase in computation time. Our best results were obtained with a walk length of 20 and a window size of 10.

The default number of epochs for which the Skip-gram model is trained in node2vec is 1. While an increase in the number of epochs leads to a substantial increase in computation time required to learn the node embeddings, it also significantly improves the clustering results. For instance, Configuration 51 trains the Skip-gram model for 1000 epochs, requiring about 10.5 hours to compute the node embeddings (see Table 5.2). Reducing the number of epochs to 1 while leaving all other parameters unchanged (Configuration 70) results in a computation time of only 5.5 minutes to compute the node embeddings. However, the quality of the clustering obtained from Configuration 70 is significantly lower than that of the clustering obtained from Configuration 51 (see Figures 15 and 16 in Appendix C). We found that increasing the number of epochs beyond 1000 does not result in significantly better clustering results (see Figures 17 and 18 in Appendix C).

5.3.3. Gromov-Wasserstein Optimal Transport

The regularization weight epsilon of the entropically regularized Gromov-Wasserstein optimal transport problem (Eq. 3.24) influences the entropy of the solution. We observed that higher values of epsilon result in more diffused Gromov-Wasserstein distance matrices, while lower values lead to more localized distance matrices. The computation time of the Sinkhorn-Knopp algorithm [25, 26] used to solve the optimization problem significantly increases for very small values of epsilon. We found that moderate regularization of the Gromov-Wasserstein optimal transport problem generally improves the accuracy and robustness of the clustering results. Our best results were achieved with an epsilon value of 0.05.

In Section 4.3, we hypothesized that "balancing" the discrete probability distributions in the source and target spaces of the Gromov-Wasserstein optimal transport problem would improve clustering results by amplifying the influence of the gene expression layer. During our experiments, however, we observed that balancing the probability distributions generally leads to a decrease in the quality of the clustering results (see Figures 19 and 20 in Appendix C). We hypothesize that this is due to the fact that the neighborhoods of the spot nodes also include cell-type nodes from the gene expression layer. Thus, the embeddings of the spot nodes comprise a significant amount of gene expression data, and balancing the probability distributions diminishes the influence of this information. Our best results were achieved with discrete uniform probability distributions.

5.4. Discussion

The results presented in Section 5.2 demonstrate that HeartNet is able to accurately and reliably cluster the myogenic and ischemic heart tissue samples into their respective groups. With well-tuned configurations, the clusterings produced by HeartNet are robust to the choice of linkage method in hierarchical clustering and to the pseudo-randomness inherent in the neighborhood sampling strategy of HeNHoE-2vec. However, HeartNet fails to discriminate fibrotic tissue samples from myogenic and ischemic samples, and it is unable to distinguish between the different zones within the class of myogenic samples (border zone, remote zone, and control). Our results do not permit concluding whether this is because HeartNet is unable to capture more subtle differences between tissue samples or whether these tissue samples in fact do not exhibit structural differences. Further investigation into the composition of the heart tissue samples is necessary to answer this question.

The results of running HeartNet only on the gene expression layers of the multilayer networks show that the incorporation of spatial information is critical in achieving accurate and reliable clustering results. Although the distance matrices produced under consideration of only the gene expression layer also display structural differences between myogenic and ischemic samples, these patterns are considerably less pronounced. This is reflected in the clustering results, which achieve lower ARI scores and exhibit significant fluctuations between runs and between linkage methods.

An extensive analysis of the effects of different hyperparameter values showed that the choice of hyperparameters significantly impacts the computation time and the clustering results of HeartNet. Given the relatively large number of hyperparameters and the computational requirements of HeartNet, a grid search to determine the optimal hyperparameters is computationally infeasible. Thus, manual hyperparameter tuning is required. We consider the results obtained with Configuration 51 to most accurately reflect the relationships between the heart tissue samples. Table 2 in Appendix B presents an overview of all configurations that were tested during our experiments.

The computation time of HeartNet with appropriate configurations lies in the range of 10 to 15 hours on the high-performance computing cluster of the Institute for Computational Genomics at RWTH Aachen University. While these computation times are considerable, they are still manageable and may be reduced through further hyperparameter tuning and further optimization for parallel computation.

5.4.1. Limitations

The time-consuming and computationally intensive manual hyperparameter tuning process is a major weakness of HeartNet. The hyperparameters must be tuned such that HeartNet accurately and reliably captures the structural similarities and differences of the multilayer networks that are to be clustered. Given that the structure of the multilayer networks can vary heavily between applications, the hyperparameters must be individually tuned for each application. This process requires the presence of ground truth labels for a representative subset of samples in order to quantify the quality of the clusterings and adjust the hyperparameters accordingly. Attaining ground truth labels may be difficult or impossible in some applications.

Another weakness is the challenge of objectively quantifying the quality of clusterings. While the adjusted Rand index provides a good indication of the agreement between a predicted clustering and the ground truth clustering, its expressiveness about the quality of a clustering is limited to the information content of the ground truth. If the ground truth does not fully reflect the relationships between samples, the ARI cannot fully reflect these relationships either. As a result, clusterings might be attained such that a clustering with a lower ARI more accurately reflects the relationships between samples than a clustering with a higher ARI (as we believe is the case for the results obtained with configurations 51 and 52, for example). This limitation results in the need to manually analyze and interpret distance matrices and dendrograms, which is time-consuming and prone to other types of inaccuracies.

Furthermore, the computation time of HeartNet may lead to restrictions in its application. Very large multilayer networks may contain too many nodes and/or edges for a feasible computation of node embeddings. This results in the need for sparsification or dimensionality reduction of the networks, which may lead to the loss of relevant information. In our specific application, the single cells from the scRNA-seq measurements were clustered into cell types to reduce the size of the networks. It is likely that a significant amount of information was lost in this process, which otherwise might have resulted in more accurate clustering outcomes. It is also important to note that the computational time requirement for the calculation of pairwise Gromov-Wasserstein distances is quadratic in the number of networks to be clustered. This will likely lead to computational challenges when attempting to cluster a much greater number of multilayer networks than in our application.

5.4.2. Generalizability

Although we developed HeartNet with the specific application of clustering heart tissue samples in mind, the clustering of multilayer networks is a task that extends to any scenario where complex entities that can be represented as multilayer networks of measured features need to be clustered. Furthermore, there might be additional modalities of single-cell data in the future which we would like to be able to incorporate into our multilayer networks as additional layers, thereby increasing their expressiveness. With these considerations in mind, we developed all stages of HeartNet such that they generalize easily to other settings where networks may be directed and/or consist of a different number of layers.

The preprocessing stages of HeartNet are applied to each layer and each set of inter-layer edges independently and are therefore easily extendable to networks with any number of layers. We developed our implementation of KNN sparsification to be applicable to both undirected and directed networks. Our HeNHoE-2vec implementation [35] was also designed to support both undirected and directed multilayer networks with an arbitrary number of layers. The remaining stages of HeartNet are independent of the structure of the multilayer networks and are therefore directly applicable in other settings.

These design choices ensure a large degree of generalizability as they allow HeartNet to be applied to any finite collection of multilayer networks which fulfill the definition of multilayer networks provided in Section 1.2.

5.4.3. Use Cases

We believe that there are three main use cases for clustering multilayer networks using HeartNet:

1. In the absence of ground truth labels, HeartNet may be used to cluster multilayer networks to discover structural similarities between the networks. This assumes, however, that the hyperparameters of HeartNet have previously been tuned to effectively cluster multilayer networks with similar structures.

2. In applications where ground truth labels for a representative subset of samples are available, HeartNet can be tuned to effectively cluster the labeled samples. The labeled samples can then be clustered together with unlabeled samples to predict the classes of the unlabeled samples based on the classes of the labeled samples that they were clustered with.

3. HeartNet may be used to unveil previously unknown relationships between labeled samples. For instance, clusterings produced by HeartNet can indicate outliers or labeling mistakes by revealing significant structural differences between samples that were previously believed to belong to the same cluster or class.

6. Conclusion

This thesis aimed to develop a general method of clustering multilayer networks based on structural similarities. To evaluate the performance of our proposed solution, we utilized a dataset of human heart tissue samples from different zones of ischemic heart disease, provided by the Institute for Computational Genomics at RWTH Aachen University Hospital. The single-cell RNA sequencing (scRNA-seq) and spatial transcriptomics data of each heart tissue sample were integrated into a unified multilayer network representation.

We introduced *HeartNet*, a highly modular and configurable pipeline that encompasses preprocessing, multilayer network embedding via the HeNHoE-2vec algorithm, and Gromov-Wasserstein distance calculations to cluster multilayer networks based on structural similarities. We demonstrated that HeartNet is able to accurately and reliably cluster the myogenic and ischemic heart tissue samples into their respective groups, and we proved its robustness against the choice of linkage method and the pseudo-randomness associated with HeNHoE-2vec. However, our experiments showed that HeartNet faces challenges in discriminating fibrotic samples and in distinguishing finer zones within the myogenic class. These limitations open opportunities for further investigation to determine whether they arise from HeartNet's inability to capture more nuanced patterns in networks or from the samples themselves lacking significant structural differences.

A key strength of HeartNet is its ability to process multilayer networks which enable rich representations of complex entities with multi-dimensional features. The multilayer network representations of heart tissue samples enabled the integration of scRNA-seq and spatial transcriptomics data. This proved to significantly enhance the accuracy and robustness of HeartNet's clustering outcomes compared to clusterings produced using scRNA-seq data alone. We also demonstrated that HeartNet is useful for the identification of outliers, as exemplified by the sample control_P8 which is part of the myogenic class but displayed unexpected similarities to ischemic samples throughout our experiments.

With HeartNet, we provide a domain-agnostic and highly customizable tool for the clustering of arbitrary entities which can be represented as multilayer networks of measured features. HeartNet is applicable to both directed and undirected multilayer networks with an arbitrary number of layers, allowing for its application in diverse settings. While HeartNet's high flexibility is one of its key strengths, it also necessitates a time-consuming hyperparameter tuning process, marking an area for future optimization.

Looking ahead, several research directions emerge. It would be interesting to investigate the composition of the fibrotic tissue samples to reveal why some fibrotic samples exhibit strong similarity to myogenic samples while others exhibit strong similarity to ischemic samples in our results. It would further be interesting to explore the reasons for sample control_P8 exhibiting strong similarity to ischemic samples despite being classified as a myogenic sample. Future work might also focus on optimizing the hyperparameter tuning process of HeartNet and determining whether further hyperparameter optimization would allow HeartNet to make finer distinctions between IHD zones. Another promising avenue is to increase the information content of the gene expression layer by replacing cell-type nodes with single-cell nodes. Although this would likely lead to higher computation times, there is a significant potential for improved clustering results as the expressiveness of the multilayer network representations of the tissue samples is increased. Finally, applying HeartNet in different domains is crucial for the validation of its general applicability and versatility.

Bibliography

[1] J. Eberwine, J.-Y. Sul, T. Bartfai, and J. Kim, "The promise of single-cell sequencing," *Nature Methods*, vol. 11, no. 1, pp. 25–27, 2014.

[2] S. Linnarsson and S. A. Teichmann, "Single-cell genomics: coming of age," *Genome Biology*, vol. 17, no. 1, p. 97, 2016.

[3] V. Marx, "Method of the year: spatially resolved transcriptomics," *Nature Methods*, vol. 18, no. 1, pp. 9–14, 2021.

[4] F. Tang, C. Barbacioru, Y. Wang, E. Nordman, C. Lee, N. Xu, X. Wang, J. Bodeau, B. B. Tuch, A. Siddiqui, K. Lao, and M. A. Surani, "mRNA-seq whole-transcriptome analysis of a single cell," *Nature Methods*, vol. 6, no. 5, pp. 377–382, 2009.

