Modeling & Analyzing Structural Brain Connectomes

Shamanth Kuthpadi
Advisor: Cameron Musco
University of Massachusetts Amherst

Abstract
This independent study aimed to explore the intersection of machine learning (ML) models, structural brain connectomes, and spectral graph theory. A significant challenge in this study was obtaining access to raw brain data. Many institutions with rights to the desired datasets either denied our requests or required specialized permissions from the National Institute of Mental Health Data Archive (NDA). To overcome this limitation, we utilized pre-processed connectomic data kindly provided by Kerepesi et al. Using this data, we visualized and analyzed brain connectomes, ultimately developing a classifier capable of mapping node-level features to specific brain regions. A binary classifier achieved 94% accuracy for hemisphere classification, while a multi-class model reached 50% accuracy for cortical structure identification. We also demonstrated that the Fiedler vector alone can classify hemispheres with an accuracy exceeding 99%. This study highlights the utility of spectral graph theory in understanding brain structure and offers avenues for future research using generative models and additional datasets.

1 Introduction

1.1 Motivation

Understanding the human brain's connectivity is crucial for diagnosing neurodevelopmental disorders, brain injuries, and neurological diseases. Structural brain connectomes represent the physical pathways of the brain and can reveal patterns of organization that correlate with cognitive function or dysfunction. Advances in spectral graph theory and machine learning have allowed for significant progress in understanding these networks. This study leverages graph theoretic techniques and ML to analyze connectomes and build predictive models that classify brain regions based on connectivity features.

1.2 Objective

In this study, our main objective was to understand how well we can distinguish between brain regions using spectral features of the Laplacian of a connectome network. For now, we have utilized the eigenvectors of the Laplacian to classify brain hemispheres and major cortical structures. However, we hope that leveraging spectral features could allow us to identify even more local brain structures/regions that can then be used to annotate brain-induced graph networks. Tools like FreeSurfer (used by Kerepesi et al.) use probabilistic approaches that heavily rely on pre-defined templates to label brain regions. However, it has been shown that machine learning (i.e. FastSurfer via convolutional neural network [2]) approaches can be much faster and just as reliable in labeling the same regions. Our work aims to propose an alternative to FreeSurfer's segmentation procedure.

1.3 Previous Work

Machine learning approaches to the analysis of connectomic data have been successful in highlighting connectomic differences between brains affected by neurodevelopmental disorders and Tardive dyskinesia (TD) brains. Previous work in this area has shown that patients with a variety of neurodevelopmental disorders are alike in that their connectomes lack the presence of highly interconnected "hubs" that are easily found in TD brains [6]. These positive and interesting insights indicate that machine learning methods can elucidate connectomic correlates of brain functions and dysfunction not known previously, such as these hubs.

Abdelnour et al. demonstrated a novel approach to predict functional brain connectivity (FC) from structural connectivity (SC) using the eigenstructure of the structural Laplacian. They derived an analytical model showing that SC and FC share eigenvectors, with their eigenvalues exhibiting an exponential relationship. This model efficiently reconstructs FC matrices using only a small number of Laplacian eigenmodes, outperforming traditional generative models that rely on time-consuming neural simulations [1]. Their findings highlight the utility of spectral graph theory in understanding the structure-function relationship in the brain and provide a mechanistic framework for deriving FC from SC using closed-form solutions.

1.4 Data

We first attempted to utilize raw brain data but there were two main issues that we faced:

1. We needed to formulate a pipeline to convert raw brain diffusion-weighted images to useable connectome networks. To do so, we turned to MRtrix3. However, MRtrix3 requires highly specific imaging conditions, including a consistent acquisition protocol for diffusion MRI sequences, which can vary across datasets. Furthermore, the pipeline for converting raw DWI data into usable connectome networks relies on accurate brain masking, fiber tractography, and parcellation steps. Each of these steps introduces potential sources of error, such as noise in the diffusion data, inaccuracies in identifying fiber pathways, or inconsistencies in parcellating brain regions.
2. While there were example raw brain images, we were not able to obtain access to full raw brain data. Many of the institutions that reserved the right to the datasets either outright denied our requests or required special permissions.

