TU WIEN Informatics

Graph Neural Network Based Classification of Biological Network Structured Systems

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Data Science

eingereicht von

Laurin Lux, Bsc.
Matrikelnummer 01554046

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ. Prof. Dr. Peter Filzmoser
Mitwirkung: Univ. Prof. Dr. Daniel Rückert
Dr. Johannes Christian Paetzold

Wien, 3. Mai 2023

Laurin Lux

Peter Filzmoser

TU WIEN Informatics

Graph Neural Network Based Classification of Biological Network Structured Systems

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Data Science

by

Laurin Lux, Bsc.
Registration Number 01554046

to the Faculty of Informatics

at the TU Wien

Advisor: Univ. Prof. Dr. Peter Filzmoser
Assistance: Univ. Prof. Dr. Daniel Rückert
Dr. Johannes Christian Paetzold

Vienna, 3rd May, 2023

Laurin Lux	Peter Filzmoser

Erklärung zur Verfassung der Arbeit

Laurin Lux, Bsc.

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 3. Mai 2023

Laurin Lux

Acknowledgements

First, I would like to thank Prof. Daniel Rückert and Prof. Peter Filzmoser for allowing me to work on this project. For his pinpoint research instinct, his motivating way of work, and constant guidance, I want to express my great appreciation to my advisor Johannes Paetzold. Moreover, I am grateful to the AIM group at TUM and the Ertürk Lab at Helmholtz Munich for their friendly and inclusive work environment.

I want to express a comprehensive thank you to Alex Berger. Thank you for making my time in Munich an all-around great experience.

Finally, thank you to my family, especially my brother Simon and my parents. Their patience, support, and understanding know no boundaries.

Kurzfassung

Graph Neural Networks (GNNs) haben in den letzten Jahren große Aufmerksamkeit erlangt. Sie stellen ein flexibles und leistungsfähiges Konzept für die multimodale Analyse von graph-strukturierten Daten dar. Die Vielfältigkeit dieser Daten ermöglicht ein breites Anwendungsspektrum in den unterschiedlichsten Bereichen. Das Potenzial von GNNs für die Analyse komplexer, biologischer, netzwerkstrukturierter Systeme wurde bisher jedoch noch nicht umfassend erforscht. In dieser Arbeit werden GNNs für die Klassifizierung von Graphen eingeführt, die das Lymphgefäßsystem und das enterische Nervensystem des Darms modellieren. Light-Sheet Fluorescence Microscopy (LSFM) in Kombination mit Immunolabeling und Tissue Clearing ermöglicht die Bildgebung solcher Strukturen in ihrer vollen Komplexität im gesamten Körper einer Maus. Allerdings ist LSFM auf drei unterscheidbare Fluorophore im sichtbaren Lichtbereich beschränkt. Diese Limitierung motiviert dazu, das Problem des Multiplexing für die Darmdaten zu lösen. Dafür, wurden GNNs eingesetzt, die die extrahierten Darmgraphen nach der Bildgebung klassifizieren. Unter Verwendung von Mehrkanal-Bilddaten, bei denen Lymphgefäße und Nerven mit verschiedenen Fluorophoren gefärbt sind, war es möglich, einen Graphen mit bekannten Gefäßannotationen zu extrahieren. Durch Training auf diesem Graphen konnte ein Modell erstellt werden, das auf Graphen angewendet werden kann, die aus Einkanal-Bilddaten stammen. Bei diesen ist die Annotation des Nerven- und Lymphsystems ansonsten nur manuell durch einen Biologen möglich. Es wurde eine auf dem SAGE GNN-Modell basierende Methode entwickelt, die eine Balanced Accuracy von 75,9% und eine Accuracy von 77,0% erreicht und damit Algorithmen wie den Random Forest (RF) (Balanced Accuracy von 71,8%) übertrifft. Über die reine Klassifizierungsgenauigkeit hinaus zeigte die Analyse der Konnektivität innerhalb der Klassen ein überlegenes Verhalten der GNN-Modelle. Der Jaccard-Index (JI) der größten Connected Component in der Ground Truth mit den beiden größten Connected Components in der SAGE-Vorhersage ergab Werte von 0,52 bzw. 0,53 für die Lymph- und Nervennetzwerke. Im Gegensatz dazu erreichte der RF-Algorithmus in JI nur 0,43 und 0,18. Dieses Ergebnis zeigt, dass die RF-Vorhersagen trotz akzeptabler Genauigkeit die Konnektivität schlecht erhalten, was für die weitere Analyse entscheidend ist. Schließlich wurden verschiedene Ansätze für die Merkmalsextraktion aus Rohbildern untersucht, um die aufgabenspezifische Klassifikationsleistung zu verbessern. Ein GNN-Modell mit einem LSTM-Merkmalsextraktor auf den Gefäßmittellinien erreichte eine Balanced Accuracy von 76,9% auf dem Testteil des Mehrkanalgraphen, während der RF-Algorithmus einen Wert von 70,6% erreichte.

Abstract

Graph Neural Networks (GNNs) have gained significant attention in recent years. They comprise a flexible and powerful deep learning concept for multimodal graph-structured data analysis. The abundance of graph-structured data facilitates a wide range of applications across vastly different domains. However, the potential of GNNs for the analysis of complex, biological, bodily network structured systems has not been comprehensively explored to this date. In this work, GNNs are introduced to the tasks of node-level classification for graphs that model the lymphatic vessel system and the enteric nervous system of the gut. Light-Sheet Fluorescence Microscopy (LSFM), in combination with immunolabeling and tissue clearing, allows the imaging of such structures in their full complexity in the entire mouse body. However, LSFM is limited to three distinguishable fluorophores in the visible light range. This limitation motivates to solve the multiplexing challenge for the gut data. To address this challenge, GNNs are employed to perform post-imaging classification on extracted gut graphs. Using multi-channel imaging data, where lymphatic vessels and nerves are stained with different fluorophores, it was possible to extract a graph representation with known labels. Training on this ground truth graph a model could be created that generalizes towards a graph extracted from single-channel imaging data, where labeling of the nervous and lymphatic system is otherwise only possible manually by a trained biologist. A method based on the SAGE GNN model was developed, which reaches a balanced accuracy of 75.9%, and an accuracy score of 77.0%, which outperforms baseline algorithms such as Random Forest (balanced accuracy of 71.8%). Beyond pure classification performance, the analysis of within-class connectivity revealed superior behavior of the GNN classifiers. The Jaccard index (JI) of the largest connected component in the ground truth with the combined two largest connected components in the SAGE prediction resulted in values of 0.52 and 0.53 for the lymph and nerve networks, respectively. In contrast, the Random Forest algorithm performs at only 0.43 and 0.18 in JI. This result shows that despite acceptable accuracy, the baseline predictions are bad at preserving connectivity, which is crucial for further analysis. Finally, different modalities for feature extraction from raw images to improve task-specific classification performance were investigated. A GNN model with an attached learnable LSTM feature extractor on the vessel centerlines achieved a balanced accuracy of 76.9% on the test partition of the multi-channel graph while the baseline Random Forest algorithm achieved a value of 70.6%.

Contents

CHAPTER 1

Introduction

Graph Neural Networks (GNNs) are among the most investigated and developed neural networks in recent years [ZCH+20]. They are designed to deal with non-euclidean data, more specifically graph-structured data [BBL+17]. This is the decisive difference to other common Deep Learning (DL) structures that are designed to work with Euclidean data representations like two-dimensional grids (e.g., Computer Vision (CV)) or one-dimensional sequences (e.g., Natural Language Processing (NLP)). GNNs successfully extract node-level features and topological information about a network, making them the models of choice for graph-structured data. Such structures are abundant in a wide range of different domains [ZCH+20]. Sometimes graphs seem to be the inherent representation of the data, while in other cases underlying graph structures might be less evident. Molecules are commonly modeled as graphs with atoms as nodes and bonds as edges. Given molecules as input, GNNs can be used to estimate molecular properties, e.g., the toxicity of a chemical or the activity of a certain drug. These tasks correspond to graph classification/regression tasks, where a graph of any given size is provided as input to generate an output of fixed size. A completely different application field are social sciences, where graphs can represent social networks with people and their relationships as nodes and edges, respectively. Here, a GNN can perform predictions on people that are likely to connect in the social network [AA03]. Such a link prediction task (i.e., a relationship prediction between entities) is another example where GNNs have excelled. [ZC18]. Lastly, a highly relevant task is the prediction of certain properties for every node in a graph. In a street network, predicting the traffic volume at all junctions, represented as nodes, is a critical task. The traffic volume at a junction is influenced by the traffic at neighboring junctions and the general structural properties of the street network.
The listed examples correspond to graph, edge, and node-level tasks. Beyond the versatility of the tasks that GNNs can handle, the examples also display the diversity of the domains, ranging from chemistry/physics over sociology to traffic planning.

Network structures are also abundant in living systems. Mammalian physiology relies

on network-structured systems, e.g., the blood vessel system, that allow a fast supply of oxygen and nutrients through the body. Other fine-grained network structures include the Central Nervous System (CNS) and the Peripheral Nervous System (PNS). Finally, the lymphatic system is network-structured and spans the whole body. The abundance of these networks suggests that GNNs are an architecture that might be useful for biomedical applications. Previous work, e.g., focusing on the analysis of the brain vasculature, has clearly demonstrated the utility of graph representations in this domain [TPS+20, PMS+21].

Light Sheet Fluorescence Microscopy (LSFM) is a technology that allows the visualization of these bodily network structures in their full complexity. LSFM relies on the principle of fluorescence, a phenomenon where the excitation of a molecule (referred to as fluorophore) results in the emission of lower energetic light with a certain time delay. This emission can then be detected and used to localize the fluorophore spatially. LSFM allows the visualization of fluorophores in 3D by slicing a light sheet through a sample that was previously rendered transparent [EBJ+12]. This method is potent in its ability to visualize biological structures in 3D dimensions with high spatial resolution. Todorov et al. [TPS+20] showed how it is possible to visualize the vasculature of a whole mouse brain. The possibilities for the visualization of single biological components are almost unlimited. Every structure that can be specifically targeted can be potentially visualized and investigated. One technique that allows precise targeting of biological structures is immunolabeling. Roughly sketched in immunolabeling, a fluorescent compound is attached to an antibody which then specifically binds to antigens in the sample. Immunolabeling on its own does not enable the imaging of 3D biological specimens due to the opacity of the sample. Opaque samples do not allow the light for excitation nor the emitted light to penetrate through the specimen. For this reason, tissue clearing is a crucial step in LSFM imaging. Tissue clearing is a process where an opaque sample is chemically altered to appear transparent in the visible light range.

While the combination of tissue clearing and LSFM allows visualizing structures of interest, a limitation for the number of different structures that can be distinctly visualized remains. This arises from the problem that only a small number of fluorophores can be perfectly differentiated in the visible light range. There are several approaches towards an instrumentation-based solution to this problem. Notably, Jahr et al. proposed a hyperspectral approach where a diffractive component allows the measurement of emitted light for different wavelengths [JSS+15]. While such approaches are rich in information, they still come with the prize of non-trivial changes to standard commercial instruments.

A more elegant solution that circumvents any changes to the instruments is a post-processing-based signal differentiation. Ideally, this eliminates the need to use different fluorophores. The classification should be possible by only considering the image information of a single channel. The traditional approach to solve such a problem is segmentation. Here, each voxel is assigned either to the background or to one of the foreground classes. Segmentation using DL in this area is well-researched and often provides compelling results [RFB15, HTN+22, SPS+21]. However, there is a multitude of

downstream tasks that have no explicit demand for segmentation and where an expressive but compact graph representation is all it needs. These representations are lightweight regarding storage demand and at the same time, regarding the models that operate on them. This allows for fast and also energy-efficient training and inference of models for downstream tasks, such as disease classification. At the same time, the reduction of learnable parameters reduces the demand for training data, which is a compelling argument for the biological and medical domain, where training data is often sparse. Graph generation is a rapidly developing field [SKW+22]. With the rising abundance of available graph data, there will be increased interest in tasks that can be directly solved on graph-structured data.

For the given reasons, this work tries to evaluate the potential of directly using graph representations for multiplexing in LSFM. Specifically, the lymphatic systems in the gut and the interwoven PNS are at the center of this work. Given the network-structured nature of both systems, it makes sense to represent them as graphs and solve the downstream multiplexing problem using the most promising group of graph learning architectures, namely GNNs.

CHAPTER 2

Background

2.1 Light Sheet Fluorescence Microscopy

LSFM is an imaging technique that emerged in the early 2000s due to light sheet generation and detection advances. LSFM is a non-destructive, high-resolution imaging technique that allows for 3D imaging of fluorescently labeled biological specimens, such as cells, tissues, and whole organisms. LSFM works by using a thin sheet of light to illuminate a sample from one angle while a camera captures images from a perpendicular angle. This allows for high-speed, high-resolution imaging of large specimens with minimal phototoxicity and photobleaching [LLYA14].

LSFM is used in various applications, from developmental biology and neuroscience to cancer research and drug discovery. It is particularly useful for imaging large, complex samples, such as whole organs, where traditional imaging techniques, like confocal microscopy, can be limited by the thickness of the specimen.

2.1.1 Tissue Clearing

Tissue clearing is a technique used in microscopy and imaging to make biological tissues transparent to light, allowing for high-resolution 3D imaging of intact tissues [RL15, UEC+20, RGM+21]. Most biological tissues are opaque to visible light due to the presence of lipids and other light-scattering molecules. This limits the achievable depth of imaging for traditional microscopy techniques. Tissue clearing involves treating the tissue with various chemicals that dissolve the light-scattering molecules while preserving the structure and fluorescence of the tissue. Once the tissue has been cleared, it can be imaged using various imaging techniques, such as light-sheet fluorescence microscopy, enabling the study of complex biological processes in healthy and diseased specimens in unprecedented detail. Tissue clearing is still rapidly evolving, with new techniques and applications constantly being developed.

There are several techniques for achieving tissue clearing, including CLARITY [CD13], CUBIC [STP+14], and DISCO [EBJ+12, PCQ+16, RWS+14, TPS+20, PSAM+19], among others. These techniques involve treating the tissue with chemicals, such as acrylamide or urea, that dissolve lipids and other light-scattering molecules while preserving the structure and fluorescence of the tissue.

2.2 Graph Neural Networks

GNNs emerged from the desire to transfer the advances made in DL for Euclidean spaces towards the Non-Euclidean domains such as graphs and manifolds. Convolutional Neural Networks (CNNs) proved to be essential for the successful application of DL [LBBH98]. Given the performance of CNNs, it was desired to generalize the convolution operation towards Non-Euclidean spaces. For graphs, the approaches to perform convolutions can be roughly separated into two strategies [BBL+17]. First, a spectral approach that relies on spectral graph theory [Chu97]. Secondly, a spatial approach that tries to define the convolution spatially as an operation on the node neighborhood. Bronstein et al. [BBL+17] point out that, however, these two methods are fundamentally different in the way the convolution is derived, it is still possible that a definition in the spectral domain boils down to the application of a filter in the spatial domain.

2.2.1 Graph Convolutional Networks

The introduction of the Graph Convolutional Network (GCN) by Kipf and Welling [KW16] was the start of a new age for the concept of the GNN. Their approach is based on spectral graph theory and uses convolutions over the spectral representations of graphs [Chu97].

The spectral definition for convolutions on graphs relies on the computation of the graph Laplacian L and its eigenvectors. The graph Laplacian is defined as $L := D - A$. Where A is the adjacency matrix, and D is the degree matrix of a graph. For undirected graphs, the degree matrix D is a positive diagonal matrix and A is a nonnegative symmetric matrix. L is a positive-semidefinite matrix with potentially positive and negative values. A normalized version of L is defined as $L_{sym} := D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$.

