GEOMETRIC DEEP LEARNING OF BRAIN CONNECTOMES OVER THE SPECTRUM OF DEMENTIA

by

Gurur Gamgam

B.S., Electrical and Electronics Engineering, Boğaziçi University, 2017

M.S., Electrical and Electronics Engineering, Boğaziçi University, 2019

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in Electrical and Electronics Engineering

Boğaziçi University

2025

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to Prof. Burak Acar for his unwavering guidance and support, from my undergraduate years to this very moment. His academic passion and constant encouragement have been a source of inspiration throughout this journey. I am deeply thankful for the many valuable lessons I have learned from him, not only about academics but also about the principles of hard work and perseverance.

I would also like to extend my sincere thanks to Assoc. Prof. Alkan Kabakçıoğlu for his invaluable guidance and constructive feedback. His insights have greatly enriched my understanding and have been instrumental in shaping this work.

My sincere gratitude goes to the members of VAVLAB—Demet, Güneş, and Göktekin—for their valuable contributions to data processing and for consistently being ready to help whenever I needed it. I am truly thankful for their support.

I also wish to thank our neurologist collaborators, especially Prof. Dr. Hakan Gürvit and Dr. Zerrin Yıldırım, for sharing their clinical expertise. Dr. Zerrin Yıldırım, who unfortunately passed away, will always be remembered for her invaluable contributions and dedication.

I owe my deepest gratitude to my family—my mother and father—for their constant support and belief in me throughout this journey. Their encouragement and the comfort they provided have been the cornerstone of my achievements. I am also deeply thankful to my lovely sister, a neurologist herself, and I hope that one day we can work together. Having such a family has always been a blessing for me.

I just want to give a special thanks to my amazing friends, İrem, Hilal, and Salih. Your constant motivation and support mean the world to me. I can't express how

grateful I am to have you all in my life. Thanks for always cheering me on and making me feel so happy.

Last but most importantly, having such a wonderful partner in my life has been the greatest source of my achievements. Dear Yaren, your presence has always been one of my greatest sources of strength. I feel very lucky to have met you.

ABSTRACT

GEOMETRIC DEEP LEARNING OF BRAIN CONNECTOMES OVER THE SPECTRUM OF DEMENTIA

Alzheimer's Disease Dementia (ADD) is a progressive neurodegenerative disorder that impairs cognitive function with different stages. Accurate diagnosis and monitoring of ADD progression are crucial for early intervention and treatment. In this thesis, we investigate the application of Geometric Deep Learning, specifically Graph Neural Networks (GNNs), to brain network analysis for ADD diagnosis, with a focus on structural and functional brain networks.

The thesis is structured around three primary objectives: (1) evaluating GNNs for structural, functional, and multi-modal brain network analysis, (2) developing a novel GNN-based biomarker quantifying structure-function relationship for ADD diagnosis, and (3) proposing a novel GNN that is capable of highlighting ADD-related subnetworks. Through these objectives, we seek to provide both improved diagnostic tools and deeper insights into the progression of ADD.

In summary, this thesis demonstrates the power and potential of GNNs in diagnosing and monitoring ADD, presenting state-of-the-art methods that offer not only improved performance but also a neurological basis for the explanations of the results. These advancements aim to enhance both the accuracy and interpretability of ADD diagnostics, ultimately contributing to a better understanding of disease progression and facilitating early intervention strategies.

ÖZET

DEMANS SPEKTRUMU ÜZERİNDE BEYİN KONNEKTOMLARININ GEOMETRİK DERİN ÖĞRENİMİ

Alzheimer Hastalığı Demansı (ADD), bilişsel fonksiyonları etkileyen ve farklı evrelere sahip ilerleyici bir nörodejeneratif bozukluktur. ADD'nin doğru bir şekilde teşhis edilmesi ve ilerleyişinin izlenmesi, erken müdahale ve tedavi için büyük önem taşımaktadır. Bu tezde, Geometrik Derin Öğrenme'nin, özellikle Çizge Sinir Ağları'nın (Graph Neural Networks - GNNs), ADD teşhisi için beyin ağı analizine uygulanmasını, yapısal ve işlevsel beyin ağlarına odaklanarak inceliyoruz.

Tez, üç ana hedef etrafında yapılandırılmıştır: (1) yapısal, işlevsel ve çoklu-modal beyin ağlarının analizi için GNN'lerin değerlendirilmesi, (2) ADD teşhisi için yapı-fonksiyon ilişkisini nicelleştiren yenilikçi bir GNN tabanlı biyobelirteç geliştirilmesi ve (3) ADD ile ilişkili alt ağları vurgulayabilen yeni bir GNN önerilmesi. Bu hedefler aracılığıyla, hem geliştirilmiş tanı araçları hem de ADD'nin ilerleyişine dair daha derin bir anlayış sunmayı amaçlıyoruz.

Sonuç olarak, bu tez, GNN'lerin ADD teşhisi ve izlenmesindeki gücünü ve potansiyelini göstermektedir. Sadece gelişmiş performans sunmakla kalmayıp aynı zamanda elde edilen sonuçların nörolojik bir temele dayandırılmasını sağlayan yöntemler sunulmaktadır. Bu ilerlemeler, ADD teşhislerinin hem doğruluğunu hem de yorumlanabilirliğini artırmayı hedeflemekte ve hastalığın ilerleyişinin daha iyi anlaşılmasına katkıda bulunarak erken müdahale stratejilerini kolaylaştırmayı amaçlamaktadır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

a_{ij}	Weight of edge connecting node i and node j
$\hat{a}_{i,j}$	Attention coefficient between node i and node j
$\mathbf{A}$	Adjacency matrix
$\hat{\mathbf{A}}$	Attention coefficient matrix
$\hat{\mathbf{A}}^{b,j}$	Attention coefficient matrix produced by head j for subject b
$\mathbf{A}_{bin}$	Binarized adjacency matrix
$\mathbf{A}_{ls}$	Log scaled adjacency matrix
$\mathbf{b}_i$	BOLD signal of node i
$\mathbf{B}$	BOLD signal matrix
$\mathbf{D}$	Degree matrix
e_{ij}	Edge connecting node i and node j
E	Set of edges
f	Function operating on node features or hidden representations
$\mathbf{g}$	Graph signal
$\mathcal{G}$	Graph
$\mathbf{h}_v^k$	Hidden representation of node v at layer k
$\mathbf{h}_{\mathcal{N}(v)}^k$	Aggregated hidden representation of the neighbors of node v at layer k
$\mathbf{H}$	Latent node representation matrix
$\mathbf{I}_N$	$N \times N$ identity matrix
$\mathcal{L}_{CE}$	Cross-entropy loss function
$\mathcal{L}_{dis}$	Disentanglement loss function
$\mathbf{L}$	Laplacian matrix
$\mathbf{L}_{norm}$	Normalized Laplacian matrix
m	Margin hyperparameter
$\mathbf{M}$	Importance mask matrix
N	Number of nodes
N_c	Number of classes

$\mathcal{N}(v)$	Set of neighbors of node v
s_θ	Spectral filter parametrized by θ
T_k	kth order Chebyshev polynomial
$\mathbf{u}_l$	Eigenvector of Laplacian corresponding to eigenvalue λ_l
$\mathbf{U}$	Eigenvector matrix of Laplacian
v_i	Node i
V	Set of vertices/nodes
$\mathbf{w}$	Node scores
x_i	Node feature vector of node i
$\mathbf{X}$	Node feature matrix
y	True probabilities
$\hat{y}$	Predicted probabilities
y_c	True probability for class c
$\hat{y}_c$	Predicted probability for class c
$\mathbf{z}$	Latent graph representation
α	Projection vector in GAT
γ	Discrepancy scores
$\mathbf{\Theta}$	Parameter matrix
λ_l	l-th smallest eigenvalue of Laplacian
λ_{var}	Variance loss coefficient
λ_{dis}	Disentanglement loss coefficient
Λ	Eigenvectors of Laplacian
μ^n	Mean discrepancy scores of negative class
μ^p	Mean discrepancy scores of positive class
σ	Nonlinear activation function
Φ_k	Update function at layer k
Ψ_k	Aggregation function at layer k

LIST OF ACRONYMS/ABBREVIATIONS

ADD	Alzheimer's Disease type Dementia
ADHD	Attention Deficit Hyperactivity Disorder
ASD	Autism Spectrum Disorder
AUC	Area Under Curve
BD	Bipolar Disorder
BOLD	Blood Oxygen Level Dependent
CNN	Convolutional Neural Network
CSF	Cerebrospinal Fluid
DAGNN	Disentangled Attention Graph Neural Network
DCNN	Diffusion-Convolutional Neural Network
DMN	Default Mode Network
DWI	Diffusion Weighted Imaging
FFE	Fast Field Echo
fMRI	Functional Magnetic Resonance Imaging
fNET	Functional Network
GAT	Graph Attention Network
GCN	Graph Convolutional Network
GDL	Geometric Deep Learning
GIN	Graph Isomorphism Network
GNN	Graph Neural Network
k-NN	k-Nearest Neighbors
ℓNET	Learned Functional Network
LSTM	Long Short-Term Memory
MCI	Mild Cognitive Impairment
MDD	Major Depressive Disorder
MLP	Multi-Layer Perceptron
MRI	Magnetic Resonance Imaging
PET	Positron Emission Tomography

PNA	Principal Neighbor Aggregation Network
PD	Parkinson's Disease
RNN	Recurrent Neural Network
rs-fMRI	Resting State Functional Magnetic Resonance Imaging
SAN	Spectral Attention Network
SCI	Subjective Cognitive Impairment
sfDLN	Structure-Function Discrepancy Learning
SGC	Simple Graph Convolution
sNET	Structural Network
T1w	T1-weighted
WL	Weisfeiler-Lehman

1. INTRODUCTION

Dementia is a broad term that describes the decline of cognitive functions, such as memory and reasoning, which negatively affects an individual's ability to perform daily activities. Common signs and symptoms of dementia include heavy memory loss, difficulty with language and familiar tasks, confusion, and personality changes. While dementia is more common among older people, it is not a natural part of aging since many people grow older by not experiencing any signs of dementia. Instead, dementia arises from alterations in the brain that cause neurons and their connections to malfunction.

There are various types of dementia, and each of them is linked to specific neurodegenerative disorders. Among these, the most common types are Alzheimer's Disease type Dementia, Vascular Dementia, Lewy body Dementia, and Frontotemporal Dementia. Each type is associated with unique characteristics; for instance, Vascular Dementia results from damage to blood vessels in the brain or reduced blood and oxygen flow, while Lewy body Dementia is caused by abnormal deposits of the protein alpha-synuclein, called Lewy bodies. Despite their differences, they all lead to progressive brain damage, causing disruption of brain functioning over time.

Among the aforementioned types, Alzheimer's Disease type Dementia (ADD) accounts for 60-80 percent of the total number of dementia cases. Therefore, it is the most frequently encountered and studied dementia type [1]. ADD is characterized by protein deposits in the form of amyloid plaques and neurofibrillary tangles containing amyloid-beta ($A\beta$) peptide and hyperphosphorylated protein tau (p-tau), respectively. These deposits lead to the destruction of neurons and their connections. As ADD progresses, it causes significant neuronal and synaptic damage, leading to disconnection syndromes that impair communication between brain regions. Disconnection syndromes are often accompanied by brain atrophy—a shrinkage of brain regions resulting in substantial loss of brain volume. Although the exact causal relationship between atrophy and

disconnection syndromes remains unclear, the two are closely intertwined and are key diagnostic markers. Diagnostic approaches often aim to evaluate both phenomena, either separately or in combination, to understand the progression and impact of ADD better.

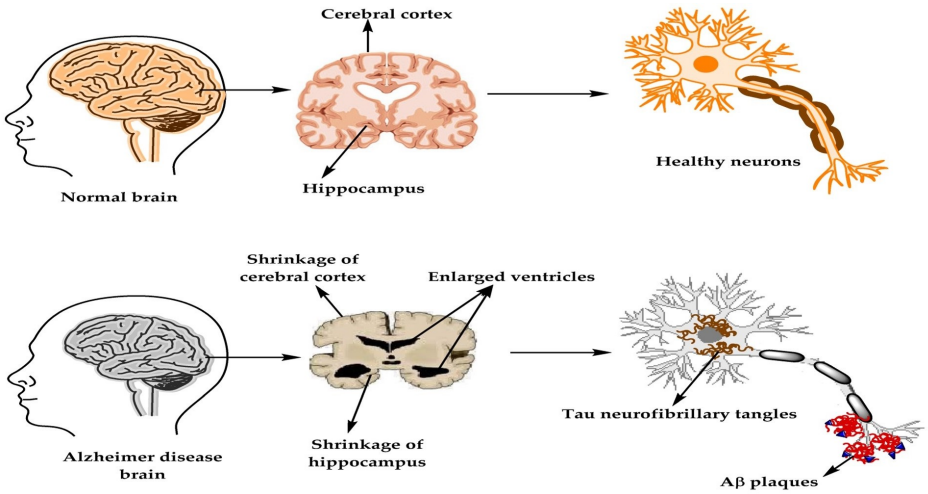

Figure 1.1. Healthy vs ADD Brain. The top panel illustrates a healthy brain with normal volume and undamaged neurons. The bottom panel shows an ADD brain with reduced volume, cortical and hippocampal shrinkage, enlarged ventricles, and disrupted neurons due to tangles and plaques.

ADD progresses over years before any clinical symptoms occur. Initial deposition of amyloid-beta peptide and protein tau begins and spreads through the brain approximately 15 to 20 years before noticeable cognitive impairments. This stage is the preclinical stage of ADD, where there are no symptoms and brain dysfunction. As amyloid plaques and neurofibrillary tangles accumulate, clinical symptoms emerge. Following the preclinical stage, individuals may face emerging memory issues that memory testing cannot identify. This stage is known as "Subjective Cognitive Impairment" (SCI) and corresponds to the early stages of ADD. As the condition progresses, the memory decline worsens and becomes detectable through memory testing. This middle stage is "Mild Cognitive Impairment" (MCI). In the later stages, as memory continues to deteriorate, other cognitive areas are also affected, resulting in multiple cognitive

deficits. When these deficits become severe enough to cause functional impairment, significantly affecting an individual's ability to perform daily activities, the condition is classified clinically as ADD.

The long preclinical stage of ADD and the absence of a cure underscore the critical importance of early diagnosis to manage symptoms effectively and delay disease progression. Detecting the preclinical stage is currently possible by analyzing biomarkers associated with amyloid plaques and neurofibrillary tangles through cerebrospinal fluid (CSF) analysis and Positron Emission Tomography (PET) scans. However, both approaches have significant limitations.

CSF analysis is highly invasive. It involves a lumbar puncture to extract a sample from the lower back, a procedure that can cause discomfort and potential side effects, such as headache and infection. These drawbacks reduce its appeal as a routine diagnostic tool. PET scans, on the other hand, provide a less invasive alternative by reflecting metabolic changes linked to neuronal injury. Despite their utility, both methods suffer from a critical limitation: their lack of specificity for ADD [2]. Many older individuals with amyloid plaques may never develop ADD, complicating the interpretation of results. This highlights the necessity of considering clinical history alongside biomarker data to improve diagnostic accuracy [3]. Consequently, while CSF and PET analyses offer insights, they fall short of providing a comprehensive and precise understanding of ADD progression.

To develop a clear understanding of ADD progression, a more detailed description of the human brain is needed. Magnetic Resonance Imaging (MRI)- based techniques offer a unique approach to studying the human brain's structure and function. MRI-based techniques are non-invasive and can track changes in brain structure and function over the course of ADD. Thus, brain models that represent the structural and functional information through the MRI, called brain connectomes, have gained popularity.

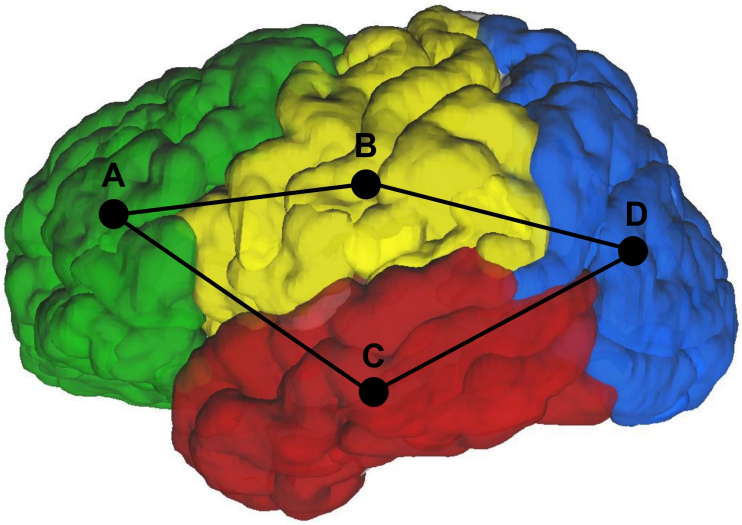

Figure 1.2. Brain as a Connectome. Here, the brain is represented as a simple graph consisting of four nodes. These nodes (A, B, C, D) correspond to brain regions, each highlighted with a distinct color. The edges between the nodes represent the relationships between brain regions, which can be either functional or structural.

A brain connectome, or simply a connectome, is a simplified, mathematical description of the elements and connections organizing the human brain, represented as a network [4]. The brain is depicted as a graph in this representation, where nodes represent its elements and edges represent their relationships. Based on this perspective, two key components to construct a connectome are the network's elements (nodes of the graph) and the connections between them (edges of the graph). Defining nodes at the microscale, such as neurons and synapses, is not feasible due to the vast number of neurons, approximately 90 billion, in the human brain. Conversely, defining nodes at the mesoscale, such as brain regions, is more practical since it reduces the number of elements to a manageable range, typically from tens to thousands. These brain regions are determined through brain parcellation using predefined anatomical atlases, such as the Destrieux Atlas [5], which relies on surface structures, or the Schaefer Atlas [6], which incorporates functional similarities in grouping regions. Identifying brain regions through such approaches is beneficial for advancing our understanding of the brain's structure and functional organization. Lastly, connectivities can be defined as either structural or functional, leading to the construction of two distinct types of networks:

structural networks (sNETs) and functional networks (fNETs).

In the context of this thesis, structural connectivity refers to the anatomical pathways formed by white matter fiber tracts linking different brain regions. Accordingly, sNETs represent the density of white matter fibers between brain regions and are reconstructed using diffusion-weighted imaging (DWI) and tractography. DWI provides data on water diffusion within tissues, which is modeled to infer the orientation of white matter fibers. Tractography, a computational method, estimates these fiber pathways. It is shown that it can align well with known brain tracts, particularly for major pathways [7]. There are two main tractography approaches: deterministic, which follows a single, most likely fiber orientation per voxel, and probabilistic, which models a range of possible orientations to account for uncertainty.

fNETs are based on statistical dependencies between blood-oxygen-level-dependent (BOLD) signals acquired through functional MRI (fMRI). The BOLD signal reflects blood oxygenation and flow changes, an indicator of underlying neural activity. Statistical measures are applied to the BOLD signals to infer functional connectivity between brain regions. These measures are primarily based on correlation measures, such as Pearson's and partial correlation.

Initial connectome analyses of ADD progression focused on network parameters using well-developed graph theory. These initial studies are based on the hypothesis that ADD progression disrupts the brain's structural and functional organization, and such disruptions affect the network's optimality, which can be detected via some network measures, such as small-worldness and modularity. Based on this assumption, group-based comparative studies identified significant alterations in the network parameters of fNETs, such as a loss of small-worldness [8–10], changes in network modularity [11,12], modifications in network hubs [13], and the presence of anomalies [14] in ADD patients when compared to control groups. Additionally, a disruption in sNETs within specific brain regions has been observed in ADD patients [15,16]. However, despite some promising results from initial studies, these studies lack specificity,

come up with conflicting results, and do not offer insights into the underlying mechanisms of ADD progression [17]. To address this limitation, researchers have focused on overcoming the challenge by leveraging new approaches to identify ADD-specific changes in the brain. They have conducted new studies using techniques such as community detection [18], tensor factorization [19], and graph kernels [20] to uncover specific brain regions affected by ADD progression.

Recently, Geometric Deep Learning (GDL) has been recognized for its success in adapting deep learning methods to non-Euclidean domains where conventional distance metrics and geometric transformations are not applicable [21]. Since brain networks are graph-structured and do not reside in the Euclidean domain, unlike image or audio signals, GDL-based methods are well-suited for their analysis. Within this domain, graph neural networks (GNNs) have gained significant popularity for studying brain networks and have been applied to various disorders, including ADD [22], Autism Spectrum Disorder (ASD) [23], and Major Depressive Disorder (MDD) [24]. GNNs are powerful tools for analyzing brain networks primarily because they can directly process graph-structured data. By leveraging graph-based representations of brain connectomes, GNNs have achieved state-of-the-art performance in classifying neurological diseases, predicting cognitive outcomes, and identifying biomarkers. Additionally, the flexibility of GNNs in handling diverse graph data types makes them highly suitable for analyzing both structural and functional brain networks.

Motivated by the potential of GNNs, this thesis aims to explore their applications to brain networks in understanding ADD progression and aiding early diagnosis. Specifically, the thesis focuses on three main objectives:

- Conducting a detailed, systematic, and comparative study of GNN models for analyzing both types of brain networks, with a particular focus on ADD, and providing a comprehensive justification for their suitability and effectiveness in this domain.

- Developing a GNN-based biomarker that quantifies the relationship between the structure and function, utilizing sNET and fNET, to support the monitoring of ADD stages.
- Proposing a novel GNN model that highlights critical brain regions, improving the detection of ADD-related changes.

