Edge-centric Graph Neural Networks for Brain Connectome Analysis

Dong Jun Yang, John Modl, Mostafa Abdelmegeed, Dr. Minjeong Kim

***Abstract*—Traditional brain network analysis using Graph Neural Networks (GNNs) focuses on node-centric analysis utilizing a correlation matrix. However, we propose a novel approach to brain network analysis that emphasizes the relationships between edges through edge functional connectivity (eFC). This method aims to classify Alzheimer's disease (AD) and Parkinson's disease (PD) more effectively.**

I. Introduction and Background

Our edge-focused Graph Neural Network (GNN) model is designed to capture meaningful data through edge-to-edge relationships dynamically over time. Building on research in the edge-focused graph within the Biological Brain Network domain [1] [2] [3], we have applied these methodologies to our GNN. Unlike traditional node-centric structures [4], which use a correlation matrix to capture static information about a node's relationship with other nodes, our GNN method employs a co-embedding approach. This approach incorporates edge attributes generated through an edge-to-edge relationship matrix, as well as node attributes generated through node-to-node embedding; integrating two matrices with different dimensions into a single graph.

Traditional methods model the brain by representing regions as nodes and similarities from one node to another, focusing on the static relationships between these nodes.[4] However, our edge-focused approach emphasizes the dynamic relationships between the edges that connect nodes, mirroring the more complex interactions found in the brain. [1] Cognitively, information processing and function result from intricate interactions not just between regions (nodes), but also between the connections (edges) linking these regions. This approach also considers co-activity, international connectivity, neural fiber measurements, and the extraction of connectivity data, including length and density from Diffusion Tensor Imaging (DTI). [2] By focusing on edge-to-edge relationships, our model captures these nuanced interactions, providing a more accurate representation of brain functionality. [3] This reflects the interconnected nature and plasticity of the brain, where the interplay between different pathways is crucial for understanding how cognitive processes and behaviors emerge. Recent claims suggest that focusing on the edge matrix rather than the node matrix is essential for making sense of the brain's complex network. [1] As a result, a GNN constructed on edge-to-edge relationships is likely to yield more meaningful and insightful data about brain activity and network dynamics.

II. Methodology

A. Edge Time Series

Edge-centric matrices are generated from time series datasets that record brain network activity across 116 regions over 210 seconds. This data is initially saved as a 210×116 matrix. From this matrix, we generate edge time series by obtaining the z-score of two matrix rows X_m and X_n. By multiplying the z-scores of these two rows, we obtain R_{mn},[3] which represents the connectivity of an edge over time. As a result, we generate a 210×6670 matrix, where 6670 is the total number of possible node combinations. Each element in this matrix represents the co-fluctuation magnitude between pairs of brain regions. The value of these co-fluctuations is positive when the activity of two regions is moving in the same direction at the same time.

B. Edge Functional Connectivity

From this edge time series (eTS) matrix, we can generate an Edge Functional Connectivity (eFC) matrix that demonstrates the dependency of each edge (co-fluctuation) and is normalized in the range of $[-1, 1]$ [2], resulting in a 6670×6670 matrix that indicates the dependency of each edge compared to other edges. The eFC can be calculated as shown in the formula below:

$$\text{eFC}_{ij,uv} = \frac{\sum_t c_{ij}(t) \cdot c_{uv}(t)}{\sqrt{\sum_t c_{ij}(t)^2}\sqrt{\sum_t c_{uv}(t)^2}}$$

where $c_{ij} = [z_i(1) \cdot z_j(1), \ldots, z_i(T) \cdot z_j(T)]$ and $c_{uv} = [z_u(1) \cdot z_v(1), \ldots, z_u(T) \cdot z_v(T)]$ are the time series for edges $\{ij\}$ and $\{uv\}$, respectively.

C. Edge Community Clustering

Dividing the group of edge-centric functional connectivity (eFC) into 10 using $k = 10$, that has similar co-fluctuation patterns, we can cluster the eFC with the k-means algorithm back to an $N_{\text{node}} \times N_{\text{node}}$ size matrix, which is 116×116. As the calculation from eFC is computationally expensive due to its size, this clustering helps to manage the complexity. The common fluctuations of edges within each community follow a similar activity trajectory (with the same properties), and each community contains different edges, which are non-overlapping communities. Since each edge is related to two nodes, overlapping node communities can be generated by mapping edges back to nodes. The same nodes may be present in each community, which means that brain regions participate in different communities for cognitive activities. The edge community is represented as a matrix whose value represents communities to which the edges between brains m and n are assigned. The matrix's mth row indicates the community in which brain region m is involved. Next, the proportion of region m in the community s is computed as follows:

$$p_{ms} = \frac{1}{N-1} \sum_{n \neq m} \delta(g_{mn}, s), \tag{1}$$

where $g_{mn} \in \{1, \ldots, k\}$ is the edge's community assignment that connects brains m and n and $\delta(x_0, y_0)$ is the Kronecker delta, whose value is 1 if $x_0 = y_0$ and zero otherwise. By definition, $\sum_s p_{ms} = 1$. The vector $\mathbf{p}_m = [p_{m1} \cdots p_{mk}]$ can be viewed as a distribution of probabilities. For $k = 5$ communities, the brain area m has five p_{ms} values. The larger the value, the greater the probability of brain area m participating in the corresponding community; that is, brain area m is more connected with this community.