The convolution of a graph in the spectral domain defines as

$$g_\theta \star s = U g_\theta U^T s \quad , \tag{2.1}$$

where g_θ is a spectral filter, $*$ is the convolution operator, $s \in \mathbb{R}^N$ is the spectral signal over the N nodes in the graph and U is the eigenvector matrix from the eigenvalue decomposition of the Laplacian $L = U \Lambda U^T$ [HVG11]. For large graphs, it is computationally expensive to calculate the eigendecomposition of the $n \times n$ Laplacian matrix because of the cubic time complexity $O(n^3)$ of the eigendecomposition.

Kipf and Welling proposed an estimation of the graph convolution with a first-order approximation using Chebyshev polynomials. Limiting the expansion of the Chebyshev polynomials to $K = 1$ results in a 1-localized convolution. This means every node is only influenced by its 1-hop neighborhood. Adding the identity matrix I is equivalent to adding self-loops to the graph. This makes every node part of its own 1-hop neighborhood. Consequently, Equation (2.1) can be approximated as

$$g_\theta \star s \approx \theta_0 s + \theta_1 (L - I) s = \theta_0 s - \theta_1 D^{-\frac{1}{2}} A D^{-\frac{1}{2}} s \tag{2.2}$$

and when the operation is further limited to a single learnable parameter θ, it can be expressed as

$$g_\theta \star s \approx \theta (I + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) s \quad , \tag{2.3}$$

where $\theta = \theta_0 = -\theta_1$.

Furthermore, Kipf and Welling replace $I + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ by $\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$. Equation 2.3 then defines as

$$g_\theta \star s \approx \theta (\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}) s \quad , \tag{2.4}$$

which resolves the exploding/vanishing gradient problem that arises when multiple convolutions are stacked. Here $\tilde{A}$ is the adjacency matrix with added self-loops $\tilde{A} = A + I$ and $\tilde{D}$ is the to $\tilde{A}$ associated degree matrix.

Finally, a single GCN layer performs the following operation

$$H^{(l+1)} = \sigma \left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} \Theta^{(l)} \right) \quad , \tag{2.5}$$

where $H^{(l+1)}$ represents the node embeddings in the $l+1$-th layer and $\Theta^{(l)}$ is the learnable weight matrix. The embedding before the first GCN layer $H^{(0)}$ is the node feature matrix $X \in \mathbb{R}^{n \times c}$, where n is the number of nodes and c is the number of features in the embedding of the nodes.

Similarity to Laplacian Filtering

Li et al. [LHW18] further investigated the GCN and made the connection to Laplacian surface smoothing that was introduced by Taubin already in 1995 [Tau95]. The comparison reveals some interesting conclusions about the GCN architecture.

Stacking GCN layers makes closely connected nodes more similar, which is the expected behavior of a smoothing operation. This can be potentially useful for further classification as clustered nodes become more similar but are not strongly influenced by nodes

from other clusters. Stacking an infinite number of GCN layers would result in equal embeddings for all nodes within a connected component. This (partly) undesirable effect is commonly referred to as the over-smoothing problem. It holds when there are no bipartite components in the graph, which is always the case because self-loops are added for a GCN layer. For detailed proof of this property, refer to Theorem 1 in the work of Li et al. [LHW18].

This effect makes it desirable to have high connectivity of nodes with the same class and low connectivity of nodes from different classes. Furthermore, it indicates that stacking many layers is not necessarily a good idea to generate good node embeddings for any given task. This is a crucial step for understanding the GCN layers in contrast to conventional CNN networks where stacking many convolutional layers is often favorable for the task due to the hierarchical feature extraction.

2.2.2 GraphSAGE

Based on the idea of the GCN Hamilton, Ying et al. [HYL17] introduced the GraphSAGE architecture in 2018. Compared to the GCN, the function of the layer is not defined using spectral graph theory. Instead, SAGE stands for SAmple and aggreGateE, representing the two major steps of a SAGE layer. While the GCN utilizes the whole neighborhood, SAGE uniformly samples a fixed number of nodes from the neighborhood. This poses a major difference in the message-passing scheme. The second step is aggregation; here, a SAGE layer can use different aggregator functions invariant to permutations of the neighbor nodes. The initial SAGE paper proposes a mean aggregator, a Long Short-Term Memory (LSTM) aggregator, and a pooling aggregator. The LSTM is used with random permutations of the neighborhood, which aims to adapt it to unordered sets. Additionally, a pooling operator (max-pooling) is proposed, where a learnable function processes each neighborhood node embedding before a max aggregation.

In contrast to a GCN-layer, the node embedding itself is concatenated to the aggregated information from the neighborhood. This allows a different transformation of the previous node embedding and the aggregated neighborhood information. In a GCN-layer, the introduction of self-loops makes the node part of its own neighborhood, and no explicit concatenation of the node embedding to the aggregate is performed.

Mixed Aggregators

With the invariance to permutation being the only requirement for the aggregation, the opportunity to use a wide range of different aggregation modes opens up. Modes such as standard deviation, maximum, minimum, mean, or other robust estimates of location and variance are all possible. Figure 2.1 illustrates how different aggregation schemes can fail to distinguish the signals from a specific neighborhood.

Corso, Cavalleri et al. [CCB+20] could show that combining different aggregation schemes improves the expressive power of GNNs. Beyond that, they validate their theory by a

boost in performance on real-world applications. The SAGE model explicitly allows the usage of such combined aggregators, which makes the model more attractive for complex tasks that can only be solved with a combination of different aggregators.

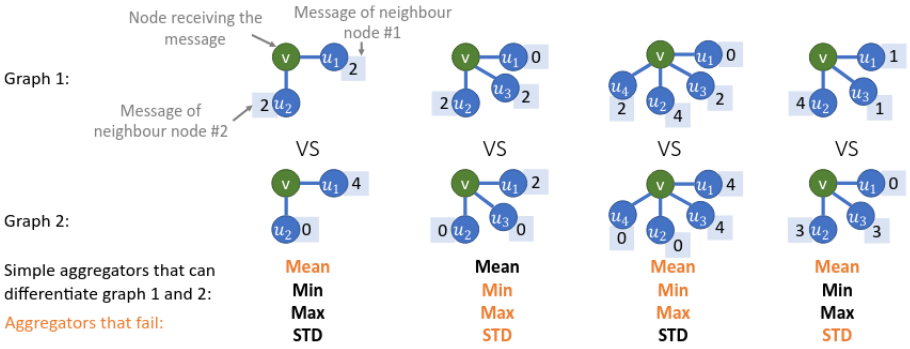

Figure 2.1: Graph structures that result in the failure of certain aggregation modes. Comparison of aggregation with mean, maximum, minimum, and standard deviation. Figure published by Corso, Cavalleri et al. [CCB+20].

2.2.3 DGCNN

The Dynamic Graph Convolutional Neural Network (DGCNN) proposed by Wang et al. [WSL+19] was inspired by PointNet [QSMG17] and is designed for the processing of 3D point clouds, but the proposed concepts have a significant impact in the field of GNNs. First, DGCNN introduces the concept of an edge convolution where a learnable edge function processes the concatenated information of both adjacent nodes; the edge function is free to define and can be, for example, a simple Multilayer Perceptron (MLP) model. In the second step, edge feature vectors are aggregated, as is typical for GNN models. Notably, the edge convolution focuses on the relationship of node embeddings regarding their neighbors, in contrast to other GNN models where the graph neighborhood only defines the message passing. The other interesting concept is the introduction of a dynamic graph structure where adjacency changes from layer to layer and is defined by proximity in the feature space. This results in a layerwise k-nearest neighbor graph that is used for the edge convolutions.

2.3 Graph Generation

Generating graph representation based on an underlying image or volume is a long-studied task. Multiple domains require graph representation for efficient processing, while the initial representation domain is an image or volume. Representative examples are road graphs for navigation tasks [HBJ+20], vascular graphs for brain analysis [PMS+21], or scene graphs for image retrieval [JKS+15]. While multi-step approaches have dominated

the approaches for graph generation in the past, novel approaches are fusing the process of object and relation prediction [SKW+22, PSP+]. Moreover, the Relationformer by Shit et al. [SKW+22] works directly on the raw input volumes, obliterating the need for in-between segmentations. In a submitted paper to ICCV, which I contributed to within my master's thesis research, we could show that single-step graph generation transformer models can be efficiently pretrained on data-rich domains such as street networks and then be fine-tuned to biological and medical domains where data is sparse. Among other approaches, this work will make graph representations more available in the biomedical domain, which motivates the further investigation of graph learning applications.

CHAPTER 3

LSFM Gut Data

3.1 Multi-Channel Gut Data

3.1.1 Data Properties

The multi-channel gut data portrays the gut of a mouse in a 3D volume and includes valuable structural information. Immunolabeling was utilized to create the data, focusing on the nervous and lymphatic system structures [MLH+23]. During the immunolabeling process, primary antibodies were employed to specifically target these structures. In the second step, two secondary antibodies, each with different fluorophores, were introduced, selectively binding to one of the two primary antibodies. Finally, incubation was performed actively by pumping the solutions containing the antibodies through the vasculature of the mouse. The antibody-fluorophore complexes that were used for the visualizations are described below:

- Lymphatic vessels that are present in the villi and the submucosa of the gastrointestinal wall [CE19] are labeled, taking advantage of the Lymphatic Vessel Endothelial Hyaluronan Receptor 1 (LYVE1) binding Anti-LYVE1 antibody. The attached fluorophore is a 647 nm (far-red) fluorophore with an emission maximum roughly at the mentioned wavelength.
- The Enteric Nervous System (ENS) that is present in the gut is visualized using a Tyrosine Hydroxylase (TH) binding primary antibody (Anti-Tyrosine Hydroxylase Antibody). TH is an enzyme that is specifically present in cells of the CNS and the PNS. The attached fluorophore is a 568 nm (orange) fluorophore with an emission maximum roughly at the mentioned wavelength.

The two complexes, consisting of the primary antibody, secondary antibody, and fluorophore, emit light at distinct wavelengths within the visible light spectrum range of

400-800 nm. These unique emissions are then measured and stored within two separate image channels. To allow the visible light used for excitation and the emitted light to penetrate tissue, the sample is rendered transparent with tissue clearing [EBJ+12].

Data acquired from LSFM is anisotropic because of differences in the spatial resolution arising from the thickness of the light sheet (z-dim) and the pixel size of the camera (x-dim, y-dim). This needs to be considered for the following downstream tasks. Table 3.1 describes the spatial characteristics of the multi-channel gut data. Regarding precision, the channel-wise data is saved as unsigned *int16* (min = 0, max = 65535).

Table 3.1: Multi-channel gut dataset characteristics.

Multi-Channel Gut Data	*x-dim*	*y-dim*	*z-dim*
Voxel Range	5853	3457	194
Spatial Resolution / µm	1.625	1.625	6.000
Size / mm	9.5	5.6	1.2

3.1.2 Volumetric Segmentation

On the previously described data, a patch of 920 × 632 × 136 voxels in x-, y-, and z-dimensions was hand-annotated by a biologist. Thereby, ground truth for segmenting the lymphatic and nervous system structures is created. The hand-annotated set covers slightly more than 2% of the complete data. Figure 3.1 a, b) show the raw data for the two imaging channels, and Figure 3.1 d, e) displays the corresponding hand-annotated segmentation.

To create a segmentation for the whole data, the hand annotations were used as ground truth for two Swin UNETR [HTN+22, HNT+22] segmentation models for the lymphatic and the nervous system. Both models were trained on the data from their associated image channels. Because of the limited ground truth data, the pre-trained UNETR models were only finetuned on the hand-annotated data. Pre-training was performed on numerous datasets containing a total of ∼ 5050 3D Computer Tomography (CT) images [1]. Although there is a significant shift from the pre-training domain (CT) to the target domain (LSFM), the pre-trained model worked excellently for the LSFM data.

3.1.3 Volume Merging

A volume synthesis step is necessary to mimic a scenario where only a single imaging channel contains the information for lymph and nerve structures. This corresponds to both the segmentation, as well as to the raw imaging data.

Segmentation: Regarding the segmentations, a simple max combination of the images results in an ideal resemblance of the single-channel scenario. After this step, the segmen-

[1] For more details refer to:
https://github.com/Project-MONAI/research-contributions/tree/main/SwinUNETR/Pretrain

tation is processed by a fill-hole algorithm followed by binary closing. This procedure can be regarded as denoising of the segmentation mask, and it is necessary due to the susceptibility of the graph extraction algorithm to holes and isolated pixels.

Raw Image: For the raw data, the combination steps pose a more difficult question. Differences in the background noise, e.g., from autofluorescence, or in the signal intensities, e.g., differences in the quantum yield of different fluorophores, make a simple max combination a poor approximation. A scenario where both lymph and nerve structures have highly comparable signal distributions would constitute the worst case for classification based on the raw image data. If such a scenario is mimicked by the merging of the channels it guarantees that the information from the raw imaging data is not artificially increasing the classification performance.

In later experiments (see Section 5.2), the effect of different merging scenarios on the classification performance is tested. The step-by-step merging procedure works as follows:

1. First, the statistical moments of the signal distribution are estimated with the robust metrics, median and Median Absolute Deviation (MAD). The signal distribution is constituted by the values of all the voxels that were masked in the segmentation step.
2. These metrics are then used to perform channel-wise centering and scaling. This centering/scaling is performed for all the relevant voxels. Meaning that the centering/scaling is performed on each raw data channel for all voxels from the merged segmentation mask.
3. Consequently, the processed data of each channel is combined. Depending on the experiment, the voxels were combined with a $max()$ or $mean()$ operation. Note that the combination is only relevant for all the voxels in the merged segmentation. Other voxels are not considered for any feature extraction.
4. Finally, the merged data needs to be centered and scaled again, which is again done using the robust median and MAD. Without the second centering, the $max()$ combination results in a right-shifted distribution, while the opposite is true for the $mean()$ combination

The proposed image merging pipeline results in two different raw data sets that are evaluated in this work:

1. Equal center and scale of both signal distributions and *max()* combination of combined mask voxel values.
2. Equal center and scale of both signal distributions and *mean()* combination of combined mask voxel values.

A schematic example of the merged image channels is displayed in Figure 3.1 c) and the corresponding merged segmentations in 3.1 f). The segmentations in the image are combined by a simple logical or, while the raw data is combined by channel-wise centering and scaling followed by a max combination.

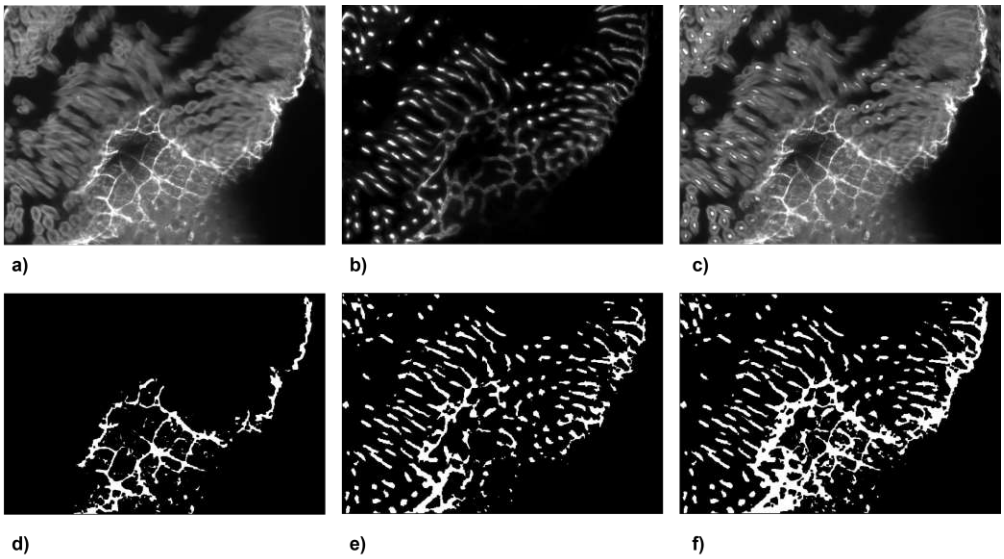

Figure 3.1: a) Raw imaging data visualizing the nervous network. b) Raw imaging data visualizing the lymphatic network. c) Combination of the channels a) and b) into a single channel. d) Segmentation of the nerves. e) Segmentation of the lymphatic vessels. f) Combined segmentations.