Instead, we turned to work done by the organizers of BrainGraph.org. Kerepesi et al. constructed structural connectomes that were processed from the Human Connectome Project (HCP) - one of the datasets that we hoped to gain access to. These connectomes were then formatted as GraphML files that included nodes, edges, and their attributes [3]. On BrainGraph.org there are many different versions but the one used for this study is called *1015 nodes set, 1064 brains, 1,000,000 streamlines, 10x repeated & averaged.*

After downloading the dataset, we then proceeded to read in the GraphMl files as NumPy/SciPy matrices and NetworkX graphs. The goal here was to create useable formats of the connectomes for further analysis in Python.

1.5 Visualization

Each edge contained a *FA_mean* attribute that represented the fractional anisotropy mean between two nodes. We considered the *FA_mean* to be the edge weights in the graph network.

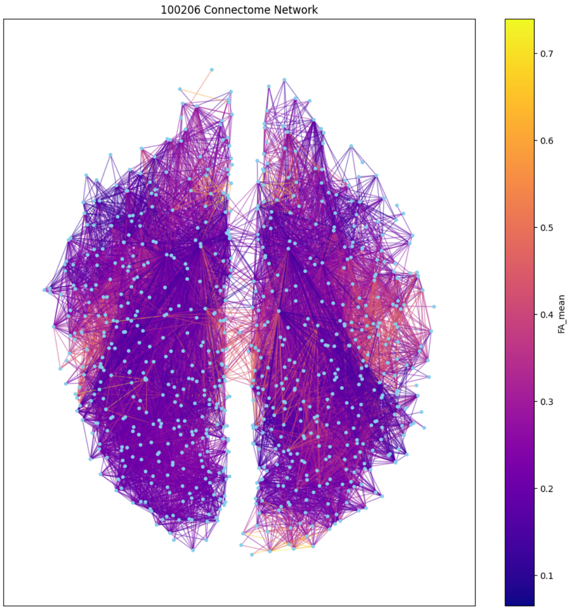

Figure 1: Example of a graph network for a given structural connectome. Nodes are blue and the edges are on a color scale mapping the *FA_mean*.

2 Methodology

2.1 Preprocessing

Firstly, we did not want isolated nodes (nodes with degree = 0) to impact or bias our downstream analysis. Hence, those nodes were removed from the graph networks.

We then gathered all of the structural connectomes and created a dataset with each row containing the features (centrality and spectral) for a given node. In other words, we extracted all the nodes and their features for a given structural connectome - we then did this for all the structural connectomes.

Finally, we created a binary classifier that maps nodes to the hemisphere within the brain structure. We also created a multi-class classifier that maps nodes to specific brain regions within the brain cortex.

2.2 Centrality Features

In a network setting, such as ours, centrality measures are commonly extracted for nodes within the graph. In particular, we chose to use:

- **Degree Centrality.** This is the most basic centrality measure and the easiest to calculate. As the name suggests, for a given node, the degree centrality is simply the degree of that node. The degree of a node is the number of edges that are incident to that node. For a directed graph, this measure is typically split into in-degree centrality and out-degree centrality where in-degree is the number of edges that are incoming and out-degree is the number of edges that are outgoing.

So, for an undirected graph, the degree centrality for a node i is:

$$d_i = \sum_{i=1}^{N} a_{ij}$$

where A is the adjacency matrix with a_{ij} being an element in the matrix corresponding to the relation between the ith and jth node [5].

- **Closeness Centrality.** The closeness centrality of a node i measures how "close" the node is to all other nodes in the graph. In particular, it is:

  $$c_i = \frac{N-1}{\sum_{j \neq i} d(i,j)}$$

  where N is the number of nodes and j is an arbitrary node (that is not the same as i) in our graph. Intuitively, nodes with higher closeness centrality measures are closer to the "center" of the network.

- **Betweeness Centrality.** Betweenness centrality is used as a measure to quantify how much control a node has during network flow. In other words, we want to check the number of times a node i falls in the path of an arbitrary node j and k:

  $$b_i = \frac{2}{(N-1)(N-2)} \sum_{i \neq j, i \neq k, j \neq k} \frac{\sigma_{jk}(i)}{\sigma_{jk}}$$

  where N is the number of nodes in the graph, σ_{jk} is the number of paths from j to k, and $\sigma_{jk}(i)$ is the number of times that i falls in their paths [5].