The remainder of this thesis is organized as follows. Chapter 2 reviews the literature on the application of GNNs to brain networks in the context of neurological disorders, identifying gaps in current research and providing the motivation for the three main objectives outlined above. Chapter 3 details the methodology used to address these objectives: it first introduces the GNNs employed in this work, followed by the development of a GNN-based biomarker for structure-function discrepancy and the corresponding model, and concludes with the introduction of a novel model that highlights critical brain regions for ADD progression. Chapter 4 provides details on the datasets, connectome construction, and experimental setup corresponding to each objective. Chapter 5 discusses the results, and Chapter 6 concludes the thesis, highlighting its contributions and suggesting future directions.

2. LITERATURE REVIEW

Applications of GNNs in the context of neurological disorders can be broadly categorized into three main approaches based on how the input graph and node features are constructed:

- Population graph-based methods: These methods do not directly process brain connectomes. Instead, they construct population graphs where the nodes represent subjects/patients, and the edges are defined based on the similarity between subjects. Node features are derived from the brain network data. In this approach, GNNs process the population graph for downstream tasks, typically node classification. While these methods do not directly analyze brain networks, they were among the earliest applications of GNNs in brain disorders. Despite not focusing on connectomes, these works contributed significantly to the initial use of GNNs in the field, with a number of studies employing this approach.
- Common graph-based methods: These methods process brain networks with GNNs, but assume a fixed underlying network structure shared by all subjects, i.e., all subjects use the same graph. Node features are extracted from brain network data, and individual variability is captured through the node features.
- Subject graph-based methods: These methods construct individual graphs and features from brain networks, which are then processed with GNNs. They are the most widely studied approach and offer a variety of models that are particularly well-suited for analyzing brain networks.

Population-graph-based methods gained prominence early on due to their foundation in the success of convolution-based GNNs. These GNNs demonstrated their effectiveness in citation network tasks, where they set state-of-the-art results for node classification. This success inspired the creation of population graphs, where nodes represent individual subjects and node features are derived from brain network data. Population-graph-based methods perform node classification on these population graphs, where

node labels correspond to the clinical labels of the subjects.

The pioneering work of Parisot et al. [23] marked a significant milestone by constructing a population graph using phenotypic features where edges were defined using phenotypic features, such as age and sex. Node features were learned representations of functional network connectivity values generated via an autoencoder. They trained a Graph Convolutional Network (GCN) in a semi-supervised manner, demonstrating its ability to handle a large proportion of unlabeled nodes while achieving promising classification results on ADD and ASD classification. Building on this foundation, Kazi et al. [25] extended the previous work by introducing an inception-inspired mechanism into GCN. This mechanism enables nodes to aggregate information from higher-order neighbors—i.e., neighbors that are not directly connected but are within a certain number of hops. By incorporating information from these higher-order neighbors, the model enhances its ability to capture more complex and long-range relationships within brain networks. Ghorbani et al. [26] tackled the class imbalance issue in population graphs by identifying and weighting key nodes to reduce bias, demonstrating this method on Parkinson's Disease (PD) classification. Lei et al. [27] enhanced the population-graph framework by incorporating multi-modal features from structural and functional networks. Their approach also incorporated higher-order neighbors through random walks to better encode local neighborhood information, thereby improving performance on ADD classification. Qiu et al. [28] extended this line of research by employing Graph Transformers, where only neighboring nodes attend to each other, on population graphs with multi-modal features from functional and structural networks. This approach led to better performance than GCN in ADD classification tasks.

However, the efficiency of population-graph-based methods depends heavily on the quality of the constructed population graph. Specifically, the choice of phenotypic similarity measures can significantly influence performance. To address this limitation, researchers have proposed various enhancements. For instance, Song et al. [29] introduced two distinct population graphs for ADD detection: one constructed using structural networks and the other using functional networks as features. The edges

of these population graphs were defined based on phenotypic measures. Two separate GCN models were applied to these graphs to predict scores. In addition, they employed an adaptive mechanism where the edges were updated based on these predicted scores. The adaptation mechanism and construction of two different population graphs improved graph definitions and enhanced the overall performance. Jiang et al. [30] computed the edge weights of a population graph by combining a GCN and a graph kernel applied to functional networks. An additional GCN module independently processed the resulting population graph using features extracted from the functional networks. This model was validated in both ADD and ASD classification tasks, demonstrating its effectiveness in both domains. Chen et al. [31] proposed adapting the population graph, where node features were vectorized fNETs, using an encoder trained on phenotypic features, achieving robust performance on ASD data. Pan et al. [32] advanced this approach by constructing two distinct population graphs: one based on phenotypic similarity learned via a pairwise autoencoder and the other on k-nearest neighbors (k-NN) similarity of functional network features. Both graphs, which shared the same node features derived from functional networks through feature selection, were jointly processed by GCNs. This enabled the extraction of shared and distinct information from each modality, facilitating effective ASD classification. Huang et al. [33] employed a pairwise encoder on phenotypic features to model population graphs, with node features extracted through recursive feature elimination on functional networks. They utilized a variation of GCNs capable of edge-specific convolution, enabling more nuanced node representations for ADD and ASD classification. Zhang et al. [34] first extracted subject-specific functional representations by applying GCNs to individual functional networks. These representations were used as features for the population graph and to construct the graph in conjunction with phenotypic measures. The proposed method was evaluated on ASD and ADD classification tasks. Hu et al. [35] utilized the correlation of functional connections to extract higher-order functional information, combining it with functional connectivities as features and similarity measures for population graph construction. The resulting population graph was processed with a GCN, achieving improved performance on an imbalanced ADD dataset. Yang et al. [36] fused phenotypic and imaging-based similarity measures

to construct population graphs, employing a spectral-attention-based GNN to model higher-order relationships for ADD classification. Lastly, Guan et al. [37] separated fNET into positive and negative components, processing each with an autoencoder to extract separate functional features. The combination of these features served as node features for a population graph, which was constructed from representations resulting from processing these features with a Graph Transformer module, where the top k nodes were assumed to be connected. The constructed population graph and its node features were then processed with another Graph Transformer, achieving success in ADD classification.

Early studies on population-graph-based methods, while foundational, come with notable limitations. First, the performance of GNN models heavily depends on the quality of the population graph, which can vary significantly based on the chosen similarity metrics and graph construction methods. Second, the entire population graph must be reconstructed for a new patient, making this approach impractical for real-time or scalable applications. Third, population-graph-based methods are not well-suited for personalized treatment, as they do not effectively account for individual-specific characteristics. To overcome these challenges, GNNs should focus on processing individual brain connectomes directly to enable more personalized and precise analyses.

Common graph-based methods have been proposed as alternatives to population graph-based approaches. Although they process brain networks directly with GNNs, they assume a shared graph structure across all subjects. A pioneering work in this area by Ktena et al. [38] constructed a k-NN graph based on the averaged functional network of all patients, treating individual fNETs as node features. They employed siamese GCNs for metric learning, achieving promising results for ASD classification. Similarly, Zhang et al. [34] constructed a k-NN graph based on the spatial distance between brain regions from their coordinates. They used a concatenation of multiple structural networks derived from different tractography algorithms as node features. Matching vs. non-matching pairs classification was then performed on PD data. Yao et al. [39] set a distinct common graph for each atlas, where each graph is the group-

wise average of all subjects. Node features were extracted from functional or structural brain networks, and each GCN, specific to an atlas, processed these features using the corresponding common graph. A contrastive learning approach was employed to train the GCNs for the classification of ADD and Attention Deficit Hyperactivity Disorder (ADHD). Lastly, Gadgil et al. [40] developed a spatio-temporal model that combines a GCN with a Long Short-Term Memory (LSTM) module to capture the dynamic nature of BOLD signals. This model processes the BOLD signals using a common graph constructed from the correlation of all concatenated BOLD signals across patients, focusing on functional differences related to sex and age.

A notable hybrid approach that integrates population-graph and common-graph strategies is the work of Cui et al. [41]. They identified common and distinct graph structures shared across all patients using a t-test on functional connections, then extracted low-dimensional representations of BOLD signal segments through sliding windows. These representations were used to construct a population graph based on feature similarity, which was subsequently processed with a convolution-based GNN for ASD classification.

While common graph-based methods avoid many drawbacks of population-graph approaches, they remain suboptimal for personalized analysis since they primarily focus on shared features rather than individual-specific connectome characteristics. As a result, these methods may lack the precision needed for early diagnosis. To process individual brain connectomes directly, subject-graph-based methods have been proposed and they become the most prevalent methods for utilizing GNNs for brain networks. Many of proposed studies fall into this category, and to present a comprehensive review, the relevant works are organized into eight key topics:

Dynamicity and Temporal Analysis of Brain Function: This category summarizes the works leveraging functional dynamics in brain networks.

- Zhang et al. [42] utilized the dynamicity of brain function by combining GCN with Recurrent Neural Network (RNN). They used structural networks as graphs and BOLD signals as node features to classify ADD.
- Xing et al. [22] extracted dynamic functional graphs using a sliding window approach to capture the temporal variability in brain function. They then combined GCN with a LSTM and introduced auxiliary tasks, such as age and sex prediction, to enhance ADD classification performance. Structural volumes were used as node features in their model.
- Kim et al. [43] integrated spatial and temporal attention mechanisms with GIN to analyze dynamic functional networks constructed from sliding windows on BOLD signals, highlighting sex-related differences. They constructed these dynamic graphs through thresholding and used one-hot encodings with timestamps as features.
- Azevedo et al. [44] employed edge-wise graph convolution in combination with temporal convolutions to model functional dynamicity alongside spatial information. Their work identified sex-related differences. Thresholded functional networks were used as the graph, with BOLD signals as the node features.
- Chen et al. [45] extracted dynamic correlation-based and Transfer Entropy-based functional networks through sliding windows, referred to as "efficient networks," and encoded them using a combination of spatial graph convolutions and Transformers to obtain functional representations for ADD classification.
- Liu et al. [46] extracted dynamic functional graphs and BOLD signal segments using sliding windows. They processed these with spatial graph convolution and LSTMs to learn spatio-temporal functional representations for the classification of ADD, MDD, and Bipolar Disorder (BD). BOLD signals are used as node features.
- Ma et al. [47] fused functional representations learned from static and dynamic functional networks, obtained through a sliding window approach, by combining GCN with an attention mechanism from multiple atlases for ADD and ASD classification. For each network, the corresponding BOLD signals were used as node features.

- Sivgin et al. [48] proposed an efficient method for modeling dynamic and lagged functional representation learning. They used thresholded functional networks, based on percentage-wise thresholding, as the graph and connectivity values as features for age and sex classification.

Multi-Modal Representations: This category summarizes multi-modal approaches leveraging both structural and functional networks.

- Souza et al. [49] proposed a multi-modal GCN where structural networks form the graph topology and functional connections are used as node features to regress clinical scores.
- Liu et al. [50] used structural networks as graphs and functional connections as node features while introducing a constraint that nodes within similar structural communities share similar representations. Their model was tested for BD classification.
- Zhang et al. [51] developed an end-to-end trainable model that learns fused multi-modal graphs from functional and structural networks and processes these multi-modal graphs with spatial graph convolutions for ADD classification. Functional connectivities were also used as node features.
- Huang et al. [52] introduced attention-diffusion modules to process neighbors of different orders independently, extracting higher-order multi-modal information, using structural networks as graphs and functional connections as features, with bilinear pooling for epilepsy classification.
- Li et al. [53] designed multi-modal graphs using block adjacency matrices, where off-diagonal blocks represented structure-function relationships learned during training. Node degree and centrality measures from fNETs and sNETs were used as node features. The resulting graphs and node features were processed with GCN for sex classification and age regression.
- Xia et al. [54] built a multi-modal graph with encoded structure-function relationships in off-diagonal blocks and used an attention mechanism to model these relationships. Node features were set as the concatenation of sNET and

fNET connectivities. The resulting multi-modal graph was processed with spatial convolution, informed by prior knowledge of functional subnetworks, for PD classification.

- Amodeo et al. [55] employed Graph Variational Autoencoders to learn multi-modal representations for ADD classification. sNETs are used as graphs and fNET connectivities are used as node features.
- Liu et al. [56] constructed multi-modal fusion graphs from structural and functional networks by selecting important connections from both modalities using a learnable method. These graphs were encoded with GCN for neuropsychiatric disorder classification, with fNET connectivities used as node features.
- Wang et al [57] learned cross-modal graph representations distilled from structural and functional networks using spatial graph convolutions for AD and PD classification. Each modality used its corresponding connections as node features.
- Kong et al. [24] encoded shared information from structural-static functional networks and static-dynamic functional networks using spatial graph convolution for MDD classification. Each modality, including dynamic functional networks, used its corresponding connections as node features.

Contrastive and Self-Supervised Learning: This category summarizes the application of contrastive and self-supervised learning to subject-graph-based methods.

- Zhang et al. [58] employed a GNN with contrastive learning and trainable Bernoulli masks to enhance functional representations for ASD and ADHD classification. A fully connected graph was assumed, with frequency fluctuations from BOLD signals used as node features.
- Wen et al. [59] proposed a masked autoencoder-based self-supervised pretraining method, where masked portions of a functional graph are reconstructed from the remaining parts. This approach improved functional representations for ASD, MDD, and BD classification.

Pooling and Hierarchical Learning: This category summarizes the works focusing on the pooling of node features to extract graph level representation.

- Huang et al. [60] employed a GNN module to learn the latent node representations, followed by clustering based on the probability distribution of these representations to obtain a structural representation for PD classification. The initial graph is constructed using a k-NN graph from sNET, with sNET connectivity values used as features.
- Wen et al. [61] employed soft assignment on functional networks, guided by prior knowledge of functional subnetworks like the salience network and the default mode network (DMN), to perform graph coarsening for ASD classification.
- Chen et al. [62] employed GNN-based soft assignment to learn multiple functional representations, each aimed to model different functional states. These learned representations are then processed with an attention module to form the final graph representation for AD classification.
- Liu et al. [63] utilized GCNs to map finer-resolution atlases to coarser ones, generating multi-resolution functional representations for ADD and ASD classification. One-hot encodings were used as features for the fNET corresponding to each resolution.

Attention Mechanisms: This category highlights the works that integrate attention mechanisms into GNNs.

- Yang et al. [36] used edge-weighted attention for multi-modal representations, where graphs are functional networks, and node features are structural and functional statistics, combined with hierarchical pooling for BD classification.
- Zhou et al. [64] utilized a combination of local and global attention mechanisms to learn functional representations and evaluate node importance for ADD diagnosis. The positive component of fNET was used as the graph, with fNET connectivities serving as node features.

- Li et al. [65] integrated GCN and Transformer architectures to capture higher-order information for ASD classification, using fNETs as both the graph and node features.
- Yu et al. [66] constructed first- and higher-order functional networks from Pearson and Spearman-based fNET. These graphs were later thresholded and binarized, then processed with GAT for schizophrenia classification.

Graph Construction and Adaptive Learning: This category summarizes the works focusing on the learning and adjusting graphs from input data.

- Zhang et al. [67] employed a GCN with adaptive graph updates to learn functional representations for depression score regression, using fNET connectivities as node features.
- Yang et al. [68] generated sparse functional graphs from BOLD signals and processed them using a Graph Attention Network (GAT) for ASD classification. fNET connectivities were used as node feaures.
- Li et al. [69] developed dynamic "effective networks" from BOLD signals using Kalman filtering under a causality assumption. Features extracted from these networks were combined with the resulting graphs and processed using a modified GAT that incorporates edge weights for ADD classification.
- Xia et al. [70] modeled functional graph construction as a reconstruction problem and applied directed graph convolution to learned graphs for ADD and ASD classification. Learned fNETs were used as graphs and BOLD signals were used as node features.
- Wang et al [71] learned functional graphs using a Transformer model and processed them with GCN for ASD classification. Learned fNETs were used as the graphs, with BOLD signals serving as node features.

Innovative Architectures and Applications: This category highlights works employing novel GNN architectures and applications.

- Kim et al. [72] applied Graph Isomorphism Network (GIN) to functional networks for identifying sex-related differences. Thresholded fNETs were used as graphs, with one-hot encoding features set as node features.
- Choi et al. [73] addressed the oversmoothing problem in GCNs by scaling node locality individually for each node and validated their approach for ADD classification using structural networks.
- Zhao et al. [74] proposed a graph convolution model that uses Euclidean distance-based node aggregation to learn functional representations for ADHD classification. Thresholded fNETs were used as graphs, with connectivity values serving as node features.
- Park et al. [75] utilized Hodge-Laplacian GNNs to extract richer topological information from structural networks for ADD classification.
- Huang [76] et al used Hodge-Laplacian-based methods to predict intelligence scores from functional networks.
- Qu et al. [77] disentangled structural and positional functional encodings within graph transformers to predict cognitive scores. Thresholded fNETs were used as graphs with node features set to connectivity values.
- Jiang et al. [78] constructed regularized functional graphs with enhanced small-world characteristics and processed them using spatial graph convolutions for cognitive score regression. k-NN fNET were constructed and connectivity values were used as node features.
- Hwang et al. [79] proposed graph convolution with separate node-wise and edge-wise learning for structural representations in ADD classification. Cortical thickness was used as a node feature.
- Xu et al. [80] learned a "contrastive graph" that highlights differences between groups and embedded this information to enhance classification performance for ASD, AD, and PD classification. Thresholding was applied to construct fNET, with connectivity values acting as node features.
- Huang et al. [81] enhanced functional representations with Cycle Convolution and edge-positional encodings to study intelligence differences.

Explainability and Subnetwork Analysis: This category highlights works focused on improving the explainability of GNN models.

- Li et al. [82] identified functional subnetworks linked to ASD-related biomarkers using a salience mapping method. This approach assigns importance scores to subgraphs by modeling the posterior probability, capturing the dependence of biomarkers on specific subgraphs.
- Cui et al. [83] enhanced model performance by learning a mask that highlights important edges for downstream tasks. They fine-tuned the models on the masked graphs, validating their approach for BD and PD classification from structural networks.
- Li et al. [84] introduced region-specific convolutions with region-selection pooling, which selects key nodes and discards the rest to produce explainable functional representations for ASD classification. Thresholded fNETs were used as graphs and fNET connectivities were used as node features.
- Zheng et al. [85] utilized prototype learning based on Graph Variational Encoders to extract independent functional subgraphs, applying this approach to classify ASD, MDD, and schizophrenia. Thresholded fNETs were constructed as graphs, with fNET connectivities serving as node features.

As discussed above, various subject-graph-based methods have been proposed. They utilize different GNN-based models. The primary advantage of GNNs lies in their ability to utilize the underlying graph structure when producing latent node or graph representations. However, this advantage is not always demonstrated, as most studies do not to compare GNNs with alternative approaches that do not rely on the graph structure. Justifying the use of GNNs is particularly important because, in some domains, GNNs have been shown to degrade performance when processing graph data compared to methods that ignore the graph structure [86]. This suggests that, in some instances, the graph structure itself might hinder node representation learning. Thus, establishing the suitability and effectiveness of GNNs for brain network analysis is crucial.

Various GNN architectures exist, each employing different strategies for leveraging graph structure and producing graph-level representations by pooling. Despite the availability of such diverse architectures, the challenge of effectively utilizing structural and functional networks in GNNs remains. Due to their sparsity, structural networks are inherently more compatible with GNNs; however, edge weights might require preprocessing to enhance GNN performance. Functional networks, on the other hand, typically undergo thresholding or k-NN processing to construct appropriate sparse graphs.

Despite the variations that exist, only a limited number of studies have systematically justified the use of GNNs by comparing them with other methodologies and assessing their performance in terms of feature selection, architectural design, and pooling strategies [87], [88]. Furthermore, these evaluations have not addressed ADD. To fill these gaps and provide valuable insights into optimal architectures for both structural and functional networks, this thesis first undertakes a comprehensive, systematic comparison of GNN architectures for analyzing structural and functional brain networks, with a particular emphasis on ADD.

More importantly, existing studies in the literature often overlook the significant personal variability inherent in brain networks when performing classifications. They typically operate under the assumption that brain networks sharing the same clinical label exhibit similarities. While this assumption holds to some extent and is essential for diagnostic purposes, it relies on comparing a brain with others as an external reference, which may differ significantly from the brain being evaluated. However, evaluating each brain based on its own internal reference—focusing on its self-consistency—can provide more personalized and meaningful insights. This approach acknowledges that brain networks may vary across individuals and emphasizes that the critical factor is the internal harmony within each brain. To measure this internal self-consistency, we propose leveraging the relationship between structural and functional networks within the brain. While many multi-modal studies treat structural and functional networks as complementary—either by fusing them or by modeling functional dynamics under

structural constraints [22,42]—they often overlook this structure-function relationship. However, it has been shown that this relationship changes as cognitive impairment progresses [89, 90]. Addressing this gap, we propose a GNN-based biomarker that quantifies the structure-function relationship within the brain. By evaluating both the structural and functional networks, this approach facilitates effective monitoring of ADD stages, offering a novel perspective for personalized diagnostics.

A GNN architecture designed for brain networks should not only achieve high classification performance but also be capable of highlighting the mesoscale structures within the brain, as understanding the underlying mechanisms of ADD is equally critical. Recognizing this need, recent explanatory GNN-based methods have been introduced. However, these approaches remain limited in their ability to provide comprehensive insights. Attention-based GNNs, on the other hand, have gained traction for their enhanced interpretability, offering the capability to identify significant connections while filtering out irrelevant ones. Despite their potential, attention-based GNNs have not yet been thoroughly explored for brain networks. To bridge this gap, we propose a novel attention-based GNN that effectively identifies and highlights critical brain regions and potentially ADD-related subnetworks, contributing both to diagnosis and understanding of underlying neural mechanisms.

In conclusion, this thesis identifies three significant gaps in the applications of GNNs to brain networks for ADD. To address these gaps, we establish three main objectives:

- Conducting a detailed, systematic, and comparative study of GNN models for analyzing both types of brain networks, with a particular focus on ADD, and providing a comprehensive justification for their suitability and effectiveness in this domain.
- Quantify the structure-function relationship within individual brain networks, proposing a novel GNN-based biomarker that evaluates each brain based on its internal coherence rather than external references, providing personalized insights

into ADD progression.