3.1.4 Graph Extraction and Labeling

The graph $\mathcal{G}_{gut}$ is extracted from the merged segmentation mask described above. To do so, the vessel-to-graph algorithm described by Drees et al. [DSH+21], implemented in the Voreen software [MSRMH09], is used. Before the segmentation is passed on to the graph extraction algorithm, it is upsampled to isotropic pixel size; this was done using third-order spline interpolation. After the interpolation, the size along the x- and the y-axis is unchanged while the size along the z-axis is increased by the factor $\frac{6.000}{1.625}$. The bulge size hyperparameter, which defines when a bulge is considered a separate node, was set to 2 for this and all other graph extractions. Additionally, to the creation of nodes, that represent bifurcation points, and edges that represent vessel-like structures, the algorithm extracts geometric edge features. These features contain rich information about the structural properties of the vessels, such as curvature, length, roundness, standard deviations, and mean values of the vessel radii. An overview of all the features that are extracted by the algorithm is provided in Appendix A (see Appendix A Table 1).

Importantly the algorithm also extracts a centerline for each edge in the graph. These

centerlines are represented by a list of variable lengths containing pixel positions of the input segmentation. The positional information of the centerline allows the labeling of the edges based on the segmentation label (lymph vs. nerve) predominant along the centerline. Thereby, a labeled ground truth data set that closely resembles the data that can be obtained from a single imaging channel is generated.

Table 3.2 summarizes the most important properties of the multi-channel gut graph. Self-loops that occurred in a negligible amount were eliminated to create a simple graph. Furthermore, isolated nodes were eliminated since they are neither corresponding to vessels nor bifurcations.

Table 3.2: Multi-channel gut graph $\mathcal{G}_{gut}$ characteristics.

Characteristic	$\mathcal{G}_{gut}$
Bifurcations (Nodes)	271100
Total Edges	340202
Lymph Edges (Lymphatic Vessels)	215674 ($\sim$ 63%)
Nerve Edges (Nerves)	124528 ($\sim$ 37%)
Avg. Degree	2.51
Connected Components	23

The biological expectation for $\mathcal{G}_{gut}$ is that it forms a single connected component. Biologically no parts of the lymphatic system or the nervous system are isolated from the rest. In fact, the extracted graph fulfills this expectation with 98.1% of the nodes (265986) being part of a single connected component. In reverse, the other connected components combined sum up to only 1.9% of the nodes (5114). These smaller connected components likely arise from a mixture of missing links in the segmentation step and imaging artifacts from the LSFM measurement.

3.1.5 Line Graph

In the biological graph of the nervous and lymphatic system, the major information lies in the vessels and not in the nodes, which only represent bifurcation points. A line graph representation $\mathcal{L}_{gut}$ where the vessels are represented as nodes and edges represent bifurcation points that connect vessels is a better choice. Thereby, the feature-rich lymphatic vessels/nerves are the instances that are classified in a node classification setting. A description of the gut line graph is displayed in Table 3.3. For further evaluation, only the largest connected component is used as the smaller connected components mostly only cover a few umpteen nodes and likely arise from artifacts along the pipeline of sample preparation, imaging, segmentation, and graph extraction.

Table 3.3: Multi-channel gut line graph $\mathcal{L}_{gut}$ characteristics.

Characteristic	$\mathcal{L}_{gut}$
Total Nodes	340202
Lymph Nodes (Lymphatic Vessels)	215674 ($\sim$ 63%)
Nerve Nodes (Nerves)	124528 ($\sim$ 37%)
Edges	641842
Avg. Degree	3.77
Connected Components	23

For a line graph $\mathcal{L}(\mathcal{G})$, the properties that are listed below hold [HNW65]. These properties give a rough insight into the characteristics of the line graph representation and explain the relationship of the properties displayed in Table 3.2 and Table 3.3.

- The cardinality of the edge set $|\mathcal{E}_\mathcal{G}|$ is equal to the cardinalty of the node set $|\mathcal{V}_\mathcal{L}|$, $|\mathcal{E}_\mathcal{G}| = |\mathcal{V}_\mathcal{L}|$. Where $\mathcal{E}_\mathcal{G}$ is the edge set of $\mathcal{G}$ and $\mathcal{V}_\mathcal{L}$ is the node set of $\mathcal{L}$.

- A graph that is connected in $\mathcal{G}$ is also connected in $\mathcal{L}$, with the exception of isolated nodes. Thereby, if $\mathcal{G}$ does not contain isolated nodes, the number of connected components in $\mathcal{G}$ is equivalent to the number of connected components in $\mathcal{L}$.

- The cardinality of the edge set $|\mathcal{E}_\mathcal{L}|$ can be calculated as half the sum of the squares of the degrees of all vertices in $\mathcal{G}$, minus $|\mathcal{E}_\mathcal{G}|$

$$|\mathcal{E}_\mathcal{L}| = \left(\frac{1}{2}\sum_{v\in\mathcal{V}_\mathcal{G}} deg(v)^2)\right) - |\mathcal{E}_\mathcal{G}| \quad .$$

3.1.6 Graph Generation Pipeline

A schematic pipeline for the creation of a line graph $\mathcal{L}(\mathcal{G})$ starting from the raw image information is displayed in Figure 3.2. Summarized the crucial steps comprise segmentation, graph extraction, and line graph creation.

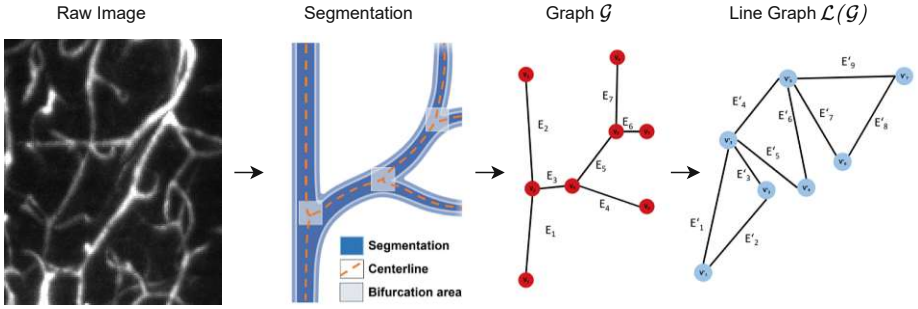

Figure 3.2: Pipeline for the creation of a line graph $\mathcal{L}(\mathcal{G})$ based on raw imaging data. Note that the images are just schematic and do not necessarily correspond to each other. The figure contains parts taken from Paetzold et al. [PMS+21].

3.2 Single-Channel Gut Data

3.2.1 Data Properties

In the single-channel gut data immunolabeling was used to simultaneously but not distinctly visualize the gut's lymphatic and nervous system structures. Primary antibodies were employed to specifically target these structures during the immunolabeling process. In the second step, two secondary antibodies with exactly the same fluorophore were introduced to bind to the two primary antibodies. This results in a single-channel image where low intensities refer to the background while high intensities refer to the presence of either lymphatic or nervous system structures.

In contrast to the multi-channel gut data, the mouse gut was extracted, and the incubation was performed passively. Here the antibody-containing solution is not pumped through the vasculature, but the gut is submerged in the solution, which is constantly stirred.

Table 3.4: Single-channel gut dataset characteristics.

Single-Channel Gut Data	x-dim	y-dim	z-dim
Spatial Resolution / µm	1.625	1.625	6.000
Section 22-05			
Voxel Range	2048	2048	157
Size / mm	3.3	3.3	0.9
Section 35-24			
Voxel Range	2048	2048	183
Size / mm	3.3	3.3	1.1
Section 27-46			
Voxel Range	2048	2048	129
Size / mm	3.3	3.3	0.8

3.2.2 Segmentation and Raw Data Normalization

The already existing UNETR segmentation models can create a segmentation of the single-channel patches. This leads to suboptimal segmentation as the models are not trained exactly for the target task. In the future, this bottleneck can be eliminated by creating hand-annotated segmentations on the single-channel data and training a UNETR segmentation model for all vessel-like structures in the image. However, at the moment, such annotations are not available in a sufficient amount.

To circumvent the problem, a part of *Section 22-05* (see Table 3.4) that was hand labeled by a biologist was directly used as segmentation. Due to the lack of annotations, the other sections can not be further evaluated as of now.

Based on the segmentation, the raw data distribution can be estimated, and a robust normalization of the raw data using median and MAD is performed. Thereby, the scale and location of the raw voxel intensities are equal to the scale and location of the generated raw data from the merging procedures described in Section 3.1.

3.2.3 Graph Extraction and Line Graph

As described in Section 3.1, a graph and a line graph representation are extracted from a segmentation. The characteristics of the graph and line graph from the annotated patch in Section 22-05 are described in Table 3.5. The graph extracted from the hand-annotated regions in *Section 22-05* is here referred to as $\mathcal{G}_{22-05}$ and the associated line graph is named $\mathcal{L}_{22-05}$.

From a total of 4744 edges in $\mathcal{G}_{22-05}$ 2979 ($\sim 63\%$) were ascribed to the nerve class and 1754 ($\sim 27\%$) were ascribed to the lymph class. The rest, consisting of 11 nodes, was not assigned to any class because more than 50% of the centerline pixels were not part of the initial segmentation. These rare events occur due to the post-processing of the annotations, with a fill-hole and binary closing step.

Figure 3.3 schematically shows the hand-annotated region in *Section 22-05* (not cyan). Notably, the annotated region covers a significant portion of the whole *Section 22-05*.

Table 3.5: Single-channel gut graph $\mathcal{G}_{22-05}$ and line graph $\mathcal{L}_{22-05}$ characteristics. The graph $\mathcal{G}_{22-05}$ was extracted directly from the hand annotations on *Section 22-05*.

Characteristic	$\mathcal{G}_{22-05}$	$\mathcal{L}_{22-05}$
Total Nodes	3330	4744
Edges	4744	10169
Avg. Degree	2.85	4.29
Connected Components	1	1

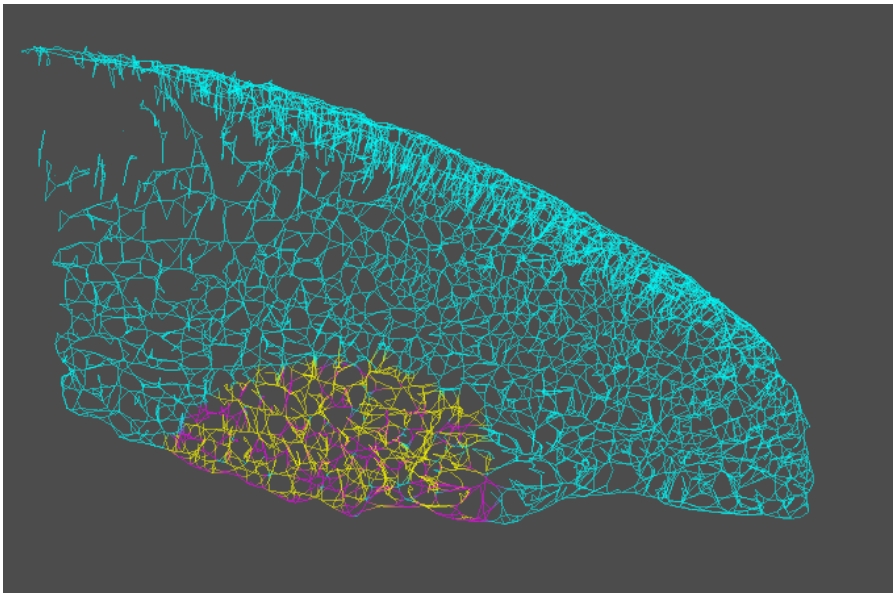

Figure 3.3: Location of the annotation in *Section 22-05*. Note that the displayed graph was extracted from a UNETR segmentation and is just a tool to visualize the annotated region. The graph $\mathcal{G}_{22-05}$ (and consequently $\mathcal{L}_{22-05}$) that is used for further evaluation was extracted directly from the hand annotations.

CHAPTER 4

Feature Engineering for Graphs

The standard Machine Learning (ML) approach for node classification on graphs combines existing node-level features with manual feature engineering to generate independent instances that are then classified using a ML algorithm. Some feature engineering approaches can be used for the inductive node representation learning setting, while others only work in the transductive setting and don't generalize to unseen graphs. In node classification, the transductive setting describes a scenario where all nodes for which inference is made are already present at training time. In contrast, in the inductive settings, the model needs to generalize to fully unseen graphs. Multiplexing the lymphatic and nervous systems corresponds to the inductive setting, as it is desired to have a model that generalizes to completely unseen and continuously generated gut graphs.

Figure 4.1 schematically describes the traditional ML node classification procedure for an inductive setting. Training, validation, and test set graphs must be disconnected before the feature generation. Otherwise, this can lead to an information leak between the nodes of each set.

Importantly the feature generation step is not obsolete also when GNN architectures instead of traditional algorithms are used.

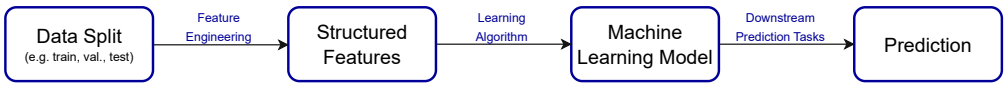

Figure 4.1: Traditional ML pipeline for node-level classification tasks.

4.1 Topological Features

Structural features about the graph that can be used are, e.g., node degree and the Graphlet Degree Vector (GDV), which represents a count vector for different graphlets that are rooted in a node. Other engineered features include node centrality features such as eigenvector, betweenness, or closeness centrality. When there are already existing node attributes, the new features can be concatenated with them. This approach generalizes to inductive node classification tasks for completely unlabeled graphs.

Many of these graph topological features can not be directly extracted by common GNNs architectures [XHLJ18, YGSYL21]. This makes the feature engineering step advantageous for the later application of GNNs as well as for other types ML classifiers.

From a biological point of view, it makes sense that structural information about the micro-environment contains helpful information for node classification. This suggests an increase in classification performance after the inclusion of these topological features.

4.2 Factorization-Based Embeddings

Another popular approach for the extraction of graph features is the generation of a low-dimensional embedding. This embedding is supposed to be a rich and compact representation of the high-dimensional node neighborhood. Famous approaches for this are, e.g., node2vec [GL16], deepwalk [PARS14] or line [TQW$^+$15]. These approaches work well for a transductive setting where the working graph is known from the start. However, they are not specifically designed for use in an inductive setting and can not guarantee to generate useful embeddings on completely unseen graphs. For that reason, node2vec and similar embeddings are not suitable for the target task of this project, which aims to create an inductive model.

4.3 Geometry Related Features

In scenarios where graphs are at the same time geometric graphs, it is possible to extract features from the Euclidean properties of these graphs. In $\mathbb{R}^2$, nodes, and edges can have areas and shape-specific properties related to them. Edges can have associated distances in Euclidean space and many more features. Such features can also be extracted from higher-dimensional Euclidean spaces.

The geometry of the structures that are modeled as graphs can be, e.g., extracted from a segmentation. In that sense, this is information transfer from a lower to a higher level of data abstraction. Many such features are extracted by the image-to-graph algorithm of Drees et al. [DSH$^+$21].

A feature not explicitly extracted by the mentioned graph generation algorithm is the directional vector of a vessel between two bifurcations/vessel endpoints. The problem

with the directional vector is that it can be defined in two ways for an undirected graph.

A unit directional vector, which is constrained to point only into one of two possible half-spaces, can overcome this problem. This direction constraint is enforced by only allowing positive x values. Furthermore, if the x value is zero, then no negative y value is permitted, and lastly, if both x and y are zero, then the z value must be positive. If one of those constraints is violated, the vector is simply inverted, i.e., it is multiplied by -1. This approach creates an additional, valuable feature that can be easily calculated for new graphs.