- **Eigenvector Centrality.** Let v be the eigenvector corresponding to the largest eigenvalue of the adjacency matrix [5]:

  $$v = \frac{1}{\lambda} Av$$

  wherein which each value of v can be represented as:

  $$v_i = \frac{1}{\lambda} \sum_{i=1}^{N} a_{ij} v_j$$

  Influence propagation is a good way to gain some intuition behind why eigenvector centrality can be a powerful measure. A node that has a high eigenvector centrality is one that has influential neighbors. "Influence" refers to a node's own centrality measure (typically this is degree centrality). In other words, a node could have the lowest degree centrality but the highest eigenvector centrality if its neighbors have high degree centrality. You can imagine that this process of analyzing the influence of a node's neighbor can recurse or propagate to neighbors of neighbors and so on.

So, for each node, we calculated the aforementioned centrality measures using NetworkX functionalities.

2.3 Spectral Features

For each of the structural connectomes, we found the corresponding un-weighted Laplacian matrix. Un-weighted in this case just refers to the fact that the adjacency matrix A has binary values indicating the existence of an edge between two nodes. The Laplacian matrix can be represented as

$$L = D - A$$

where L is our Laplacian matrix, D is the degree matrix, and A is our adjacency matrix. In the context of the Laplacian matrix, the eigenvectors corresponding to the smallest non-zero eigenvalues are informative. Hence, we chose the smallest k eigenvectors $v_{n-1}, ..., v_{n-k}$ of L and then represented each node by it's corresponding row in $V \in R^{n \times k}$ [4].

To find the Laplacian matrix and the eigenspectra, we took advantage of functionalities within SciPy.

To ensure that the eigenvectors would indeed be useful in downstream analysis, we decided to perform a few key tests (as seen in Figure 2 and Figure 3). The Fiedler vector has a clear transition zone in the middle indicating that it can be used to clearly cluster the nodes in the network. To confirm this, we also plotted the clusters that are formed by simply using the Fiedler vector on the average graph. We found that, just by visual analysis, it does a fantastic job of separating the left and right hemispheres. Consequently, we decided to utilize the spectral features during classification.

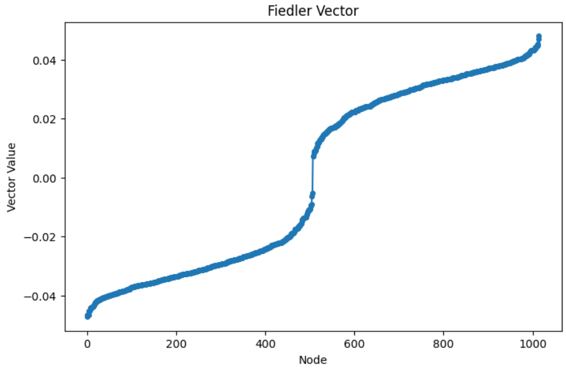

Figure 2: This is the plot of a Fiedler vector (second smallest eigenvector) in sorted order. There is a clear transition zone in the middle indicating that there is a clear clustering of nodes.

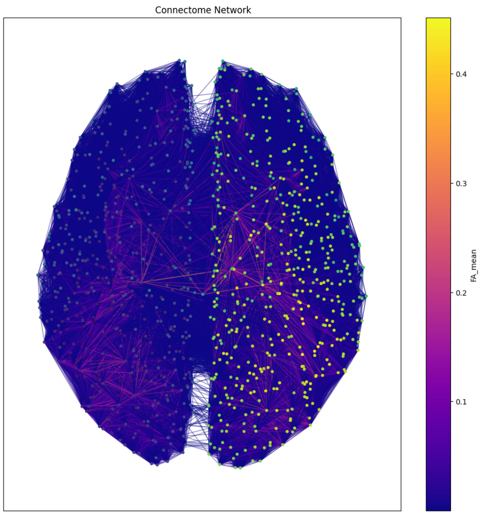

Figure 3: Each node is colored based on its respective value in the Fiedler vector. It is clear to see that the Fiedler vector is extremely informative when it comes to separating the brain into the left and right hemispheres

2.4 Model Selection

We made the empirical choice to use extreme gradient-boosted trees as our underlying model. Boosting is an excellent choice when minimizing bias and variance which can then lead to highly accurate models. They also can provide insights into the importance of individual features and this can be particularly useful when trying to gain intuition about the effectiveness of feature choices.

The brain regions (labels) were encoded using Scikit-learn's label encoder. The dataset was then split (stratified) into a training set (80%) and a testing set (20%). To utilize Extreme Gradient Boosting, we used the XGBClassifier library from the XGBoost Python package. The hyperparameters for the model were selected using stratified k-fold cross-validation in conjunction with a randomized search.