- Develop an attention-based GNN to highlight critical brain regions and identify ADD-related subnetworks, enhancing the detection of ADD-related changes and advancing diagnostic capabilities.

3. METHODOLOGY

The thesis addresses three main research objectives in the application of GNNs to brain networks for ADD. These objectives are:

- Conducting a detailed, systematic, and comparative study of GNN models for analyzing both types of brain networks, with a particular focus on ADD, and providing a comprehensive justification for their suitability and effectiveness in this domain.
- Quantify the structure-function relationship within individual brain networks, proposing a novel GNN-based biomarker that evaluates each brain based on its internal coherence rather than external references, providing personalized insights into ADD progression.
- Develop an attention-based GNN to highlight critical brain regions and identify ADD-related subnetworks, enhancing the detection of ADD-related changes and advancing diagnostic capabilities.

Based on these objectives, this chapter is organized as the following : The first section provides an introduction to the fundamentals of GNNs and explores various GNN models that will be used across all objectives. The second section focuses on quantifying the structure-function relationship within individual brain networks, proposing a novel GNN-based biomarker for personalized ADD progression insights. The third section introduces the attention-based GNN, designed to highlight critical brain regions and identify ADD-related subnetworks.

3.1. Graph Neural Networks

An undirected, weighted graph $\mathcal{G} = \{V, E, \mathbf{A}\}$ is defined by three components: a set of N vertices or nodes ($V = \{v_i\}$), a set of edges ($E = \{e_{ij}\}$), and an $N \times N$ adjacency matrix $\mathbf{A}$, where a_{ij} represents the weight of the edge e_{ij}, or is zero if no such

edge exists. Existence of the e_{ij} means that the node v_i and v_j are neighbors. In the context of brain networks, the nodes represent cortical brain regions, also referred as region of interests (ROIs), and the edges signify the structural or functional connections between these regions. Each node v_i is associated with an M-dimensional row vector $\mathbf{x}_i$ (either an embedding or feature vector), forming the $N \times M$ feature matrix $\mathbf{X}$.

GNNs are a class of neural network architectures designed to process data represented as graphs. They operate on graph-structured data by learning representations of nodes, edges, or entire graphs through a combination of graph-specific operations and neural network techniques. GNNs extend traditional neural networks to graph domains by leveraging graph topology and associated features to generate descriptive latent node representations enriched with structural information. Depending on their architecture, GNNs can be applied to a variety of tasks, including node classification, link prediction, and graph classification, while supporting both local and global operations. Formally defining, a GNN can be represented as

$$\mathbf{H} = \mathrm{GNN}(\mathbf{X}, \mathbf{A}), \tag{3.1}$$

where GNN represents the GNN layer, and $\mathbf{H}$ is the latent node representation matrix, with each row $\mathbf{h}_i$ corresponding to the latent representation of node i. After applying a graph-pooling operation, a graph-level latent representation can be obtained as

$$\mathbf{z} = \mathrm{Pool}(\mathbf{H}), \tag{3.2}$$

where Pool refers to an operation (e.g., sum, mean, or parametric function) over the latent node representations, and $\mathbf{z}$ is the resulting graph-level latent representation. This representation can then be used for downstream tasks as

$$\hat{y} = \mathrm{Task}(\mathbf{z}), \tag{3.3}$$

where Task represents the task-specific layer and $\hat{y}$ is the prediction. Since brain network analysis primarily involves classification, task-specific layer predicts probability distribution over classes. The entire model, including the GNN layer, pooling, and Task layer, is trained using cross-entropy loss, $\mathcal{L}_{\mathrm{CE}}$, defined as

$$\mathcal{L}_{\mathrm{CE}}(\hat{y}, y) = -\sum_{c=1}^{N_c} y_c \log(\hat{y}_c), \tag{3.4}$$

where N_C is the number of classes, y_c is the true probability for class c, and $\hat{y}_c$ is the predicted probability for class c.

Most GNNs operate locally, constructing latent node representations by aggregating features from neighboring nodes and updating these representations using parametric functions. GNNs that rely on local operations can be referred as local GNNs. Their design is based on the assumption that local information is crucial for constructing a node's representation. This locality assumption is widely applied in other domains, such as image and speech processing, where convolution operations are used to extract localized features. While graphs inherently reside in a non-Euclidean domain, leveraging local information remains essential for learning compact and meaningful node representations.

Local GNNs adopt a message-passing framework to learn latent node representations [91]. In this framework, each node iteratively updates its representation by aggregating information from its neighbors. The forward propagation process in Local GNNs can be expressed through two operations: *aggregation* (Ψ) and *update* (Φ), applied iteratively across multiple layers

$$\mathbf{h}^k_{\mathcal{N}(v)} = \Psi_k(\{\mathbf{h}^{k-1}_u\}_{u\in V}) : \text{aggregate}, \tag{3.5}$$

$$\mathbf{h}^k_v = \Phi_k(\mathbf{h}^{k-1}_v, \mathbf{h}^k_{\mathcal{N}(v)}) : \text{update}. \tag{3.6}$$

Here, $\mathcal{N}(v)$ represents the set of neighbors of node v, $\mathbf{h}^k_v$ is the hidden representation of node v at layer k and $\mathbf{h}^k_{\mathcal{N}(v)}$ denotes the aggregated representation of the neighbors of node v at layer k. During the aggregation step, information (messages) from neighboring nodes is processed using the aggregation function Ψ. This process, commonly referred to as message passing, is followed by the update step, where the aggregated messages and the node's current representation are transformed into a new latent node representation using the update function Φ. Both the aggregation and update functions are typically parametric, and their parameters are learned during training.

In contrast to local GNNs, some variants of GNNs operate globally, aggregating features from all nodes in the graph rather than restricting the aggregation to local neighborhoods. These models, referred to as global GNNs, aim to capture non-local dependencies and global context. Despite the non-locality, global GNNs can still be described within the message-passing framework using analogous *aggregation* (Ψ) and *update* (Φ) operations, as shown below

$$\mathbf{h}^k_{\mathcal{N}(v)} = \Psi_k(\{\mathbf{h}^{k-1}_u\}_{u\in V}) : \text{aggregate}, \tag{3.7}$$

$$\mathbf{h}^k_v = \Phi_k(\mathbf{h}^{k-1}_v, \mathbf{h}^k_{\mathcal{N}(v)}) : \text{update}. \tag{3.8}$$

In this case, V denotes the set of all nodes in the graph, and the aggregation function Ψ operates over all node features to capture global context. To explicitly distinguish between local and global GNNs, the aggregation step can be reformulated as follows

$$\mathbf{h}^k_{\mathcal{N}(v)} = \begin{cases} \Psi_k(\{\mathbf{h}^{k-1}_u\}_{u\in\mathcal{N}(v)}) & \text{(local aggregation, local GNN)} \\ \Psi_k(\{\mathbf{h}^{k-1}_u\}_{u\in\mathcal{V}}) & \text{(global aggregation, global GNN).} \end{cases} \tag{3.9}$$

This formulation emphasizes the key distinction: local GNNs aggregate information from a node's immediate neighborhood ($\mathcal{N}(v)$), whereas global GNNs aggregate information from all nodes in the graph ($\mathcal{V}$).

While the distinction between local and global aggregation provides insight into the scope of information utilized by GNNs, it does not sufficiently capture the diversity of GNN architectures. For a more detailed and meaningful classification, GNNs can be broadly grouped into three categories based on their underlying design principles: Convolution-based Graph Neural Networks, Expressivity-based Graph Neural Networks, and Attention-based Graph Neural Networks. Convolution-based Graph Neural Networks adapt convolution operations to the graph domain, aiming to extract local information efficiently. On the other hand, Expressivity-based GNNs aim to maximize the model's ability to distinguish between different graphs by mapping them to distinct representations. Finally, Attention-based GNNs integrate attention mecha-

nisms, which enhance the aggregation process by emphasizing critical connections and filtering out less relevant ones. Among these, Convolution-based GNNs serve as the foundation for many modern GNN architectures and will be introduced first.

3.1.1. Convolution-based Graph Neural Networks

Convolutional Neural Networks (CNNs) have achieved tremendous success in fields such as image, video, and audio processing by leveraging localized convolutional filters to learn expressive local patterns from data [92, 93]. These filters are inherently translation-equivariant, enabling CNNs to recognize patterns regardless of their spatial location, a property that enhances their robustness and generalization capabilities. Moreover, CNNs are versatile tools capable of handling data of varying sizes and resolutions, making them indispensable in applications ranging from computer vision to speech processing.

Motivated by CNNs' ability to capture localized features effectively, researchers sought to adapt convolutional operations for non-Euclidean domains such as graphs. Unlike images or audio, graphs have irregular structures that inherently represent relationships between entities, making them suitable for tasks where information lies in a node's local neighborhood. Convolutions, being inherently local operators, provide a natural way to process graph data by aggregating information from a node's immediate neighbors. This connection between the locality of convolution and the structure of graphs inspired the emergence of convolution-based GNNs.

As convolution-based GNNs gained traction, two primary approaches emerged to define and apply convolutions on graphs: spatial convolution-based GNNs and spectral convolution-based GNNs. Spatial convolution-based GNNs focus on leveraging the local neighborhood structure of graphs, applying convolution operations directly in the spatial domain to aggregate features from nearby nodes. In contrast, spectral convolution-based GNNs draw from spectral graph theory, aiming to connect the graph's spectral properties with its spatial structure through filters defined in terms

of eigenvalues and eigenvectors. Together, these approaches represent complementary strategies for extending convolutional operations to non-Euclidean domains, each offering unique perspectives on graph data processing.

The initial wave of GNN research focused on spatial convolution, where the convolutional operation was applied directly in the spatial domain of the graph. These methods aimed to leverage the graph's local structure by treating the neighborhood of each node as the domain for feature aggregation. This approach provided a simple yet effective way to define graph convolutions by mimicking the locality principle observed in CNNs.

Spatial convolutional-based GNNs were appealing because of their simplicity and their natural alignment with the local nature of graph structures. By aggregating information from neighboring nodes, these methods efficiently captured localized patterns and built expressive node representations while preserving the graph's inherent topology. The spatial convolutional approach proved particularly well suited for tasks where local interactions within the graph play a central role, such as node classification and link prediction.

Atwood et al. [94] introduced the first model to operate on graphs based on the principle of convolution, known as the Diffusion-Convolutional Neural Network (DCNN). DCNN performs a novel diffusion-convolution operation to build node representations by incorporating information from neighbors across multiple hops in a graph-structured input. The model is motivated by the idea that leveraging higher-order information through graph diffusion can produce strong latent representations.

The diffusion process is represented as a power series of the degree-normalized transition matrix, which provides a systematic mechanism for including contextual information from neighbors at various distances (hops) around each node.

DCNN takes as input a feature matrix ($\mathbf{X}$) (node features) and an adjacency matrix ($\mathbf{A}$) (graph structure). To perform the diffusion-convolution operation, the model computes a tensor $\mathbf{P}$ of dimensions $N \times N_{hop} \times N$ where N is the node number, and N_{hop} is the maximum hop distance. This tensor encodes a power series of the degree-normalized transition matrix up to order N_{hop}, serving as the foundation for extracting higher-order neighborhood information. The propagation rule for DCNN can be summarized as

$$\mathbf{H} = \sigma(\mathbf{\Theta} \odot \mathbf{PX}), \tag{3.10}$$

where σ is nonlinear function, $\mathbf{\Theta}$ is a $H \times F$ matrix containing learnable parameters of the model and F is the feature dimension.

In forward pass, tensor multiplication is performed between node features and power series tensor. Node features, $\mathbf{X}$, is a $N \times F$ matrix and $\mathbf{P}$ is a $N \times H \times N$ tensor. Multiplication of those gives us a $N \times H \times F$ tensor. Then, this tensor is multiplied elemenet-wise by a weight matrix $\mathbf{\Theta}$. Resulting tensor of activations is a $N \times H \times F$ tensor where each node has a latent representation of size $H \times F$. DCNN only entails $H \times F$ parameters making the size of the latent diffusion-convolutional representation independent of the size of the input.

A pioneering contribution to spatial convolution-based GNNs was made by Hamilton et al. [95], who proposed the GraphSAGE model. GraphSAGE introduces an efficient method for aggregating features from neighboring nodes to compute latent node representations using a set of aggregation functions. The core idea of GraphSAGE is to apply an aggregation function to encode information from neighboring nodes into a single vector. This aggregated vector is then combined with the node's own representation to compute an updated latent representation. The propagation rule of GraphSAGE is

defined as follows

$$\begin{aligned} \mathbf{h}_{\mathcal{N}(v)}^{k} &= \Psi_k(\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)), \\ \mathbf{h}_v^k &= \sigma(\mathbf{\Theta}^k.CONCAT(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)), \\ \mathbf{h}_v^k &= \mathbf{h}_v^k / ||\mathbf{h}_v^k||_2. \end{aligned} \tag{3.11}$$

Here, the model computes embeddings iteratively over K layers starting with $k = 0$, where the initial node embeddings are the input features. Iterating over K layers allows model to encode information coming from higher order neighbors up to K hop, enriching the learned embeddings with multi-hop contextual information.

In each layer, a node first aggregates the representations of its neighbors from the previous layer, $(\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v))$ into a single vector $\mathbf{h}_{\mathcal{N}(v)}^k$. After aggregating the neighboring feature vectors, the node's current representation, $\mathbf{h}_v^{k-1}$ is concatenated with the aggregated neighborhood vector, $\mathbf{h}_{\mathcal{N}(v)}^k$ and this concatenated vector is transformed using a fully connected layer, parametrized by $\mathbf{\Theta}^k$, with a nonlinear activation function σ to produce the updated node embedding. Finally, the latent node representations are normalized to ensure stability.

GraphSAGE introduces several types of aggregator functions to encode neighborhood information efficiently, including mean aggregators, LSTM-based aggregators, and pooling aggregators. The general form of the aggregation function, regardless of its type, is given as

$$\Psi_k = f(\sigma(\{\mathbf{\Theta}^k \mathbf{h}_u^k\}), \tag{3.12}$$

σ is nonlinear activation function, $\mathbf{\Theta}^k$ stores parameters specific to the aggregator, and f is the mean, max, or LSTM function applied to the set of neighbor features.

The learnable parameters in GraphSAGE include the weights and biases of the aggregator functions and the transformation layer applied after the concatenation operation. These parameters are shared across all nodes, making the model scalable

and independent of the graph size. This efficient design makes GraphSAGE particularly suitable for graph classification tasks and for predicting node labels in dynamic or evolving graphs, where the graph structure changes over time or contains a large number of nodes.

To summarize, spatial convolution-based GNNs excel at leveraging local graph structures by aggregating information from neighboring nodes, making them particularly effective for tasks focused on localized patterns. However, these methods inherently rely on spatial operations without explicitly accounting for the spectral properties of graphs. While approaches like DCNN and GraphSAGE have advanced the field by introducing efficient mechanisms for capturing higher-order neighborhood information, their focus remains primarily on the spatial domain. This leaves room for methods that can bridge the gap between local graph structures and spectral characteristics, offering a complementary perspective for graph representation learning.

Spatial convolution-based GNNs effectively leverage local graph structures; however, they do not have a direct connection to the spectral properties of graph convolution. In the Euclidean domain, such as with images, the spatial and spectral aspects of convolution are well understood and are linked through mathematical tools like the Fourier transform. In contrast, applying spatial convolution directly to graphs does not inherently align with their spectral properties. This disconnection limits the ability to effectively capture the characteristics of the graph signal.

Spectral graph theory provides a rigorous mathematical foundation for analyzing graphs in the spectral domain [96]. Within this framework, convolutional filters can be defined using using the eigenvalues and eigenvectors of the graph Laplacian, which allows for operations that process signals based on their frequency components. Spectral filters are particularly powerful for capturing global patterns and addressing tasks such as graph signal processing and network-wide pattern recognition, offering a theoretical underpinning for frequency-aware graph operations.

However, a key limitation of traditional spectral convolutional filters is their global nature. These filters act across the entire graph and do not inherently preserve spatial locality, which can lead to unintended aggregation of information from distant, unrelated nodes. This lack of spatial focus can hinder the model's ability to capture fine-grained, localized features that are critical in many practical applications.

To overcome this limitation, spectral convolution-based GNNs introduced methods for spatially localized spectral convolutions, which aim to balance the benefits of global spectral analysis with localized feature aggregation. These approaches adapt spectral filters to focus on a node's immediate neighborhood while maintaining a theoretical foundation rooted in spectral graph theory. By bridging the gap between spatial and spectral domains, these models provide a more comprehensive framework for graph convolutions.

Bruna et al. [97] made the first attempt to define spectral convolutional filters with spatial support by introducing cubic spline kernels. However, their approach fell short of providing a clear solution for efficiently combining spatial locality with spectral properties. The pioneering work of Defferrard et al. [98] addressed this challenge by proposing ChebyNet, the first practical implementation of spatially localized spectral convolutions. ChebyNet demonstrated that it is possible to achieve spatial localization within a theoretical spectral framework, marking a significant step forward in graph convolutional network design.

To gain a better understanding of ChebyNet, it's essential to clarify the definition of graph convolution in the spectral domain. In this context, convolution on graphs is defined using the graph Fourier basis [99]. This basis allows us to project graph signals into a frequency domain, much like the Fourier transform functions in Euclidean space. The graph Fourier basis is obtained from the eigendecomposition of the graph Laplacian, which represents the graph's underlying structure. The graph Laplacian is defined as

$$\mathbf{L} = \mathbf{I}_N - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T, \tag{3.13}$$

where $\mathbf{D} = \{d_{ii} = \sum_j a_{ij}\}$ is the diagonal degree matrix and $\mathbf{I}_N$ is the $N \times N$ identity matrix. The graph Laplacian $\mathbf{L}$ is a real, symmetric, positive semi-definite matrix, therefore, it has a complete set of orthonormal eigenvectors, $(\mathbf{u}_l)_{l=0,1...N-1}$ associated with non negative eigenvalues $(\lambda_l)_{l=0,1...N-1}$. These eigenvalues can be ordered as $0 = \lambda_0 < \lambda_1 \leq \lambda_2 ..... \leq \lambda_{N-1}$ with $\lambda_0 = 0$ corresponding to the trivial eigenvector $\mathbf{1}$, satisfying $\mathbf{L1} = 0$. These eigenvalues are placed along the diagonal of $\mathbf{\Lambda}$, while the eigenvectors are organized as the columns of $\mathbf{U}$.

The graph Laplacian can also be interpreted as a second-order differentiation operator. For a signal $\mathbf{g} \in \mathbb{R}^N$ defined over the nodes of the graph, the Laplacian acts as

$$\mathbf{g}^T \mathbf{L} \mathbf{g} = \sum_{i \in V} \sum_{j \in V} a_{i,j} (g(i) - g(j))^2, \tag{3.14}$$

where $a_{i,j}$ represents the edge weight connection node i and node j and $g(i)$ is the value of the signal $\mathbf{g}$ on node i. This operation captures how the signal $\mathbf{g}$ varies across the graph by comparing the values of neighboring nodes, i.e it measures the smoothness of the signal over the graph.

A smooth signal is characterized by similar values at strongly connected nodes (nodes connected by a large edge weight). In this case, Equation 3.14 will return a small value, indicating that the signal does not vary much between these nodes. In contrast, if strongly connected nodes have significantly different values, the operation will return a larger value, signaling a greater difference in the signal across the connection.

Equation 3.14 can be also seen as a specialized form of eigenvector-eigenvalue equation. Recall the standard eigenvector-eigenvalue equation

$$\mathbf{L} \mathbf{u}_l = \lambda_l \mathbf{u}_l. \tag{3.15}$$

Multiplying both sides of this equation by $\mathbf{u}_l^T$ results in

$$\mathbf{u}_l^T \mathbf{L} \mathbf{u}_l = \lambda_l. \tag{3.16}$$

Since $\mathbf{u}_l^T$ is unit vector, this equation shows that the eigenvalues λ_l represent a measure

of the smoothness of the corresponding eigenvector $\mathbf{u}_l$.

Following the Equation 3.16, low eigenvalues correspond to smooth, slowly varying eigenvectors, while high eigenvalues correspond to eigenvector that changes abruptly. Eigenvalue-eigenvector pair carries the notion of frequency-complex exponentials in the conventional Fourier transform defined in the Euclidean domain. The eigendecomposition of the graph Laplacian provides the graph Fourier basis, where the eigenvectors $\mathbf{u}_l$ resemble the complex exponentials, and the eigenvalues λ_l are analogous to the corresponding frequencies.

By representing a graph signal as a linear combination of eigenvectors of Laplacian, we can define the graph Fourier transform. The graph Fourier transform of a graph signal $\mathbf{g}$ can be defined as $\hat{\mathbf{g}} = \mathbf{U}^T\mathbf{g}$ and the inverse transform as $\mathbf{g} = \mathbf{U}\hat{\mathbf{g}}$. One can then define a spectral filter $s_\theta(\mathbf{\Lambda})$ in the Fourier domain, a parametric function of the eigenvalues $\mathbf{\Lambda}$ of $\mathbf{L}$, which acts on the graph signal $\mathbf{g}$ as

$$s_\theta * \mathbf{g} = \mathbf{U}s_\theta(\mathbf{\Lambda})\mathbf{U}^T\mathbf{g}. \tag{3.17}$$

This filters amplifies or attenuating the contributions of some of the graph Fourier components, eigenvectors.