[5] M. D. Luecken and F. J. Theis, "Current best practices in single-cell RNA-seq analysis: a tutorial," *Molecular Systems Biology*, vol. 15, Jun 2019.

[6] J. A. Briggs, C. Weinreb, D. E. Wagner, S. Megason, L. Peshkin, M. W. Kirschner, and A. M. Klein, "The dynamics of gene expression in vertebrate embryogenesis at single-cell resolution," *Science*, vol. 360, no. 6392, p. eaar5780, 2018.

[7] L. Li, F. Xiong, Y. Wang, S. Zhang, Z. Gong, X. Li, Y. He, L. Shi, F. Wang, Q. Liao, B. Xiang, M. Zhou, X. Li, Y. Li, G. Li, Z. Zeng, W. Xiong, and C. Guo, "What are the applications of single-cell RNA sequencing in cancer research: a systematic review," *Journal of Experimental & Clinical Cancer Research*, vol. 40, no. 1, p. 163, 2021.

[8] C. Kuppe, R. O. Ramirez Flores, Z. Li, S. Hayat, R. T. Levinson, X. Liao, M. T. Hannani, J. Tanevski, F. Wünnemann, J. S. Nagai, M. Halder, D. Schumacher, S. Menzel, G. Schäfer, K. Hoeft, M. Cheng, S. Ziegler, X. Zhang, F. Peisker, N. Kaesler, T. Saritas, Y. Xu, A. Kassner, J. Gummert, M. Morshuis, J. Amrute, R. J. A. Veltrop, P. Boor, K. Klingel, L. W. Van Laake, A. Vink, R. M. Hoogenboezem, E. M. J. Bindels, L. Schurgers, S. Sattler, D. Schapiro, R. K. Schneider, K. Lavine, H. Milting, I. G. Costa, J. Saez-Rodriguez, and R. Kramann, "Spatial multi-omic map of human myocardial infarction," *Nature*, vol. 608, no. 7924, pp. 766–777, 2022.

[9] M. Nitzan, N. Karaiskos, N. Friedman, and N. Rajewsky, "Gene expression cartography," *Nature*, vol. 576, no. 7785, pp. 132–137, 2019.

[10] N. Moriel, E. Senel, N. Friedman, N. Rajewsky, N. Karaiskos, and M. Nitzan, "NovoSpaRc: flexible spatial reconstruction of single-cell gene expression with optimal transport," *Nature Protocols*, vol. 16, no. 9, pp. 4177–4200, 2021.

[11] P. Demetci, R. Santorella, B. Sandstede, W. S. Noble, and R. Singh, "SCOT: Single-cell multi-omics alignment with optimal transport," *Journal of Computational Biology*, vol. 29, pp. 3–18, Jan. 2022.

[12] F. Mémoli, "On the use of Gromov–Hausdorff distances for shape comparison," *Eurographics Symposium on Point-Based Graphics*, 2007.

[13] G. Peyre, M. Cuturi, and J. Solomon, "Gromov-Wasserstein averaging of kernel and distance matrices," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, 2016.

[14] "Institute for Computational Genomics at RWTH Aachen University Hospital." https://www.costalab.org. Accessed: 2023-04-06.

[15] "10x Genomics." https://www.10xgenomics.com/support/spatial-gene-expression-fresh-frozen. Accessed: 2023-04-06.

[16] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[17] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pp. 701–710, ACM, 2014.

[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *Proceedings of Workshop at ICLR*, vol. 2013, 2013.

[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems* (C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, eds.), vol. 26, Curran Associates, Inc., 2013.

[20] G. Valentini, E. Casiraghi, L. Cappelletti, V. Ravanmehr, T. Fontana, J. Reese, and P. Robinson, "Het-node2vec: second order random walk sampling for heterogeneous multigraphs embedding," 2021.

[21] L. V. Kantorovich, "On the translocation of masses," *Journal of Mathematical Sciences*, vol. 133, no. 4, pp. 1381–1382, 2006.

[22] L. V. Kantorovich, "Mathematical methods of organizing and planning production," *Management Science*, vol. 6, pp. 366–422, July 1960.

[23] L. N. Vaserstein, "Markov processes over denumerable products of spaces, describing large systems of automata," *Probl. Peredachi Inf.*, vol. 5, pp. 64–72, 1969.

[24] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Advances in Neural Information Processing Systems* (C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, eds.), vol. 26, Curran Associates, Inc., 2013.

[25] R. Sinkhorn and P. Knopp, "Concerning nonnegative matrices and doubly stochastic matrices," *Pacific Journal of Mathematics*, vol. 21, no. 2, pp. 343 – 348, 1967.

[26] G. Peyré and M. Cuturi, "Computational optimal transport," *Foundations and Trends in Machine Learning*, vol. 11, no. 5–6, pp. 355–607, 2019.

[27] F. Mémoli, "Gromov–Wasserstein distances and the metric approach to object matching," *Foundations of Computational Mathematics*, vol. 11, no. 4, pp. 417–487, 2011.

[28] F. Nielsen, *Hierarchical Clustering*, pp. 195–211. Cham: Springer International Publishing, 2016.

[29] R. Sibson, "SLINK: an optimally efficient algorithm for the single-link cluster method," *The computer journal*, vol. 16, no. 1, pp. 30–34, 1973.

[30] D. Defays, "An efficient algorithm for a complete link method," *The computer journal*, vol. 20, no. 4, pp. 364–366, 1977.

[31] R. R. Sokal and C. D. Michener, "A statistical method for evaluating systematic relationships," *University of Kansas Science Bulletin*, vol. 38, pp. 1409–1438, 1958.

[32] J. H. Ward Jr, "Hierarchical grouping to optimize an objective function," *Journal of the American statistical association*, vol. 58, no. 301, pp. 236–244, 1963.

[33] G. N. Lance and W. T. Williams, "A generalized sorting strategy for computer classifications," *Nature*, vol. 212, no. 5058, pp. 218–218, 1966.

[34] R. M. Cormack, "A review of classification," *Journal of the Royal Statistical Society: Series A (General)*, vol. 134, no. 3, pp. 321–353, 1971.

[35] R. M. Giesler, "HeNHoE-2vec." https://github.com/RobertGiesler/HeNHoE-2vec, 2023. Accessed: 2023-08-23.

[36] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proceedings of the 7th Python in Science Conference* (G. Varoquaux, T. Vaught, and J. Millman, eds.), (Pasadena, CA USA), pp. 11 – 15, 2008.

[37] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, (Valletta, Malta), pp. 45–50, ELRA, May 2010.

[38] H. Krekel, B. Oliveira, R. Pfannschmidt, F. Bruynooghe, B. Laugher, and F. Bruhin, "pytest." https://github.com/pytest-dev/pytest, 2004. Accessed: 2023-08-23.

[39] R. Flamary, N. Courty, A. Gramfort, M. Z. Alaya, A. Boisbunon, S. Chambon, L. Chapel, A. Corenflos, K. Fatras, N. Fournier, L. Gautheron, N. T. Gayraud, H. Janati, A. Rakotomamonjy, I. Redko, A. Rolet, A. Schutz, V. Seguy, D. J. Sutherland, R. Tavenard, A. Tong, and T. Vayer, "POT: Python Optimal Transport," *Journal of Machine Learning Research*, vol. 22, no. 78, pp. 1–8, 2021.

[40] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics Quarterly*, vol. 3, pp. 95–110, Mar. 1956.

[41] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[42] M. L. Waskom, "seaborn: statistical data visualization," *Journal of Open Source Software*, vol. 6, no. 60, p. 3021, 2021.

[43] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[44] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, no. 336, pp. 846–850, 1971.

[45] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193–218, 1985.

Appendices

A. HeartNet Parameters

Parameter	Type	Description	Notes
Sparsification			
`spot k`	Integer or `None`	k used in the KNN sparsification of the spatial layer.	If `None`, spatial layer is not sparsified.
`celltype k`	Integer or `None`	k used in the KNN sparsification of the gene expression layer.	If `None`, gene expression layer is not sparsified.
`interlayer method`	"`knn`", "`threshold`", or `None`	Sparsification method for the inter-layer edges.	If `None`, inter-layer edges are not sparsified. If set to "`threshold`", the threshold value is defined according to Eq. 4.33.
`interlayer k`	Integer	k used in the KNN sparsification of the inter-layer edges.	Only applies if `interlayer method` = "`knn`".
HeNHoE-2vec			
`p`	Float	Return parameter.	
`q`	Float	In-out parameter.	
`s`	Float or list of dictionaries	Switching parameter(s).	If different switching parameters are used for different layer pairs, these are defined in a list of dictionaries.
`dims`	Integer	Dimensionality of the node embeddings.	
`num walks`	Integer	Number of random walks per node.	
`walk length`	Integer	Length of each random walk.	
`window size`	Integer	Skip-gram window size.	
`epochs`	Integer	Number of Skip-gram epochs used to learn the embeddings.	
Gromov-Wasserstein			
`epsilon`	Float	Regularization weight of the regularized Gromov-Wasserstein optimal transport problem.	
`distribution`	"`uniform`" or "`balanced`"	Whether each node has the same "mass" (`uniform`) or whether each layer has a fixed fraction of the total mass which is distributed amongst its nodes (`balanced`).	Defines the probability distributions in the source and target space.
`alpha`	Float	The fraction of the total mass divided amongst cell-type nodes. E.g., if `alpha = 0.8`, the total mass of all cell-type nodes sums up to 0.8 and the total mass of all spot nodes sums up to 0.2.	Must be between 0 and 1. Only applies if `distribution` = "`balanced`".
Hierarchical Clustering			
`linkage method`	"`single`", "`complete`", "`average`", or "`ward`"	Linkage method used in hierarchical clustering.	
`n clusters`	Integer	Number of clusters to be attained when cutting the dendrogram.	

Table 1. | Configuration parameters of HeartNet.

B. HeartNet Configurations

Table 2. | All HeartNet configurations tested during our experiments.