4.4 Raw Data Features

Graphs are often generated at a higher abstraction level from an underlying information-rich representation. E.g., spatial graphs can be created from an underlying image/volume representation. In such cases, node/edge features can be created from the associated raw data representation. Notably, the useful information from the raw data representation is very much task-dependent.

An excellent example of such cases can be found in street network graphs. Information about the vegetation bordering the road might be crucial for graph-level tasks, like the projected service time. In contrast, this information is probably irrelevant to other tasks, such as travel time predictions. Due to the task dependency of feature quality, feature extraction is ideally learned and targeted towards a specific prediction task.

In this work, first, a "handcrafted" image intensity feature extraction is proposed. This approach extracts the median, mean, max, min, 25% quantile, 75% quantile, and standard deviation from the centerline voxel intensity values. This approach catches the distributional information of the raw image data. However, all these descriptive statistics are order invariant, which means it is impossible to differentiate sequences with the same values but in a different order. Both sequences, $[1, 1, 2, 2, 3]$ and $[1, 2, 3, 2, 1]$, will yield the same results and are inseparable based on the mentioned statistics. To discriminate sequences based on intensity patterns, the simple distributional approach will thus fail.

This weakness was addressed with a learnable feature extractor that captures sequence-specific patterns. Furthermore, the feature extractor is directly integrated into the GNN classification architecture, allowing the backpropagation to the feature extractor. A detailed description of this approach follows in Section 5.2.3. From a biological perspective, such a feature extractor is useful when the binding of the antibodies follows specific patterns that are distinct for different vessel types.

CHAPTER 5

Experiments and Results

5.1 Data Split on Graphs

In contrast to traditional ML approaches, in graph learning, the data split needs to be done with care to the effects on the graph structure. Moreover, all edges between the training graphs and the evaluation graphs must be cut. Otherwise, message-passing occurs between them.

Various sampling methods to induce subgraphs are proposed in the literature, such as a random node or edge selection, random walks, or random walks with jumping probabilities [LF06]. Beyond simple sampling strategies, the problem boils down to graph partitioning, which is an active branch of research [BMS+16].

5.1.1 Random Sampling

Random node/edge sampling or random walks/jumps are one potential way to partition a graph. Here, the partitioning is created by the induced subgraph $\mathcal{G}_s$ of e.g., a sampled node set $\mathcal{V}_s$. The disadvantage of random node/edge sampling and random walks with jumping is that it does not ensure connectivity of the induced subgraph $\mathcal{G}_s$. While simple random walks ensure connectivity of $\mathcal{G}_s$, they do not ensure the connectivity of the induced subgraph $\mathcal{G}_r$ from the complementary nodes $\mathcal{V}_r = \mathcal{V} \setminus \mathcal{V}_s$.

Intact connectivity is crucial for GNNs to work properly, which makes random sampling approaches mostly unfavorable. Figure 5.1 shows how random node sampling creates a subgraph that has almost a full loss of connectivity between its nodes.

5.1.2 Breadth First Search

Breadth First Search (BFS) is another possibility to partition a graph, in contrast to Depth First Search (DFS), it ensures that the node subset is spreading out evenly.

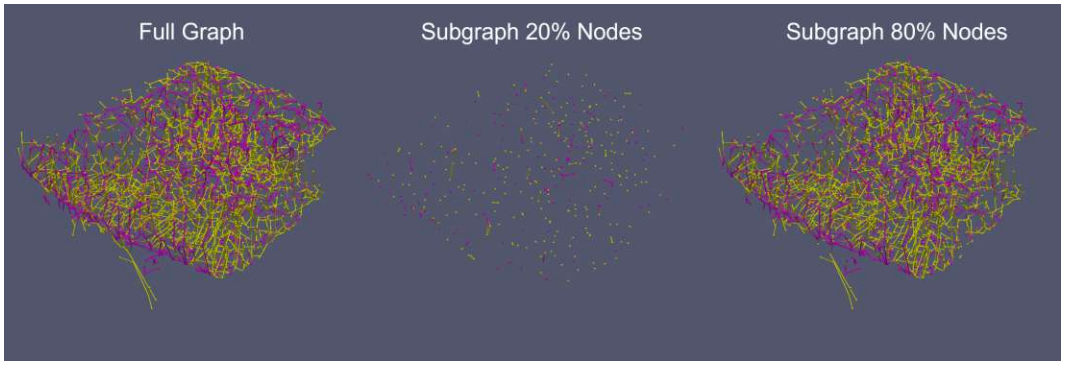

Figure 5.1: Example for a data split by random node selection. Random sampling does not guarantee a similar degree of connectivity as a geometric split. This is especially visible for the smaller set that contains only 20% of the total nodes. Random sampling-induced graph partitions are, therefore, highly undesirable for the creation of data split on a graph.

However, there are two major problems with BFS. First, the selection of a proper start node is necessary to create a meaningful partition. Second, the induced partition is potentially leading to a large cut size, which deteriorates the structural integrity of the subgraphs induced by the partitioning. A visualization of the shortcomings of BFS compared to spectral partitioning is displayed in Appendix A in Figure 1.

5.1.3 Geometric Partitioning

In the specific context of the gut graph, the nodes also represent points in $\mathbb{R}^3$. This allows for a data split based on the spatial position of nodes, by the elimination of all edges that connect two node sets $\mathcal{V}_1$ and $\mathcal{V}_2$. Where $\mathcal{V}_1$ and $\mathcal{V}_2$ are the subsets of $\mathcal{V}$ that lie on two different sides of a splitting plane H, and $\mathcal{V} = \mathcal{V}_1 \cup \mathcal{V}_2$.

In the biological context, such a split ideally causes a model to measure the generalization from one part of the gut to another part of the gut. While in a perfect scenario, the generalization of the models towards new specimens, i.e., new gut graphs, would be measured, this seems to be the best possible approximation when only a single specimen is available.

An ideal splitting hyperplane would create a partition of the graph that eliminates as few edges as possible and finding such a hyperplane is not a trivial task. Figure 5.2 visualizes different splitting hyperplanes on a small fraction of a gut graph. Notably, different positions for the hyperplanes have a huge impact on the structure of the partitioned graph. In Figure 5.2, the split along the y-axis would be desired as it creates a partition perpendicular to the direction of the lumen. However, for larger graphs, which consist of many gut loops with different orientations of the lumen, it is impossible to induce such a split with a geometric hyperplane.

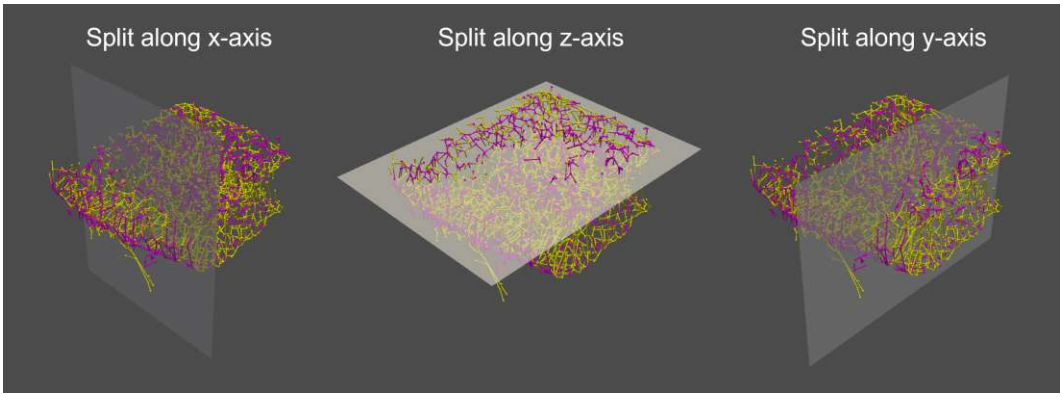

Figure 5.2: Example for a data split of a geometric graph along the x, y, and z-axis. Notably, the structural integrity is affected to varying degrees. The given split planes divide the data into two fractions with 20% and 80% of the data. When the graph consists of numerous gut loops, which is the case for $\mathcal{G}_{gut}$, such a split is likely to severely harm the structural integrity.

5.1.4 Spectral Partitioning

Spectral partitioning [DH73, Fie75, Che15] is derived from the need to create a bipartition that minimizes the number of cut edges. Formally a vector for the bipartition of the graph can be defined as

$$(x_i) := \begin{cases} 1, & i \in V_1 \\ -1, & i \in V_2 \end{cases} \quad , \tag{5.1}$$

with $x \in \{1, -1\}^n$ and a bipartition $(V_1, V_2 = V \setminus V_1)$ for a graph G. The number of edges that are cut by such a bipartition is proportional to

$$\mathbf{x}^T L \mathbf{x} = \sum_{\{i,j\} \in E} (x_i - x_j)^2 = 4 \cdot |E(V_1, V_2)| \quad , \tag{5.2}$$

where $E(V_1, V_2)$ is the set of edges that need to be cut to bipartition the node sets V_1 and V_2. This gives rise to the following optimization problem

$$\begin{aligned} &\min \mathbf{x}^T L \mathbf{x} \\ &\text{subject to } \mathbf{x}^T \mathbf{1} = 0 \\ &\qquad x_i \in \{-1, 1\} \quad , \end{aligned} \tag{5.3}$$

where $\mathbf{1}$ is a vector of ones.

Interestingly this problem can be solved considering the second smallest eigenvector of the eigendecomposition of the graph Laplacian L already, which was already introduced in

Section 2.2. In spectral graph theory, the number of zero eigenvalues corresponds to the number of connected components in a graph. This means the second smallest eigenvector can only then be used to induce a split if the graph is connected. This eigenvector is commonly referred to as Fiedler Vector.

The Fielder Vector can be used for partitioning the node sets according to the sorted weights in the vector. Notably, this partitioning does not guarantee the creation of two connected components, but through a simple transfer of nodes that induce a small graph to the other part of the bipartition, this problem can be solved.

Summarized, spectral partitioning generates subgraphs with minimal destruction of the structural integrity of the full graph. Additionally, it does not need visual supervision in contrast to a geometric splitting approach. Since spatial proximity is closely correlated with connectedness in the gut graph, the split from spectral partitioning results in a split that also closely resembles a spatial split. This leads to the desired characteristic of the split, that it separates different parts of the gut while maintaining the structural integrity of the partitioned parts.

5.1.5 Train-Validation-Test Split

Due to the given reasons spectral partitioning was used to generate a training, validation, and test set for the multi-channel gut graph (see 3.1 for details). The split was directly performed on the line graph, which ensures that no vessels are lost in the process of the split. Regarding the spatial partition that is induced by spectral partitioning, experiments showed that partitioning the graph or the dual graph results in almost identical splits.

For the split, only the largest connected component which covers $> 98\%$ of nodes (334411) was considered. First, a split separating the graph into two subgraphs of size 260840 (78%) and 73571 (22%) was induced. This induces a cut of size 7 on the graph, which suggests that the spectral partitioning finds a partition where the two parts of the graph are almost disconnected. The larger of the two splits resulted in the training graph, while the other graph was split again to create a test and a validation graph. A visual representation of the first split is displayed in Figure 5.3 where the yellow part displays the training graph and the cyan part contains the part which is split further. The displayed representation is the line graph representation $\mathcal{L}$ where the spatial position of the nodes is chosen as the center point of the connecting line segment between two bifurcations.

In a second split on the smaller graph, the validation and test graph are created. This split creates two subgraphs of size 30017 (9%) and 43554 (13%), that represent the validation and test graph, respectively. The cut size of this second split amounts to 118, which shows that more connections are destroyed than for the first split. However, compared to the total number of edges in both graphs, 118 is a relatively small number of lost edges. The unequal sizes of the two graphs are a result of the search for a split that maintains structural integrity. In contrast to the structural integrity of the graphs, perfect equality of the partitions' sizes is only of minor importance. Training, validation,

and test graph are displayed in Figure 5.4, with the associated colors yellow, purple, and cyan respectively.

It is important to mention that any feature engineering on the graph, that extracts topological information from the graph must be done strictly after the data split. Otherwise, this leads to an information leak between the train, validation, and test set.

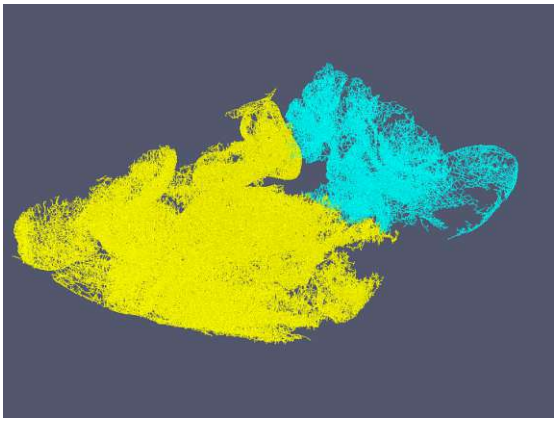

Figure 5.3: Graphical representation of the first split (training vs. rest) using spectral partitioning. The yellow part indicates the training graph. The displayed graph is the line graph $\mathcal{L}$ generated from the multi-channel gut data.

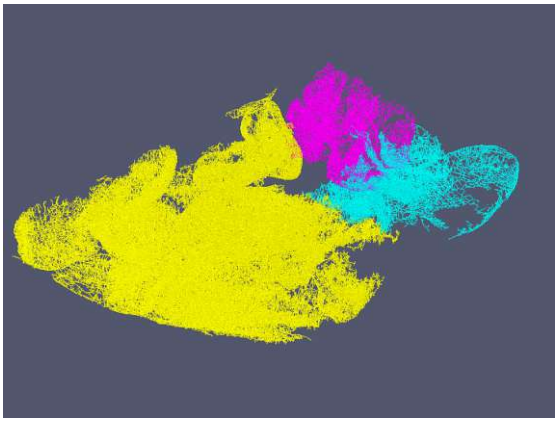

Figure 5.4: Graphical representation of the training (yellow), validation (purple), and test (cyan) graphs that are created by spectral partitioning. The displayed graph is the line graph $\mathcal{L}$ generated from the multi-channel gut data.

Table 5.1: Characteristics of the training, validation, and test graph created from spectral partitioning of the multichannel gut line graph $\mathcal{L}_{gut}$.

Line Graphs $\mathcal{L}$	*Count*	*Fraction*
Training Partition		
Nodes	260840	100%
Nerve Nodes	90040	35%
Lymph Nodes	170800	65%
Validation Partition		
Nodes	30017	100%
Nerve Nodes	14430	48%
Lymph Nodes	15587	52%
Test Partition		
Nodes	43554	100%
Nerve Nodes	18391	42%
Lymph Nodes	25163	58%

5.2 Classification with GNNs

Before the application of any classification algorithm, all instance features were centered and scaled. The parameters for centering and scaling are extracted exclusively from the training graph, which prevents information to leak from the training graph to the validation or test graph. Scaling and centering of the instance features from validation, test, and single-channel graphs are then done with the parameters extracted from the training graph.

5.2.1 Notes on the Computation Graph

The generated input data from the lymphatic and nervous system is already graph-structured and can potentially be directly used as input for GNN models. However, the existing graph structure is not necessarily optimal for the generation of expressive (in terms of classification performance) node embeddings. Based on the working principle of GNNs, the structure of the graph that is used for computation is essential for performance and needs to be chosen with care. In this work, different approaches that change the structure of the initial biological representation were investigated.

- **Line Graph:** GNNs generally rely on passing information between nodes. This means that effectively the node features are crucial for the message-passing step, while edges only indicate where the message passing occurs. In the biological graph of the nervous and lymphatic system, the major information lies in the vessels and not in the nodes which only represent bifurcation points, but no structural information about the vessels. A line graph representation, where the vessels are represented as nodes and edges represent bifurcation points that connect edges, is, therefore, a better alternative. The line graph transformation for the gut graph is discussed in detail in the dataset description.
- **k-NN Graph:** The Dynamic Edge Convolution layer proposed by Wang et al. [WSL+19] updates not only the node embeddings but also the connectivity of the graph. After every layer, the nodes are reconnected with the k-nearest neighbors in the features space, thereby creating a k-NN graph. This non-static behavior of the graph structure can be advantageous to cluster nodes that have structurally similar roles in the initial graph. This, in turn, potentially eases the classification task.