However an arbitrary filter does not guarantee the spatial localization because resulting matrix from $\mathbf{U}s_\theta(\mathbf{\Lambda})\mathbf{U}^T$ may include connections that are not present in the adjacency matrix. In order to achieve spatial localization, ChebyNet define g_θ in terms of a truncated Chebyshev expansion [98]

$$s_\theta(\Lambda) = \sum_{k=0}^{K} \theta_k T_k(\tilde{\Lambda}) \Rightarrow s_\theta * \mathbf{g} = \sum_{k=0}^{K} \theta_k \mathbf{U} T_k(\tilde{\Lambda})\mathbf{U}^T\mathbf{g}, \tag{3.18}$$

where $T_k(x)$ are the orthogonal Chebyshev polynomials with support $[-1, 1]$, $\tilde{\Lambda} = \frac{2}{\lambda_{max}}\Lambda - I_N$ maps the eigenvalues of $\mathbf{L}$ (the maximum being λ_{max}) to this domain, and K is the cutoff on the polynomial order which allows messages from neighbors up to K hops away to be aggregated.

To define filters consistent with the support of Chebyshev polynomials, ChebyNet uses normalized graph Laplacian which is defined as the

$$\mathbf{L}_{norm} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}, \tag{3.19}$$

where $\mathbf{L}_{norm}$ denotes the normalized graph Laplacian. Like the graph Laplacian, the normalized graph Laplacian also possesses a complete set of eigenvectors and eigenvalues, carrying a similar notion of frequency-complex exponentials. However, a key difference is that the eigenvalues of the normalized graph Laplacian are bounded within the interval $[0, 2]$. Therefore ChebyNet assumes $\lambda_{max} = 2$ for practical uses. Final formulation of ChebyNet is

$$\mathbf{H} = [\mathbf{X},\ \mathbf{L}_{norm}\mathbf{X},\ T_2(\mathbf{L}_{norm})\mathbf{X} \dots T_K(\mathbf{L}_{norm})\mathbf{X}]\mathbf{\Theta}, \tag{3.20}$$

where $T_k(\mathbf{L}_{norm}) = \mathbf{U}T_k(\tilde{\Lambda})\mathbf{U}^T$ and $\mathbf{\Theta} \in \mathbb{R}^{M(K+1)\times F}$ are the filter parameters to be learned. Choice of K allows model to inject higher order relationships. ChebyNet provides a first attempt to define spatially localized convolutional filters for GNNs.

To further improve ChebyNet, Kipf et al. [100] proposed a simplified version of ChebyNet called as Graph Convolutional Network (GCN). GCN is the strictly-localized version of ChebyNet, where the assumptions of $K = 1$ and $\theta_0 = -\theta_1 = \theta$ is made. GCN formulation leads to

$$s_\theta * \mathbf{g} = \theta(\mathbf{I}_N + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{g}. \tag{3.21}$$

This filtered signal can be further normalized to map the eigenvalues of the convolution operator to the interval $[-1, 1]$, so that vanishing/exploding gradients in a deep architecture are avoided. The final formulation is given as

$$\mathbf{H} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}\mathbf{\Theta}, \tag{3.22}$$

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, $\tilde{\mathbf{D}}$ is the corresponding degree matrix and $\mathbf{\Theta} \in \mathbb{R}^{M\times F}$ are the filter parameters to be learned.

Compared to ChebyNet, GCN is significantly simpler, but its primary strength lies in its ability to form deeper models by stacking multiple GCN layers. By employing the operation $\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$, GCN prevents vanishing or exploding gradients in deep

models. This is achieved since eigenvalues of the $\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ operator are scaled to the interval $[-1, 1]$. As a result, stacking GCN layers allows for effective extraction of information from higher-order neighbors without the risk of losing information due to vanishing or exploding gradients during training.

GCN and ChebyNet stand as the two most pioneering works in spectral convolution-based GNNs, laying the foundation for subsequent advancements. Their success has inspired numerous efforts to enhance the effectiveness and efficiency of spectral convolution-based GNNs. For instance, Wu et al. [101] proposed Simple Graph Convolution (SGC), which simplifies the GCN architecture by reducing the complexity of stacked GCN layers. They argued that the primary innovation of GCN lies in its message-passing mechanism, making the non-linearity introduced by activation functions less critical. SGC replaces the repeated multiplication of $\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ across layers with a single matrix raised to a power. This approach not only reduces computational overhead but also preserves the effectiveness of GCN. Both SGC and GCN exhibit low-pass filter characteristics, enabling them to effectively reduce noise in node features.

While the low-pass filtering nature of GCN is advantageous in many applications, it can be suboptimal for tasks requiring higher-frequency information. To address this limitation, Levie et al. [102] proposed using Cayley Polynomials in place of Chebyshev Polynomials, while He et al. [103] introduced Bernstein Polynomials to design more robust spectral filters. These methods provide greater flexibility in defining spectral filters, though their advantages across diverse downstream tasks remain limited. In a different approach, Li et al. [104] proposed Adaptive GCN, which learns the graph structure in a task-driven way by adapting the Laplacian matrix during training. Adaptive GCN updates the underlying graph to better align with downstream tasks, demonstrating an effective method for customizing graph structures during the training process.

Although spectral convolution-based GNNs are primarily designed for node classification tasks on graphs with rich features, their message-passing nature allows them

to handle graph classification tasks involving multiple input graphs effectively. This versatility arises from their ability to extract information across each graph through the lens of the Laplacian matrix, even in the presence of variations in the Laplacian between graphs. The Laplacian governs the aggregation process, and updating node representations using the Laplacian is mathematically analogous to applying a second-order differentiation operation, enabling the model to capture intricate structural information within graphs.

3.1.2. Expressivity-based Graph Neural Networks

Earlier GNNs primarily focused on learning latent node representations by adapting the convolution operation to the graph domain. These models were designed to work effectively with graphs that have rich node features. While they could be utilized to learn latent graph representations, they were not primarily focused on this aspect, leaving it relatively underexplored. Although learning rich latent node representations may naturally lead to rich latent graph representations, this connection is not well studied, particularly for cases where graphs lack features. Consequently, the challenge of effectively handling featureless graphs remained an open question in the early GNN literature.

Despite the limited focus on latent graph representations within GNN research, extracting information solely from graph structures has been a well-studied problem for many years. Researchers developed algorithms to identify distinct graphs, some of which were applied to machine learning in the form of graph kernels. These kernels, which measure the similarity between graphs, became a powerful tool for extracting information from graphs. Among them, Weisfeiler-Lehman (WL) graph kernels [105] gained particular prominence. Based on the WL graph isomorphism test, WL graph kernels extract distinct information from graphs. Graph isomorphism, which tests whether two graphs are structurally identical, played a central role in the success of these kernels and is worth emphasizing due to its importance in graph representation learning.

Building on the principles of graph isomorphism, Xu et al. introduced a pioneering approach that integrated this concept into GNNs [106]. Their work resulted in the Graph Isomorphism Network (GIN), a novel architecture designed to map different graphs to distinct points in the latent space, thereby enabling the learning of optimal latent graph representations. By leveraging graph isomorphism, GIN offered increased expressive power and effectively addressed the challenge of featureless graphs. Moreover, Xu et al. demonstrated the inherent connection between message-passing algorithms and the WL isomorphism test, providing a theoretical basis for their approach. The development of GIN marked a pivotal moment in the evolution of expressivity-based GNNs, inspiring further research into the expressive power of GNNs for graph classification through the lens of graph isomorphism.

GIN is inspired by WL graph isomorphism test. The primary purpose of the WL test is to determine whether two graphs are isomorphic—that is, whether they share the same structural form. The key idea behind the WL test lies in iteratively updating node representations using an injective function. This function takes as input a node (or a tuple of nodes) along with the representations of its neighbors (or the neighbors of the tuple) and outputs a new representation. This process is repeated until the node representations stabilize, meaning no further changes occur. By comparing the final node representations of two graphs, the WL test can identify if they are isomorphic.

GIN implements the simplest form of the WL test, known as the 1-dimensional WL test or 'color refinement', which has proven to be highly effective in many practical scenarios. However, it is important to note that the WL test, including its 1-dimensional version, is not a definitive solution for all cases of graph isomorphism. Certain non-isomorphic graphs may be indistinguishable under the WL test, and this limitation applies to the 1-dimensional WL test as well, being its most basic and simplified version. Here is the algorithmic flow of the simple 1-dimensional WL test :

- All nodes initially have the same color. $c^0(v) = c^0(w)$ for all $v, w \in V$
- Each nodes' color is updated by the following : $c^t(v) = \phi(c^{t-1}(v), \{c^{t-1}(w) : w \in \mathcal{N}(v)\})$ where $\{.\}$ denotes the multiset and ϕ is an injective function.
- Color refinement stops when there is no update.

1-dimensional WL test and local GNNs share a conceptual similarity in how they iteratively process node information to capture structural differences in graphs. In the 1-dimensional WL test, nodes are assigned initial colors, and their colors are iteratively refined based on their own state and the multiset of neighbor colors. This process ensures that nodes with structurally different neighborhoods eventually receive distinct color representations. Similarly, local GNNs iteratively update node features using a message-passing mechanism, where each node aggregates information from its neighbors and combines it with its own state.

This similarity can be expressed mathematically. At each iteration, a node's representation is updated using an aggregation and update function, analogous to the color refinement in the 1-dimensional WL test. The update equation for a local GNN and color update function are given by

$$\mathbf{h}_v^k = \phi(\mathbf{h}_v^{k-1}, f(\{\mathbf{h}_u^{k-1} : u \in \mathcal{N}(v) : \text{Local GNN}, \tag{3.23}$$

$$c^t(v) = \phi(c^{t-1}(v), \{c^{t-1}(w) : w \in \mathcal{N}(v) : \text{Color Update}. \tag{3.24}$$

In this analogy, the node colors in the 1-dimensional WL test correspond to the node features in GNNs, while the multiset of neighbor colors represents the aggregated messages exchanged during message-passing.

Building on this connection, GIN is designed to achieve the same expressive power as the 1-dimensional WL test. Its goal is to learn latent representations of graphs by mapping distinct graph structures to unique latent embeddings. To accomplish this, GIN identifies three key requirements: injective aggregation, injective update, and injective graph pooling functions.

To formally define let $\mathcal{A} : \mathcal{G} \to \mathbb{R}^d$ be a GNN. With a sufficient number of GNN layers, $\mathcal{A}$ maps any graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ that the 1-dimensional WL test of isomorphism decides as non-isomorphic, to different embeddings if the following condition holds :

- Aggregation and update function, f and ϕ, are injective.
- Graph pooling function is injective

These properties ensure that the GNN can generate distinct latent node representations for structurally different nodes and produce unique graph-level embeddings for non-isomorphic graphs. To achieve this, GIN proposes the use of injective functions at all stages of the learning process.

GIN simplifies the design of injective aggregation and update functions by assuming features belong to a countable space. It utilizes a theoretical result stating that, there exists a function $f : \mathcal{X} \to \mathbb{R}^n$ so that for infinitely many choices of ϵ, including irrational numbers, $(1+\epsilon).f(c)+\sum_{x\in X} f(x)$ is unique for each pair (c, X) where $c \in X$, $X \subset \mathcal{X}$ is a multiset of bounded size and $\mathcal{X}$ denotes the countable input feature space. Moreover any other function $\hat{f}$ over such pairs can be decomposed as

$$\hat{f}(c, X) = \phi((1 + \epsilon).f(c) + \sum_{x\in X} f(x)), \tag{3.25}$$

for some function ϕ. This theorem states that node features, $(f(c))$, and their aggregated messages, $(\sum_{x\in X} f(x))$ can be processed with an injective function following the above formulation.

This decomposition allows GIN to implement an injective aggregation step by summing all messages, and an injective update step using the Universal Approximation Theorem. This theorem ensures that a multi-layer perceptron (MLP) can approximate any function, enabling the MLP to model the injective update function. Based on these principles, GIN's forward propagation rule is defined as

$$\mathbf{h}_v^k = MLP^k((1 + \epsilon^k).\mathbf{h}_v^{k-1} + \sum_{u\in\mathcal{N}(v)} \mathbf{h}_u^{k-1}), \tag{3.26}$$

where MLP^k denotes multilayer perceptron at layer k. While the parameter ϵ is

theoretically essential, practical experiments show that setting $\epsilon = 0$ does not impact performance. Consequently, GIN simplifies its propagation to

$$\mathbf{h}_v^k = MLP^k(\mathbf{h}_v^{k-1} + \sum_{u \in \mathcal{N}(v)} \mathbf{h}_u^{k-1}), \tag{3.27}$$

$$\mathbf{H} = MLP^k((\mathbf{A} + \mathbf{I})\mathbf{X}). \tag{3.28}$$

Unlike GIN, many popular GNN architectures, such as GCN and GraphSAGE, do not meet the conditions necessary to achieve the same expressive power as the 1-dimensional WL test. These models typically use non-injective aggregation functions, such as mean or max-pooling, and rely on simpler, one-layer perceptrons rather than the more expressive MLPs used in GIN. As a result, they are unable to distinguish certain non-isomorphic graphs that the WL test can identify. However, these architectures still perform well in tasks like node classification, where distinguishing entire graph structures is less important, especially when node features are rich. In these cases, mean-based aggregators are effective at capturing the distribution of neighbor features, enabling strong performance on node-level tasks. In contrast, GIN might be better suited for graph-level tasks, where distinguishing structural differences between entire graphs is crucial.

The theoretical foundations of the GIN are based on countable sets. However, in many practical scenarios, node features are drawn from continuous spaces, and finding an injective function for multisets from uncountable sets is considerably more challenging. To address this issue, Corso et al. [107] introduced Principal Neighbor Aggregation Network (PNA). PNA emphasizes that at least n aggregators are required to to distinguish multisets of size n when defined over $\mathbb{R}$. They further argue that moments of a multiset, such as mean, standard deviation, and higher-order moments, serve as valid examples of how n aggregators can be employed. This establishes the need for multiple aggregators to enhance the expressive power of the network.

While most GNN models rely on a single aggregator—mean for GCN, sum for GIN, and max for GraphSAGE—PNA combines several aggregators, including mean,

standard deviation, max, and min, to increase its expressive capacity. Additionally, PNA incorporates degree scalers that modify the aggregated values, enabling amplification or attenuation of the incoming messages. The propagation rule in PNA is given by

$$\mathbf{h}_i^k = \Phi^k(\mathbf{h}_i^{k-1}, \Psi^k(\{\mathbf{h}_v^{k-1}, v \in \mathcal{N}(v)\})), \tag{3.29}$$

where Φ^k is MLP and aggregation operation Ψ^k can use various functions defined as the

$$\Psi^k = \begin{bmatrix} I \\ S(\mathbf{D}, \alpha = 1) \\ S(\mathbf{D}, \alpha = -1) \end{bmatrix} \otimes \begin{bmatrix} \mu \\ std \\ max \\ min \end{bmatrix}, \tag{3.30}$$

where $\otimes$ denotes tensor product , $S(\mathbf{D}, \alpha) = (\frac{log(\mathbf{D}+1)}{\delta})^\alpha$ represents the scalers for aggregation functions, $\mathbf{D}$ is the node degree and δ is the normalization coefficient computed over training set. Mean, standard deviation, maximum, and minimum functions are employed as aggregator functions. The use of a logarithmic scaler is motivated by the fact that node degrees, when expressed in a linear scale, may not capture the full range of influence, especially when the node degrees vary significantly. This logarithmic scaling ensures that the influence of node degrees is more appropriately modeled.

Numerous studies have extended the exploration the expressive power of GNNs through the lens of graph isomorphism. Some approaches aimed to enhance expressivity by enriching node features. For example, Bouritsas et al. [108] demonstrated that incorporating substructure information—such as the number of cycles, triangles, and other motifs—into node features significantly improves GNN expressivity. Similarly, Sato et al. [109] proposed using random node features to increase the capacity of GNNs by providing additional information for distinguishing graph structures.

Other research has focused on surpassing the expressive power of the 1-dimensional WL test, which fails to identify certain basic substructures, such as triangles. Maron et

al. [110] developed a GNN composed of a maximal collection of permutation-invariant and equivariant linear layers to increase expressivity. Morris et al. [111] made the first attempt to generalize GNNs to the k-dimensional WL test by introducing k-GNNs, which correspond to the k-WL test. Following this, Maron et al. [112] introduced the first practical GNN that surpasses the 1-dimensional WL test in expressivity. Their model, known as 3-WL GNN, matches the expressiveness of the 3-WL test. However, while these methods are inspired by graph isomorphism, some, like the 3-WL GNN, treat graphs as higher-dimensional tensors, and their operations cannot be easily expressed using the standard aggregate-and-update framework of GNNs.

Another notable approach is that of Bevilacqua et al. [113], who proposed extracting bags of subgraphs from the input graph and processing them with expressive GNNs to further enhance model capacity. All of the aforementioned methods have shown promising results on datasets where structural differences in graphs are critical, such as social networks—where triangles play an important role—and molecular graphs, where cycles and substructures are essential.

However, this focus on graph substructures may not directly translate to brain networks, where it is unclear whether structures such as cycles and triangles carry meaningful information for extracting graph-level insights. Despite this, the enhanced expressive power of these advanced GNNs, particularly GIN, could still offer benefits when applied to brain network analysis.

3.1.3. Attention-based Graph Neural Networks

Attention mechanisms have emerged as a powerful and versatile tool for extracting relevant information from diverse data. By dynamically assigning importance weights to different inputs, they enable models to focus on the most relevant elements within complex and variable dependencies. This data-dependent weighted aggregation allows attention mechanisms to capture intricate relationships in a principled manner, making them an essential component in many machine learning domains.

Initially developed for sequence-based tasks, attention mechanisms revolutionized fields like machine translation [114,115], machine reading [116], and sentence representation learning [117]. The introduction of the self-attention mechanism within the Transformer architecture [115] was a landmark development. Self-attention, which models relationships within a single input sequence, demonstrated that attention alone could build robust models. Its ability to efficiently capture long-range dependencies and scalability set the foundation for state-of-the-art performance in sequence-based tasks.

Beyond sequence-based tasks, attention mechanisms have also made a significant impact in the vision domain. Models like Vision Transformers [118] leverage self-attention to capture relationships between patches of an image, enabling superior performance in image classification, object detection, and segmentation tasks. These advancements highlight the versatility of attention mechanisms in handling diverse data representations.

Inspired by their success in various models, attention mechanisms have been extended to graph-structured data, leading to the emergence of attention-based GNNs. This transition leverages the natural conceptual analogy between sequences and graphs: tokens and their dependencies in sequences correspond to nodes and their relationships (edges) in graphs. In graph contexts, attention dynamically computes latent node representations by weighting the contributions of neighboring nodes, enabling models to adaptively aggregate information.

The seamless integration of attention mechanisms into GNNs empowers these models to capture intricate structural and relational patterns. By focusing on relevant substructures and dynamically adjusting the aggregation process, attention-based GNNs provide a scalable and flexible solution for graph representation learning. These models are particularly effective in scenarios where adaptive modeling of node relationships is essential, offering a principled approach to processing graph-structured data.

Attention-based GNNs can be broadly categorized into two main approaches, reflecting different ways of leveraging attention mechanisms to model relationships in graph data: local attention-based GNNs and global attention-based GNNs. Local attention-based GNNs applies attention mechanisms to directly connected nodes in the graph, respecting the explicit graph topology. By focusing on neighbors defined by the graph structure, the model emphasizes local patterns and structural dependencies encoded in the edges. Neighbor-based attention is particularly suited for tasks where the graph's predefined structure is critical to capturing meaningful relationships. In contrast, global attention-based GNNs assumes a complete graph, where all node pairs are considered, allowing the model to infer latent relationships directly from the node features. This setup captures all pairwise relationships, making it ideal for scenarios where the graph structure is unknown, incomplete, or insufficiently informative. However, the flexibility of fully connected attention comes at the cost of increased computational complexity, as the number of pairwise relationships scales quadratically with the number of nodes.

These two approaches reflect a trade-off between leveraging predefined structural knowledge and allowing the model to learn relationships in a data-driven manner. While the local attention-based GNNs emphasize the explicit graph topology, global attention-based GNNs offer greater flexibility by capturing potential hidden dependencies that might not be encoded in the graph structure.

Local Attention-based GNNs restrict the attention mechanism to operate only on the neighborhood of each node. These models are a subset of local GNNs, where the aggregation process considers only the node's direct neighbors. The general aggregation function for local Attention-based GNNs can be expressed as

$$\mathbf{h}_{\mathcal{N}(v)}^{k} = \Psi_k(\{\mathbf{h}_u^{k-1}\}_{u\in\mathcal{N}(v)}). \tag{3.31}$$

Here, Ψ_k is a function that implements the attention mechanism, dynamically determining the contribution of each neighboring node u to the aggregation for node v at layer k.

The pioneering work by Veličković et al. [119] introduced the Graph Attention Network (GAT), which was the first model to incorporate attention mechanisms into GNNs. GAT is a quintessential example of a local Attention-based GNN. The forward propagation of GAT is given by

$$\mathbf{H} = \sigma(\hat{\mathbf{A}}\mathbf{\Theta}\mathbf{X}), \tag{3.32}$$

where $\mathbf{\Theta}$ stores parameters for projection and $\hat{\mathbf{A}} \in \mathbb{R}^{N\times N}$ is the matrix that contains the attention coefficients. The attention coefficients are computed as follows

$$\hat{a}_{i,j} = \frac{e^{(\text{LeakyReLU}(\alpha^\top \mathbf{\Theta h_i} + \alpha^\top \mathbf{\Theta h_j}))}}{\sum_{k\in\mathcal{N}(i)\cup\{i\}} e^{\text{LeakyReLU}(\alpha^\top \mathbf{\Theta h_i} + \alpha^\top \mathbf{\Theta h_k})}}, \tag{3.33}$$

where $\hat{a}_{i,j}$ is the attention coefficient between node i and node j, $\alpha^k \in \mathbb{R}^{2\cdot d_{attn}}$ is learnable vector, $\mathbf{h_i}$ is the representation of node i and Leaky-ReLU is introduced as non-linearity function to prevent cancelling exponentials when $k = i$. GAT learns spatially localized representations since it only computes the relations between neighbors. To enrich the latent representation, multi-head architecture can employed into GAT. Each head can be modeled with independent parameters, $\mathbf{\Theta}^k$ and ${\alpha^k}^\top$. Resulting formulation for attention coefficients in case is

$$\hat{a}^k_{i,j} = \frac{e^{(LeakyReLU({\alpha^k}^\top \mathbf{\Theta}^k \mathbf{h_i} + {\alpha_k}^T \mathbf{\Theta}^k \mathbf{h_j}))}}{\sum_{k\in\mathcal{N}(i)\cup\{i\}} e^{LeakyReLU({\alpha^k}^\top \mathbf{\Theta}^k \mathbf{h_j} + {\alpha_k}^T \mathbf{\Theta}^k \mathbf{h_k})}}, \tag{3.34}$$

where $\mathbf{\Theta}^k$ and ${\alpha^k}^\top$ are the learnable parameters of attention head k and $\hat{a}^k_{i,j}$ is the attention coefficient between node i and node j computed at head k. Therefore multi-head GAT can be expressed as

$$\mathbf{H}^k = \sigma(\hat{\mathbf{A}}^k \mathbf{\Theta}^k \mathbf{X}), \tag{3.35}$$

where $\mathbf{H}^k$ is the latent node representation matrix generated by head k. By concatenation all output matrices along head dimension, final latent node representations can be obtained.