3 Results

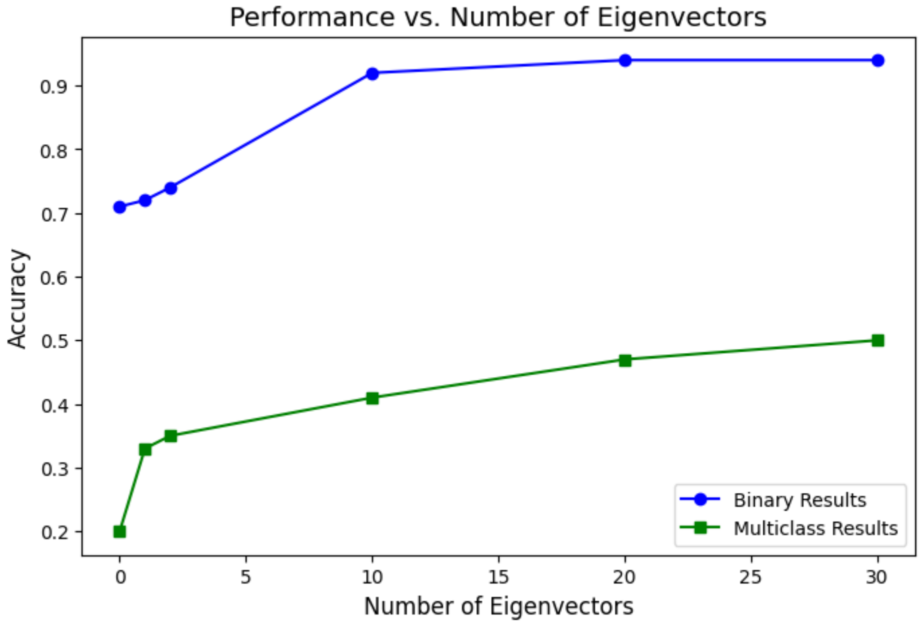

Figure 4: The above are the performance results for the binary classifier and multi-class classifier for different numbers of eigenvectors.

The best binary classifier has an outstanding accuracy of 94%. The Laplacian eigenspectrum seemed to boost performance by a significant amount and we were able to validate this by analyzing the feature importance F-scores. On the other hand, the best multi-class classifier only has an accuracy of 50%. Inherently, the task is harder as there are many more classes (49 classes to be exact) to classify and so spectrum-identified clusters need to also be very precise.

3.1 Discussion

From the plot, we can observe several key characteristics:

- For the binary classifier, beyond 10 eigenvectors there wasn't much change in accuracy. Whereas, for the multi-class classifier, there seems to be an increasing trend in performance as we increase the number of eigenvectors in the feature set past 20. This is probably due to the fact that lower eigenvectors (those corresponding to smaller eigenvalues) typically capture global structures, such as clusters or communities in the graph and higher eigenvectors capture finer, localized details in the graph. Hence, when detecting hemispheres we don't need as many eigenvectors but for specific brain regions, we will need much higher eigenvectors.

- The importance of centrality measures fades away as the eigenvectors are introduced in the feature set. Maybe this is because centrality measures encode a node's structural importance as a scalar but including feature space from the eigenspectra is a richer (because of the multiple dimensions) version.

To confirm if the introduction of more eigenvectors (past 30) would improve the performance of the multi-class model, we included a total of 100 eigenvectors in the feature space and measured the same statistics.