#	spot k	celltype k	interlayer k	interlayer stds	p	q	s c2s	s s2c	dims	num walks	walk length	window size	epochs	epsilon	distribution	alpha
0	10	5	10	-	1	0.5	3	1	128	10	20	10	1	0	uniform	-
1	10	5	10	-	1	0.5	3	1	128	10	10	10	1	0	uniform	-
2	10	5	10	-	1	0.5	3	1	128	10	10	10	1	0	uniform	-
3	10	5	10	-	1	0.5	3	1	128	10	30	15	10	0	uniform	-
4	10	5	10	-	1	0.5	3	1	128	10	30	15	1	0	uniform	-
5	10	5	10	-	1	0.5	3	3	128	10	20	10	1	0	uniform	-
6	10	5	10	-	1	0.5	3	3	128	10	20	10	1	0	uniform	-
7	10	5	10	-	1	0.5	3	1	64	10	20	10	1	0	uniform	-
8	10	10	10	-	1	0.5	3	1	128	10	20	10	1	0	uniform	-
(9)	10	5	10	-	1	0.5	3	1	128	10	20	10	1	0	balanced	1
(10)	10	5	10	-	1	0.5	3	1	128	10	20	10	1	0	balanced	0.25
11	10	5	10	-	1	0.5	3	1	128	10	20	10	1	0	balanced	0.5
12	10	5	10	-	1	0.5	3	1	128	10	20	10	1	0	balanced	1
13	10	5	10	-	1	0.5	3	1	128	10	20	10	1	0	balanced	0
14	10	5	-	1	1	0.5	3	1	128	10	20	10	1	0	balanced	0.5
15	10	5	-	1	1	0.5	3	1	128	10	20	10	1	0	uniform	-
16	10	10	-	1	1	0.5	3	1	128	20	30	20	1	0	balanced	0.5
17	10	10	-	1	1	0.5	3	1	128	10	20	10	1	0	balanced	0.5
18	10	5	-	1	0.5	0.2	3	1	128	10	20	10	1	0	balanced	0.5
19	10	10	-	1	0.5	0.2	10	10	128	10	20	10	1	0	balanced	0.5
20	10	10	-	1	1	0.5	10	10	128	10	20	10	1	0	balanced	0.5
21	10	10	-	1	1	0.5	10	10	128	10	20	10	1	0	uniform	-
22	10	10	-	1	2	0.5	10	5	128	20	10	10	1	0	balanced	0.5
23	10	5	-	1	2	0.5	10	5	128	20	10	10	1	0	balanced	0.5
24	10	10	-	1	2	0.5	5	0.001	128	20	10	10	1	0	balanced	1
25	10	5	-	1	1	0.5	3	1	128	40	5	5	1	0	balanced	0.5
26	10	5	-	1	1	0.5	3	1	128	40	5	5	1	0.0005	uniform	-
27	10	5	-	1	1	0.5	3	1	128	10	20	10	1000	0	balanced	0.5
28	10	5	10	-	1	0.5	3	1	128	20	20	10	1000	0.05	uniform	-
29	10	5	10	-	1	0.5	3	1	128	10	20	10	1000	0.05	uniform	-
30	10	5	10	-	1	0.5	3	1	128	10	20	10	500	0.05	uniform	-
31	10	5	10	-	1	0.5	3	1	128	10	20	10	100	0.05	uniform	-
32	10	5	10	-	1	0.5	3	1	128	10	20	10	1	0.05	uniform	-
33	10	5	10	-	1	0.5	3	1	128	10	20	10	1	0.005	uniform	-
34	10	5	10	-	1	0.5	3	1	128	5	20	10	1	0.05	uniform	-
35	10	5	10	-	1	0.5	3	1	128	20	20	10	1	0.05	uniform	-
36	10	5	10	-	1	0.5	3	1	128	10	20	5	1	0.05	uniform	-
37	10	5	10	-	1	0.5	3	1	128	10	20	20	1	0.05	uniform	-
38	10	5	10	-	1	0.5	3	1	128	10	30	30	1	0.05	uniform	-
39	10	5	10	-	1	0.5	3	1	128	10	20	10	1	0.1	uniform	-
40	10	5	-	1	1	0.5	10	10	128	10	20	10	1	0.05	uniform	-
41	10	5	-	1	1	0.5	10	10	128	10	20	5	1	0.05	uniform	-
42	10	5	10	-	1	0.5	10	10	128	10	20	10	1	0.05	uniform	-
43	10	5	10	-	1	0.5	100	100	128	10	20	10	1	0.05	uniform	-
44	10	5	-	2	1	0.5	3	1	128	10	20	10	1	0.05	uniform	-
45	10	5	10	-	1	0.5	10	10	128	10	20	10	1000	0.05	uniform	-
46	10	5	10	-	1	0.2	10	10	128	10	20	10	1	0.05	uniform	-

Continued on next page

#	spot k	celltype k	interlayer k	interlayer stds	p	q	s c2s	s s2c	dims	num walks	walk length	window size	epochs	epsilon	distribution	alpha
47	10	10	10	-	1	0.5	10	10	128	10	20	10	1	0.05	uniform	-
48	10	10	10	-	1	0.5	10	10	128	10	20	10	1000	0.05	uniform	-
49	10	10	10	-	1	0.5	10	10	128	20	20	10	1000	0.05	uniform	-
50	10	-	10	-	1	0.5	10	10	128	10	20	10	1000	0.05	uniform	-
51	10	-	10	-	1	0.5	10	10	128	20	20	10	1000	0.05	uniform	-
52	10	-	10	-	1	0.5	3	3	128	10	20	10	1000	0.05	uniform	-
53	10	-	10	-	1	0.5	10^6	10^6	128	10	20	10	1000	0.05	uniform	-
54	10	-	10	-	1	0.5	10^6	10^6	128	10	20	10	1000	0.05	balanced	1
55	10	-	10	-	1	0.5	10	3	128	20	20	10	1000	0.05	uniform	-
56	10	-	-	1	1	0.5	10	10	128	20	20	10	1000	0.05	uniform	-
57	10	-	-	1	1	0.5	10	10	128	20	30	20	1000	0.05	uniform	-
58	10	-	10	-	1	0.5	10	10	128	20	20	10	1000	0.05	balanced	0.5
59	10	-	10	-	1	0.5	10	10	128	10	10	10	1000	0.05	uniform	-
60	20	-	10	-	1	0.5	10	10	128	20	20	10	1000	0.05	uniform	-
61	10	-	10	-	1	0.5	10	10	128	20	20	10	500	0.05	uniform	-
62	10	-	10	-	1	0.5	10	10	128	20	20	10	100	0.05	uniform	-
63	10	-	10	-	1	0.5	10	10	128	20	20	20	1000	0.05	uniform	-
64	10	-	10	-	1	0.5	10	10	128	20	20	20	1000	0.005	uniform	-
65	100	-	10	-	1	0.5	10	10	128	20	20	10	1000	0.05	uniform	-
(66)	-	-	10	-	1	0.5	10	10	128	20	20	10	1000	0.05	uniform	-
67	10	-	10	-	1	0.5	100	100	128	20	20	10	1000	0.05	uniform	-
68	10	-	10	-	1	0.5	10	10	128	30	20	10	1000	0.05	uniform	-
69	10	-	10	-	1	0.5	10	10	128	20	20	10	2000	0.05	uniform	-
70	10	-	10	-	1	0.5	10	10	128	20	20	10	1	0.05	uniform	-

Table 2 shows all HeartNet configurations which were tested during our experiments. The index # indicates the configuration number assigned to each configuration. For the parameters "spot k" and "celltype k", a missing value indicates that the corresponding layer was not sparsified. A missing value in "interlayer k" indicates that the inter-layer edges were sparsified using the thresholding method outlined in Section 4.1.1. The value in "interlayer stds" indicates the number of standard deviations away from the mean where the threshold value was set, i.e., if interlayer stds $= 2$, we set $t = \mu(A) + 2\sigma(A)$ instead of $t = \mu(A) + \sigma(A)$ (see Eq. 4.33). The value in "s c2s" indicates the value of the HeNHoE-2vec switching parameter from the gene expression layer to the spatial layer, and the value in "s s2c" indicates the value of the switching parameter from the spatial layer to the gene expression layer. For configurations where epsilon $= 0$, only the unregularized Gromov-Wasserstein optimal transport optimization problem (Eq. 3.23) was solved. We do not list the parameters for hierarchical clustering because any form of hierarchical clustering can easily be applied to the Gromov-Wasserstein distances resulting from the above configurations. Experiments with configuration numbers in parentheses (9, 10, 66) could not be completed due to time constraints or other issues.

C. Distance Matrices and Clustermaps

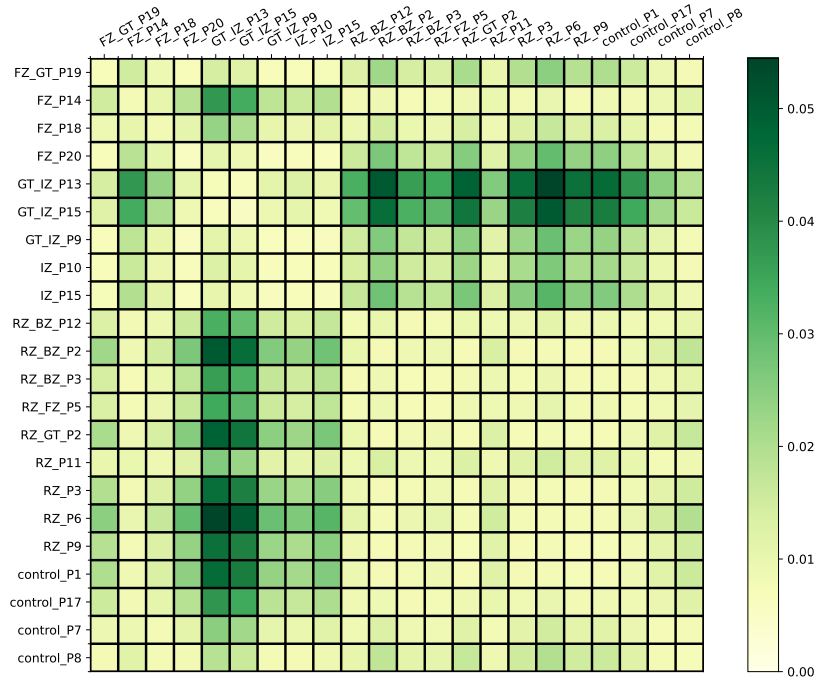


Figure 1. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 52**.

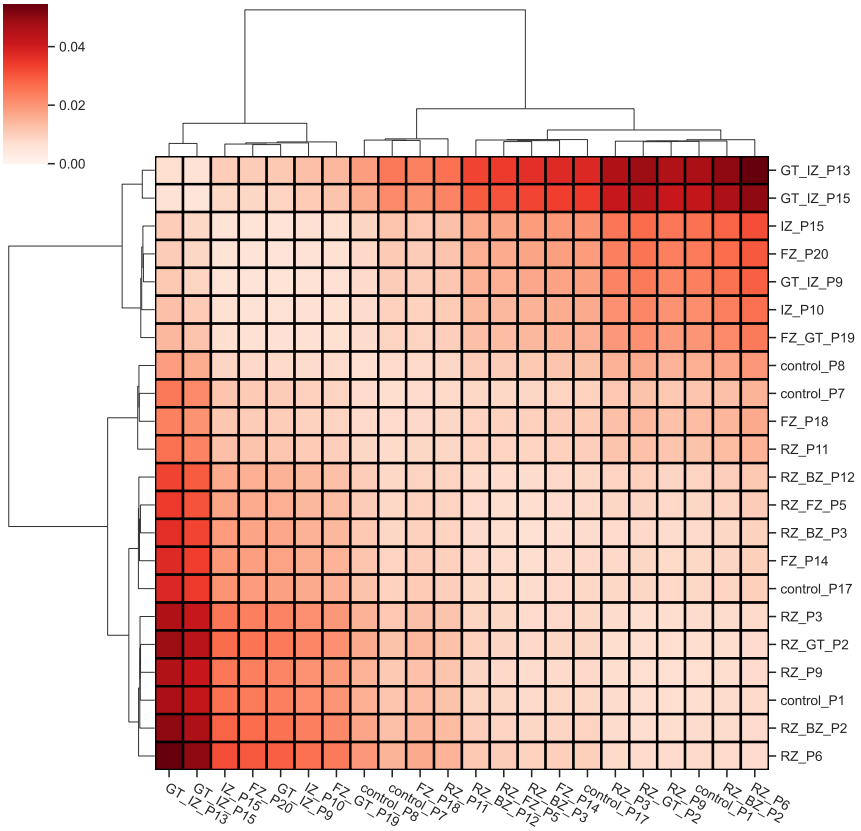


Figure 2. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 52**.