5.2.2 Base GNN Architecture

The base architecture is a GNN that operates directly on the line graph representation of the gut datasets. It uses the refined embeddings from the feature generation steps that are outlined in Chapter 4. An overview of the basic hyperparameters for the GNN architecture with a short description is provided in Table 5.2.

From the vast number of different GNN layer types, this work focuses on some of the predominant architectures that were partly discussed in detail in Section 2.2. Namely,

these layer types are SAGE, GCN, Graph Attention Network (GAT), and, later on, also EdgeConv and the associated DGCNN model.

Other prominent architectures, such as ClusterGCN [CLS+19], were not used due to their focus on large graphs that require a batching procedure for efficient training. This is especially important for huge graphs that do not fit on GPU memory. This memory limitation does not pose a problem for processing the gut graph. Moreover, ClusterGCN works best if intuitive clusterings of the graph are available. Since the gut graph represents a continuous structure, this is not the case.

Results:

Using Bayesian optimization, the ideal model parameters were identified with a hyperparameter sweep consisting of 50 individual runs. The sweeps were always performed with a fixed layer type, while the other parameters were adjustable for optimization.

For all parameter sweeps, WANDB (Weights and Biases) was used [Bie20]. Model selection was performed according to the balanced accuracy on the validation graph. Then the best models from the hyperparameter tuning were evaluated for their performance on the test graph and for some models also on the single-channel graph $\mathcal{L}_{22-05}$. During the training, the AdamW [KB15, LH17] optimizer was used, and the loss was defined using the Binary Cross Entropy (BCE).

An overview of the results achieved with the ideal hyperparameter settings for different layer types is provided in Table 5.3, and more detailed results are in Appendix A in Table 3 and 4. Table 5.3 summarizes the performance regarding the accuracy, balanced accuracy, and the F_1 score for the nerve classification and the lymph classification, denoted as F_1 - N and F_1 - L. Out of all the evaluated layer types, SAGE generated the best results on the validation graph. In the first evaluation, all models were compared on the data set without centerline statistics. Then in the second step, SAGE was further evaluated with the centerline statistics, due to the favorable performance compared to the other GNN layer types.

The SAGE model trained on the training graph from the multichannel gut data also generated excellent results on the graph $\mathcal{L}_{22-05}$ from the single-channel data. A further evaluation regarding their Receiver Operating Characteristic (ROC) curves and the Area Under the Curve (AUC) scores on the test set and the $\mathcal{L}_{22-05}$ graph is displayed in Figure 5.5 and 5.6. The hyperparameter setting for this SAGE model was as follows:

Learning Rate: 0.004802, *Weight Decay: 0.00023*, *Dropout: 0.5*, *No. Layers: 4*, *Hidden Channels: 64*, *MLP: Yes*, *Skip Connections: Yes* and *Aggregation: Mean*.

An insightful visual comparison of the SAGE predictions with the ground truth from the hand annotation for the graph $\mathcal{L}_{22-05}$ is displayed in Figure 5.7 and 5.8. The predictions that are shown in Figure 5.8 are from the best SAGE model without the use of any centerline information.

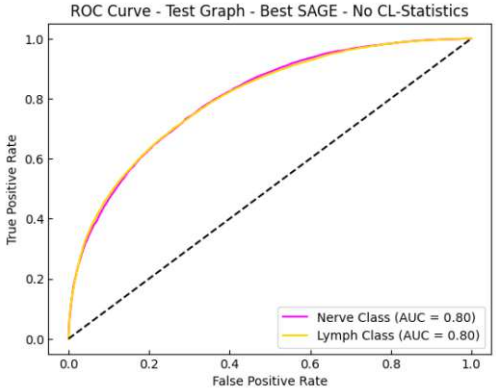

Figure 5.5: ROC Curve on the test graph for the best SAGE classifier that does not use any raw data information.

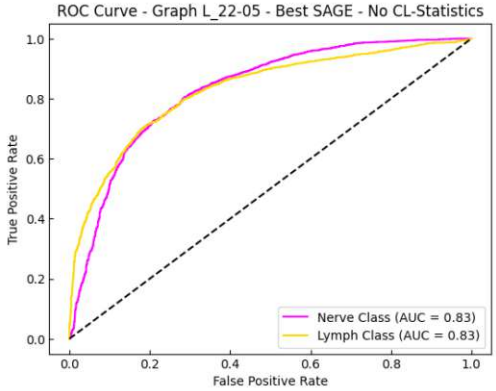

Figure 5.6: ROC Curve on the graph $\mathcal{L}_{22-05}$ for the best SAGE classifier that does not use any raw data information.

Both displayed plots are line graphs. To create a visualization of the line graphs in $\mathbb{R}^3$, the position of the nodes is chosen as the middle of the straight line segment that defines an edge in the initial graph $\mathcal{G}$. Notably, the model succeeds in creating a prediction where the lymph network, as well as the nerve network on its own, stay nicely connected. This is a desired property as the biological expectation is full connectivity within each network. Alterations from this expectation in the ground truth can occur due to the limited size of the annotated patch and errors along the graph generation pipeline. A more detailed discussion of this behavior follows in the main results section.

In Appendix A, Figure 3 visually shows the correctly and wrongly classified nodes. This visualization impressively displays how errors occur almost exclusively at the border regions of both networks. It can be concluded that the interaction areas of the two biological networks are the most difficult part of the classification tasks.

Upon evaluating the covered total length of accurately classified vessel-like structures, it was observed that the fraction of the total vessel length that is correctly classified is comparable to the accuracy scores. Specifically, 74.88% of the total length is correctly classified for the graph $\mathcal{L}_{22-05}$. This suggests that there is no significant bias towards either short or long vessels being correctly classified.

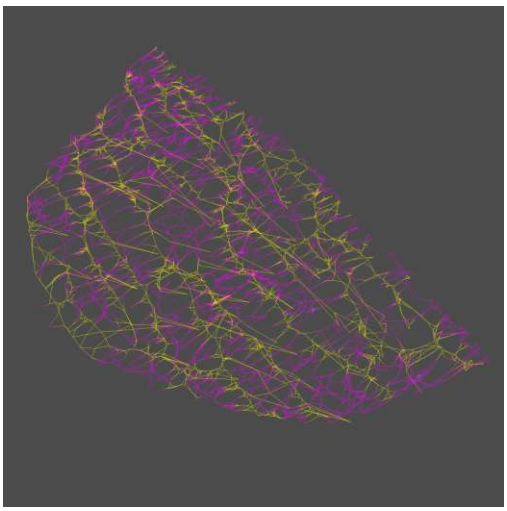

Figure 5.7: Ground truth line graph from the annotations on *Section 22-05*. Magenta corresponds to nerve nodes, yellow corresponds to lymph nodes, and cyan to unlabeled nodes.

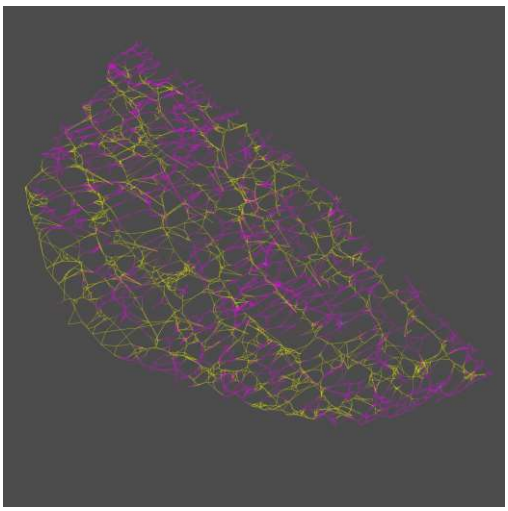

Figure 5.8: Predictions for the line graph from the annotations on *Section 22-05*. Magenta corresponds to nerve nodes, yellow corresponds to lymph nodes, and cyan to unlabeled nodes.

Table 5.2: Hyperparameters for the GNN model for the lymph/nerve classification. The table gives an overview of all the optimized hyperparameters.

Hyperparameter	*Description*
No. of Layers	The number of GNN layers that are in series. A large number of layers makes the embedding of the nodes dependent on a large neighborhood.
Layer Type	The type of GNN layer that is used in the model. One of SAGE, EdgeConv, GCN or GAT [VCC+17].
Multilayer Perceptron	Whether or not a MLP (two-layers) is attached before and after the GNN layers. This makes architecture more expressive while maintaining the perceptive field for each node.
Hidden Channels	The number of hidden channels for the GNN layers.
Aggregation Scheme	Defines the aggregation type performed in the SAGE layers. It is also possible that multiple different aggregation schemes are used. Common aggregation modes are mean, max, standard deviation, and LSTMs on random permutations as proposed in the SAGE paper[HYL17].
Skip Connections	Whether or not skip connections in the form of additions[HZRS16] are realized between consecutive GNN layers. In SAGE layers, the concatenation of the old node embedding already comes close to a skip connection.
Dropout	Compared to classical CNNs where dropout is generally only applied on the final MLP head that works on top of the convolutional feature extractor [SHK+14], it is common to include dropout layerwise in GNN architectures. The indicated dropout ratio is introduced after every layer of the GNN architecture.
Learning Rate	The learning rate for the used AdamW [KB15, LH17]. To optimize the training process for the hybrid architecture, the learning rate for the LSTM is separately adjustable by parameter grouping.
Weight Decay	The weight decay, again, the parameter can be separately set for LSTM.
Hidden Channels LSTM	The number of hidden channels for the LSTM feature extractor.

Table 5.3: Performance of the GNN models without a coupled feature extractor. In some experiments, the centerline statistics were used as features while other models are completely unaware of the raw intensity data. F_1 - N and F_1 - L denote the F_1 scores for the nerve and lymph classification.

Dataset		**Performance Metrics**			
Graph	**Raw Data**	**Acc.**	**Bal. Acc.**	**F_1 - N**	**F_1 - L**
SAGE CL-Statistics					
Validation Graph	Max Comb	0.7007	0.7021	0.7039	0.6974
	Mean Comb	0.7240	0.7251	0.7239	0.7241
Test Graph	Max Comb	0.7740	0.7678	0.7311	0.8051
	Mean Comb	0.8079	0.7993	0.7658	0.8372
$\mathcal{L}_{22-05}$	Max Comb	0.7435	0.6924	0.8137	0.5885
SAGE No CL-Statistics					
Validation Graph	Not Used	0.6707	0.6728	0.6805	0.6599
Test Graph	Not Used	0.7185	0.7175	0.6807	0.7483
$\mathcal{L}_{22-05}$	Not Used	0.7699	0.7586	0.8144	0.6972
GCN No CL-Statistics					
Validation Graph	Not Used	0.6548	0.6589	0.6813	0.6234
Test Graph	Not Used	0.7013	0.7080	0.6799	0.7200
GAT No CL-Statistics					
Validation Graph	Not Used	0.6043	0.6022	0.5711	0.6327
Test Graph	Not Used	0.6752	0.6607	0.5961	0.7284

5.2.3 Hybrid Architecture

The idea of the hybrid architecture is to combine a GNN operating on the graph representation with a feature extractor that is directly operating on the raw volume data. A direct combination of a feature extractor with the GNN eliminates the need for the repeated hand-crafting of features for different use cases. Figure 5.9 shows the working principle of the proposed architecture.

The centerline feature extractor can be realized by any given model that is capable of extracting a fixed-length feature vector from an input with a variable sequence length. One of the possible models consists of 1D convolutions followed by a pooling operation, e.g., mean or max pooling. This architecture allows the extraction of specific patterns along the centerline that are characteristic of a certain class. However, problems arise with extremely short sequences that limit the length of the possible 1D convolution kernels. An alternative to the 1D convolution is an LSTM model [HS97] as feature extractor. These networks are well known to show excellent performance in the classification and feature extraction of variable-length sequences. LSTM networks are a form of Recurrent Neural Network (RNN). From the final state, a fixed-length feature vector can be extracted and used for concatenation with the existing embeddings. Whereas with 1D convolutions, it is necessary to perform a pooling step to achieve a predefined length, in the LSTM it is simply possible to use the features of the last state of the LSTM. Moreover, the LSTM can be made bidirectional, making it invariant towards the centerline's direction.

The biological reasoning for the proposed architecture is the hypothesis that the antibody binding is not uniform along a certain vessel structure. This would result in different intensity patterns along a vessel that can then be leveraged to improve classification performance.

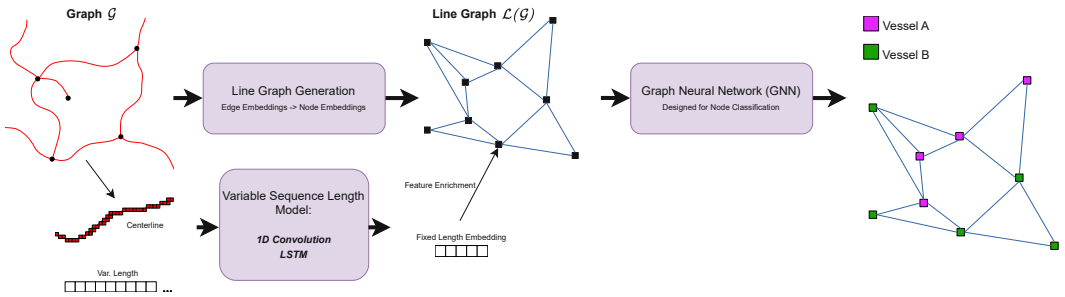

Figure 5.9: Architecture for the extraction of raw data features from the centerline using a variable length deep learning model (RNN, 1D convolution) combined with consecutive node classification using a GNN.

Results:

Table 5.4 summarizes the results for the LSTM coupled GNN. The performance is roughly comparable to the use of just centerline statistics (see Table 5.3).

Table 5.4: Performance of the SAGE models with a coupled LSTM feature extractor. F_1 - N and F_1 - L denote the F_1 scores for the nerve and lymph classification.

Dataset		Performance Metrics			
Graph	**Raw Data**	**Acc.**	**Bal. Acc.**	**F_1 - N**	**F_1 - L**
SAGE LSTM					
Validation Graph	Max Comb	0.7026	0.7045	0.7095	0.6952
	Mean Comb	0.7210	0.7224	0.7231	0.7189
Test Graph	Max Comb	0.7746	0.7687	0.7325	0.8052
	Mean Comb	0.8094	0.7987	0.7639	0.8402

5.2.4 DGCNN Architecture

The DGCNN architecture is expected to work well when the relationship of node embeddings to all its neighbors is more important than just the aggregated information from the neighborhood embeddings. This is enforced by the EdgeConv GNN layer that is used in DGCNN. Moreover, the dynamic change of the graph connectivity using a k-NN graph might be beneficial to allow for a better clustering of same-type nodes. This concept has been shown to be efficient for whole brain vessel graphs [WPP+23].

A disadvantage of the DGCNN structure lies in the increase of computation time that arises from the dynamic change of the graph structure, which makes a neighborhood search for every node necessary after every single layer. Implementation-wise, this is realized by the construction of a KD-Tree in $O(n \log n)$ and the neighborhood querying of every point in the KD-Tree in $O(k \cdot n \log n)$.

Results:

First, the network was evaluated only with the simplest configuration, meaning that no centerline statistics were used and also no learnable feature extraction in form of an LSTM was used.

Simply using the edge convolution while maintaining the static graph did not show favorable results in a hyperparameter optimization sweep with 25 separate evaluated models. The best model yielded a balanced accuracy of 0.6555 and an accuracy of 0.6515 on the validation set (see Table 5.5), which is clearly no improvement to the evaluated SAGE models. For the dynamic graph structure, a single promising hyperparameter setting was evaluated but again failed to improve performance. Due to the increased training time of the dynamically changing graph structure and the lack of performance, this approach was not considered further.

Table 5.5: Performance of the EdgeConv models with a static graph structure. F_1 - N and F_1 - L denote the F_1 scores for the nerve and lymph classification.