GAT employs a variation of the self-attention mechanism where the query, key, and value projections are coupled, i.e., they are equal to each other. In addition, GAT introduces a new parameter, $\mathbf{a}^k$, to compute attention coefficients, alongside a weight matrix that models the key and query projections. In contrast, the classical self-attention mechanism, as used in Transformers, decouples the query, key, and value

projections to model relationships more flexibly. Specifically, given query, key, and value projection matrices $\mathbf{\Theta^q}$, $\mathbf{\Theta^k}$, $\mathbf{\Theta^v}$ the classical self-attention mechanism operates on node features as follows

$$\hat{\mathbf{A}}^k = \text{softmax}(\frac{(\mathbf{\Theta^q X})^T(\mathbf{\Theta^k X})}{\sqrt{d}}), \tag{3.36}$$

$$\hat{\mathbf{H}}^k = \hat{\mathbf{A}}^k(\mathbf{\Theta^v X}), \tag{3.37}$$

where d is the scaling factor used to stabilize gradients when the softmax produces large values. If this mechanism is constrained to operate only on neighboring nodes, it takes the following form

$$\hat{a}_{i,j} = \frac{e^{(\mathbf{\Theta^q x}_i)^T(\mathbf{\Theta^k x}_j)}}{\sum_{\mathcal{N}(i)\cup\{i\}} e^{(\mathbf{\Theta^q x}_i)^T(\mathbf{\Theta^k x}_k)}}, \tag{3.38}$$

$$\mathbf{Z}^k = \sigma(\hat{\mathbf{A}}^k\mathbf{\Theta}^k(\mathbf{\Theta^v X})). \tag{3.39}$$

Here, node features are projected into query and value representations using the query ($\mathbf{\Theta^q}$) and value ($\mathbf{\Theta^v}$) projection matrices, while neighboring node features are treated as keys and projected using the key projection matrix ($\mathbf{\Theta^k}$) . GAT assumes equal projection matrices, while the formulation above is more similar to the classical self attention mechanism employed in Transofermers. GAT simplifies this setup by assuming equal projection matrices for the query, key, and value, whereas the formulation above aligns more closely with the classical self-attention mechanism used in Transformers. A similar decoupled approach was explored by Shi et al. [120], showing that separating the query, key, and value projections can enhance the capacity of GAT. This variant will be referred to as GAT-Transformer. Dwivedi et al. [121] also extended the concept of Transformers to graphs, resulting in the Graph Transformer model. They treated node features as an input sequence, similar to how tokens are handled in Transformers, and applied a positional encoding scheme to incorporate structural information from the graph. Specifically, they proposed using the eigenvectors of the graph Laplacian as positional encodings, as these eigenvectors capture distance-aware information and provide a natural way to embed the graph structure into the model.

Another notable improvement over the original GAT architecture was proposed by Brody et al. [122], who introduced the GATv2 model. They addressed a key limitation of GAT, the attention coefficients are unconditioned on the query node. GATv2 resolves this issue with a simple yet effective modification to the attention computation, as given by

$$\hat{a}_{i,j} = \frac{e^{(\alpha^\top \text{LeakyReLU}(\mathbf{\Theta h_i} + \mathbf{\Theta h_j}))}}{\sum_{k \in \mathcal{N}(i) \cup \{i\}} e^{(\alpha^\top \text{LeakyReLU}(\mathbf{\Theta h_i} + \mathbf{\Theta h_k}))}}, \tag{3.40}$$

where the attention coefficients explicitly depend on both the query node i and the neighboring node j, ensuring a more dynamic and conditioned computation.

Regardless of the specific attention mechanism, local attention-based GNNs extract information by dynamically adjusting edge weights instead of relying on a static adjacency matrix, as in many convolution-based approaches. This dynamic re-weighting allows the model to capture local dependencies in a data-driven manner, effectively highlighting the most important local features and enhancing the capacity of GNNs.

Global Attention-based GNNs extend the attention mechanism to allow nodes to attend to any other node in the graph, even if there is no direct connection between them. These models fall under the category of global GNNs, where the aggregation process considers all possible node pairs in the graph. The general aggregation function for global Attention-based GNNs can be expressed as

$$h^k_{\mathcal{N}(v)} = \Psi_k(\{h^{k-1}_u\}_{u \in (V)}). \tag{3.41}$$

Here, Ψ_k is a function that implements the attention mechanism, dynamically determining the contribution of each other node u in the set of all nodes to the aggregation for node v at layer k.

One of the most notable attempts to integrate global attention into GNNs is the work by Rampasek et al., who introduced the GraphGPS model [123]. To build a powerful and scalable model, they enhanced node features with structural and positional information derived from random walks and Laplacian matrices of the graph.

By combining the Transformer architecture with local GNNs, they were able to extract both local and global information from the graph, enhancing the learned latent graph representation.

In addition, Kreuzer et al. [124] utilized spectral graph theory to develop a Global Attention-based GNN known as the Spectral Attention Network (SAN). SAN also uses the first m eigenvectors to build initial positional encodings but proposes learning these encodings through a linear mapping over the layers. Furthermore, SAN treats the edges from the real graph and the added edges differently in the attention computation, allowing for more efficient capture of both local and global information.

Global attention-based GNNs can be viewed as instances of Transformers, where the tokens correspond to the nodes in the graph. While they are capable of extracting global information, local information remains crucial and more specifically tailored for GNNs.

3.2. Structure Function Discrepancy

Current GNN-based models for brain networks primarily focus on assigning clinical labels to brain networks, operating under the assumption that networks within the same clinical population share significant structural and functional similarities. While this assumption holds to some extent, it overlooks the critical role of personal variability in understanding disease progression and patient-specific patterns. This limitation results in suboptimal performance and a lack of personalized insights into the disruptions caused by ADD.

Importantly, most existing studies evaluate brain networks by comparing them to those of others with the same clinical label. While this approach can be informative, it fails to account for the significant personal variability inherent in brain networks. Instead, evaluating a brain with respect to its own internal reference—focusing on its self-consistency—can provide more personalized and meaningful insights. This person-

alized approach acknowledges that each brain exhibits unique structural and functional patterns, emphasizing that the critical factor is the internal harmony within the brain rather than its similarity to others.

The structure-function relationship serves as an ideal metric for such personalized assessment, as it captures the interplay between two distinct yet interconnected modalities within the same brain. However, most multi-modal studies fail to fully model this complexity. Structural and functional networks are typically treated as complementary sources of information, fused superficially, or analyzed independently. These approaches overlook the interdependence between structure and function and fail to account for the progressive dissimilarity between these networks during ADD progression.

To address these limitations, we propose the brain connectome structure-function discrepancy learning (sfDLN) model, a GNN-based framework designed to quantify the structure-function discrepancy as a biomarker for ADD severity. This discrepancy measures the divergence between structural and functional networks within an individual brain, offering a personalized metric for evaluating disease progression. By leveraging each brain's self-consistency as a reference, the sfDLN model mitigates the confounding effects of personal variability and enables a more nuanced understanding of ADD.

In healthy brains, structural connectivity provides a foundation for functional synchronization, reflecting the intricate interplay between anatomical organization and brain activity. However, this structure-function relationship changes during ADD progression [90]. We conjecture that the discrepancy—or mismatch—between structural and functional connectivity patterns increases as cognitive impairment advances from SCI towards ADD. Furthermore, we hypothesize that this structure-function relationship serves as a decisive and discriminative feature for diagnosing and staging ADD.

The sfDLN model adopts a siamese architecture, comprising two identical GNN-based branches to process structural and functional networks separately. The architecture of the sfDLN model is depicted in Figure 3.1.

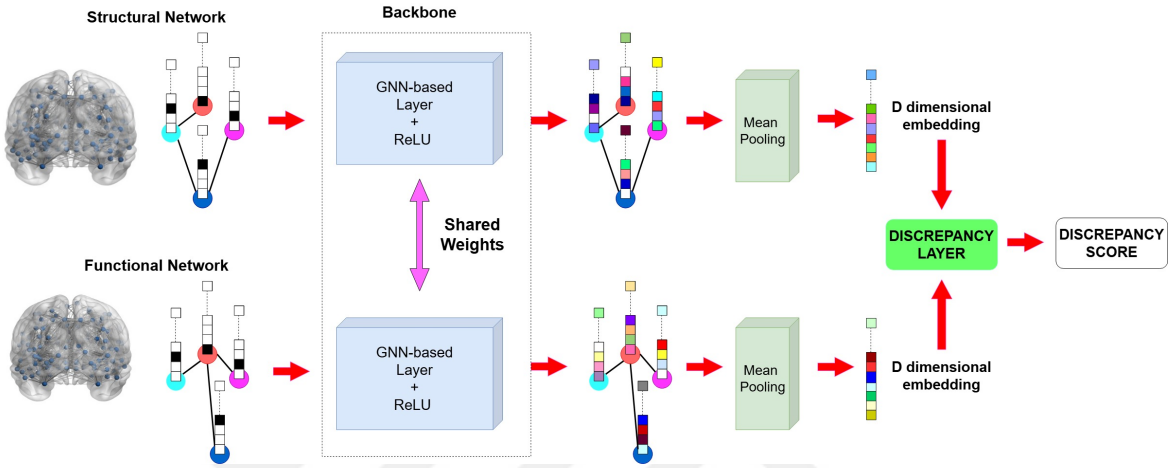

Figure 3.1. sfDLN Architecture [125].

Any GNN model can be used as a backbone of sfDLN to encode structural and functional networks. For simplicity lets assume GCN is used in sfDLN architecture. sfDLN generates latent graph representations as the following

$$\mathbf{H} = \text{ReLU}(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}\mathbf{\Theta}), \tag{3.42}$$

$$\mathbf{z} = \frac{1}{N}\sum_{i\in V}\mathbf{h}_i, \tag{3.43}$$

where ReLU is used as non-linear activation function, defined as the Typically, the ReLU function is used as the activation, which is defined as $\text{ReLU}(x) = \max(0, x)$, and mean pooling is used at the output of GNN branches to produce d dimensional graph embeddings ($\mathbf{z}$). Then, the scaled dot product of sNET embeddings ($\mathbf{z}_s$) and fNET embeddings ($\mathbf{z}_f$) is used to generate the bounded structure-function discrepancy score (γ) as

$$\gamma = 1 - sigm\left((\mathbf{z}_s \cdot \mathbf{z}_f)/Z\right), \tag{3.44}$$

where $sigm()$ is the sigmoid function and Z is a scaling factor. Sigmoid function is

used to bound the structure-function discrepancy score and scaling factor is used to ensure training convergence.

The sfDLN model is trained by minimizing the following loss function, which simultaneously minimizes the intra-class variance and maximizes the inter-class distance

$$J = \lambda_{var}(\text{std}^p + \text{std}^n) + \max(0, m - (\mu^p - \mu^n)) \ . \tag{3.45}$$

Above, μ and std are the mean and the standard deviation of the discrepancy scores (γ) across the corresponding classes. Meanwhile, λ_{var} and m are hyperparameters that control the influence of the standard deviation and margin score in distinguishing the means of the populations, respectively. The superscripts p and n refer to the positive and negative classes, where the positive class corresponds to the stage with a higher structure-function discrepancy. Our hypothesis posits that later stages of cognitive decline correspond to the positive class, based on the assumption that the mismatch between structure and function increases as ADD progresses.

By explicitly modeling the structure-function discrepancy, focusing on each brain's unique internal reference, and hypothesizing its progressive deterioration during dementia, the sfDLN framework offers a novel perspective on ADD diagnostics. It provides deeper, individualized insights into ADD progression, paving the way for more effective diagnostics and personalized therapeutic interventions.

3.3. Disentangled Attention Graph Neural Network

Mesoscale analysis of brain networks is crucial for understanding the progression of ADD, as the brain consists of distinct, non-overlapping subnetworks, each with specific structural and functional roles. The brain consists of distinct, non-overlapping subnetworks, each with specific structural and functional roles, and these subnetworks are targeted differently as the disease progresses. Understanding how these subnetworks change across stages is key to diagnosing and tracking ADD, highlighting the importance of analyzing the brain at the mesoscale level.

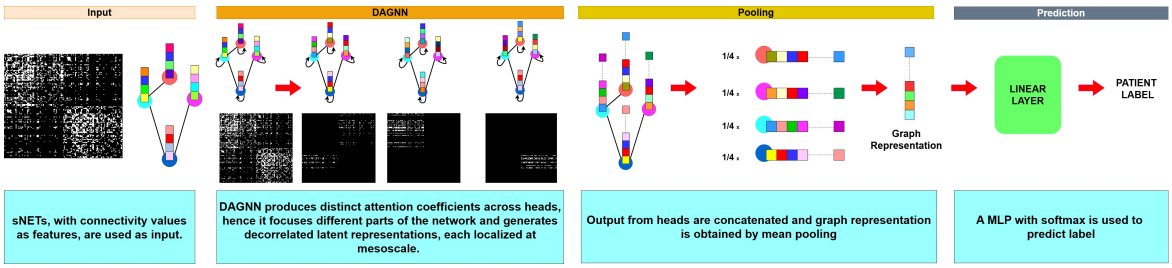

Figure 3.2. DAGNN [126]. Each head is independent from each other and compute a set of different attention coefficients along with node features. Node features from heads are concatenated. Mean pooling is performed to obtain graph-level representation. A MLP layer with softmax is used to predict label.

Attention-based GNNs have gained significant attention due to their ability to highlight important connections within the brain. This makes them capable of identifying subnetworks through their multi-head attention architecture. Multi-head mechanism allows for the exploration of different parts of the brain simultaneously, enhancing the model's ability to find relevant subnetworks. However, we have identified a key limitation: although each attention head operates independently, these models often focus on similar brain regions when classifying ADD. This can lead to the overlooking of subnetworks that may be crucial for ADD progression. To address this issue, we propose a novel model, the Disentangled Attention Graph Neural Network (DAGNN), which is designed to more effectively extract information from distinct subnetworks, using a unique disentanglement loss.

DAGNN model, which we propose here, is depicted in Figure 3.2. DAGNN generates latent node representations based on a attention-based GNN and obtain latent graph representation by a mean pooling

$$\mathbf{H} = \mathrm{ReLU}(\mathrm{GNN}(\mathbf{X}, \mathbf{A})), \tag{3.46}$$

$$\mathbf{z} = \frac{1}{N} \sum_{i \in V} \mathbf{h}_i. \tag{3.47}$$

GNN could be any attention-based GNN with multiple heads. For simplification, lets

assume multi-head GAT is used in DAGNN. Resulting DAGNN propagation can be expressed as

$$\hat{a}_{i,j}^{k} = \frac{e^{(LeakyReLU(\alpha^{k^\top}\mathbf{\Theta}^{k}\mathbf{h_i}+\alpha^{k^\top}\mathbf{\Theta}^{k}\mathbf{h_j}))}}{\sum_{k\in\mathcal{N}(i)\cup\{i\}} e^{LeakyReLU(\alpha^{k^\top}\mathbf{\Theta}^{k}\mathbf{h_j}+\alpha^{k^\top}\mathbf{\Theta}^{k}\mathbf{h_k})}}, \tag{3.48}$$

$$\mathbf{H}^{k} = \text{ReLU}(\hat{\mathbf{A}}^{k}\mathbf{\Theta}^{k}\mathbf{X}), \tag{3.49}$$

$$\mathbf{H} = \text{concat}(\mathbf{H}^{k}), \tag{3.50}$$

$$\mathbf{z} = \frac{1}{N}\sum_{i\in V}\mathbf{h}_i, \tag{3.51}$$

where ReLU is used as non-linear activation function and concat operation concatenates all latent node representations produced by each head along head dimension to form final latent node representation matrix, $\mathbf{H}$.

We identified a limitation in the multi-head GAT, where the attention coefficients computed by each head, denoted as $\hat{\mathbf{A}}^{k}$, are surprisingly similar. This observation points to a potential drawback: GAT may not fully leverage its multi-head structure, leading to the neglect of important variations across heads. To address this issue, we propose a novel loss function called disentanglement loss. This loss aims to differentiate each head by maximizing the distance between the attention coefficient matrices produced by each head. The disentanglement loss is defined as

$$\mathcal{L}_{dis} = ReLU(m - \frac{1}{N_{pairs}}\sum_{b=1}^{N_B}\sum_{j=1}^{N_H}\sum_{k=j+1}^{N_H} dist(\hat{\mathbf{A}}^{b,j}, \hat{\mathbf{A}}^{b,k})), \tag{3.52}$$

where $\hat{\mathbf{A}}^{b,j}$ represents the attention coefficients of the b-th patient in a batch produced by j-th head of GAT, N_B is the batch size, N_{pairs} is the number of attention coefficients pairs in the batch, N_H is the number of heads of GAT, $dist$ is the function for measuring distance and m is the margin hyperparameter. his loss function aims to push the pairwise distance between attention coefficient matrices to a value as large as m, promoting differentiation between the attention coefficients produced by different heads.

We propose to use the mean of the L1 distance as our distance measure for the disentanglement loss, as it has two main advantages.

- Boundedness : The column-wise L1 distance is bounded by 2, as the attention coefficient matrix is column-wise stochastic. This property ensures that the margin hyperparameter m remains bounded, regardless of the number of attention heads. This simplifies the optimization process by not requiring adjustments based on the number of heads.
- Sparsity Promotion : The L1 distance encourages sparsity in the attention coefficients, as it penalizes the similarity between attention coefficient matrices more strongly when their elements are closer to each other. This promotes the learning of distinct, meaningful attention patterns across different heads.

The proposed distance function is defined as

$$dist(\hat{\mathbf{A}}^{b,j}, \hat{\mathbf{A}}^{b,j}) = \frac{1}{N} \sum_{n=1}^{n=N} \sum_{l=1}^{n=N} |\mathbf{C}_{n,l}|, \tag{3.53}$$

where $\mathbf{C} = \mathbf{A}^{b,j} - \mathbf{A}^{b,k}$ represents the difference between the attention coefficient matrices. The L1 distance between two matrices is computed column-wise, and then the average column-wise L1 distance is taken as the overall distance between the two matrices. This column-wise computation ensures that the resulting maximum distance m is bounded by 2.

We trained our model with a combination of classification loss and disentanglement loss. Overall loss function can be written as

$$\mathcal{L}_{total} = \mathcal{L}_{CE} + \lambda_{dis}\mathcal{L}_{dis}, \tag{3.54}$$

where λ_{dis} is hyperparameter that balances classification and disentanglement loss.

4. EXPERIMENTS AND RESULTS

4.1. Connectome Construction and Dataset

Our dataset consists of sNETs and fNETs from 88 volunteers. To construct brain networks, DWI, resting-state functional MRI (rs-fMRI) and T1-weighted (T1w) MRI data were acquired with written consent in a single session using the Philips Achieva 3T MRI system (Netherlands) with a 32-channel head coil at Istanbul University, Istanbul Faculty of Medicine, Neuroimaging Unit of Hulusi Behçet Life Sciences Research Laboratory[1] . Here are the imaging parameters for the each modality :

- DWI: Data was acquired with $TE/TR = 92\,ms$ / $9032\,ms$, $FOV = 200 \times 236\,mm^2$ at $2.27\,mm$ isotropic voxel size. 120 weighting gradients with a maximum gradient strength of $40\,mT/m$, $200\,mT/m/ms$ slew rate were obtained at 6 shells in q-space using a single-shot, pulse-gradient spin echo (PGSE) EPI sequence.
- rs-fMRI: data was collected by using the Fast Field Echo (FFE) technique with MS mode, single-shot EPI, and real-time reconstruction. The imaging parameters were $TE/TR = 30\,ms/3000\,ms$, $FOV = 212 \times 199\,mm^2$, at $3.31\,mm$ isotropic voxel size.
- T1w : 3D FFE pulse sequence with multi-shot TFE (Turbo Field Echo) imaging mode is used with $TE/TR = 3.8\,ms/8.3\,ms$, $FOV = 220 \times 240\,mm^2$, at $1.0\,mm$ isotropic voxel size.