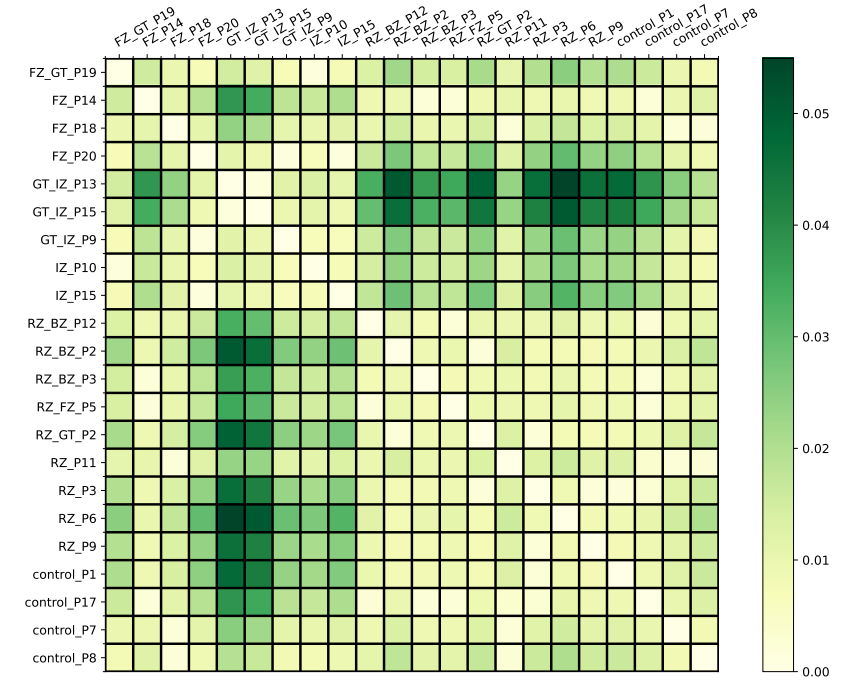


Figure 3. | The **unregularized** Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 52**.

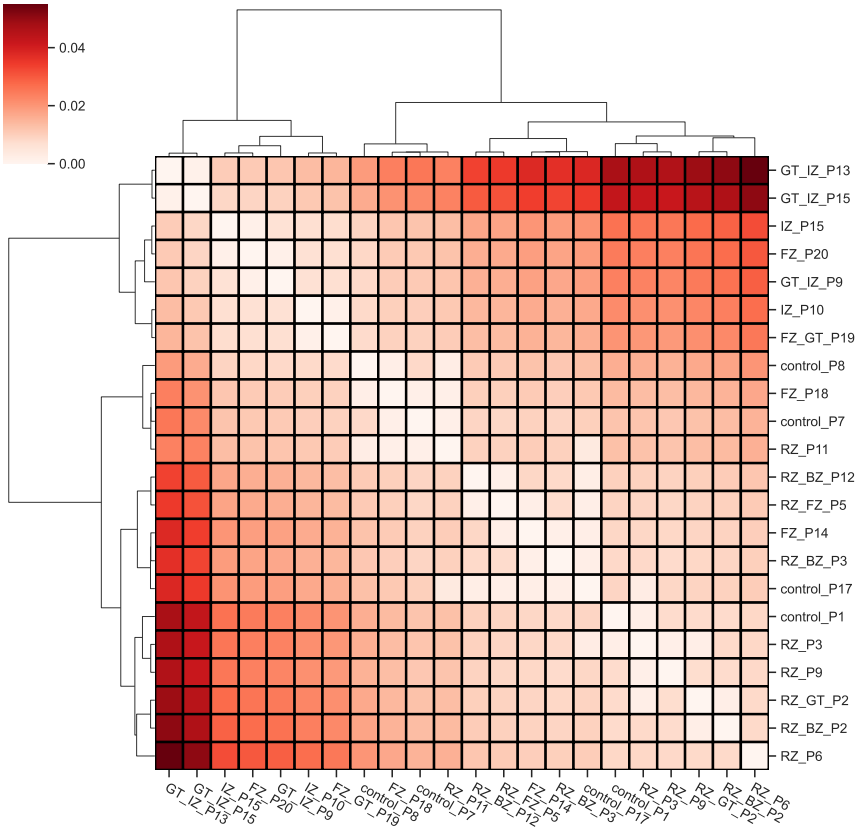


Figure 4. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the **unregularized** Gromov-Wasserstein distance calculated using HeartNet with **Configuration 52**.

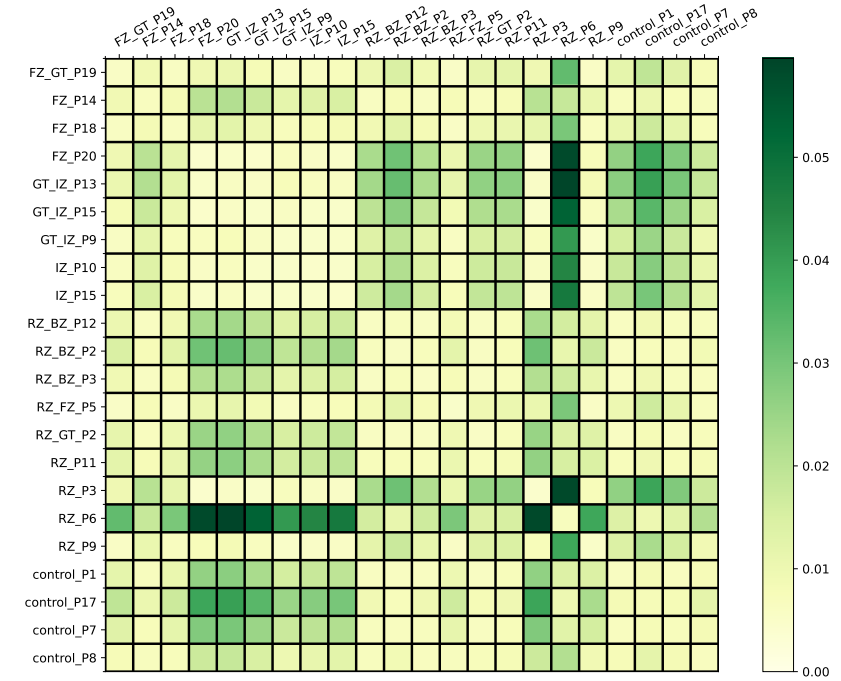


Figure 5. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 45**. Notably, this configuration sparsifies the gene expression layer with $k = 5$.

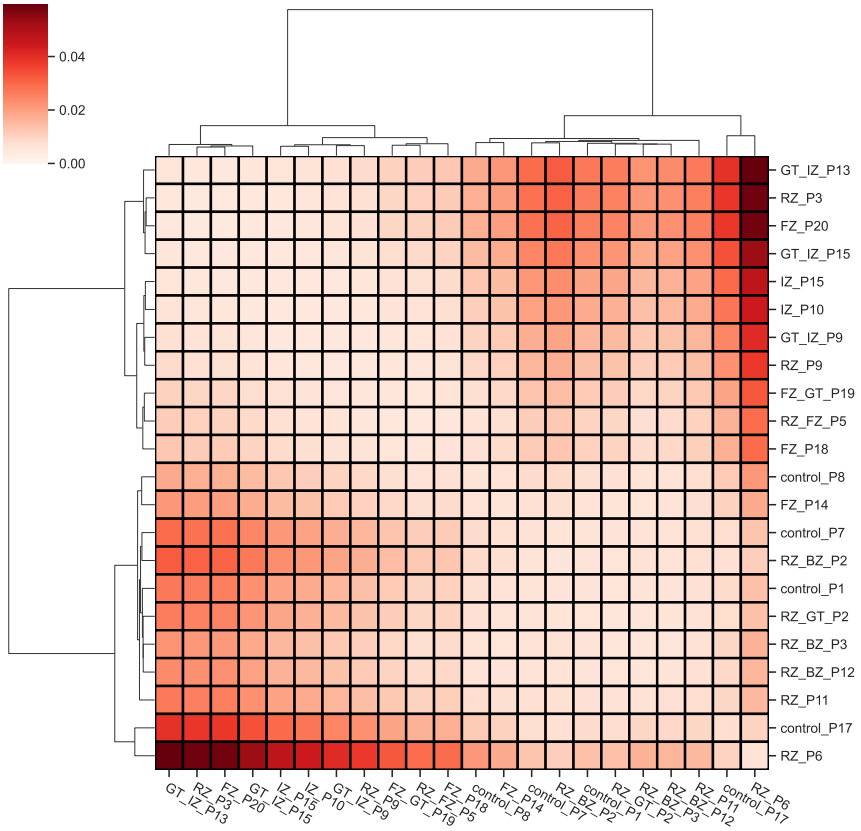


Figure 6. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 45**. Notably, this configuration sparsifies the gene expression layer with $k = 5$.

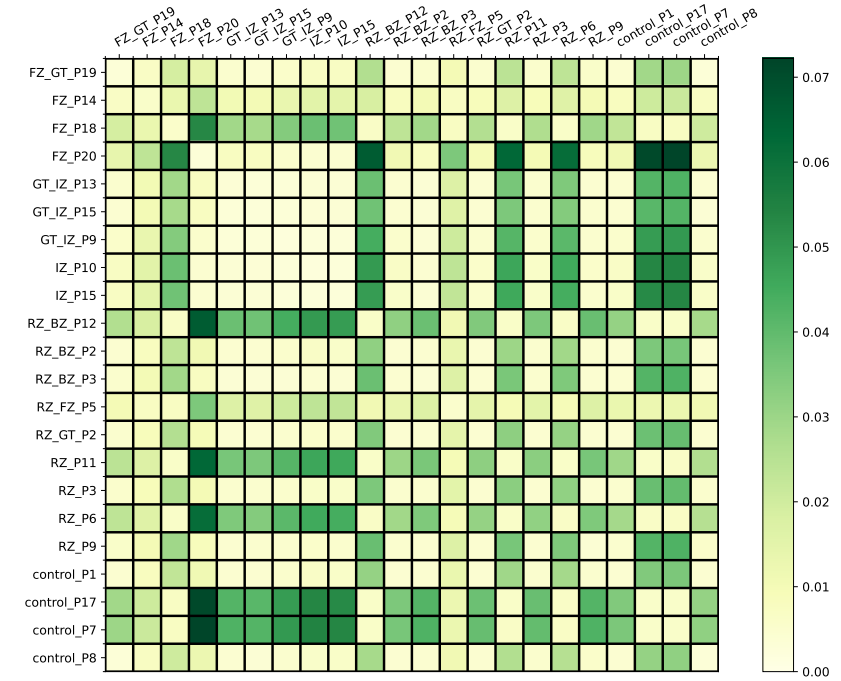


Figure 7. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 65**. Notably, this configuration sparsifies the spatial layer with $k = 100$.