Dataset	**Performance Metrics**			
Graph	**Acc.**	**Bal. Acc.**	**F_1 - N**	**F_1 - L**
EdgeConv (Static Graph) No CL-Statistics				
Validation Graph	0.6515	0.6555	0.6766	0.6222
Test Graph	0.7044	0.7090	0.6785	0.7264

5.3 Classification with Baseline Algorithms

Factorization followed by classification is a well-established and high-performing approach in the transductive setting. However, this does not work in an inductive setting and can therefore not be considered a baseline.

The alternative is the extraction of topological features from the graph to enrich node embeddings, followed by traditional ML approaches that consider the nodes as independent instances as described in Chapter 4. The baselines were evaluated with the same feature vectors that were already used for the classification with GNNs.

Due to the comparably smaller computational cost, these models could be evaluated based on Cross-Validation (CV), which ensures higher confidence in the results. However, this comes at the expense of comparability to the GNN models. Moreover, as described earlier, the partitioning of a graph is a non-trivial task that prevents the out-of-the-box application of CV. For that reason, the same procedure as for the GNN models was chosen. Namely, the multi-channel gut graph split into a training, validation, and test graph was used, followed by feature generation, model selection on the validation graph, and performance estimation on the test graph and the single-channel gut graph.

5.3.1 Random Forest

Random Forest (RF) is a versatile classification algorithm that provides good performance on a wide range of classification tasks. The hyperparameter grid that was used to find the optimal model is displayed in 5.6. For the model selection, balanced accuracy was used. The results for the model that performed best on the validation graph are described in Table 5.7. For the full results refer to Table 2 in Appendix A.

Table 5.6: Hyperparameter grid search for the RF model.

RF Parameters	*Values*
Number of Decision Trees	$\{10, 20, 100, 200\}$
Tree Depth	$\{6, 10, 14, 18, 20\}$
Features per Split	$\{\log_2(p), \sqrt{p}\}$

Table 5.7: Evaluation of the Random Forest Models with the optimal hyperparameter setting. Setting with features from max combined raw data: (Trees: 200 Depth: 20, Features per Split: $\sqrt{p}$). Setting with features from mean combined raw data: (Trees: 200, Depth: 18, Features per Split: $\sqrt{p}$). Setting without centerline features: (Trees: 100, Depth: 20, Features per Split: $\sqrt{p}$). F_1 - N and F_1 - L denote the F_1 scores for the nerve and lymph classification.

Dataset		**Performance Metrics**			
Graph	**Raw Data**	**Acc.**	**Bal. Acc.**	**F_1 - N**	**F_1 - L**
Random Forest CL-Statistics					
Validation Graph	Max Comb	0.6494	0.6464	0.6094	0.6820
	Mean Comb	0.6718	0.6690	0.6362	0.7011
Test Graph	Max Comb	0.7239	0.7058	0.6432	0.7748
	Mean Comb	0.7541	0.7284	0.6592	0.8077
Random Forest No CL-Statistics					
Validation Graph	Not used	0.6000	0.5936	0.5071	0.6635
Test Graph	Not used	0.6638	0.6343	0.5277	0.7389
$\mathcal{L}_{22-05}$	Not used	0.7467	0.7179	0.8047	0.6396

5.3.2 SVM

Support Vector Machines (SVMs) are among the best-performing non-deep learning models in ML. Considering this, it makes sense to use them as a baseline classification algorithm. SVMs perform a classification of the data based on a separating hyperplane. However, this separation is done in higher dimensionality using kernelization, which allows non-linear classification boundaries. An overview of the hyperparameter grid is provided in Table 5.8. The results for the model that performed best on the validation graph regarding balanced accuracy are described in Table 5.9. For the full results refer to Table 2 in Appendix A.

Table 5.8: Hyperparameter grid search for the SVM model.

SVM Parameters	*Values*
Reg. Param. C	{0.1, 1}
Kernel	{*Linear*, *Radial Basis Function (RBF)* }

Table 5.9: Evaluation of the SVM model with the optimal hyperparameter setting. Setting with features from max combined raw data: (Reg. Param. C: 1 Kernel: RBF) Setting with features from mean combined raw data:: (Reg. Param. C: 1, Kernel: RBF). Setting without centerline features: (Reg. Param. C: 1, Kernel: RBF). F_1 - N and F_1 - L denote the F_1 scores for the nerve and lymph classification.

Dataset		**Performance Metrics**			
Graph	**Raw Data**	**Acc.**	**Bal. Acc.**	**F_1 - N**	**F_1 - L**
Support Vector Machine CL-Statistics					
Validation Graph	Max Comb	0.6325	0.6277	0.5682	0.6801
	Mean Comb	0.6590	0.6556	0.6151	0.6939
Test Graph	Max Comb	0.6990	0.6727	0.5857	0.7637
	Mean Comb	0.7427	0.7152	0.6386	0.8002
Support Vector Machine No CL-Statistics					
Validation Graph	Not used	0.5779	0.5693	0.4405	0.6612
Test Graph	Not used	0.6486	0.6108	0.4691	0.7374
$\mathcal{L}_{22-05}$	Not used	0.7418	0.6730	0.8207	0.5389

5.4 Main Results

5.4.1 Baseline vs GNN

A comparison of the classification performance with exactly the same features shows the superiority of GNN models compared to standard algorithms (SVM and RF) that disregard network information for message passing. Consistently the classification performance in terms of accuracy, balanced accuracy, and F_1-scores shows that the SAGE architecture is the superior model. Notably, the optimization effort for the baseline models is not comparable to the optimization effort of the GNN models. However, the quite drastic performance difference of often close to 5% indicates that there is an unbridgeable performance difference between standard ML classifiers and GNNs for the multiplexing task.

The comparison to the standard algorithms also showed that hyperparameter tuning and the selection of the right GNN architecture is crucial to achieving top performance. Often, wrong settings of the hyperparameters, such as the wrong layer type or a wrong aggregation mode already nullify the advantages compared to simple algorithms. Efforts to easily find the ideal GNN architecture like the GraphGym approach by You et al.[YYL20] are therefore highly relevant to the field of geometric deep learning.

A detailed investigation of the differences in the SAGE model (without CL-Statistics) predictions to the baseline predictions showed that the SAGE model is better at enforcing connectivity among the lymph and nerve network. Table 5.10 shows the differences between the SAGE and RF predictions for the within-call connectivity on the graph $\mathcal{L}_{22-05}$. With only the two largest nerve components for the SAGE predictions, already

$\sim 58\%$ of the nerve nodes are correctly classified, for the lymph components the respective value is $\sim 69\%$. This stands in strong contrast to the RF predictions where the two values are $\sim 44\%$ and $\sim 28\%$ respectively. The trend stays the same for all the k values ranging from 1 to 10. The k values represent the number of considered connected components ordered by decreasing size.

The induced nerve subgraph on $\mathcal{L}_{22-05}$ of the ground truth nerve nodes consists of 33 connected components, but the largest among the connected components covers already 90% of all nerve nodes. The induced lymph subgraph consists of 12 connected components with the largest connected component covering 94% of all lymph nodes. A comparison of the number of connected components for the induced subgraphs, from SAGE and RF predictions and also from the ground truth labels is exhibited in Table 5.11. It shows that the SAGE predictions produce fewer connected components, even at some point over-connecting the networks. However, the number of connected components for the within-class subgraphs is still better approximated by the SAGE predictions.

An analysis that combines aspects of connectivity with classification accuracy is the Jaccard index

$$J(A, B_k) = \frac{|A \cap B_k|}{|A \cap B_k|} \quad , \tag{5.4}$$

when A is the node set of the largest connected component of the induced subgraph of a certain class based on the ground truth and B_k is the union of the k largest connected components of the subgraph from the respective class that is created from the predictions. The results displayed in Appendix A in Table 5 underline the superior behavior of the SAGE model compared to a simple baseline when it comes to the simultaneous evaluation of accuracy and connectivity. The observed trends are similar to the results that are depcited in Table 5.10.

The SAGE algorithm with mean aggregation exhibits far superior connectivity compared to other models due to its message passing and aggregation scheme. As previously discussed in the Background Chapter, a GCN functions similarly to a smoothing operator where neighboring node embeddings gradually converge to the same embedding. This behavior is also observed in the SAGE model with a mean aggregation mode, which closely resembles the GCN model. Consequently, the SAGE model with mean aggregation mode enhances connectivity within a class, which is desirable for the given task. Figure 5.10 illustrates this behavior.

Preserving connectivity is crucial, especially for common graph analysis tasks like detecting the shortest path or calculating flow. Future biological assessments of the lymphatic system graph and the nervous system graph are therefore dependent on intact connectivity within the systems.

Table 5.10: Connectivity analysis using connected components. For the analysis, the induced subgraph from all the predicted nerve nodes and the induced subgraph from all the predicted lymph nodes are used.

$\mathcal{L}_{22-05}$ Connect. Analysis	SAGE No CL		RF No CL	
Con. Comps. k	Recall@k Nerve Subgraph	Recall@k Lymph Subgraph	Recall@k Nerve Subgraph	Recall@k Lymph Subgraph
1	0.3273	0.6140	0.2779	0.0895
2	0.5750	0.6916	0.4710	0.1807
3	0.6230	0.7109	0.5445	0.2531
4	0.6465	0.7109	0.5737	0.2970
5	0.6700	0.7109	0.6029	0.3284
6	0.6868	0.7109	0.6318	0.3637
7	0.7043	0.7127	0.6566	0.3905
8	0.7190	0.7138	0.6761	0.4156
9	0.7328	0.7144	0.6888	0.4379
10	0.7402	0.7149	0.6986	0.4527

Table 5.11: Connected component comparison of the induced subgraphs of classified nerve and lymph nodes. The SAGE and the RF models are the models without centerline statistics that performed best on the validation graph.

$\mathcal{L}_{22-05}$ Comparison	Connected Components	Isolated Nodes
Induced Nerve Subgraph		
Ground Truth	33	11
SAGE Predictions	10	2
RF Predictions	187	87
Induced Lymph Subgraph		
Ground Truth	12	6
SAGE Predictions	77	26
RF Predictions	255	154

5.4.2 GNN Model Comparison

The SAGE models outperformed the GAT, GCN, and EdgeConv approaches that were evaluated. The comparison was based on the input graph with the least information content (no raw data information), and since GAT, GCN, EdgeConv did not perform well, they were omitted in favor of SAGE for further analysis on the single-channel graph and experiments with a learnable raw data feature extractor.

The favorable performance compared to the GCN suggests that the initial node embedding is important for the classification since SAGE concatenates the old node embedding in every layer while the GCN only introduces self-loops. Another hint towards this

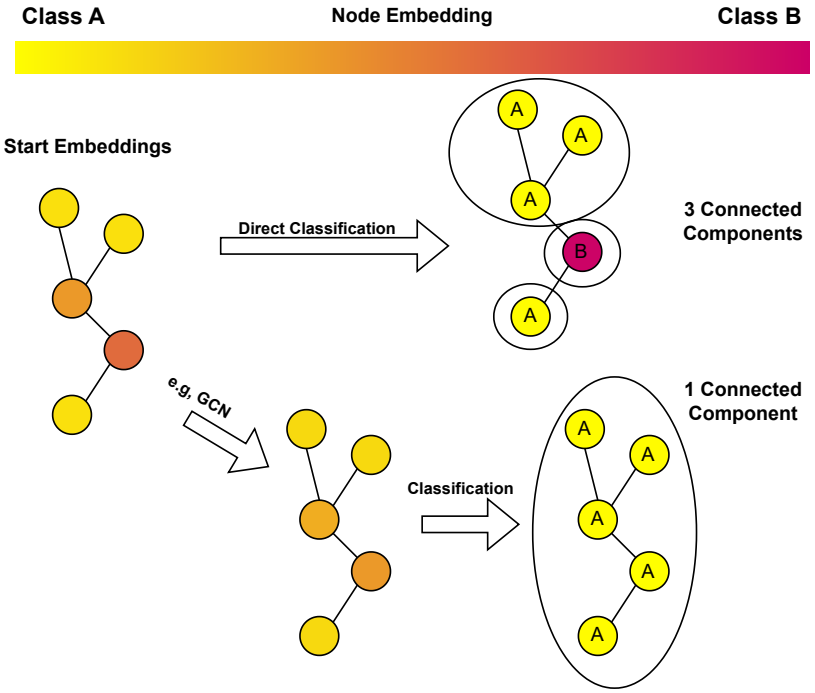

Figure 5.10: Illustration of how a GCN or SAGE with mean aggregation are enhancing the within-class connectivity.

assumption is that in the best-performing hyperparameter settings, skip connections were consistently used across all layer types. Interestingly, even for the SAGE layers that inherently used skip connections by concatenation, the skip connections in form of additions were still favorable.

From the comparison to EdgeConv, where the relationship of node embeddings is at focus, it can be concluded that not the relationship of embeddings is decisive for the classification but the pooled neighborhood information.

Finally, the trend towards good performance with the mean aggregator compared to standard, deviation, max, or combined aggregators indicate that no single nodes dominate the classification in a certain neighborhood, and the averaged features in the neighborhoods are more decisive than the variability of the neighborhood embeddings.

5.4.3 Impact of Raw Image Information

Both proposed merging scenarios result in useful information for classification, as evidenced by the performance gain achieved through the introduction of voxel intensity information along the centerline in the form of descriptive statistics. The mean combination mode produces a stronger performance increase than the max combination mode, indicating that the latter is a poor choice for estimating the worst-case performance

increase of raw data information for the single-channel data set.

On the multi-channel gut graph, the introduction of centerline statistics leads to a relatively consistent increase of approximately 5% in balanced accuracy and accuracy for all the evaluated GNN and baseline classifiers. This observation roughly estimates the potential performance increase achievable when properly utilizing raw data information. However, as expected, directly applying these trained models to the single-channel gut graph $\mathcal{L}_{22-05}$ did not yield the same results. Detailed results can be found in Appendix A in Table 3 and in Figure 2 that shows the ROC for the predictions on the $\mathcal{L}_{22-05}$ graph. The lacking performance can be partly attributed to the different intensity distributions of the lymph and nerve voxels in the single-channel volume compared to the assumed distributions in the merged multi-channel volume. This is evident from the voxel intensity distributions of the hand annotations of the single-channel volume *Section 22-05* (refer to Figure 5.11), which show that the lymph voxel intensity distribution and the nerve voxel density distribution neither share the same location nor scale.

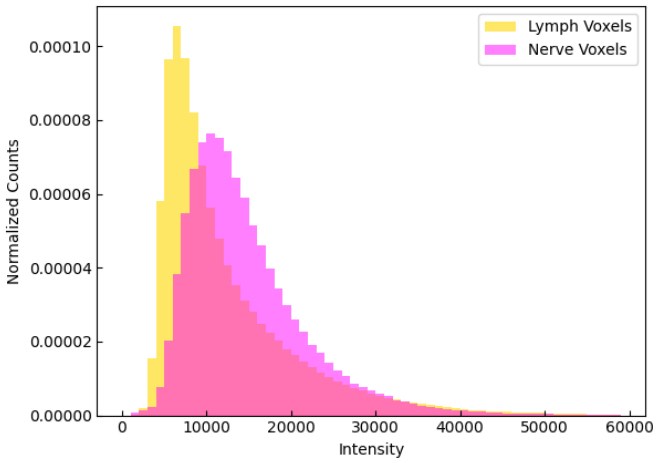

Figure 5.11: Comparison of the voxel intensities of nerve and lymph voxels from the single channel data (*Section 22-05*). The considered voxels were hand-annotated by a biologist and assigned to the respective class. The intensity is mapped within the *int16* data range. Both histograms are normalized.

One approach to address this issue is to leverage the pre-existing information of the different locations and scales from the single-channel data for the scaling of the distributions in the multi-channel data. Experiments that applied this approach demonstrated a significant improvement in performance on the single-channel graph, but it still fell short of the classifiers trained without any centerline information. It is highly probable that the GNN classifier relies heavily on centerline information for classification, and even minor changes in the intensity distribution can greatly impair the classifier's performance. This assumption is reinforced by the number of layers employed in the SAGE model with and without centerline statistics. The majority of the top 5 SAGE models that lack

centerline statistics used four layers, while the majority of the top 5 models with centerline information (raw data max combined) had only a single GNN layer. This suggests that the models without centerline statistics first need to refine the node embeddings based on the neighborhood, while those with centerline information already possess expressive enough node embeddings for classification.