Each subject's T1w-MRI, fMRI, and DWI volumes were isotropically resampled at $1.5\,mm$ resolution and co-registered using the FreeSurfer, FSL, and Tortoise toolboxes. The co-registered volumes were manually verified by the neurologists on board. T1w-MRI volumes were parcelled with 148-parcel Destrieux atlas ($N = 148$) [5] to delineate the cortical regions. Following parcellation, sNET and fNET were constructed

[1]Istanbul University, Istanbul Faculty of Medicine, Ethics Committee Approval: 877/30.05.2014; Bogazici University, Ethics Committee Approval: 2014-1/17.02.2014

for each subject.

sNET, $\mathcal{G}_s = \{V, E_s, \mathbf{A}_s\}$, was constructed using deterministic principal diffusion direction (PDD) tractography, which was based on 4th-order Runge-Kutta integration. The tractography process involved the following parameters [127]:

- Step size: $0.7\,mm$
- Maximum Curvature: $35°$
- Minimum fractional anisotropy (FA) threshold as stopping criteria : 0.15
- Minimum fiber length : $20\,mm$
- Number of seeds per voxel: 30

The edge weights, a_{ij}, were defined as

$$a_{ij} = \frac{2}{Vol_i + Vol_j} \sum_k W_{ik} W_{jk}, \tag{4.1}$$

where Vol_i is the volume of the i^{th} parcel and W_{ik} quantifies the association between the i^{th} parcel and the k^{th} fiber as described in [19] :

$$W_{ik} = \sum_{r_j \in N_i} \frac{1}{(2\pi)^{\frac{3}{2}} * \text{std}^3} e^{\frac{-|r_j - e_k|^2}{2*\text{std}^2}}, \tag{4.2}$$

$$|r_j - e_k| < 2 * \text{std}, \tag{4.3}$$

where N_i is the parcel i, e_k is the end point of fiber k, r_j is the position of voxel j inside the parcel. std value is chosen as 0.155.

The edge weights of sNETs are computed with the normalization as shown in the Equation 4.1. This normalization is advantageous for deep learning models, which perform well on smoother intervals. However, as shown in Figure 1, these normalized weights often result in very low values.

These low connectivity values may hinder the GNN's ability to learn meaningful patterns. To address this, three different sNET variants are considered:

- Original sNET: Does not change the edge weights, A is used directly.
- Binary sNET: Applies binarization to the edge weights, setting all positive weights to 1: $A_{bin} = (A[A > 0] = 1)$
- Log-Scaled sNET: Applies a logarithmic transformation to the edge weights such that the resulting weights are linear in terms of the logarithmic scale, $A_{ls} = \frac{(\log_{10} A - min(\log_{10} A))}{(max(\log_{10} A) - min(\log_{10} A))}$

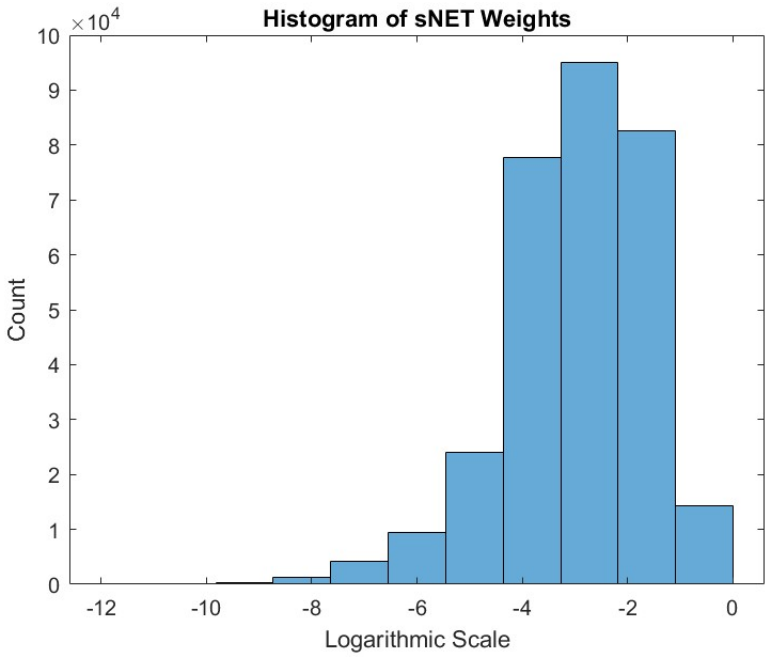

Figure 4.1. Histogram of sNET Edge Weights.

Binarization highlights the existance of edge weight and significantly increases the edge weight values compared to the original sNET. However, this approach may lead to a loss of information about the scale of connectivity values. To retain scale information while benefiting the amplified edge weights, Log-Scaled sNETs are introduced.

On the other hand, fNET is constructed based on BOLD signals. These BOLD signals are four-dimensional time-series data defined over the brain's voxels. To summarize the BOLD signal for a particular brain region, we compute the average of the time-series data across all the voxels within that region. This averaged time-series

represents the BOLD signal for the entire region, capturing the overall functional activity of the region. These region-wise BOLD signals are then used to construct the functional network, where edges represent the functional connectivity between different brain regions, reflecting the temporal correlations between the regions' activity.

fNET is derived from BOLD signals of the brain regions, denoted as $\mathbf{B} \in \mathbb{R}^{N \times T}$ where T is the number of time points, and let $\mathbf{b}_i$ is the BOLD signal of node i. Each $\mathbf{b}_i$ is centered to have zero mean.

Traditional approaches construct fNETs based on Pearson or partial correlation coefficients. While simple and popular, these methods result in fully connected graphs, which degrade GNN performance. Alternatively, sparse fNETs can be constructed using graph learning methods, leveraging smoothness assumptions to optimize graph structures directly from BOLD signals. These learning methods assumes that the BOLD signals should vary smoothly on the learned graph.

Therefore three fNET construction methods are evaluated: Pearson fNET, partial fNET and Learned fNET, we refer it to as ℓNET. Pearson and Partial fNET are correlation-based fNETs while ℓNET are learned through the BOLD signals.

- Pearson fNET : Edges, (a_{ij}) represent the sample Pearson correlation coefficient between $\mathbf{b}_i$ and $\mathbf{b}_j$

$$a_{ij} = \frac{\sum_{t=1}^{T} \mathbf{b}_{i,t}\mathbf{b}_{j,t}}{\sqrt{\sum_{t=1}^{T}(\mathbf{b}_{i,t})^2}\sqrt{\sum_{t=1}^{T}(\mathbf{b}_{j,t})^2}}. \tag{4.4}$$

- Partial fNET : Edges (a_{ij}) corresponds to the partial correlation coefficient of $\mathbf{b}_i$ and $\mathbf{b}_j$. Formally defining

$$\Sigma_{i,j} = \sum_{t=1}^{T} \mathbf{b}_{i,t}\mathbf{b}_{j,t}, \tag{4.5}$$

$$a_{ij} = \frac{\Sigma_{i,j}^{-1}}{\sqrt{\Sigma_{i,i}^{-1}}\sqrt{\Sigma_{j,j}^{-1}}}, \tag{4.6}$$

where Σ is the sample covariance matrix. Partial correlation is derived from inverse of sample covariance matrix called as precision matrix.

- ℓNET : ℓNET are sparse graphs which is the solution of an optimization problem aiming to find optimal graph from signal with the smoothness assumption [128]. The optimization problem can be defined as

$$\mathbf{A} = \underset{\mathbf{A}\in\mathcal{A}}{\arg\min} \left(\mathrm{Tr}\left(\mathbf{B}^T\mathbf{L}\mathbf{B}\right) + f(\mathbf{A})\right), \tag{4.7}$$

$$f(\mathbf{A}) = -\mathbf{1}^T \log(\mathbf{A}\mathbf{1}) + ||\mathbf{A}||_F^2, \tag{4.8}$$

where $\mathbf{L}$ is the Laplacian constructed from $\mathbf{A}$ and $\mathbf{1}$ is the column vector of 1s, $||\cdot||_F$ is the Frobenius norm. $f(\mathbf{A})$ controls the positivity of the node degrees and the network sparsity, respectively. The first term in Equation 4.7 imposes smoothness of the BOLD signals $\mathbf{B}$ on ℓNET, as is evident from the identity $\mathbf{b}^T\mathbf{L}\mathbf{b} = \sum_{i,j} a_{i,j}(\mathbf{b}_i - \mathbf{b}_j)^2$ favoring strong connections between similar nodes. The above optimization problem was solved as described in the literature [129] which employs convex optimization to the objective function.

Clinical diagnostic labels of the subjects are given following the NIA-AA guidelines [130–132]. by the neurologists involved in the study. The resulting dataset is referred to as the CAPA dataset. A detailed summary of the dataset is provided in Table 4.1.

Table 4.1. CAPA Dataset Description.

CAPA DATASET	
Characteristic	**Value**
Label Distribution	18 ADD, 46 MCI, 24 SCI
Age	69.8±8.3 ADD, 62.0±9.7 MCI, 56.6±9.7 SCI
Gender	10M/8F ADD, 23M/23F MCI, 12M/12F SCI
Number of Brain Regions	148

4.2. Diagnostics using GNNs

GNNs hold significant promise for diagnosing different stages of ADD by analyzing structural and functional brain networks. Despite the proposal of numerous models, the literature notably lacks a detailed comparative study of GNNs applied to brain networks for ADD diagnosis. This gap underscores the need for systematic evaluation to better understand their potential and limitations. The aim of this section is to conduct a comprehensive comparative study through a series of experiments designed to justify and explore the utility of GNNs for ADD diagnostics. These experiments focus on classifying ADD from early stages : ADD-SCI and ADD-MCI using the CAPA dataset. The analysis is structured into three categories: sNET experiments, exploring structural brain networks; fNET experiments, focusing on functional brain networks; and multi-modal experiments, integrating structural and functional modalities.

To measure and compare the performance of the models, three metrics are used as :

- Accuracy : The percentage of correct predictions made by the model, calculated as the ratio of correctly predicted samples to the total number of samples.
- F1-Score : Harmonic mean of precision and recall, defined as

$$\text{F1-Score} = \frac{2TP}{2TP + FP + FN}, \tag{4.9}$$

  where TP denotes the true positives, FP is the false positives and the FN is the number of false negatives. F1-Score is more sensitive to the positive classes, in our case ADD. It may help us to quantify cases where ADD subjects are scarce.
- Area Under Curve (AUC) : The AUC is computed from the Receiver Operating Characteristics (ROC) curve, which plots the true positive rate (sensitivity) against the false positive rate (1-specificity) across different decision thresholds. The AUC represents the degree of separability between the classes, with higher values indicating better discrimination between positive and negative classes.

For the ADD-SCI classification, the CAPA dataset consists of 42 subjects, including 18 ADD and 24 SCI cases. Since this dataset is relatively balanced, all the metrics proposed above offer comparable reliability in evaluating model performance.

In contrast, for the ADD-MCI classification, the CAPA dataset is more imbalanced, with only 18 of the 64 subjects classified as ADD, while the remaining 46 subjects are MCI. In this case, the F1-Score serves as a better performance indicator compared to Accuracy or AUC, as it places greater emphasis on the positive class (ADD), making it particularly useful in scenarios with class imbalance.

To ensure a robust foundation for this comparative study, the initial phase involves setting an MLP-based baseline that does not utilize graph structure. This baseline serves as a reference for determining which network best represents the corresponding modality.

Subsequently, the focus shifts to systematically comparing GNNs both with each other and with the baseline. This comparative study evaluates the performance of various GNN models in diagnosing ADD stages. The experiments then progress to identifying optimal configurations for the proposed architecture, taking into account critical factors such as feature selection, model selection, pooling methods, and potential enhancements drawn from recent literature.

This structured approach not only addresses the current gap in the literature but also provides valuable insights into the design and application of GNNs for ADD diagnostics. By systematically analyzing structural, functional, and multi-modal brain networks, this section aims to establish a robust methodological framework and provide valuable insights into the application of GNNs to brain networks for ADD diagnosis.

4.2.1. sNET Experiments

In the experiments, two classification tasks are performed: ADD-SCI and ADD-MCI. Performance on these tasks is quantified using accuracy, F1-score, and AUC metrics. Cross-validation is employed to evaluate performance, and each cross-validation experiment is repeated 20 times to mitigate the impact of random seeds. For the ADD-SCI task, 21-fold cross-validation is utilized due to the limited sample size. It is observed that using larger folds reduces the generalizability of models for ADD-SCI classification. The 21-fold cross-validation strikes a balance between improved generalization and runtime efficiency, being twice as fast as leave-one-out cross-validation while yielding comparable results. For ADD-MCI classification tasks, 7-fold cross-validation is applied. Detailed description for both experiments including hyperparameters are given in the Table 4.2.

Table 4.2. Summary of sNET Experimental Configurations for Classification Tasks.

Description	**ADD-SCI**	**ADD-MCI**
Label Distribution	18 ADD/24 SCI	18 ADD/46 SCI
Number of Folds	21	7
Batch Size	6	6
Number of Epochs	100	100
Learning Rate	0.001	0.001
Optimizer	Adam	Adam
Weight Decay	0.0001	0.001

Structural brain networks represent the brain's physical connectivity by modeling the existence and strength of connections between distinct brain regions. These networks are typically represented as sparse graphs, where nodes correspond to brain regions and edges encode structural connectivity values. The inherent sparsity of sNETs makes them particularly well-suited for GNNs, which can effectively leverage the graph structure to propagate and aggregate information across connected nodes.

One of the primary challenges in analyzing sNETs is the lack of intrinsic node features. Since the input data consists solely of graph representations, meaningful features must either be derived from the graph structure itself or the connectivity values must be used as node features in conjunction with the defined graph topology. This feature engineering step is critical for enabling GNNs to fully capture the structural properties of the brain network.

To justify the application of GNNs for structural network analysis, it is important to demonstrate that their node aggregation mechanisms provide improved classification or prediction performance compared to models that do not explicitly utilize graph structures. A carefully designed baseline model, which processes features independently of the graph structure, is essential to establish a fair comparison and highlight the benefits of GNN-based approaches.

In the sNET experiments, these objectives will be explored through the following steps:

- Baseline Model Evaluation: A baseline model that does not utilize graph structure and relies solely on node features will be implemented. This baseline will provide a reference point for assessing the effectiveness of different feature construction methods.
- GNN Model Comparison: Various GNN layers with diverse configurations will be evaluated by integrating them into the baseline model (replacing one layer). The goal is to identify the GNN models that yields significant improvement over the baseline. The top-performing models will be shortlisted for further investigation.
- GNN Architecture Optimization: The shortlisted GNN models will undergo optimization through experiments with different setups, such as varying the number of layers and directly applying the GNN without an intermediate baseline model. The best-performing architecture and configuration will be selected for subsequent steps.

- Pooling Method Analysis: The selected GNN architecture will be tested with various pooling strategies to identify the most suitable approach for generating latent graph representation.
- Model Refinement: Building on insights from the literature, the selected GNN architecture will be enhanced with minor modifications to test potential performance improvements.

The baseline model should be invariant to node ordering, ensuring that its output does not depend on the specific ordering of nodes in the input. GNNs naturally achieve this invariance through their equivariant message-passing mechanisms and invariant graph pooling layers, which allow them to produce the same graph representations regardless of node ordering. A similar principle is well established in the context of set-based models, where elements are treated as unordered entities. When no specific relationships between nodes are assumed, the nodes of a graph can be viewed as elements of a set, and the task of designing an invariant baseline model becomes analogous to defining a model that represents a function invariant to permutations.

Zaheer et al. demonstrated that any permutation-invariant function can be constructed by independently processing the features of each element (or node) and combining their outputs through an order-invariant operation, such as summation [133]. Based on this theoretical foundation, we propose the DeepSet model as our baseline.

DeepSet uses a MLP to independently process node features. This ensures equivariance at the node level. A pooling operation, such as summation, is then applied over node features to achieve full invariance to node ordering. This combination of equivariant processing and invariant pooling allows DeepSet to focus solely on feature learning without leveraging graph connectivity, making it a fair comparison to GNNs.

The propagation mechanism of a K-layer DeepSet for obtaining latent graph representations is defined as

$$\mathbf{H} = \sum_{i \in V} f^K(\mathbf{X}), \tag{4.10}$$

$$\mathbf{z} = \sum_{i \in V} \mathbf{h}_i, \tag{4.11}$$

where f is an MLP, and K denotes the number of stacked MLP layers. Specifically, f^K is a composition of K MLPs

$$f^K = f_K \circ f_{K-1} \circ \cdots \circ f_1. \tag{4.12}$$

Each MLP f^k maps $\mathbf{h}_i^{k-1} \in \mathbb{R}^{d_{k-1}}$ to $\mathbf{h}_i^k \in \mathbb{R}^{d_k}$ as follows

$$h_i^k = \sigma(\mathbf{\Theta}_2^k \sigma(\mathbf{\Theta}_1^k h_i^{k-1})), \tag{4.13}$$

where $\mathbf{\Theta}_1^k \in \mathbb{R}^{d_k \times d_{k-1}}$, $\mathbf{\Theta}_2^k \in \mathbb{R}^{d_k \times d_k}$ are the learnable parameters of the MLP, and σ is an activation function. Typically, the ReLU function is used as the non-linear activation function. Finally, to predict labels from the latent graph representation z, a task-specific MLP, f_{task} is applied

$$\hat{y} = f_{task}(\mathbf{z}). \tag{4.14}$$

DeepSet not only provides a robust baseline for comparison with GNNs but also aids in understanding how to select features from sNETs. While DeepSet does not perform node-wise aggregation, it still conducts feature-wise aggregation. This allows us to determine which feature selection method is the most effective.

A common practice in feature selection for brain networks is the direct use of connectivity values as features [87]. Since structural networks lack additional information beyond connectivity, these values serve as a reliable and straightforward baseline. In earlier studies, alternative network measures, such as node degree and centrality, were proposed as node features. However, these measures can also be derived from connectivity values. While such features may provide supplementary information, it is likely that the underlying networks can inherently capture these distinctions directly

from the connectivity values.

Table 4.3. Performance of 1-Layer and 2-Layer DeepSet Models on original, binary and log-scaled sNETs.

ADD-SCI Classification						
Network	**1-Layer DeepSet(DS^1)**			**2-Layer DeepSet(DS^2)**		
	Acc	F1	AUC	Acc	F1	AUC
Original	65.9±2.4	56.6±3.3	68.2±3.0	64.5±2.2	55.0±3.9	67.6±2.9
Binary	84.0±2.6	**81.6±3.1**	87.3±1.9	**85.1±2.1**	**83.0±2.5**	87.7±1.7
Log-Scaled	**84.4±2.7**	81.4±3.4	**89.1±1.9**	83.9±2.6	80.9±3.2	**90.0±1.6**
ADD-MCI Classification						
Network	**1-Layer DeepSet(DS^1)**			**2-Layer DeepSet(DS^2)**		
	Acc	F1	AUC	Acc	F1	AUC
Original	71.4±1.2	1.4±4.6	45.0±1.0	68.5±2.6	24.7±5.1	52.8±2.5
Binary	71.7±1.7	41.1±4.4	74.5±2.4	71.1±1.4	41.0±4.0	**75.5±2.0**
Log-Scaled	**74.4±2.1**	**49.5±4.8**	**74.7±1.8**	**73.8±2.1**	**50.0±4.1**	75.2±1.5

As illustrated in the connectome construction, there types of sNET are considered: original sNET, binary sNET and log-scaled sNET. Therefore, three different feature selection methods are employed in the baseline experiments.

Both 1-layer and 2-layer DeepSet models are tested on ADD-SCI and ADD-MCI classification tasks for three variants. 1-Layer and 2-layer Deepset models will be referred to as $\mathbf{DS}^1$ and $\mathbf{DS}^2$, respectively. For both models, the hidden dimensions are set to 128, i.e, $d_1 = d_2 = 128$.

Table 4.3 compares the performance of original, binary and log-scaled sNET. Both Binary and Log-Scaled sNETs demonstrate significantly better performance than Original sNETs for both tasks. This suggests that amplifying the edge weights improves the representational quality of sNETs, enabling more effective feature learning.

The Log-Scaled sNET demonstrates an improvement over the Binary sNET, particularly in classifying ADD-MCI. However, the significant performance boost of Binary sNET compared to the Original sNET underscores the importance of highlighting the presence of edges. The further enhancement seen with Log-Scaled sNET in ADD-MCI classification suggests that the scale of edge weights provides valuable information that can refine the model's learning even more. Additionally, the increase in performance with the addition of layers is generally modest.

With the baseline model established and initial experiments conducted for sNET selection, we now compare the most common GNN models to evaluate their performance against the baseline model, DeepSet. The primary goal is to replace the second DeepSet layer with different GNN models and assess whether they improve results, thereby demonstrating the effectiveness of the message-passing scheme in GNNs.

To ensure fair comparisons, we systematically replace the second DeepSet layer with various GNN models while keeping all other parameters constant. As observed in earlier experiments (Table 4.3), the second DeepSet layer does not enhance performance. Therefore, if replacing it with a GNN layer results in improved outcomes, it validates the benefits of message aggregation in GNNs. Importantly, for each GNN model, we evaluate multiple configurations and select the best-performing one to serve as a baseline for subsequent experiments. This approach ensures that each GNN model is tested under optimal conditions, allowing for meaningful comparisons.

The construction of node features ($\mathbf{X}$) and the underlying graph ($\mathbf{A}$) plays a crucial role in the performance of GNNs. When node features are absent, two common approaches are:

- One-hot Encoding: Setting $\mathbf{X} = \mathbf{I}$
- Connection Profiles: Using binary or log-scaled adjacency matrices as features.

To illustrate the limitations of one-hot encoding, consider the propagation equations for a 1-layer DeepSet and a 1-layer GIN

$$\mathbf{H} = \sigma(\mathbf{\Theta}_2^k \sigma(\mathbf{\Theta}_1^1 \mathbf{X})), \tag{4.15}$$

$$\mathbf{H} = MLP((\mathbf{A}_{bin} + \mathbf{I})\mathbf{X}). \tag{4.16}$$

Using one-hot encodings ($\mathbf{X} = \mathbf{I}$), GIN is equivalent to 1-Layer DeepSet using binary connection profile, binary sNET, with self connections are added ($\mathbf{X} = \mathbf{A}_{bin} + \mathbf{I}$). This demonstrates that one-hot encoding alone fails to capture meaningful neighbor information in 1-layer GNNs. Therefore we will stick to using connectivity values as features in the experiments.