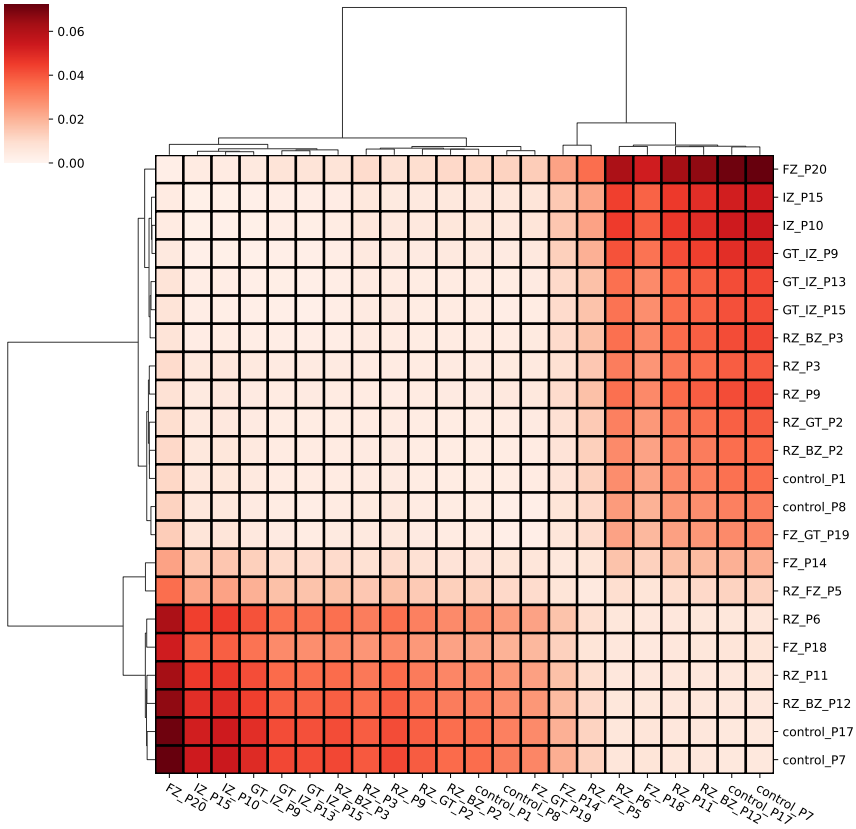


Figure 8. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 65**. Notably, this configuration sparsifies the spatial layer with $k = 100$.

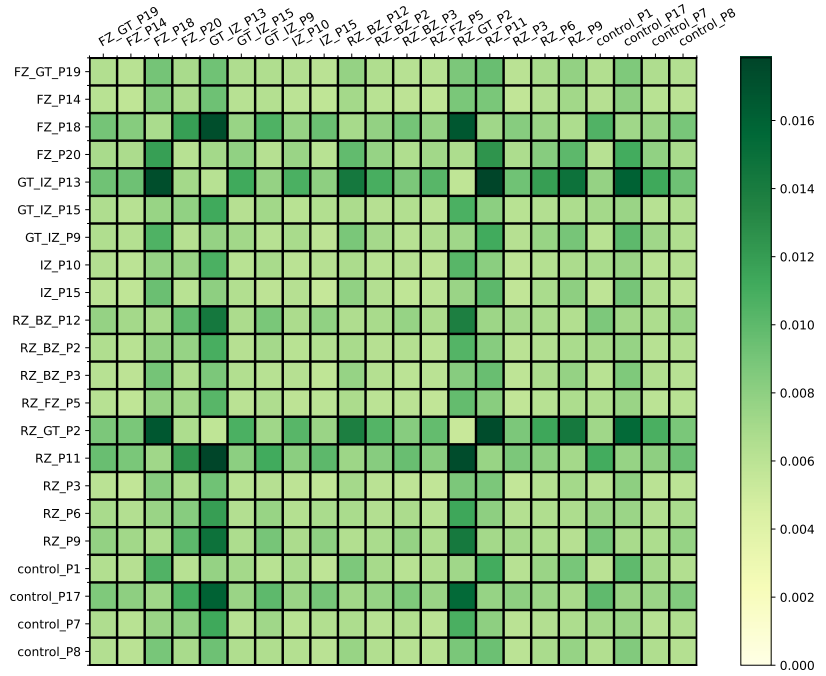


Figure 9. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 56**. Notably, this configuration applies threshold sparsification to the inter-layer edges.

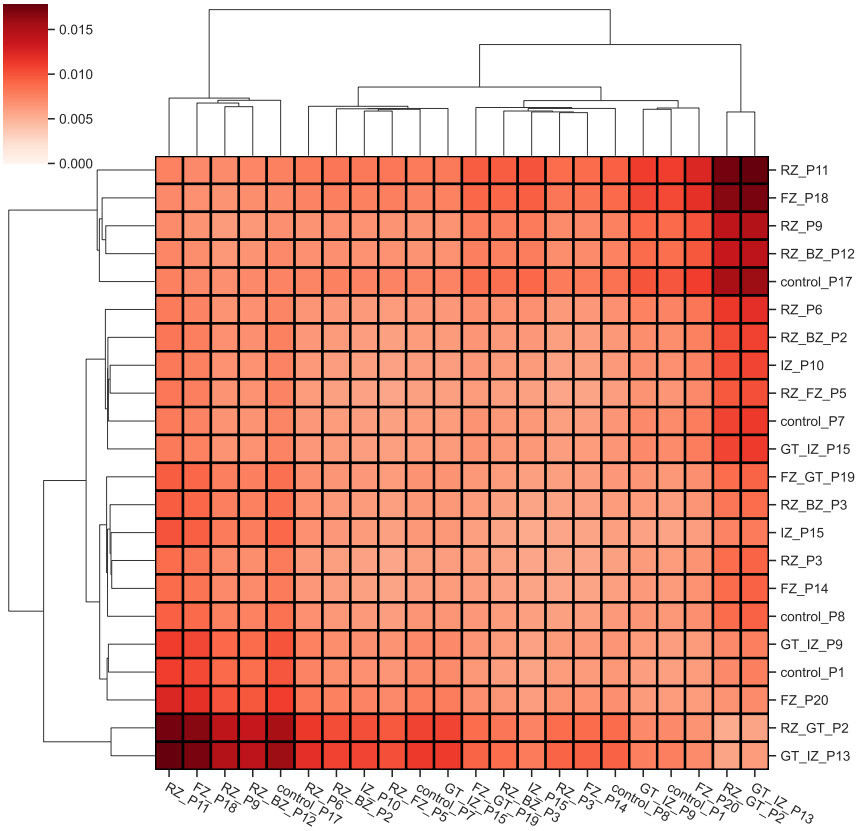


Figure 10. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 56**. Notably, this configuration applies threshold sparsification to the inter-layer edges.

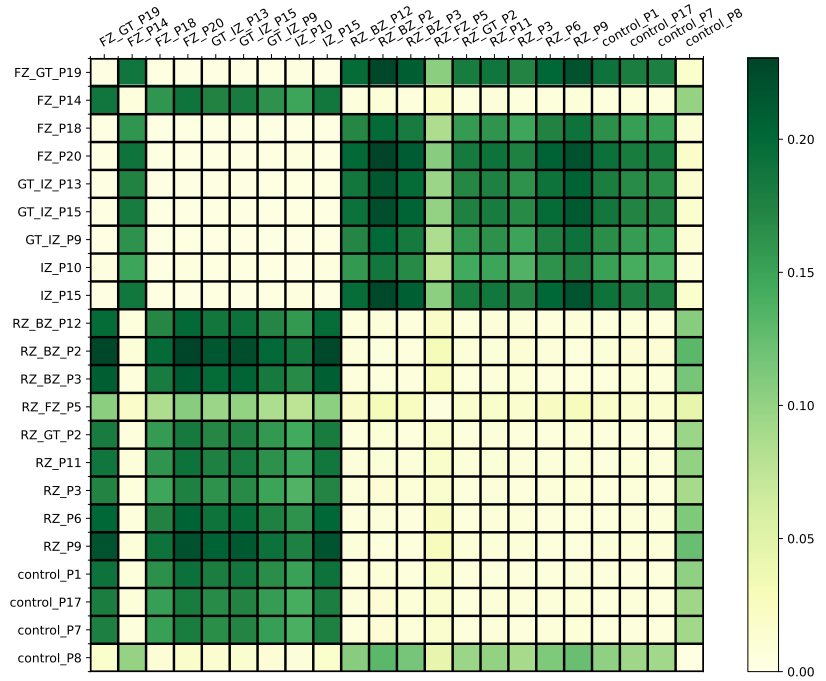


Figure 11. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 53**. Notably, this configuration sets $s = 10^6$ for both directions of switching between layers.

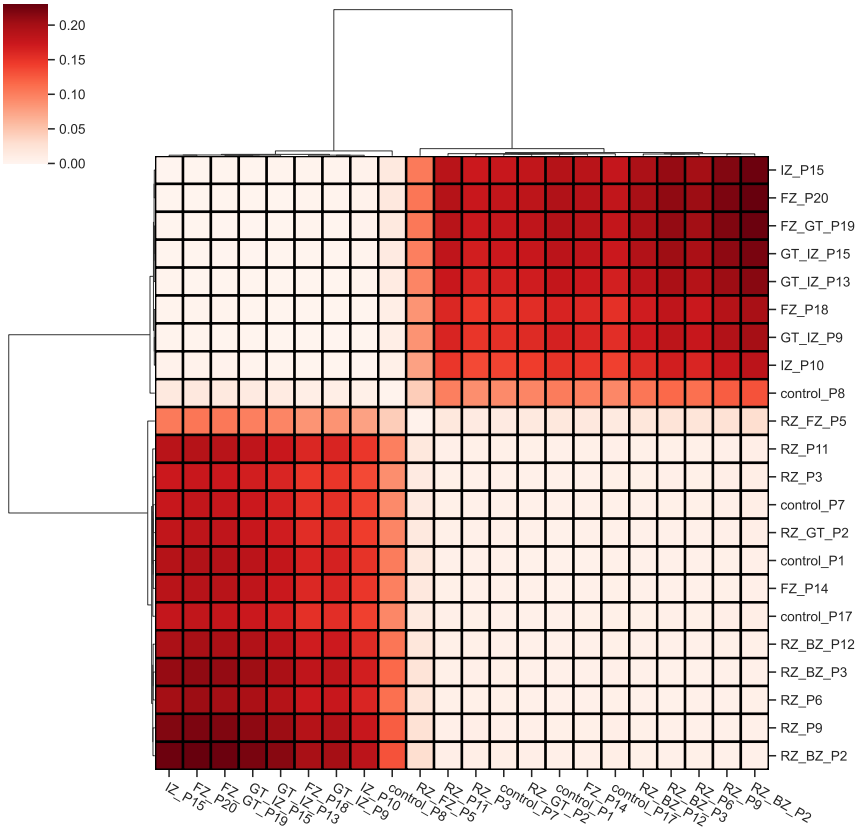


Figure 12. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 53**. Notably, this configuration sets $s = 10^6$ for both directions of switching between layers.

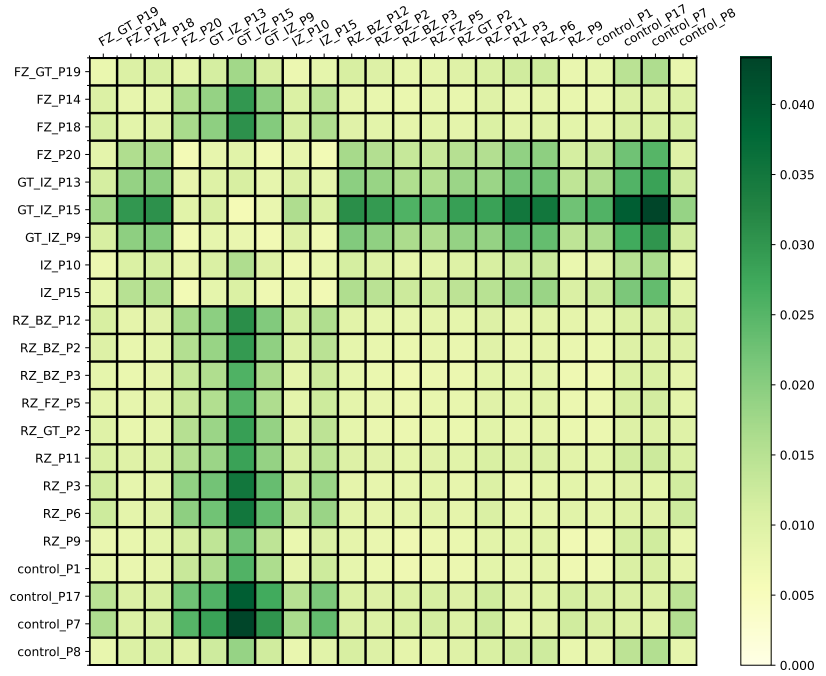


Figure 13. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 63**. Notably, this configuration sets the window size for Skip-gram to 20.