In the future, fine-tuning the model on annotated single-channel gut data could potentially solve this issue, but presently, such data is unavailable.

5.4.4 Impact of the Hybrid Architecture

The hybrid architecture may not yield drastic performance benefits, but it is a versatile approach that can adapt to different tasks. Specifically, in scenarios where distributional information is insufficient, an LSTM can still extract useful information. Although the benefits of the hybrid architecture may not be immediately apparent in terms of performance metrics, the general concept of a GNN model with the capacity for targeted and learnable feature extraction in a different data representation domain is highly intriguing and warrants further investigation.

5.4.5 DGCNN vs Other GNNs

Based on the limited number of experiments, it appears that there is no apparent benefit in utilizing edge convolutions and a dynamic graph structure. The nearest neighbor approach used to create the dynamic k-NN graph sampling was found to be detrimental to the runtime. Furthermore, the dynamic graph calculation does not enforce within-class connectivity on the initial input graph since connectivity is only promoted within the dynamic k-nearest neighbors graph and not within the initial graph. In conclusion, the DGCNN structure does not appear to be suitable for the given task as it does not improve classification performance or intra-class connectivity.

CHAPTER 6

Conclusion and Outlook

6.1 Conclusion

Despite the success of GNNs in the biomedical domain, e.g., in applications such as drug development and discovery, disease prediction in bioinformatics, and advanced image processing [ZLLT21, ZCH+20, WPC+20], research on using GNNs for tasks on biological network graphs is limited. This work explores a use-case where GNNs are applied to a graph representing the lymphatic and nervous systems in the gut of mice, by solving the multiplexing problem with a graph representation. The results show that GNNs constitute an excellent group of methods for this type of graph representation. Regarding traditional performance metrics such as accuracy, balanced accuracy, and F_1 scores, the best GNN model clearly outperformed the baselines. Beyond that, a more detailed analysis of connectivity revealed the favorable properties of the GNN predictions.

Given the drastic progress in LSFM and other biomedical imaging technologies, it is likely that similar tasks will arise frequently in the near future. In this regard, the presented work provides a first reference point for architectural choices and preprocessing routines that have proven to be promising.

6.2 Outlook

The presented work is just a starting point towards a whole range of biomedical tasks that inherently deal with graph-structured data and promise to be solvable with GNNs. Future works could include investigating pathological changes in biological networks, such as the classification of bowel diseases like Crohn's disease or ulcerative colitis. Previous studies have already shown a connection between structural changes in the lymphatic system with inflammations, e.g., caused by Inflammatory Bowel Disease (IBD) [RDBD+11, RSR+18]. Combined with these graph-level tasks, there are also node-level and subgraph-level

tasks, such as detecting single pathological vessels or regions that show pathological alterations. Comparable tasks exist in the nervous system, where it could be beneficial to identify changes in the network that are associated with pathological conditions. In addition to pure predictive tasks, the graph representations will allow for a more detailed understanding of structural properties that are potentially associated with pathological changes. Approaches such as GNNExplainer [YBY+19], which identify critical subgraph structures for predictions, will facilitate interpretability and explainability for these applications of GNNs.

Another medical application where GNN will most likely be of significance soon is Ophthalmology. In the evaluation of Optical Coherence Tomography Angiography (OCTA) data, that visualizes the vessel structures in the eye, GNN are potentially an ideal architecture for predictions on diseases and disease progressions such as Diabetic Retinopathy (DR) and Age-Related Macular degeneration (AMD) [MHL+23].

Appendix A

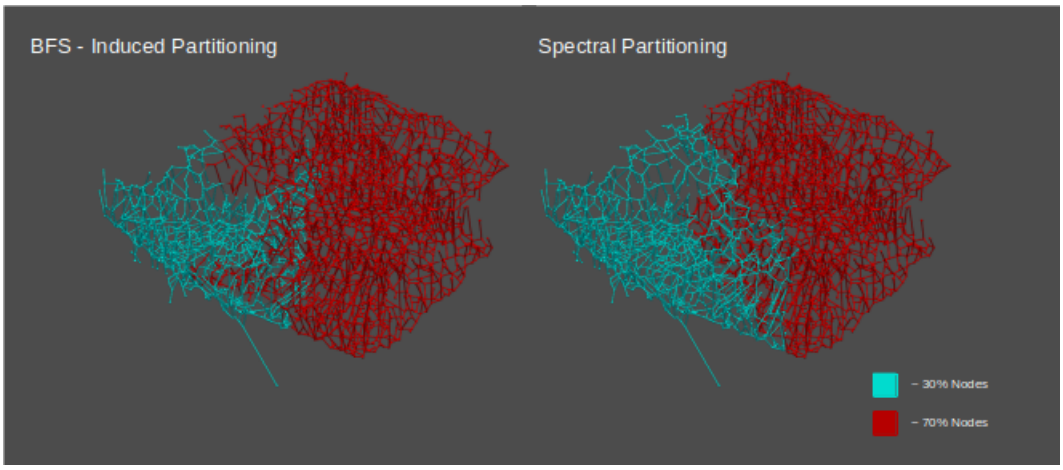

Figure 1: Comparison of the induced splits on a graph by BFS (starting from one of the left outer nodes in the image) versus spectral partitioning. Notably, even after selecting a good start node for BFS the split created by spectral partitioning is more even and induces a smaller cut. The cut sizes (= number of deleted edges to induce the bipartition) are 75 and 50 for the BFS and spectral partitioning respectively. Consequently, BFS is inducing a bipartition by cutting 50% more edges.

Table 1: Overview over the features in the initial node embeddings in the line graph $\mathcal{L}(\mathcal{G})$. Features are extracted from different domains and are extracted at different steps of the line graph generation. For details on the features that are generated in the segmentation-to-graph step refer to the work by Drees et al. [DSH+21].

Feature	**Source Domain**	**Generation Step**	**Feature Type**
Length	Segmentation	Graph Gen.	Geom.
Distance	Segmentation	Graph Gen.	Geom.
Curveness	Segmentation	Graph Gen.	Geom.
Mean Min. CL Surf. Dist.	Segmentation	Graph Gen.	Geom.
Std. Min. CL Surf. Dist.	Segmentation	Graph Gen.	Geom.
Mean Max. CL Surf. Dist.	Segmentation	Graph Gen.	Geom.
Std. Max. CL Surf. Dist.	Segmentation	Graph Gen.	Geom.
Mean Mean CL Surf. Dist.	Segmentation	Graph Gen.	Geom.
Std. Mean CL Surf. Dist.	Segmentation	Graph Gen.	Geom.
Mean Roundness CL Surf. Dist.	Segmentation	Graph Gen.	Geom.
Std. Roundness CL Surf. Dist.	Segmentation	Graph Gen.	Geom.
X-Orientation	Spatial Graph $\mathcal{G}$	Handcrafted	Geom.
Y-Orientation	Spatial Graph $\mathcal{G}$	Handcrafted	Geom.
Z-Orientation	Spatial Graph $\mathcal{G}$	Handcrafted	Geom.
Degree (Line Graph)	Line Graph $\mathcal{L}(\mathcal{G})$	Handcrafted	Struct.
G_2 Graphlet Count (Triangles)	Line Graph $\mathcal{L}(\mathcal{G})$	Handcrafted	Struct.
G_5 Graphlet Count (Squares)	Line Graph $\mathcal{L}(\mathcal{G})$	Handcrafted	Struct.
Betweenness Centrality	Line Graph $\mathcal{L}(\mathcal{G})$	Handcrafted	Struct.
Closeness Centrality	Line Graph $\mathcal{L}(\mathcal{G})$	Handcrafted	Struct.
Mean CL Voxel Value	Raw Data	Handcrafted	Image
Min CL Voxel Value	Raw Data	Handcrafted	Image
Max CL Voxel Value	Raw Data	Handcrafted	Image
Std CL Voxel Value	Raw Data	Handcrafted	Image
Median CL Voxel Value	Raw Data	Handcrafted	Image
25% Quantile CL Voxel Value	Raw Data	Handcrafted	Image
75% Quantile CL Voxel Value	Raw Data	Handcrafted	Image

Table 2: Evaluation of the SVM and RF model with the optimal hyperparameter setting. SVM setting with features from max combined raw data: (Reg. Param. C: 1 Kernel: RBF), mean combined raw data: (Reg. Param. C: 1, Kernel: RBF) and without CL-Statistics: (Reg. Param. C: 1, Kernel: RBF). RF setting with features from max combined raw data: (Trees: 200, Depth: 20, Features per Split: $\sqrt{p}$), mean combined raw data: (Trees: 200, Depth: 18, Features per Split: $\sqrt{p}$) and without CL-Statistics: (Trees: 100, Depth: 20, Features per Split: $\sqrt{p}$). F_1 - N and F_1 - L denote the F_1 scores for the nerve and lymph classification.

Dataset		**Performance Metrics**							
Graph	**Raw Data**	**Acc.**	**Bal. Acc.**	**F_1 - N**	**F_1 - L**	**Prec. - N**	**Prec. - L**	**Rec. - N**	**Rec. - L**
Support Vector Machine CL-Statistics									
Validation	Max Comb	0.6325	0.6277	0.5682	0.6801	0.6528	0.6205	0.5029	0.7524
Graph	Mean Comb	0.6590	0.6556	0.6151	0.6939	0.6724	0.6499	0.5668	0.7444
Test	Max Comb	0.6990	0.6727	0.5857	0.7637	0.6994	0.6989	0.5038	0.8417
Graph	Mean Comb	0.7427	0.7152	0.6386	0.8002	0.7847	0.7256	0.5384	0.8920
Support Vector Machine No CL-Statistics									
Validation Graph	Not used	0.5779	0.5693	0.4405	0.6612	0.6072	0.5669	0.3457	0.7930
Test Graph	Not used	0.6486	0.6108	0.4691	0.7374	0.6479	0.6489	0.3677	0.8540
$\mathcal{L}_{22-05}$	Not used	0.7418	0.6730	0.8207	0.5389	0.7290	0.7969	0.9389	0.4071
Random Forest CL-Statistics									
Validation	Max Comb	0.6494	0.6464	0.6094	0.6820	0.6561	0.6446	0.5690	0.7239
Graph	Mean Comb	0.6718	0.6690	0.6362	0.7011	0.6810	0.6651	0.5969	0.7412
Test	Max Comb	0.7239	0.7058	0.6432	0.7748	0.7078	0.7326	0.5895	0.8222
Graph	Mean Comb	0.7541	0.7284	0.6592	0.8077	0.7948	0.7368	0.5632	0.8937
Random Forest No CL-Statistics									
Validation Graph	Not used	0.6000	0.5936	0.5071	0.6635	0.6221	0.5891	0.4280	0.7593
Test Graph	Not used	0.6638	0.6343	0.5277	0.7389	0.6484	0.6700	0.4449	0.8237
$\mathcal{L}_{22-05}$	Not used	0.7467	0.7179	0.8047	0.6396	0.7816	0.6764	0.8291	0.6066

Table 3: Performance of the SAGE GNN models with different feature settings. In some experiments, the centerline intensities were used for centerline statistics and by a learnable feature extractor (LSTM) while other models are completely unaware of the raw intensity information. F_1 - N and F_1 - L denote the F_1 scores for the nerve and lymph classification.

Dataset		**Performance Metrics**							
Graph	**Raw Data**	**Acc.**	**Bal. Acc.**	**F_1 - N**	**F_1 - L**	**Prec. - N**	**Prec. - L**	**Rec. - N**	**Rec. - L**
SAGE No CL-Statistics									
Validation Graph	Not Used	0.6707	0.6728	0.6805	0.6599	0.6374	0.7111	0.7299	0.6156
Test Graph	Not Used	0.7185	0.7175	0.6807	0.7483	0.6533	0.7740	0.7106	0.7243
$\mathcal{L}_{22-05}$	Not Used	0.7699	0.7586	0.8144	0.6972	0.8270	0.6804	0.8023	0.7149
SAGE CL-Statistics									
Validation Graph	Max Comb	0.7007	0.7021	0.7039	0.6974	0.6711	0.7340	0.7401	0.6642
	Mean Comb	0.7240	0.7251	0.7239	0.7241	0.6973	0.7528	0.7526	0.6976
Test Graph	Max Comb	0.7740	0.7678	0.7311	0.8051	0.7345	0.8024	0.7277	0.8078
	Mean Comb	0.8079	0.7993	0.7658	0.8372	0.7892	0.8203	0.7438	0.8548
$\mathcal{L}_{22-05}$	Max Comb	0.7435	0.6924	0.8137	0.5885	0.7495	0.7258	0.8899	0.4949
SAGE LSTM									
Validation Graph	Max Comb	0.7026	0.7045	0.7095	0.6952	0.6687	0.7429	0.7557	0.6534
	Mean Comb	0.7210	0.7224	0.7231	0.7189	0.6915	0.7539	0.7577	0.6870
Test Graph	Max Comb	0.7746	0.7687	0.7325	0.8052	0.7341	0.8040	0.7310	0.8065
	Mean Comb	0.8094	0.7987	0.7639	0.8402	0.8010	0.8147	0.7300	0.8675

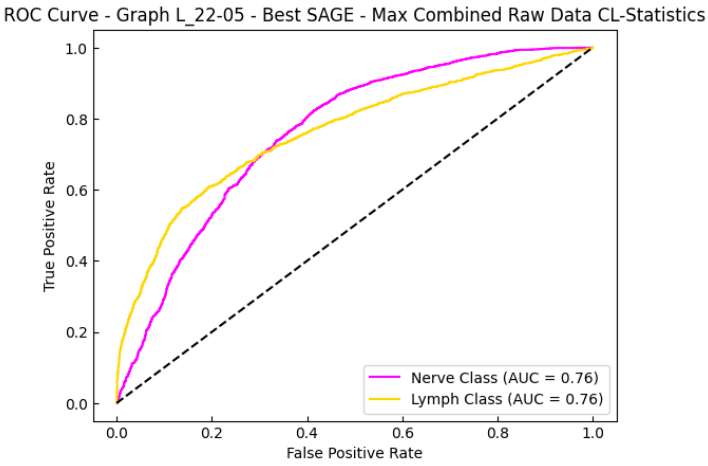

Figure 2: ROC Curve on the graph $\mathcal{L}_{22-05}$ for the best SAGE classifier that used the max combined raw data in the training stage.

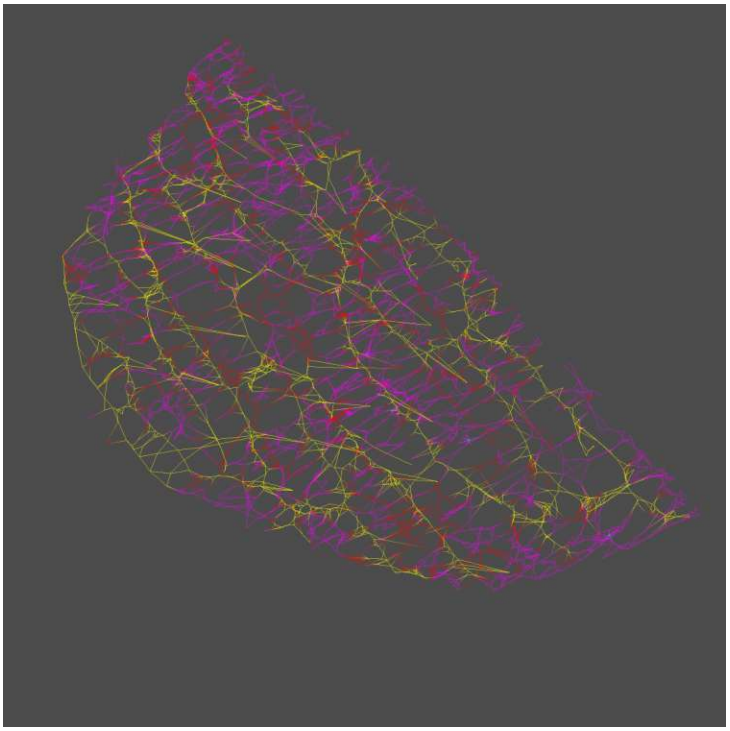

Figure 3: Visualization of the misclassified nodes on the graph $\mathcal{L}_{22-05}$. Magenta indicates correctly classified nerve nodes, yellow indicates correctly classified lymph nodes and red indicates false classifications. The predictions are made by the best-evaluated SAGE model without the use of centerline statistics.