D. Graph Nerual Network Computation

Edge-centric matrix calculations can be integrated into a Graph Neural Network (GNN) using two different matrices: the community matrix or the eFC matrix. Using the community matrix is computationally cheaper but may result in the loss of detail as it reduces the dimensionality of the original eFC matrix. Conversely, computation using the eFC matrix is more expensive but retains all the detailed information. Both methods involve assigning the current matrix as the edge feature to the GNN graph and using the mean of each region, calculated from the original time series matrix, as a node attribute. However, for the eFC matrix with dimensions of 6670×6670, it is necessary to use a co-embedding method to combine it into a single graph for training in the GNN.

E. Co-embedding of Node and Edge Features

Traditional GNN approaches often focus on node features only and are susceptible to the loss of information that is contained in edges that could be extracted. To address this issue, a multi-dimensional approach can be used, and edge features can be added, encoded, and utilized in the graph learning process. This process, called co-embedding, ensures proper representation for classification and prediction tasks [5] [2].

Figure 1 shows the Edge Aggregated Graph Neural Network (EAGNN), a process of taking both nodes and edges and combining them into a node-representation through co-embedding layers. It is designed to be a unified framework, where layers compute low-dimensional representations for nodes by considering both the nodes' features and the features of their connecting edges. While GNNs aggregate just node features, EAGNN also aggregates edge features. After aggregation, node representations are updated by combining the aggregated edge information with the node features from the previous layer.

Once Node representations are obtained through the co-embedding layers, edge classifiers use these representations to organize edges into categories.

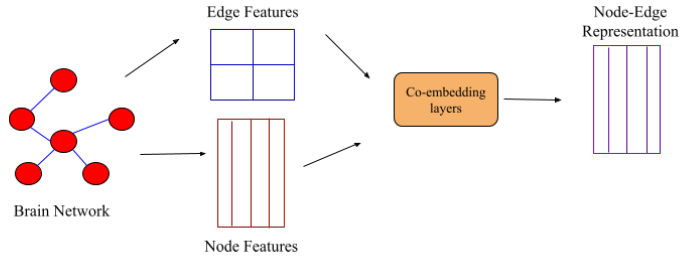

Fig. 1. The EAGNN co-embedding process allows for node and edge graphs of different sizes to be embedded without the loss of features, taking the form of node-edge representations.

The outcome of the co-embedding process are node-edge representations, which allow the model to better capture relationships between local nodes and edges, leading to more accurate predictions [2].

F. Result

Our current research has demonstrated brain network graph classification using Graph Convolutional Networks (GCN) [6] with the community matrix [2]. Future work will focus on co-embedding using the eFC matrix to investigate potential improvements compared to traditional node-based brain network construction.

References

[1] J. Faskowitz, F. Z. Esfahlani, Y. Jo, O. Sporns, and R. F. Betzel, "Edge-centric functional network representations of human cerebral cortex reveal overlapping system-level architecture," *Nature Neuroscience*, vol. 23, pp. 1644–1654, Dec. 2020.

[2] T. Hasegawa, H. Arvidsson, N. Tudzarovski, K. Meinke, R. V. Sugars, and A. Ashok Nair, "Edge-Based Graph Neural Networks for Cell-Graph Modeling and Prediction," in *Information Processing in Medical Imaging* (A. Frangi, M. De Bruijne, D. Wassermann, and N. Navab, eds.), vol. 13939, pp. 265–277, Cham: Springer Nature Switzerland, 2023. Series Title: Lecture Notes in Computer Science.

[3] W. Wang, R. Du, Z. Wang, X. Luo, H. Zhao, P. Luan, J. Ouyang, and S. Liu, "Edge-centric functional network reveals new spatiotemporal biomarkers of early mild cognitive impairment," *Brain-X*, vol. 1, p. e35, Sept. 2023.

[4] X. Li, Y. Zhou, N. Dvornek, M. Zhang, S. Gao, J. Zhuang, D. Scheinost, L. H. Staib, P. Ventola, and J. S. Duncan, "Braingnn: Interpretable brain graph neural network for fmri analysis," *Medical Image Analysis*, vol. 74, p. 102233, 2021.

[5] Y. Zhou, H. Huo, Z. Hou, L. Bu, J. Mao, Y. Wang, X. Lv, and F. Bu, "Co-embedding of edges and nodes with deep graph convolutional neural networks," *Scientific Reports*, vol. 13, p. 16966, Oct. 2023.

[6] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," Feb. 2018. arXiv:1710.10903 [cs, stat].