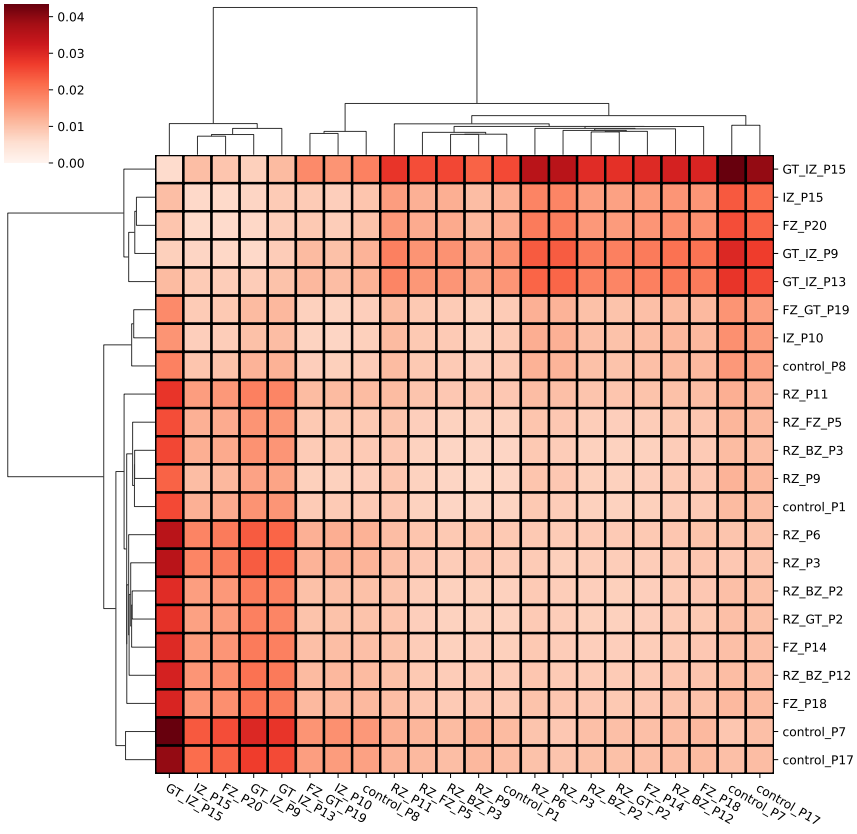


Figure 14. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 63**. Notably, this configuration sets the window size for Skip-gram to 20.

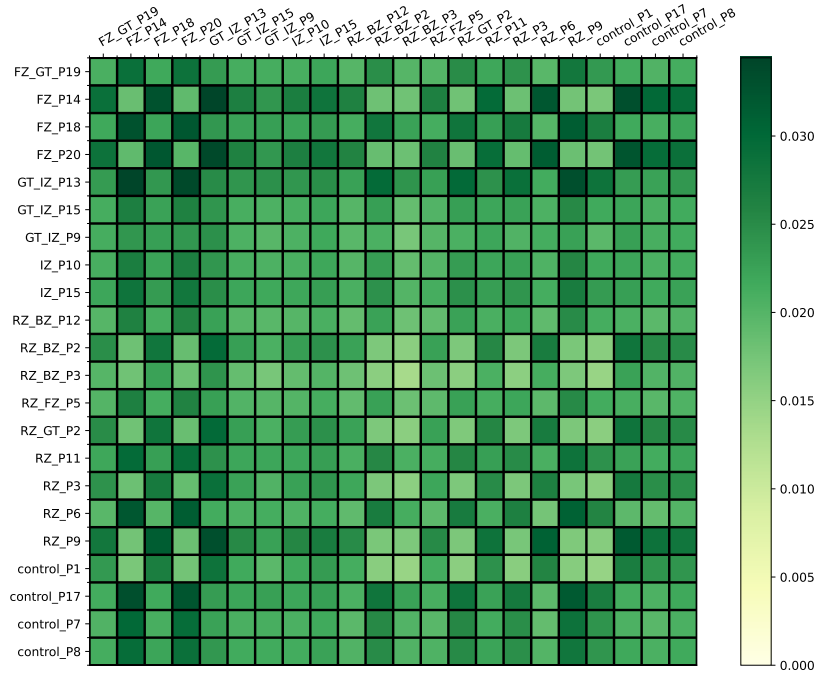


Figure 15. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 70**. Notably, this configuration trains the Skip-gram model for 1 epoch.

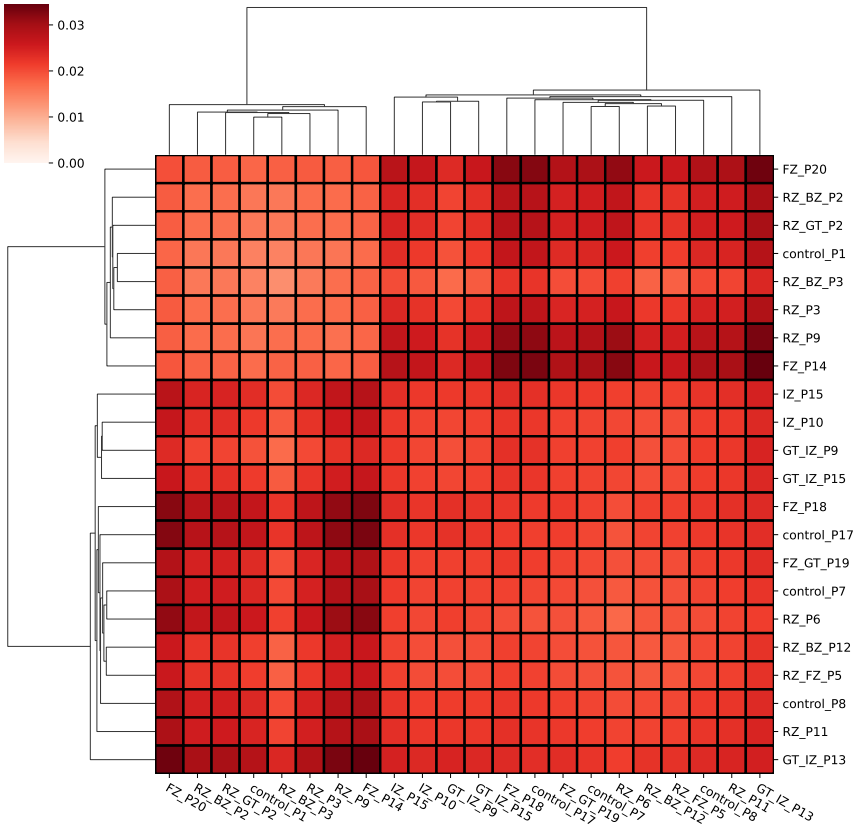


Figure 16. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 70**. Notably, this configuration trains the Skip-gram model for 1 epoch.

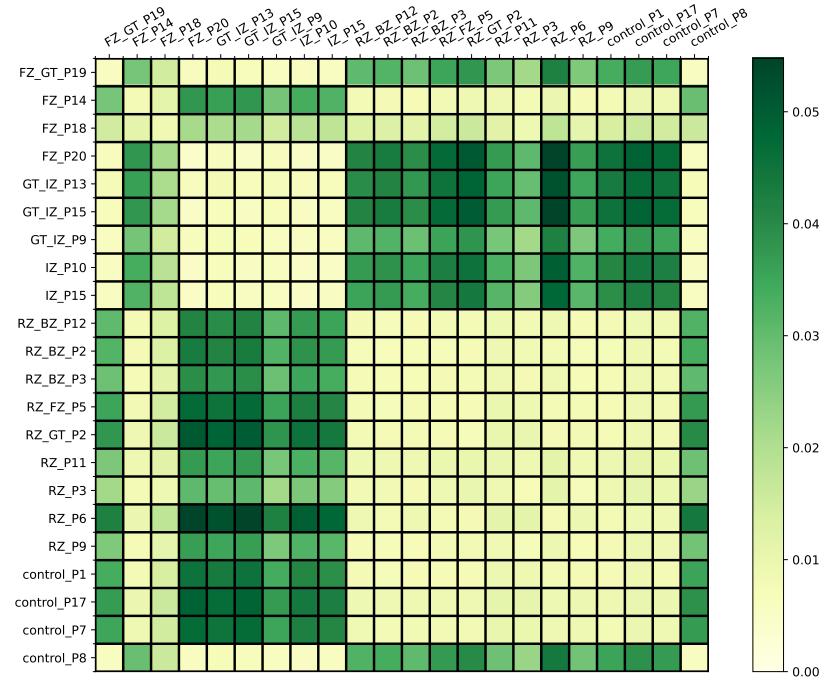


Figure 17. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 69**. Notably, this configuration trains the Skip-gram model for 2000 epochs.

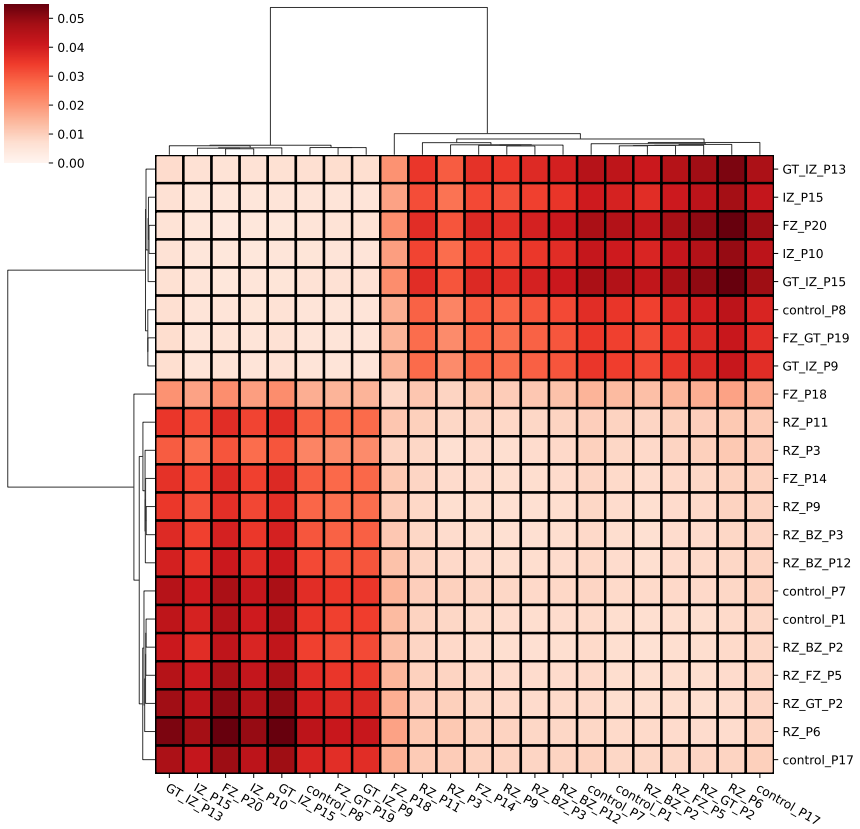


Figure 18. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 69**. Notably, this configuration trains the Skip-gram model for 2000 epochs.