Table 4: Performance of different GNN layer types in the simplest feature configuration (no raw data information in the form of centerline statistics). F_1 - N and F_1 - L denote the F_1 scores for the nerve and lymph classification.

Dataset		**Performance Metrics**							
Graph	**Raw Data**	**Acc.**	**Bal. Acc.**	**F_1 - N**	**F_1 - L**	**Prec. - N**	**Prec. - L**	**Rec. - N**	**Rec. - L**
GCN No CL-Statistics									
Validation Graph	Not Used	0.6548	0.6589	0.6813	0.6234	0.6124	0.7189	0.7676	0.5503
Test Graph	Not Used	0.7013	0.7080	0.6799	0.7200	0.6209	0.7853	0.7512	0.6648
GAT No CL-Statistics									
Validation Graph	Not Used	0.6043	0.6022	0.5711	0.6327	0.5962	0.6107	0.5481	0.6563
Test Graph	Not Used	0.6752	0.6607	0.5961	0.7284	0.6276	0.7046	0.5676	0.7539
EdgeConv No CL-Statistics									
Validation Graph	Not Used	0.6515	0.6555	0.6766	0.6222	0.6108	0.7118	0.7584	0.5526
Test Graph	Not Used	0.7044	0.7090	0.6785	0.7264	0.6273	0.7806	0.7388	0.6792

Table 5: Connectivity analysis with the Jaccard index using connected components. For the analysis, the induced subgraph from all the predicted nerve nodes and the induced subgraph from all the predicted lymph nodes are used.

$\mathcal{L}_{22-05}$ Connect. Analysis	SAGE No CL		RF No CL	
Con. Comps. k	Jaccard@k Nerve Subgraph	Jaccard@k Lymph Subgraph	Jaccard@k Nerve Subgraph	Jaccard@k Lymph Subgraph
1	0.3080	0.4821	0.2608	0.0887
2	0.5195	0.5267	0.4327	0.1775
3	0.5364	0.5405	0.4897	0.2429
4	0.5554	0.5391	0.5122	0.2838
5	0.5756	0.5382	0.5340	0.3087
6	0.5655	0.5375	0.5318	0.3404
7	0.5810	0.5388	0.5515	0.3646
8	0.5932	0.5397	0.5675	0.3874
9	0.6051	0.5401	0.5585	0.4076
10	0.6011	0.5405	0.5652	0.4214

List of Figures

List of Tables

Acronyms

AMD Age-Related Macular degeneration. 48

AUC Area Under the Curve. 31

BCE Binary Cross Entropy. 31

BFS Breadth First Search. 25, 26, 49, 57

CNN Convolutional Neural Network. 6, 8, 34

CNS Central Nervous System. 2, 11

CT Computer Tomography. 12

CV Computer Vision. 1

CV Cross-Validation. 38

DFS Depth First Search. 25

DGCNN Dynamic Graph Convolutional Neural Network. 9, 31, 37, 45

DL Deep Learning. 1, 2, 6

DR Diabetic Retinopathy. 48

ENS Enteric Nervous System. 11

GAT Graph Attention Network. 31, 34, 42

GCN Graph Convolutional Network. 6–8, 31, 34, 41–43, 57

GDV Graphlet Degree Vector. 22

GNN Graph Neural Network. 1–3, 6, 8, 9, 21–23, 25, 30, 31, 34–38, 40, 44, 45, 47, 48, 52, 54, 57, 59, 60

IBD Inflammatory Bowel Disease. 47

LSFM Light Sheet Fluorescence Microscopy. 2, 3, 5, 12, 15, 47

LSTM Long Short-Term Memory. 8, 34, 36, 37, 45, 52, 59, 60

LYVE1 Lymphatic Vessel Endothelial Hyaluronan Receptor 1. 11

MAD Median Absolute Deviation. 13, 18

ML Machine Learning. 21, 22, 25, 38–40, 56

MLP Multilayer Perceptron. 9, 34

NLP Natural Language Processing. 1

OCTA Optical Coherence Tomography Angiography. 48

PNS Peripheral Nervous System. 2, 3, 11

RBF Radial Basis Function. 39, 40, 51, 59, 60

RF Random Forest. 38, 40–42, 51, 59, 60

RNN Recurrent Neural Network. 36, 57

ROC Receiver Operating Characteristic. 31

SVM Support Vector Machine. 39, 40, 51, 59, 60

TH Tyrosine Hydroxylase. 11

Bibliography

[AA03] Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003.

[BBL+17] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[Bie20] Lukas Biewald. Experiment tracking with weights and biases, 2020. *Software available from wandb. com*, 2(5), 2020.

[BMS+16] Aydın Buluç, Henning Meyerhenke, Ilya Safro, Peter Sanders, and Christian Schulz. *Recent Advances in Graph Partitioning*. Springer, 2016.

[CCB+20] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. *Advances in Neural Information Processing Systems*, 33:13260–13271, 2020.

[CD13] Kwanghun Chung and Karl Deisseroth. Clarity for mapping the nervous system. *Nature Methods*, 10(6):508–513, 2013.

[CE19] Vincenza Cifarelli and Anne Eichmann. The intestinal lymphatic system: functions and metabolic implications. *Cellular and Molecular Gastroenterology and Hepatology*, 7(3):503–513, 2019.

[Che15] Jeff Cheeger. A lower bound for the smallest eigenvalue of the laplacian. In *Problems in Analysis*, pages 195–200. Princeton University Press, 2015.

[Chu97] Fan RK Chung. *Spectral Graph Theory*, volume 92. American Mathematical Soc., 1997.

[CLS+19] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 257–266, 2019.

[DH73] William E Donath and Alan J Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5):420–425, 1973.

[DSH+21] Dominik Drees, Aaron Scherzinger, René Hägerling, Friedemann Kiefer, and Xiaoyi Jiang. Scalable robust graph and feature extraction for arbitrary vessel networks in large volumetric datasets. *BMC Bioinformatics*, 22(1):1–28, 2021.

[EBJ+12] Ali Ertürk, Klaus Becker, Nina Jährling, Christoph P Mauch, Caroline D Hojer, Jackson G Egen, Farida Hellal, Frank Bradke, Morgan Sheng, and Hans-Ulrich Dodt. Three-dimensional imaging of solvent-cleared organs using 3disco. *Nature Protocols*, 7(11):1983–1995, 2012.

[Fie75] Miroslav Fiedler. A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal*, 25(4):619–633, 1975.

[GL16] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864, 2016.

[HBJ+20] Songtao He, Favyen Bastani, Satvat Jagwani, Mohammad Alizadeh, Hari Balakrishnan, Sanjay Chawla, Mohamed M Elshrif, Samuel Madden, and Mohammad Amin Sadeghi. Sat2graph: Road graph extraction through graph-tensor encoding. In *European Conference on Computer Vision*, pages 51–67. Springer, 2020.

[HNT+22] Ali Hatamizadeh, Vishwesh Nath, Yucheng Tang, Dong Yang, Holger R Roth, and Daguang Xu. Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 7th International Workshop, BrainLes 2021, Held in Conjunction with MICCAI 2021, Virtual Event, September 27, 2021, Revised Selected Papers, Part I*, pages 272–284. Springer, 2022.

[HNW65] Frank Harary and C St JA Nash-Williams. On eulerian and hamiltonian graphs and line graphs. *Canadian Mathematical Bulletin*, 8(6):701–709, 1965.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[HTN+22] Ali Hatamizadeh, Yucheng Tang, Vishwesh Nath, Dong Yang, Andriy Myronenko, Bennett Landman, Holger R Roth, and Daguang Xu. Unetr: Transformers for 3d medical image segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 574–584, 2022.

[HVG11] David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

[HYL17] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, 2017.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[JKS+15] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[JSS+15] Wiebke Jahr, Benjamin Schmid, Christopher Schmied, Florian O Fahrbach, and Jan Huisken. Hyperspectral light sheet microscopy. *Nature Communications*, 6(1):7990, 2015.

[KB15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

[KW16] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LF06] Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 631–636, 2006.

[LH17] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

[LHW18] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[LLYA14] John Lim, Hwee Kuan Lee, Weimiao Yu, and Sohail Ahmed. Light sheet fluorescence microscopy (lsfm): past, present and future. *Analyst*, 139(19):4758–4768, 2014.

[MHL+23] Martin J Menten, Robbie Holland, Oliver Leingang, Hrvoje Bogunovic, Ahmed M Hagag, Rebecca Kaye, Sophie Riedl, Ghislaine L Traber, Osama N Hassan, Nick Pawlowski, et al. Exploring healthy retinal aging with deep learning. *Ophthalmology Science*, page 100294, 2023.

[MLH+23] Hongcheng Mai, Jie Luo, Luciano Hoeher, Rami Al Maskari, Izabela Horvath, Johannes Paetzold, Mihail Todorov, Farida Hellal, and Ali Erturk. Whole mouse body histology using standard igg antibodies. *bioRxiv*, pages 2023–02, 2023.

[MSRMH09] Jennis Meyer-Spradow, Timo Ropinski, Jörg Mensmann, and Klaus Hinrichs. Voreen: A rapid-prototyping environment for ray-casting-based volume visualizations. *IEEE Computer Graphics and Applications*, 29(6):6–13, 2009.

[PARS14] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 701–710, 2014.

[PCQ+16] Chenchen Pan, Ruiyao Cai, Francesca Paola Quacquarelli, Alireza Ghasemigharagoz, Athanasios Lourbopoulos, Paweł Matryba, Nikolaus Plesnila, Martin Dichgans, Farida Hellal, and Ali Ertürk. Shrinkage-mediated imaging of entire organs and organisms using udisco. *Nature Methods*, 13(10):859–867, 2016.

[PMS+21] Johannes C Paetzold, Julian McGinnis, Suprosanna Shit, Ivan Ezhov, Paul Büschl, Chinmay Prabhakar, Anjany Sekuboyina, Mihail Todorov, Georgios Kaissis, Ali Ertürk, et al. Whole brain vessel graphs: A dataset and benchmark for graph learning and neuroscience. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[PSAM+19] Johannes C Paetzold, Oliver Schoppe, Rami Al-Maskari, Giles Tetteh, Velizar Efremov, Mihail I Todorov, Ruiyao Cai, Hongcheng Mai, Zhouyi Rong, Ali Ertuerk, et al. Transfer learning from synthetic data reduces need for labels to segment brain vasculature and neural pathways in 3d. 2019.

[PSP+] Chinmay Prabhakar, Suprosanna Shit, Johannes C Paetzold, Ivan Ezhov, Rajat Koner, Hongwei Li, Florian Sebastian Kofler, et al. Vesselformer: Towards complete 3d vessel graph generation from images. In *Medical Imaging with Deep Learning*.

[QSMG17] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings*

of the IEEE Conference on Computer Vision and Pattern Recognition, pages 652–660, 2017.

[RDBD+11] J-F Rahier, S De Beauce, L Dubuquoy, E Erdual, J-F Colombel, Anne Jouret-Mourin, K Geboes, and P Desreumaux. Increased lymphatic vessel density and lymphangiogenesis in inflammatory bowel disease. *Alimentary Pharmacology & Therapeutics*, 34(5):533–543, 2011.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[RGM+21] Douglas S Richardson, Webster Guan, Katsuhiko Matsumoto, Chenchen Pan, Kwanghun Chung, Ali Ertürk, Hiroki R Ueda, and Jeff W Lichtman. Tissue clearing. *Nature Reviews Methods Primers*, 1(1):84, 2021.

[RL15] Douglas S Richardson and Jeff W Lichtman. Clarifying tissue clearing. *Cell*, 162(2):246–257, 2015.

[RSR+18] Sonia Rehal, Matthew Stephens, Simon Roizes, Shan Liao, and Pierre-Yves von der Weid. Acute small intestinal inflammation results in persistent lymphatic alterations. *American Journal of Physiology-Gastrointestinal and Liver Physiology*, 314(3):G408–G417, 2018.

[RWS+14] Nicolas Renier, Zhuhao Wu, David J Simon, Jing Yang, Pablo Ariel, and Marc Tessier-Lavigne. idisco: a simple, rapid method to immunolabel large tissue samples for volume imaging. *Cell*, 159(4):896–910, 2014.

[SHK+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[SKW+22] Suprosanna Shit, Rajat Koner, Bastian Wittmann, Johannes Paetzold, Ivan Ezhov, Hongwei Li, Jiazhen Pan, Sahand Sharifzadeh, Georgios Kaissis, Volker Tresp, et al. Relationformer: A unified framework for image-to-graph generation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVII*, pages 422–439. Springer, 2022.

[SPS+21] Suprosanna Shit, Johannes C Paetzold, Anjany Sekuboyina, Ivan Ezhov, Alexander Unger, Andrey Zhylka, Josien PW Pluim, Ulrich Bauer, and Bjoern H Menze. cldice-a novel topology-preserving loss function for tubular structure segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16560–16569, 2021.

[STP+14] Etsuo A Susaki, Kazuki Tainaka, Dimitri Perrin, Fumiaki Kishino, Takehiro Tawara, Tomonobu M Watanabe, Chihiro Yokoyama, Hirotaka Onoe, Megumi Eguchi, Shun Yamaguchi, et al. Whole-brain imaging with single-cell resolution using chemical cocktails and computational analysis. *Cell*, 157(3):726–739, 2014.

[Tau95] Gabriel Taubin. A signal processing approach to fair surface design. In *Proceedings of the 22nd Annual Conference on Computer graphics and Interactive Techniques*, pages 351–358, 1995.

[TPS+20] Mihail Ivilinov Todorov, Johannes Christian Paetzold, Oliver Schoppe, Giles Tetteh, Suprosanna Shit, Velizar Efremov, Katalin Todorov-Völgyi, Marco Düring, Martin Dichgans, Marie Piraud, et al. Machine learning analysis of whole mouse brain vasculature. *Nature Methods*, 17(4):442–449, 2020.

[TQW+15] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077, 2015.

[UEC+20] Hiroki R Ueda, Ali Ertürk, Kwanghun Chung, Viviana Gradinaru, Alain Chédotal, Pavel Tomancak, and Philipp J Keller. Tissue clearing and its applications in neuroscience. *Nature Reviews Neuroscience*, 21(2):61–79, 2020.

[VCC+17] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *6th International Conference on Learning Representations*, 2017.

[WPC+20] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2020.

[WPP+23] Bastian Wittmann, Johannes C Paetzold, Chinmay Prabhakar, Daniel Rueckert, and Bjoern Menze. Link prediction for flow-driven spatial networks. *arXiv preprint arXiv:2303.14501*, 2023.

[WSL+19] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[XHLJ18] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[YBY+19] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[YGSYL21] Jiaxuan You, Jonathan M Gomes-Selman, Rex Ying, and Jure Leskovec. Identity-aware graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10737–10745, 2021.

[YYL20] Jiaxuan You, Zhitao Ying, and Jure Leskovec. Design space for graph neural networks. *Advances in Neural Information Processing Systems*, 33:17009–17021, 2020.

[ZC18] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. *Advances in Neural Information Processing Systems*, 31, 2018.

[ZCH+20] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81, 2020.

[ZLLT21] Xiao-Meng Zhang, Li Liang, Lin Liu, and Ming-Jing Tang. Graph neural networks and their current applications in bioinformatics. *Frontiers in Genetics*, 12:690049, 2021.