The following GNN models, chosen for their diversity in design, were tested: GraphSAGE, ChebyNet, GCN, GIN, and GAT. Specific configurations for each model are as follows:

- GraphSAGE:

  Aggregators: Mean, Max, and LSTM.
- ChebyNet:

  Polynomial Orders : $K = 2$ and $K = 3$

  Adjacency Matrices : Binary and log-scaled
- GCN:

  Adjacency Matrices : Binary and log-scaled
- GIN:

  No configuration
- GAT:

  Number of Heads : 1 and 4

Each GNN replaces the second DeepSet layer, and its performance is evaluated on binary and log-scaled sNETs under consistent conditions. These architectures will be referred to as "DS + GNN," where GNN denotes the specific model being used.

The DS + GNN architecture obtains the latent graph representations as follows

$$\mathbf{z} = \sum_{i \in V} \text{GNN}(f^1(\mathbf{X})), \tag{4.17}$$

where f^1 represents the first DeepSet layer and GNN denotes the GNN propagation mechanism, which varies depending on the specific GNN model (e.g., GCN, GIN). This experimental setup facilitates direct comparisons of the effectiveness of GNN architectures in leveraging message passing. Additionally, analyzing adjacency matrix variants (binary vs. log-scaled) and aggregators (e.g., mean, max) provides insights into optimal configurations for sNET-based tasks. The GNN model comparison experiments begin with the GraphSAGE. Recall GraphSAGE propagation as

$$\begin{aligned} \mathbf{h}^k_{\mathcal{N}(v)} &= \Psi_k(\mathbf{h}^{k-1}_u, \forall u \in \mathcal{N}(v)), \\ \mathbf{h}^k_v &= \sigma(\mathbf{\Theta}^k . CONCAT(\mathbf{h}^{k-1}_v, \mathbf{h}^k_{\mathcal{N}(v)})), \\ \mathbf{h}^k_v &= \mathbf{h}^k_v / ||\mathbf{h}^k_v||_2, \end{aligned} \tag{4.18}$$

where Ψ_k function can vary depending on the type of aggregation used. It is expressed as

$$\Psi_k = f(\sigma(\{\mathbf{\Theta}^k \mathbf{h}^k_u\}), \tag{4.19}$$

where f is the aggregation operation, which could be either *mean*, *max*, or an *LSTM*-based operation. The specific GraphSAGE variant is referred to based on the aggregation function used; for example, GraphSAGE with mean aggregation is called Mean GraphSAGE.

In the experiments, the normalization step in Equation 4.10 is omitted, and the learnable parameters are not used in the aggregation functions for Mean GraphSAGE and Max GraphSAGE. Consequently, Mean GraphSAGE is expressed as

$$\mathbf{h}^k_i = \mathbf{\Theta_1} \mathbf{h}^{k-1}_i + \mathbf{\Theta_2} \frac{1}{|N(i)|} \sum_{j \in N(i)} \mathbf{h}^{k-1}_j, \tag{4.20}$$

where $|N(i)|$ denotes the number of neighbors of node i, i.e., the cardinality of $N(i)$.

Max GraphSAGE is expressed as

$$\mathbf{h}_i^k = \mathbf{\Theta_1}\mathbf{h}_i^{k-1} + \mathbf{\Theta_2}\max(\{\mathbf{h}_j^{k-1}\}), \tag{4.21}$$

where max function operates along the feature dimension.

Table 4.4. Performance of DS + GraphSAGE Configurations Compared to 2-Layer DeepSet on Binary and Log-Scaled sNETs.

ADD-SCI Classification						
Network	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
Mean	85.0±3.0	82.8±3.6	89.4±2.8	88.3±1.5	86.5±1.7	91.5±1.2
Max	**86.7±3.5**	**83.6±4.5**	**92.1±2.5**	**89.1±2.3**	**87.0±3.0**	**92.9±1.7**
LSTM	83.3±1.5	81.0±2.0	87.6±2.0	83.8±3.8	80.7±4.5	89.3±3.5
DS^2	85.1±2.1	83.0±2.5	87.7±1.7	83.9±2.6	80.9±3.2	90.0±1.6
ADD-MCI Classification						
Network	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
Mean	74.0±1.9	**47.2±5.0**	78.7±1.5	76.2±2.5	**52.3±5.0**	78.4±2.0
Max	**76.0±2.7**	45.1±7.8	**81.7±2.8**	**77.3±3.0**	51.2±6.7	**80.7±3.3**
LSTM	72.3±1.9	43.0±4.6	75.7±2.6	74.0±1.1	48.7±1.8	74.7±2.8
DS^2	71.1±2.3	41.0±4.0	75.5±2.0	73.8±2.1	50.0±4.1	75.2±1.5

Three GraphSAGE variants—DS + Mean GraphSAGE, DS + Max GraphSAGE, and DS + LSTM GraphSAGE—were evaluated alongside the DeepSet model. The results are summarized in Table 4.4. Among the tested GraphSAGE variants:

- DS + Max GraphSAGE consistently outperformed the DeepSet model across all metrics for both binary and log-scaled sNETs.
- DS + Mean GraphSAGE improved over DeepSet in most scenarios but fell short for ADD-SCI classification on binary sNETs.

- DS + LSTM GraphSAGE underperformed compared to DeepSet, likely due to its reliance on ordered sequences, which is not well-suited for the unordered nature of graph nodes.

Table 4.5. Performance of ChebyNet Configurations Compared to 2-Layer DeepSet and DS + Max GraphSAGE on Binary and Log-Scaled sNETs.

ADD-SCI Classification						
Model	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
K = 2, binary	85.9±2.5	**83.7±2.9**	88.8±1.9	85.9±2.6	83.3±3.3	90.0±1.3
K = 3, binary	84.1±3.3	81.5±4.0	89.2±2.0	86.4±2.5	83.8±3.3	91.2±1.3
K = 2, weighted	N/A	N/A	N/A	85.0±2.8	81.9±3.6	89.8±1.9
K = 3, weighted	N/A	N/A	N/A	87.5±2.1	85.2±3.6	91.0±1.5
DS^2	85.1±2.1	83.0±2.5	87.7±1.7	83.9±2.6	80.9±3.2	90.0±1.6
DS + Max GraphSAGE	**86.7±3.5**	83.6±4.5	**92.1±2.5**	**89.1±2.3**	**87.0±3.0**	**92.9±1.7**
ADD-MCI Classification						
Model	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
K = 2, binary	72.7±2.3	42.9±6.0	77.3±2.0	75.3±2.1	**51.3±4.5**	76.6±1.6
K = 3, binary	72.3±2.5	41.7±6.1	77.3±1.9	73.4±2.3	48.6±4.4	78.3±2.2
K = 2, weighted	N/A	N/A	N/A	75.7±2.3	51.2±5.4	77.3±2.4
K = 3, weighted	N/A	N/A	N/A	75.3±1.7	50.8±3.9	78.3±1.9
DS^2	71.1±2.3	41.0±4.0	75.5±2.0	73.8±2.1	50.0±4.1	75.2±1.5
DS + Max GraphSAGE	**76.0±2.7**	**45.1±7.8**	**81.7±2.8**	**77.3±3.0**	51.2±6.7	**80.7±3.3**

These results highlight the potential of GraphSAGE, particularly in leveraging log-scaled sNETs for enhanced classification performance, and emphasize the importance of the aggregation mechanism. DS + Max GraphSAGE emerged as the most effective model, demonstrating robust performance across tasks and configurations. Consequently, DS + Max GraphSAGE will be adopted as a baseline for subsequent

experiments involving other GNN architectures. This iterative evaluation and selection process ensures that future comparisons are based on the best-performing configurations for each GNN model. Building on this analysis, the next experiment delves into the performance of ChebyNet, which employs polynomial based spectral approximation for neighborhood aggregation. Recall ChebyNet propagation as:

$$\hat{\mathbf{X}} = [\mathbf{X},\ \mathbf{L}_{norm}\mathbf{X},\ T_2(\mathbf{L}_{norm})\mathbf{X} \dots\ T_K(\mathbf{L}_{norm})\mathbf{X}]\mathbf{\Theta}. \tag{4.22}$$

Experiments were conducted with polynomial orders $K = 2$ and $K = 3$. For $K = 2$, ChebyNet aggregates messages from immediate neighbors, while $K = 3$ enables message aggregation from 2-hop neighbors, capturing a broader neighborhood context.

In addition, two different normalized Laplacian construction is experimented :

- Binary : The normalized Laplacian $\mathbf{L}$ is constructed using a binarized adjacency matrix $\mathbf{A}_{bin}$
- Weighted : The normalized Laplacian $\mathbf{L}$ is constructed using a log-scaled adjacency matrix $\mathbf{A}_{ls}$

This setup provides insights into how different feature selection and edge weighting schemes influence DS + ChebyNet's performance. The results for binary and log-scaled sNETs are presented in Table 4.3. (K=2,binary) row means that the ChebyNet is used with $K = 2$ and Laplacian $\mathbf{L}$ is constructed by binary adjacency matrix while (K=3,weighted) row means that the ChebyNet is used with $K = 3$ and Laplacian $\mathbf{L}$ is constructed by log-scaled adjacency matrix.

DS + ChebyNet's performance with various configurations was evaluated and compared with the 2-Layer DeepSet model. The results are summarized in Table 4.3. Based on these results, the following observations were made:

- ChebyNet configurations generally demonstrate improved results compared to the 2-Layer DeepSet model for both ADD-SCI and ADD-MCI classifications,

indicating its ability to effectively leverage graph structure.

- The polynomial order K affects performance. For ADD-MCI classification, $K = 2$ slightly outperforms $K = 3$ with log-scaled features, while for ADD-SCI classification, $K = 3$ performs marginally better than $K = 2$ with log-scaled features. For binary features, $K = 2$ consistently achieves better results than $K = 3$ for both tasks.
- In terms of feature representation, log-scaled features tend to outperform binary features for both tasks across all configurations, indicating that edge weight information enhances the model's performance.
- Weighted Laplacians yield better results than binary Laplacians when $K = 3$, suggesting that higher-order relationships are more sensitive to edge weights than immediate connections.
- DS + ChebyNet configurations generally perform less effectively, particularly when compared to Max GraphSAGE. This indicates that while ChebyNet effectively captures neighborhood information, its expressiveness might be limited, especially distinguishing ADD and SCI.

Overall, the results indicate that DS + ChebyNet demonstrates competitive performance compared to the 2-Layer DeepSet model, particularly benefiting from its ability to capture higher-order relationships through polynomial filters. The choice of polynomial order (K) and feature representation significantly impacts its effectiveness, with log-scaled features and lower polynomial orders generally yielding better results. Despite these strengths, DS + ChebyNet struggles to match the performance of DS + Max GraphSAGE. Building on this analysis, the next experimnt delves into the performance of GCN, which employs a first-order spectral approximation for neighborhood aggregation and provides a complementary perspective on leveraging graph structure for these tasks. Recall GCN propagation as

$$\hat{\mathbf{X}} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}\mathbf{\Theta}. \tag{4.23}$$

Table 4.6. Performance of DS + GCN Configurations Compared to 2-Layer DeepSet and DS + Max GraphSAGE on Binary and Log-Scaled sNETs.

ADD-SCI Classification						
Network	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
Binary	85.3±2.0	83.0±2.4	88.7±1.5	86.9±2.9	84.5±3.7	91.0±2.5
Weighted	-	-	-	87.5±2.3	85.2±2.9	91.1±2.1
DS^2	85.1±2.1	83.0±2.5	87.7±1.7	83.9±2.6	80.9±3.2	90.0±1.6
DS + Max GraphSAGE	**86.7±3.5**	83.6±4.5	**92.1±2.5**	**89.1±2.3**	**87.0±3.0**	**92.9±1.7**
ADD-MCI Classification						
Network	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
Binary	71.8±1.7	41.7±4.9	76.8±1.9	74.4±2.5	50.3±4.5	77.8±2.2
Weighted	-	-	-	74.6±2.1	49.8±5.0	77.3±2.0
DS^2	71.1±2.3	41.0±4.0	75.5±2.0	73.8±2.1	50.0±4.1	75.2±1.5
DS + Max GraphSAGE	**76.0±2.7**	**45.1±7.8**	**81.7±2.8**	**77.3±3.0**	**51.2±6.7**	**80.7±3.3**

GCN can be viewed as a simplified version of ChebyNet, where the polynomial order K is fixed to 2. Consequently, experiments involving varying polynomial orders were not conducted. Instead, evaluations were performed using binary and weighted adjacency matrices. The experimental procedure was aligned with that of ChebyNet. DS + GCN performance with binary and weighted adjacency matrix was evaluated and compared with the 2-Layer DeepSet and DS + Max GraphSAGE model. The results are summarized in Table 4.6. Based on these results, the following conclusions were made:

- DS + GCN outperforms DeepSet across all configurations and classification tasks, showcasing its effectiveness in leveraging graph structures.
- For ADD-SCI classification, weighted adjacency matrices achieve better performance than binary graphs. However, for ADD-MCI classification, weighted

graphs perform slightly worse than binary graphs. The varying influence of edge weights across tasks may reflect differences in task complexity or the underlying characteristics of the classification problems.

- Log-scaled features consistently outperform binary features across all configurations and tasks, confirming their superior ability to represent node and edge information.
- Despite its strengths, DS + GCN does not match the performance of DS + Max GraphSAGE, which achieves the highest results in both classification tasks. This trend is consistent with the observations for DS + ChebyNet, emphasizing that while DS + GCN offers meaningful improvements over simpler models like DeepSet, it cannot surpass DS +Max GraphSAGE.

The experimental results highlight GCN's ability to effectively leverage graph structures, consistently outperforming the DeepSet model across various configurations. Its sensitivity to adjacency matrix design and feature representation underscores the critical role of task-specific preprocessing and input choices. However, similar to ChebyNet, GCN falls short of surpassing Max GraphSAGE, which remains the top-performing model. Building on these findings, the subsequent experiments focus on the GIN, which introduces a more expressive approach to graph representation through its injective aggregation mechanism.

GIN employs a unique propagation mechanism, defined as

$$\mathbf{H} = MLP((\mathbf{A}_{bin} + \mathbf{I})\mathbf{X}). \tag{4.24}$$

The key distinctions of GIN compared to other models are its use of the summation aggregation function and its reliance on a MLP. Experimental results for GIN, alongside comparisons to DeepSet and DS + Max GraphSAGE, on binary and log-scaled sNETs are presented in Table 4.7. Here are the takes from the experiments :

- DS + GIN outperforms DeepSet in both classification tasks with log-scaled features. For binary features, GIN exhibits lower performance, indicating its reliance on enriched feature representations like log-scaled features to maximize its po-

tential.

- In ADD-MCI classification, DS + GIN exhibits a more nuanced performance profile. While it does not surpass DS + Max GraphSAGE in overall metrics such as accuracy and AUC, it achieves the highest F1-score for both binary and log-scaled features, outperforming both DS + Max GraphSAGE and DeepSet in this critical metric.

Table 4.7. Performance of DS + GIN compared to 2-Layer DeepSet and Max DS + GraphSAGE on Binary and Log-Scaled sNETs.

ADD-SCI Classification						
Network	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
GIN	83.2±3.6	80.2±4.4	88.6±2.1	85.6±3.0	83.0±3.6	90.0±2.4
DS^2	85.1±2.1	83.0±2.5	87.7±1.7	83.9±2.6	80.9±3.2	90.0±1.6
DS + Max GraphSAGE	**86.7±3.5**	**83.6±4.5**	**92.1±2.5**	**89.1±2.3**	**87.0±3.0**	**92.9±1.7**
ADD-MCI Classification						
Network	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
GIN	74.9±3.0	**46.5±7.4**	78.7±2.4	77.2±2.9	**54.0±7.6**	79.0±2.5
DS^2	71.1±2.3	41.0±4.0	75.5±2.0	73.8±2.1	50.0±4.1	75.2±1.5
DS + Max GraphSAGE	**76.0±2.7**	45.1±7.8	**81.7±2.8**	**77.3±3.0**	51.2±6.7	**80.7±3.3**

Despite its limited performance in ADD-SCI classification, DS + GIN establishes a strong baseline for ADD-MCI classification. Given the class imbalance inherent in ADD-MCI classification, where F1-score serves as the most critical evaluation metric, DS + GIN demonstrates a clear advantage in this context. Consequently, DS + GIN will be utilized as an additional baseline for the final GNN experiment involving the GAT.

GAT introduces an attention mechanism into the message-passing framework, defined as follows

$$\mathbf{H} = \sigma(\hat{\mathbf{A}}^k \mathbf{\Theta}^k \mathbf{X}), \tag{4.25}$$

$$\hat{a}^k_{i,j} = \frac{e^{(LeakyReLU(\alpha^{k\top} \mathbf{\Theta}^k \mathbf{h_i} + \alpha^{k\top} \mathbf{\Theta}^k \mathbf{h_j}))}}{\sum_{k \in \mathcal{N}(i) \cup \{i\}} e^{LeakyReLU(\alpha^{k\top} \mathbf{\Theta}^k \mathbf{h_j} + \alpha^{k\top} \mathbf{\Theta}^k \mathbf{h_k})}}. \tag{4.26}$$

Table 4.8 presents the performance of DS + GAT, with configurations using one attention head (1-Head) and four attention heads (4-Head), compared to DeepSet, DS + Max GraphSAGE, and DS + GIN on binary and log-scaled sNETs for ADD-SCI and ADD-MCI classification tasks. The key conclusions from this experiment are as follows:

- The 4-Head GAT model demonstrates a significant advantage over DeepSet, highlighting the effectiveness of attention mechanisms in message-passing frameworks. 4-Head GAT configuration surpasses Max GraphSAGE in most metrics for both tasks when using binary features. With log-scaled features, 4-Head GAT achieves the highest F1-score and AUC performance for ADD-SCI classification and secures a clear margin in F1-score for ADD-MCI classification.
- The superior performance of 4-Head GAT compared to 1-Head GAT underscores the benefits of employing multiple independent attention mechanisms, which enhance the model's representational capacity.
- Additionally, both GAT configurations exhibit minimal performance discrepancies between binary and log-scaled features for ADD-MSCI classification. This consistency indicates GAT's robustness to feature scaling in this task.

GAT's ability to dynamically assign importance to node features via attention mechanisms significantly enhances its performance compared to DeepSet and other GNN models, especially for tasks with imbalanced data such as ADD-MCI classification. The superior F1-scores achieved by DS + 4-Head GAT demonstrate its capacity

to better handle minority classes, which is critical in clinical applications. Furthermore, its robustness across binary and log-scaled features sets it apart from other GNN models, making it a highly versatile and reliable approach for sNET-based classification tasks.

Table 4.8. Performance of DS + GAT compared to 2-Layer DeepSet, DS + Max GraphSAGE and DS + GIN on Binary and Log-Scaled sNETs.

ADD-SCI Classification						
Network	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
1-Head	85.0±3.0	82.5±3.7	90.8±1.7	88.1±2.4	86.8±2.9	93.0±1.6
4-Head	**87.5±2.4**	**85.4±2.9**	**93.4±1.5**	88.1±2.1	**87.0±2.4**	**94.2±1.4**
DS^2	85.1±2.1	83.0±2.5	87.7±1.7	83.9±2.6	80.9±3.2	90.0±1.6
DS + Max GraphSAGE	86.7±3.5	83.6±4.5	92.1±2.5	**89.1±2.3**	87.0±3.0	92.9±1.7
DS + GIN	83.2±3.6	80.2±4.4	88.6±2.1	85.6±3.0	83.0±3.6	90.0±2.4
ADD-MCI Classification						
Network	**Binary Features**			**Log-Scaled Features**		
	Acc	F1	AUC	Acc	F1	AUC
1-Head	75.9±3.3	54.1±6.9	79.4±2.0	75.8±3.8	54.7±6.7	78.6±1.8
4-Head	**77.0±2.0**	**55.9±3.9**	79.8±1.8	76.6±2.6	**55.7±4.4**	79.7±1.3
DS^2	71.1±2.3	41.0±4.0	75.5±2.0	73.8±2.1	50.0±4.1	75.2±1.5
DS + Max GrapSAGE	76.0±2.7	45.1±7.8	**81.7±2.8**	**77.3±3.0**	51.2±6.7	**80.7±3.3**
DS + GIN	74.9±3.0	46.5±7.4	78.7±2.4	77.2±2.9	54.0±7.6	79.0±2.5

From the GNN comparison experiments, three models emerged as top-performing candidates: Max GraphSAGE, GIN, and 4-Head GAT. These models will be further investigated to identify the optimal architecture that maximizes classification performance.

Building on the previous experiments, Max GraphSAGE, GIN, and 4-Head GAT were selected for further analysis. These models were tested in combination with a 1-layer DeepSet, forming a hybrid DS + GNN architecture. In each experiment, only 1-layer GNNs were used to precisely measure the contribution of the GNN component. However, while this configuration allows for a more controlled evaluation, the proposed DS + GNN architecture may not represent the optimal setup. To thoroughly investigate these top-performing models, two configurations will be considered:

- Plain GNN Architectures: In this setup, GNN models were directly applied to process input without a preceding DeepSet layer. Different numbers of GNN layers, specifically 1 and 2, were evaluated. These architectures are denoted as GNN^k where GNN refers to Max GraphSAGE, GIN, or 4-Head GAT, and k represents the number of GNN layers. The plain GNN architectures are defined as

$$\mathbf{z} = \sum_{i \in V} GNN^k(\mathbf{X}). \tag{4.27}$$

  Here, GNN^k represents a stack of k GNN layers.
- Adding another layer to DS + GNN architecture: This setup extends the previously explored DS + GNN architecture by adding an additional GNN layer. These configurations are denoted as $DS + \text{GNN}^2$, where DS represents the initial DeepSet layer, and GNN^2 denotes a 2-layer GNN. This experiment aimed to determine whether increasing the depth of the GNN component would enhance performance. However, to mitigate the well-known issue of oversmoothing in GNNs [134], experiments were limited to two GNN layers.