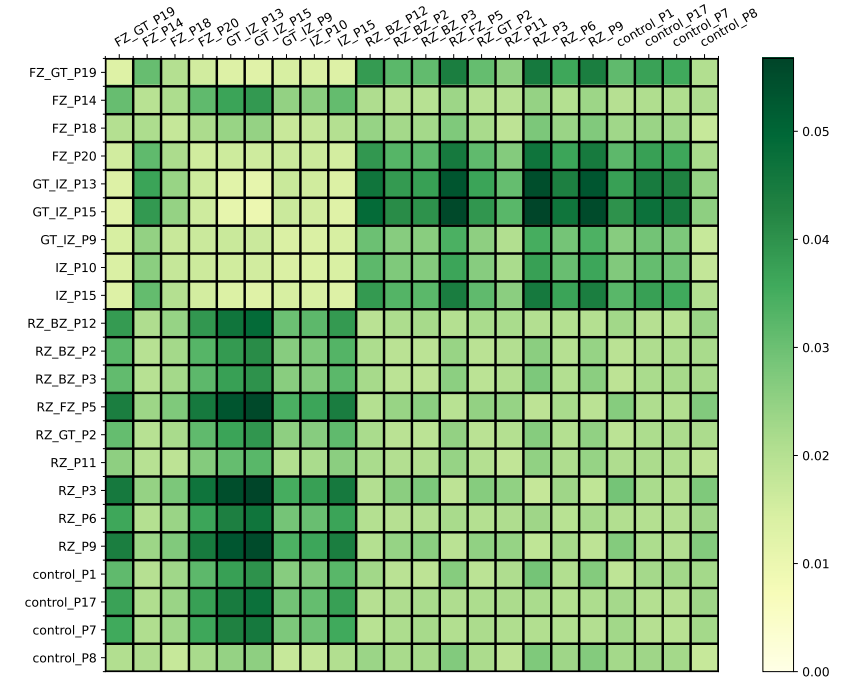


Figure 19. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 58**. Notably, this configuration balances the Gromov-Wasserstein probability distributions with $\alpha = 0.5$.

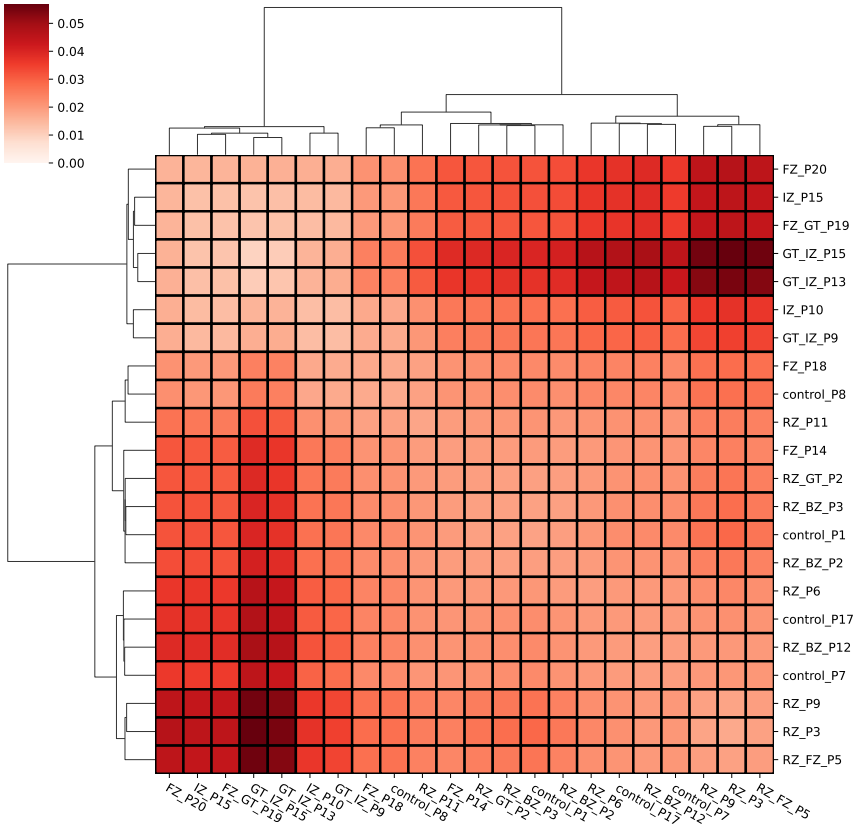


Figure 20. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 58**. Notably, this configuration balances the Gromov-Wasserstein probability distributions with $\alpha = 0.5$.

Excluding Spatial Information

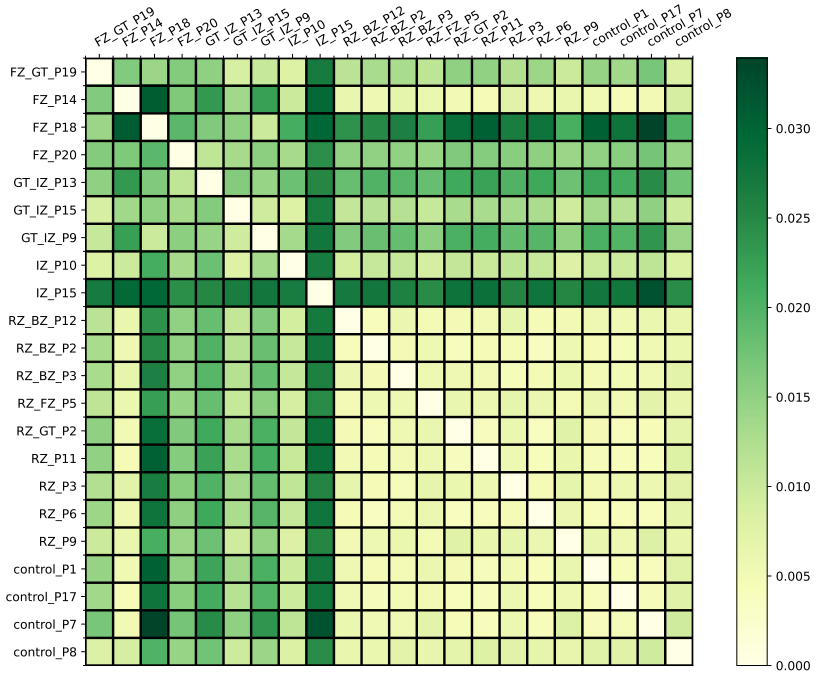


Figure 21. | The entropically regularized Gromov-Wasserstein distance matrix between the node embeddings of all samples, calculated using HeartNet with **Configuration 64** on **only the gene expression layers**.

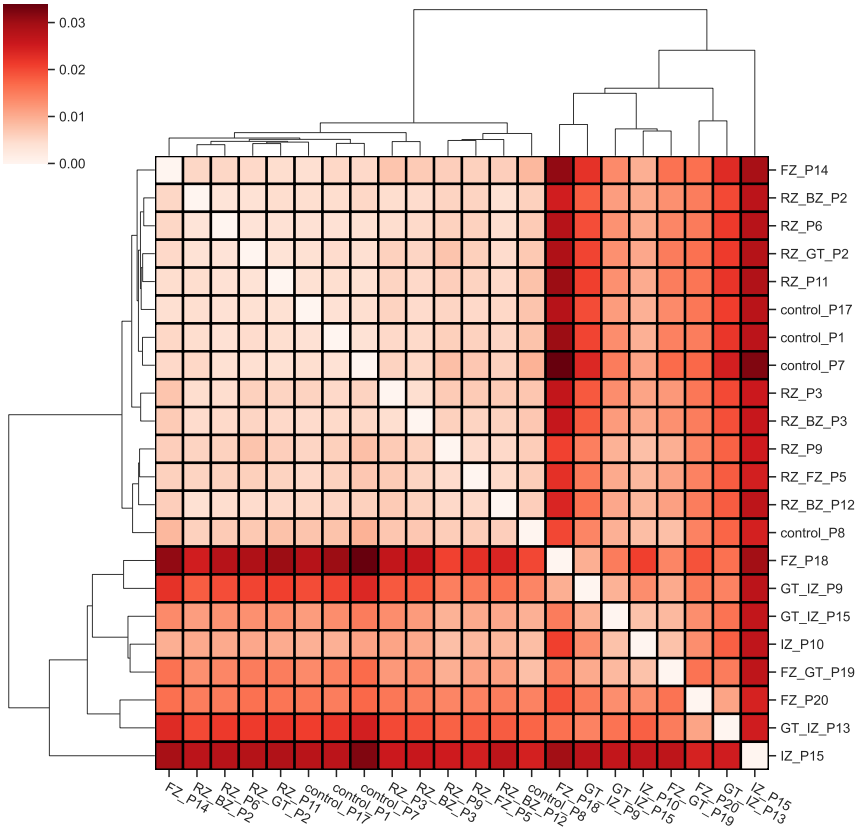


Figure 22. | The clustermap resulting from hierarchically clustering the samples with Ward linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 64** on **only the gene expression layers**.

D. Dendrograms

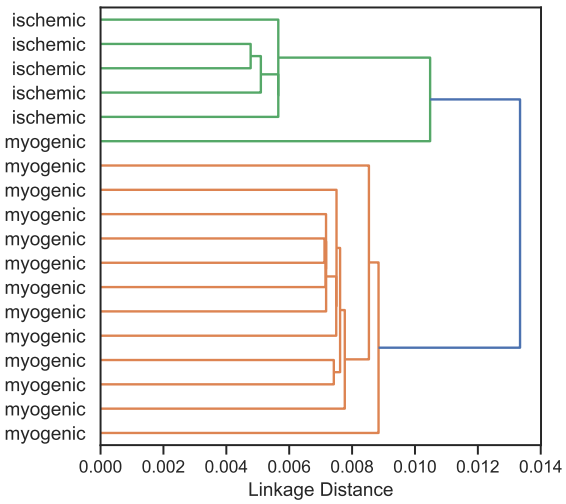


Figure 23. | The dendrogram resulting from hierarchically clustering only the myogenic and ischemic samples with **single** linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 51**.

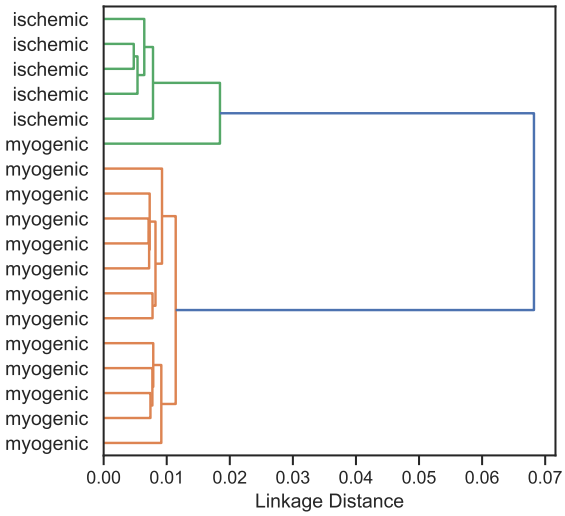


Figure 24. | The dendrogram resulting from hierarchically clustering only the myogenic and ischemic samples with **complete** linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 51**.