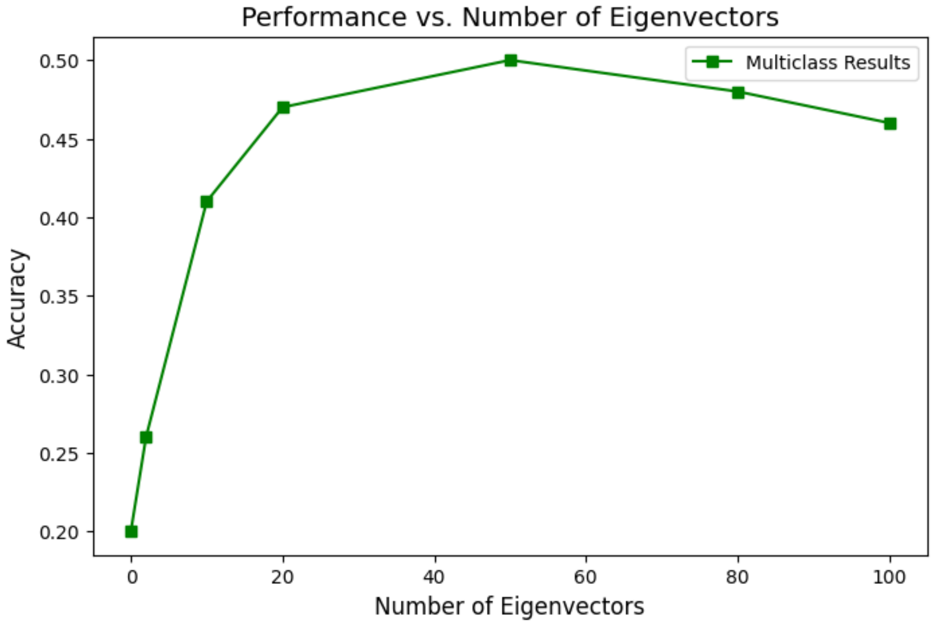

Figure 5: The above are the performance results for the multi-class classifier for different numbers of eigenvectors (up to 100).

We can observe that our multi-class model saturates somewhere between 50 and 70 eigenvectors. This saturation suggests that while the initial eigenvectors capture global structural information, higher-order eigenvectors may introduce redundancy or noise. Beyond a certain point, the additional eigenvectors no longer provide significant new information about the graph structure, as their contribution to distinguishing finer details becomes negligible. This aligns with the known property of Laplacian eigenvectors, where lower eigenvectors describe global patterns and higher ones capture localized nuances that may not always aid classification. As such, the best accuracy remains approximately 50 %.

We also tried to weight the Laplacian and then extract the spectral features but this didn't seem to help either. Weighted in this case just refers to the fact that the adjacency matrix A has values indicating the weight of an edge between two nodes. The edge weights correspond to the fractional anisotropy mean between two nodes and so it could be that this is simply not giving us any useful information about the local structures of the brain.

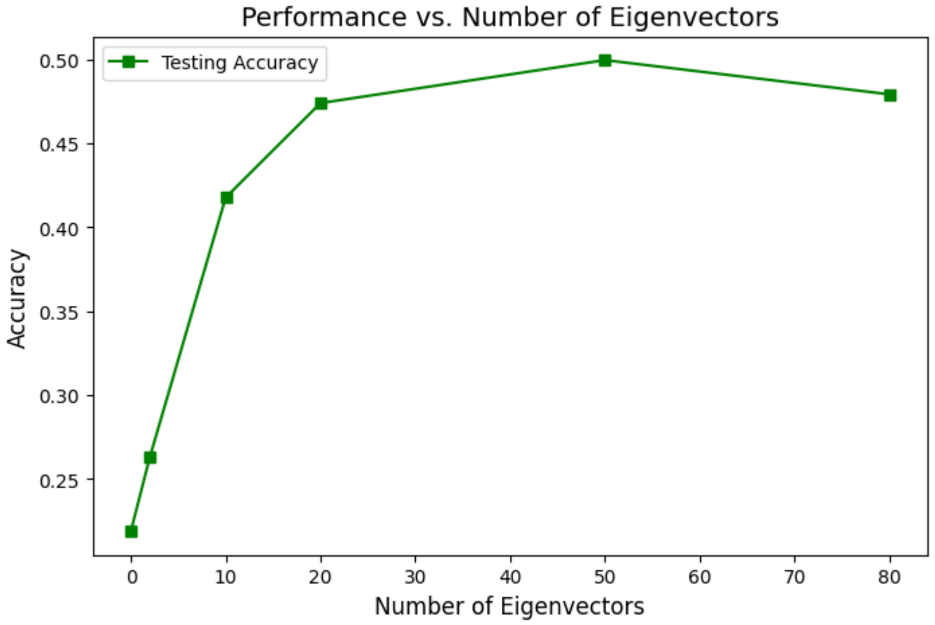

Figure 6: The above are the performance results for the multi-class classifier for different number of eigenvectors of the weighted Laplacian (up to 80).

3.2 Improving the Binary Classifier

The Fiedler vector, corresponding to the second smallest eigenvalue of the Laplacian, naturally divides a graph into two partitions. This property stems from its role in minimizing the normalized graph cut, making it ideal for identifying large-scale structures like brain hemispheres. Since our Fiedler vector seems to be doing a very good job at separating the nodes in the network into the two hemispheres we aim to classify, we decided to just create our own classifier that only uses the Fiedler vector. In particular, we can leverage the fact that the second smallest eigenvector of the Laplacian partitions a network where positive values of the eigenvector correspond to one cluster and negative values correspond to the other cluster.

By simply checking a condition to see if a node has a corresponding negative/positive value in the Fiedler vector we were able to classify nodes with an impressive accuracy of over 99%.

4 Conclusion

Over the course of this independent study, we were able to delve into the realm of machine learning and graph theory in the context of structural brain connectomes. We took existing structural connectomes in the format of GraphML and visualized them as NetworkX networks. Using the newly formatted connectomes, we applied graph theoretic approaches to relate centrality measures to the roles of nodes in a biological setting. We then also performed basic computational operations in the realm of spectral analysis to get the Laplacian matrices, create symmetric matrices, and extract eigenvectors features. Finally, we used machine learning techniques to identify brain regions such as hemispheres and cortical structures.

In regards to our results, the high accuracy of the binary classifier confirms the effectiveness of spectral graph theory for identifying large-scale brain structures. This finding has important implications for clinical and research applications. For instance, accurately identifying brain hemispheres based on connectivity could enhance automated brain mapping in neuroimaging pipelines, reducing manual effort and potential errors. In clinical settings, this accuracy could aid in detecting anomalies or asymmetries in brain structure associated with neurological disorders such as epilepsy, stroke, or neurodevelopmental conditions. On the other hand, all of our strategies to try and improve the accuracy of the multiclass classifier only produced a best accuracy of 94 %.

4.1 Future Work

- Thus far, we have built two classifiers (1) a binary classifier to distinguish between hemispheres (2) a multi-class classifier to distinguish between more localized brain regions within the cortex. While the binary classifier does well, the multi-class classifier does not. We can try to dissect where the issue might be and whether identifying cortical regions is a much harder task than we previously anticipated.
- There are a myriad of generative models, and I would like to train a model so that it can generate a reasonable connectome network given some prior information about the structure. In other words, given sets of nodes and edges pertaining to clusters in a connectome network, can a generative model generate the rest of the network/cluster? Kerepesi et al. propose the *Brain Evolution Workflow*, maybe I could test the validity of this workflow [3].
- There are many more versions of the structural connectome data that BrainGraph.org has made available. It would be interesting to conduct further analyses separately on these versions or even some combination of them.
- There is a lot of work being done to predict functional connectomes from structural connectomes using eigendecomposition and other various spectral methodologies. It would be interesting to see what the limitations of those methods are.

References

[1] Farras Abdelnour et al. "Functional Brain Connectivity Is Predictable from Anatomic Network's Laplacian Eigen-Structure". In: *NeuroImage* 172 (May 2018), pp. 728–739. ISSN: 1053-8119. DOI: `10.1016/j.neuroimage.2018.02.016`. (Visited on 11/21/2024).

[2] Leonie Henschel et al. "FastSurfer - A Fast and Accurate Deep Learning Based Neuroimaging Pipeline". In: *NeuroImage* 219 (Oct. 2020), p. 117012. ISSN: 1053-8119. DOI: `10.1016/j.neuroimage.2020.117012`. (Visited on 12/17/2024).

[3] Csaba Kerepesi et al. "The Braingraph.Org Database of High Resolution Structural Connectomes and the Brain Graph Tools". In: *Cognitive Neurodynamics* 11.5 (Oct. 2017), pp. 483–486. ISSN: 1871-4099. DOI: `10.1007/s11571-017-9445-1`. (Visited on 10/25/2024).

[4] Cameron Musco. *COMPSCI 514 Lecture 18: Algorithms for Data Science*. `https://people.cs.umass.edu/~cmusco/CS514F24/slides/lecture18/lecture18Compressed.pdf`. Accessed: 2024-13-12. 2024.

[5] Salma Salhi et al. "Network Analysis of the Human Structural Connectome Including the Brainstem". In: *PLOS ONE* 18.4 (Apr. 2023), e0272688. ISSN: 1932-6203. DOI: `10.1371/journal.pone.0272688`. (Visited on 10/09/2024).

[6] Roma Siugzdaite et al. "Transdiagnostic brain mapping in developmental disorders". In: *Current Biology* 30.7 (2020), pp. 1245–1257. DOI: `10.1016/j.cub.2020.01.078`. URL: `https://doi.org/10.1016/j.cub.2020.01.078`.