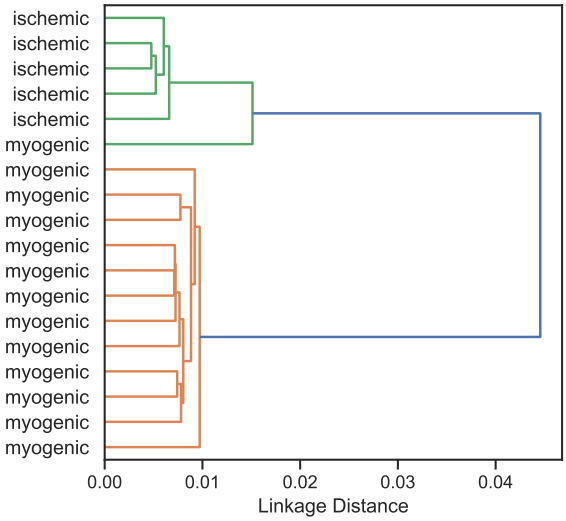


Figure 25. | The dendrogram resulting from hierarchically clustering only the myogenic and ischemic samples with **average** linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 51**.

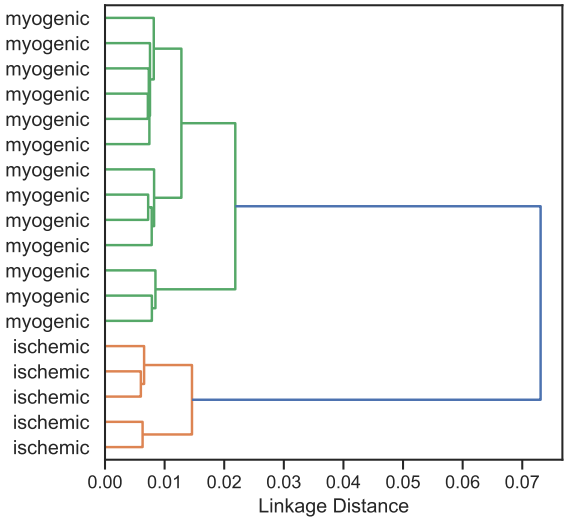


Figure 26. | The dendrogram resulting from hierarchically clustering only the myogenic and ischemic samples with **Ward** linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 52**.

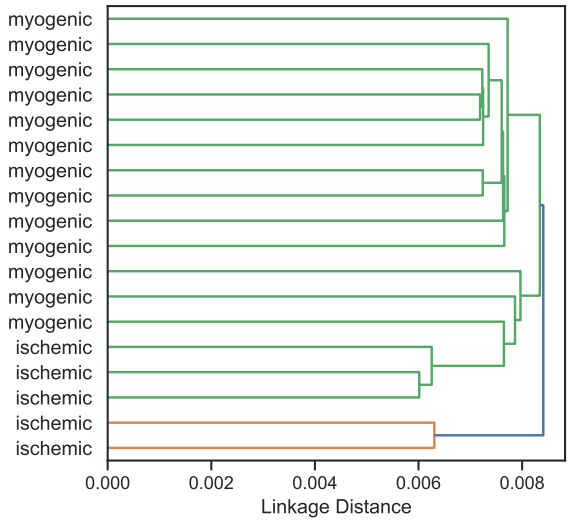


Figure 27. | The dendrogram resulting from hierarchically clustering only the myogenic and ischemic samples with **single** linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 52**.

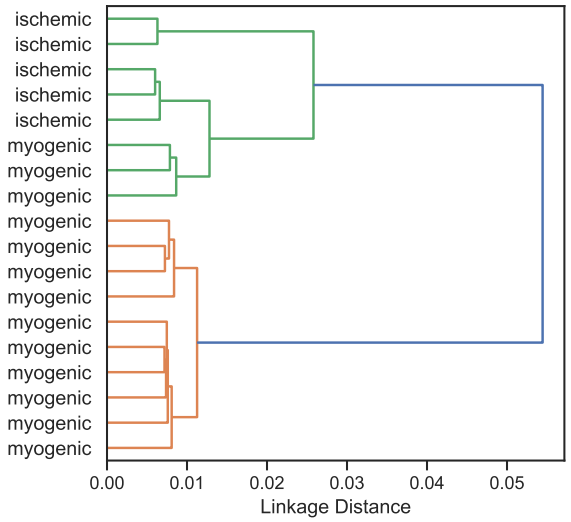


Figure 28. | The dendrogram resulting from hierarchically clustering only the myogenic and ischemic samples with **complete** linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 52**.

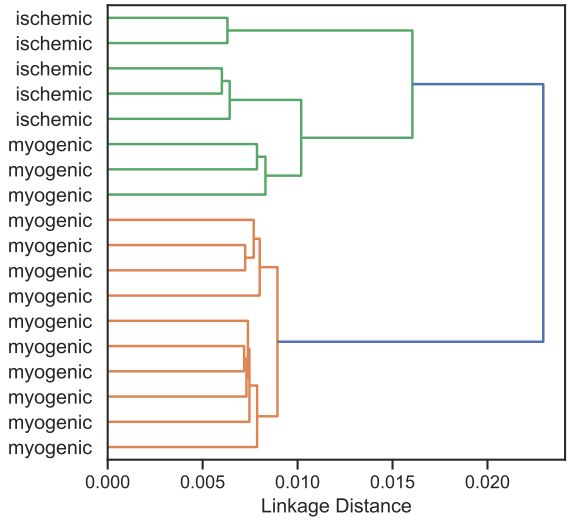


Figure 29. | The dendrogram resulting from hierarchically clustering only the myogenic and ischemic samples with **average** linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 52**.

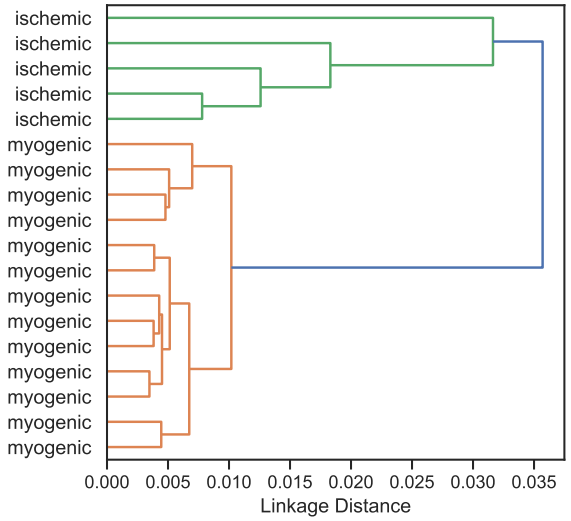


Figure 30. | The dendrogram resulting from hierarchically clustering only the myogenic and ischemic samples with **Ward** linkage, where distances between samples are given by the regularized Gromov-Wasserstein distance calculated using HeartNet with **Configuration 64** on only the gene expression layers.

E. ARI Plots

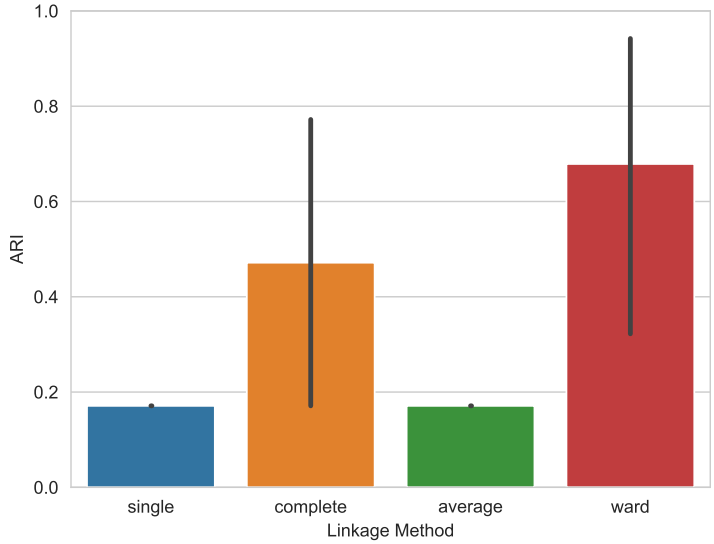


Figure 31. | Linkage method vs ARI for the clusterings resulting from four separate HeartNet runs with **Configuration 64** on only the gene expression layers. Fibrotic samples are excluded, and the myogenic and ischemic samples are clustered into two clusters. The height of each bar indicates the average ARI over the four runs, and the error bars indicate uncertainty.

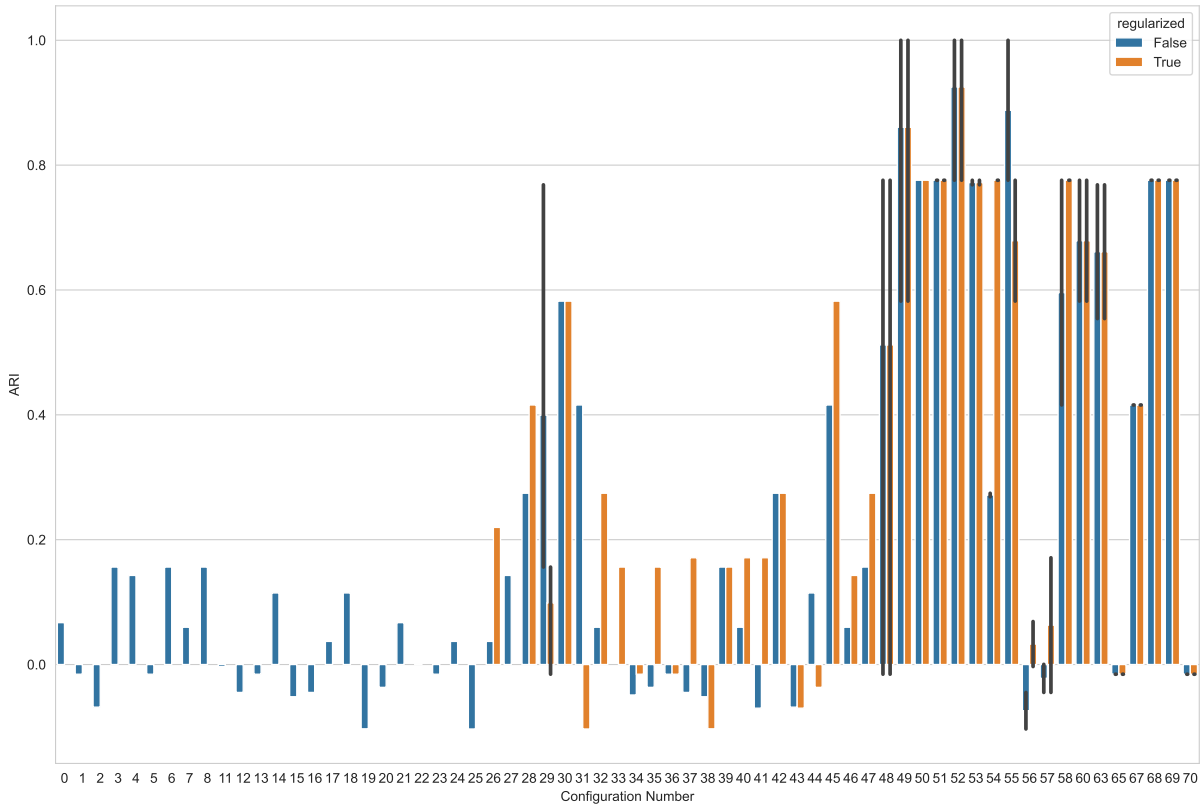


Figure 32. | ARI values resulting from running HeartNet with different configurations and hierarchically clustering the myogenic and ischemic samples into two clusters using Ward linkage. The height of each bar indicates the average ARI over all HeartNet runs with that configuration, and the error bars indicate uncertainty. For configurations 0 through 25, the entropically regularized Gromov-Wasserstein distance was not calculated. Some configurations were excluded because they were used only for experiments where the spatial information was excluded or because the experiments that used these configurations failed due to time constraints or other issues.