Experimental results for both GNN^k and DS + GNN^2 configurations, along with the top-performing DS + GNN architectures from previous experiments, can be found in Table 4.9. Log-scaled features were used in these experiments due to their superior performance over binary features.

Table 4.9. Performance of Plain GNN and GNN^k architectures compared to DS + Max GraphSAGE, DS + GIN and DS +4-Head GAT on Binary and Log-Scaled sNETs.

ADD-SCI Classification			
Network	**Log-Scaled Features**		
	Acc	F1	AUC
Max GraphSAGE[1]	85.9±3.7	82.8±4.9	91.3±1.9
GIN[1]	80.0±3.3	76.5±4.2	86.1±3.1
4-Head GAT[1]	83.8±3.2	80.4±4.6	90.8±1.6
Max GraphSAGE[2]	86.0±2.9	82.7±3.9	92.5±2.0
GIN[2]	77.7±6.0	74.5±7.0	82.8±5.2
4-HeadGAT[2]	83.3±3.5	80.5±4.1	89.7±1.8
DS + Max GraphSAGE[2]	88.3±3.1	86.1±3.9	91.8±1.7
DS + GIN[2]	88.0±2.1	86.0±2.5	91.7±1.9
DS + 4-Head GAT[2]	87.6±3.0	85.6±3.6	91.6±1.6
DS + Max GraphSage	**89.1±2.3**	87.0±3.0	92.9±1.7
DS + GIN	85.6±3.0	83.0±3.6	90.0±2.4
DS + 4-Head GAT	88.1±2.1	**87.0±2.4**	**94.2±1.4**
ADD-MCI Classification			
Network	**Log-Scaled Features**		
	Acc	F1	AUC
Max GraphSAGE[1]	**79.1±2.1**	53.9±4.1	**82.0±2.5**
GIN[1]	72.7±4.4	48.7±9.1	74.6±3.5
4-Head GAT[1]	75.3±4.7	52.8±9.0	77.9±2.0
Max GraphSAGE[2]	78.1±2.1	52.2±6.4	81.0±2.5
GIN[2]	72.8±4.8	48.9±8.4	73.7±1.8
4-Head GAT[2]	73.9±3.9	50.3±8.4	76.4±1.6
DS + Max GraphSAGE[2]	75.6±3.1	46.2±8.3	79.9±3.3
DS + GIN[2]	75.2±2.5	51.6±5.2	78.6±1.8
DS +4-Head GAT[2]	75.0±2.9	49.9±5.8	76.2±1.5
DS + Max GraphSage	77.3±3.0	51.2±6.7	80.7±3.3
DS + GIN	77.2±2.9	54.0±7.6	79.0±2.5
DS + 4-Head GAT	76.6±2.6	**55.7±4.4**	79.7±1.3

The summarized findings are as follows:

- Adding another GNN layer to DS + GNN architectures negatively affected results. It shows that these GNN models extract most of the information within the immediate neighbors.

- GNN^k underperforms compared to the DS + GNN^k models in most setups. It shows that encoding the connectivity values before using GNN positively affects the performance. Only the Max GraphSAGE improved the results in the ADD-MCI classification for accuracy and AUC.
- DS + 4-Head GAT emerged as the best architecture for log-scaled sNET processing, outperforming the other architectures.

Table 4.10. Performance of GAT variants on Log-Scaled sNETs.

ADD-SCI Classification			
Network	**Log-Scaled Features**		
	Acc	F1	AUC
DS + 4-Head GAT	**88.1±2.1**	**87.0±2.4**	**94.2±1.4**
DS + 4-Head GATv2	88.0±2.4	86.2±2.8	92.3±1.4
DS + 4-Head TransformerGAT	87.1±2.2	85.1±2.6	91.8±1.2
ADD-MCI Classification			
Network	**Log-Scaled Features**		
	Acc	F1	AUC
DS + 4-Head GAT	**76.6±2.6**	**55.7±4.4**	**79.7±1.3**
DS + 4-Head GATv2	76.5±2.0	55.4±3.1	78.9±2.0
DS + 4-Head TransformerGAT	75.3±1.9	52.9±3.9	79.0±1.8

The DS + 4-Head GAT model proved to be the most effective architecture for processing log-scaled sNETs. To further enhance its performance, the attention mechanism will be explored to refine this architecture. Several attention-based approaches will be tested and compared with the original GAT, including:

- GATv2 : A simple modification to the GAT model. It computes attention coefficients as the following

$$\hat{a}^k_{i,j} = \frac{e^{(\alpha^{k\top}\,\mathrm{LeakyReLU}(\boldsymbol{\Theta}^k\mathbf{h_i}+\boldsymbol{\Theta}^k\mathbf{h_j}))}}{\sum_{k\in\mathcal{N}(i)\cup\{i\}} e^{(\alpha^{k\top}\,\mathrm{LeakyReLU}(\boldsymbol{\Theta}^k\mathbf{h_i}+\boldsymbol{\Theta}^k\mathbf{h_k}))}}. \tag{4.28}$$

Main difference with GAT is that the activation function is used before projection.

- TransformerGAT : Conventional attention approach is employed. It computes the attention coefficients as the following

$$a_{i,j}^{k} = \frac{e^{(\mathbf{\Theta^{q}}^{k}\mathbf{x}_i)^T(\mathbf{\Theta^{k}}^{k}\mathbf{x}_j)}}{\sum_{\mathcal{N}(i)\cup\{i\}} e^{(\mathbf{\Theta^{q}}^{k}\mathbf{x}_i)^T(\mathbf{\Theta^{k}}^{k}\mathbf{x}_k)}}. \tag{4.29}$$

These two variants of GAT is used to evaluate on both ADD-SCI and ADD-MCI classification and compared with the original GAT. Experimental results are shown in 4.10.

Experimental results in Table 4.10 shows that the GAT is superior to the other attention-based GNN models. To further enhance the model's performance, the pooling module, which aggregates node representations into a latent graph representation, will be explored. The following pooling methods will be investigated:

- Mean Pool: A straightforward pooling method that computes the average of node representations, providing a simple yet effective aggregation approach.
- Attention Pool : A more sophisticated, attention-based pooling technique that computes a node's importance score to perform a weighted summation of node representations, enabling the model to focus on the most informative nodes with the following equation

$$\mathbf{w} = MLP(\mathbf{H}), \tag{4.30}$$

$$\mathbf{z} = \sum_{i\in V} w_i \mathbf{h}_i. \tag{4.31}$$

These pooling techniques will be evaluated to determine their impact on the final graph representation and overall performance. Experimental results are shown in Table 4.11.

The results indicate that, despite its simplicity and similarity to Sum Pooling, Mean Pooling consistently provides the best overall performance across both tasks.

While Attention Pooling shows competitive results in the ADD-SCI classification task, achieving the highest accuracy, it underperforms in the ADD-MCI classification. Furthermore, the high performance observed in ADD-SCI is accompanied by greater variability, suggesting that Attention Pooling may not offer consistent results across different validations. This variability makes it less reliable compared to other pooling methods.

Table 4.11. Performance of pooling methods on Log-Scaled sNETs.

ADD-SCI Classification			
Network	**Log-Scaled Features**		
	Acc	F1	AUC
Sum Pooling	88.1±2.1	87.0±2.4	94.2±1.4
Mean Pooling	88.2±1.9	**87.1±2.2**	**95.2±1.3**
Attention Pooling	**88.5±2.2**	86.9±5.2	94.3±2.0
ADD-MCI Classification			
Network	**Log-Scaled Features**		
	Acc	F1	AUC
Sum Pooling	76.6±2.6	55.7±4.4	79.7±1.3
Mean Pooling	**77.1±1.3**	**56.2±3.2**	**80.6±1.1**
Attention Pooling	71.9±1.2	52.0±2.6	77.8±0.8

One potential explanation for Mean Pooling's superior performance over Sum Pooling could be its ability to smooth latent node representations when aggregating them into latent graph representations. By averaging the node features, Mean Pooling may better capture the overall structure and reduce the risk of overemphasizing outlier values.

In light of all the conducted sNET experiments, we conclude that GAT consistently outperforms other GNN models across both ADD-SCI and ADD-MCI classification tasks. Among all tested architectures, the combination of DeepSet preprocess-

ing with GAT (DS + GAT), followed by Mean Pooling trained on log-scaled sNETs, emerges as the best-performing approach. This underscores the importance of feature selection, encoding, and attention mechanisms in extracting meaningful patterns from structural brain networks.

The use of DeepSet plays a critical role in encoding connectivity values, transforming raw structural data into representations that are more amenable to processing by GNNs. This capability allows GAT to optimize message passing, making it particularly effective in identifying key structural patterns that differentiate between various stages of ADD.

The performance of the DS + GAT architecture highlights the value of adaptive mechanisms like attention in brain network analysis. By focusing on the most relevant interactions within the network, GAT achieves significant improvements in classification metrics while also offering interpretability by highlighting critical connections. This is especially relevant for ADD, where explainability provides valuable insights into the progression of the disease.

4.2.2. fNET Experiments

In the experiments, two classification tasks are performed: ADD-SCI and ADD-MCI. Performance on these tasks is quantified using accuracy, F1-score, and AUC metrics. Cross-validation is employed to evaluate performance, and each cross-validation experiment is repeated 20 times to mitigate the impact of random seeds. For the ADD-SCI task, 21-fold cross-validation is utilized due to the limited sample size. It is observed that using larger folds reduces the generalizability of models for ADD-SCI classification. The 21-fold cross-validation strikes a balance between improved generalization and runtime efficiency, being twice as fast as leave-one-out cross-validation while yielding comparable results. For ADD-MCI classification tasks, 7-fold cross-validation is applied. Detailed description for both experiments including hyperparameters are given in the Table 4.12.

Table 4.12. Summary of fNET Experimental Configurations for Classification Tasks.

Description	**ADD-SCI**	**ADD-MCI**
Label Distribution	18 ADD/24 SCI	18 ADD/46 SCI
Number of Folds	21	7
Batch Size	6	6
Number of Epochs	100	100
Learning Rate	0.001	0.001
Optimizer	Adam	Adam
Weight Decay	0.0001	0.0001

Functional brain networks represent the brain's functional profile by modeling the statistical dependencies, such as correlations or coherence, between the activity patterns of distinct brain regions. Unlike structural networks, these networks are typically represented as fully connected graphs, where nodes correspond to brain regions and edges encode functional connectivity values derived from pairwise measures.

The fully connected nature of fNETs poses unique challenges for GNNs, which are inherently designed to operate on sparse graphs. Fully connected graphs can introduce redundancy and dilute the graph's informative content, potentially leading to computational inefficiencies and suboptimal learning performance. To address this, preprocessing steps are often required, such as thresholding to sparsify the graph, filtering to retain only significant connections, or converting the graph into a more interpretable form tailored for GNN input.

Similar to sNETs, functional networks also face the challenge of lacking node features. As a result, it becomes essential to either derive meaningful features from the functional connectivity matrix or use the connectivity values themselves as features. This limitation necessitates careful design of graph representations and feature engineering to ensure GNNs can effectively capture the functional characteristics of the

brain.

To justify the use of GNNs for analyzing fNETs, it is essential to demonstrate their ability to leverage graph structures for improved classification or prediction performance. A fair comparison against baseline models that do not explicitly use graph structure is crucial to highlight the advantages of GNNs in functional network analysis. In order to

In the fNET experiments, these objectives will be explored through the following steps similar to the sNET experiments:

- Baseline Model Evaluation: DeepSet will be used to assess the effectiveness of different feature construction methods.
- GNN Model Comparison: Various GNN models with diverse configurations will be evaluated by integrating them into the Deepset (replacing one layer). This configurations include graph construction strategies for fully connected fNETs and model parameters. The top-performing models will be shortlisted for further investigation.
- Optimization of Selected GNNs: The shortlisted GNN models will undergo optimization through experiments with different setups, such as varying the number of layers and directly applying the GNN without an intermediate baseline model. The best-performing architecture and configuration will be selected for subsequent steps.
- Pooling Method Analysis: The selected GNN architecture with the best configuration will be tested with various pooling strategies to identify the most suitable approach for aggregating node-level information.
- Model Refinement: Building on insights from the literature, the selected GNN architecture will be enhanced with minor modifications to test potential performance improvements.

DeepSet is used as a baseline to compare GNN models and evaluate fNET construction methods, following the same procedure with the sNET experiments. Three fNET construction methods are evaluated: partial fNET, Pearson fNET and ℓNET. Partial and Pearson fNET are correlation-based fNETs while ℓNET.

Table 4.13. Performance of 1-Layer and 2-Layer DeepSet Models on fNETs.

ADD-SCI Classification						
Network	**1-Layer DeepSet**			**2-Layer DeepSet**		
	Acc	F1	AUC	Acc	F1	AUC
Pearson	66.5±3.1	60.6±3.5	69.1±2.5	66.7±2.8	60.8±3.6	69.5±2.3
Partial	51.2±7.5	27.4±11.0	51.1±5.5	52.1±6.3	34.8±8.2	51.0±5.2
ℓNET	**76.5±2.5**	**71.2±3.0**	**78.2±1.9**	**74.6±2.7**	**69.7±3.2**	**77.5±1.5**
ADD-MCI Classification						
Network	**1-Layer DeepSet**			**2-Layer DeepSet**		
	Acc	F1	AUC	Acc	F1	AUC
Pearson	70.8±1.9	**39.3±5.1**	63.4±1.3	72.6±3.1	**44.5±5.6**	65.1±1.7
Partial	71.7±4.7	0.0±0.0	55.4±5.7	70.6±1.5	0.0±0.0	54.8±5.7
ℓNET	**75.3±2.6**	38.8±7.1	**68.9±2.0**	**76.9±2.6**	40.7±6.3	**72.3±1.8**

To evaluate the performance of different functional network types, experiments were conducted using the DeepSet model. The results for each type of fNET are summarized in Table 4.13. Unlike sNETs, where binarization and scaling were applied due to the small edge weights, ℓNETs were used directly without binarization since their edge weights were well-distributed and of significant magnitude. Similarly, the other fNET types were also utilized in their raw form without additional preprocessing of edge weights.

As presented in Table 4.13, among the correlation-based fNETs, the partial fNET exhibited the poorest performance across all metrics and models, indicating its limited utility in this context. In contrast, the Pearson fNET demonstrated significantly better

results compared to the partial fNET, showcasing its robustness in capturing functional connectivity.

The ℓNET, however, outperformed both correlation-based fNETs, particularly in the ADD-SCI classification task, where it showed superior results on all metrics. For the ADD-MCI classification, the Pearson fNET achieved a higher F1-Score, while the ℓNET maintained an advantage in terms of AUC.

These findings underscore the strengths of both the Pearson fNET and ℓNET in capturing meaningful relationships within functional brain networks. Consequently, these two fNET types were selected for subsequent GNN experiments, as they provide the most promising foundation for advancing the classification performance.

Initial experiments with DeepSet indicated that Pearson fNET and ℓNET hold promise for functional network analysis, paving the way for further exploration with GNNs. This section systematically evaluates the performance of various GNN architectures when applied to Pearson fNET and ℓNET , following a similar procedure that used in the sNET experiments.

Pearson fNETs differ from structural networks in that they are typically represented as fully connected graphs, making them less straightforward to adapt for GNN input. To make the Pearson fNET adjacency matrix suitable for GNN input, a sparsification step is required. A common practice in the literature is thresholding, where only a small percentage of the highest-weighted connections are retained. For this study, two sparsification thresholds were selected to construct adjacency matrices for Pearson fNET experiments:

- 5p-Pearson: The adjacency matrix $\mathbf{A}$ includes the top 5 percent of positive connections. Node features are directly set to the corresponding Pearson fNET values without any thresholding.

- 20p-Pearson: The adjacency matrix $\mathbf{A}$ includes the top 20 percent of positive connections. Node features are directly set to the corresponding Pearson fNET values without any thresholding.

This preprocessing ensures that the sparsified Pearson fNETs provide a manageable yet meaningful connections for GNN training. On the other hand, ℓNETs are sparse graphs by design, making them directly suitable for GNN input without requiring additional sparsification. Following the same procedure as in the sNET experiments, we use the node feature matrix $\mathbf{X}$ to represent the connectivity values of ℓNET.

To maintain methodological consistency and comparability, all GNN architectures were tested under identical conditions. Specifically, the second DeepSet layer was replaced with the respective GNN model while other parameters remained fixed. This consistency ensures fair comparisons and provides a robust framework for evaluating the advantages of GNNs, especially their ability to aggregate messages and learn more expressive graph representations. The best-performing configurations from the sNET experiments were utilized as a baseline, ensuring a solid foundation for evaluating the performance of GNN models on fNETs.

The following GNN architectures were selected based on their diversity in design and their success in the sNET experiments:

- Max and Mean GraphSAGE
- GCN
- GIN
- 4-Head GAT

To systematically evaluate the ability of GNNs to improve upon the baseline DeepSet model, two experimental setups were designed. The first setup focused on Pearson fNET, exploring how sparsification impacts GNN performance. The second

setup investigated ℓNET, leveraging its inherent sparsity to directly test GNN architectures without additional preprocessing. The primary goal of both setups was to identify the GNN architectures that could outperform DeepSet in modeling functional connectivity data.

Table 4.14. Performance of DS + GNN models for Pearson fNET.

ADD-SCI Classification						
Network	**5p-Pearson**			**20p-Pearson**		
	Acc	F1	AUC	Acc	F1	AUC
DS + Max GraphSAGE	64.5±4.7	58.5±5.2	68.1±3.2	**66.9±1.7**	60.3±2.8	69.2±1.2
DS + Mean GraphSAGE	65.2±4.0	59.5±4.1	67.8±2.3	66.9±2.5	60.5±3.3	69.4±2.3
DS + GCN	65.9±3.8	58.5±4.8	**70.4±2.7**	65.0±1.8	60.0±2.7	69.3±1.3
DS + GIN	64.5±2.9	58.7±2.9	66.1±1.8	63.6±5.2	56.0±6.9	67.7±2.9
DS + 4-Head GAT	66.4±3.7	**61.4±3.8**	68.7±1.8	64.8±2.3	57.8±3.9	69.4±1.4
DS^2	**66.7±2.8**	60.8±3.6	69.5±2.3	66.7±2.8	**60.8±3.6**	**69.5±2.3**
ADD-MCI Classification						
Network	**5p-Pearson**			**20p-Pearson**		
	Acc	F1	AUC	Acc	F1	AUC
DS + Max GraphSAGE	70.0±2.8	34.2±5.9	63.6±1.8	71.4±3.0	38.1±7.7	63.3±2.0
DS + Mean GraphSAGE	71.6±2.6	42.5±6.7	63.7±1.7	70.3±5.2	40.5±10.3	62.9±2.2
e DS + GCN	71.7±2.6	41.8±4.4	64.7±1.4	71.4±2.7	41.8±5.6	64.6±1.3
DS + GIN	67.2±3.7	37.7±7.5	57.5±2.5	67.0±3.6	38.8±6.0	59.5±2.0
DS + 4-Head GAT	71.2±2.6	42.0±6.8	62.7±1.8	70.1±2.3	38.3±6.5	61.7±1.3
DS^2	**72.6±3.1**	**44.5±5.6**	**65.1±1.7**	**72.6±3.1**	**44.5±5.6**	**65.1±1.7**

In the first experimental setup, each GNN architecture was trained and evaluated on the sparsified 5p-Pearson and 20p-Pearson configurations. These configurations were designed to explore how varying the sparsity of the graph, as determined by the percentage of top-weighted connections retained, impacts the ability of GNNs to learn meaningful representations from functional connectivity data. Experimental results

can be seen in Table 4.14.

The experimental results revealed that sparsification had a task-dependent impact on GNN performance. Convolution-based GNNs (DS + GraphSAGE and DS + GCN) performed slightly better with the 20p-Pearson configuration for ADD-SCI classification. In contrast, DS + GIN and DS + GAT showed a decrease in performance with increased graph density, suggesting that these models may struggle with the additional complexity introduced by denser graphs. For ADD-MCI classification, only DS + Max GraphSAGE showed improvement with the 20p-Pearson configuration, while DS + GAT's performance dropped relative to the 5p-Pearson configuration. However, across both tasks, the performance differences between sparsification levels were not substantial.

Importantly, none of the GNN architectures managed to outperform the baseline DeepSet model for ADD-MCI classification. Although certain GNNs achieved marginally better results in ADD-SCI classification, the improvements were not significant. This outcome highlights a potential limitation of the thresholding approach used to construct adjacency matrices for functional networks. While thresholding simplifies the graph structure, it may inadvertently discard critical information embedded in the weaker connections or distort the underlying relationships in the data. Consequently, the constructed sparse adjacency matrices may fail to provide GNNs with the necessary structural information to outperform simpler models like DeepSet.

In the second experimental setup, GNNs were evaluated directly on ℓNET, leveraging its inherent sparsity without requiring additional preprocessing. This setup aimed to determine whether GNNs could effectively capitalize on ℓNET's sparse structure to outperform DeepSet.

The results, presented in Table 4.15, revealed that ℓNET's sparsity did not provide a significant advantage for GNNs over DeepSet. For ADD-SCI classification, Max GraphSAGE and GAT showed slight improvements over the baseline, while for ADD-

MCI classification, Mean GraphSAGE and GCN outperformed the baseline DeepSet by a small margin. However, these improvements were minimal and did not indicate a consistent advantage for GNNs across tasks.

Table 4.15. Performance of DS + GNN models for ℓNET.

ADD-SCI Classification			
Network	**ℓNET**		
	Acc	F1	AUC
DS + Max GraphSAGE	**76.9±3.2**	**70.8±4.3**	**80.6±1.9**
DS + Mean GraphSAGE	73.1±1.1	68.0±0.9	76.3±0.4
DS + GCN	73.8±2.1	68.4±3.6	76.3±1.3
DS + GIN	70.7±3.5	64.2±4.8	69.8±1.7
DS + 4-Head GAT	75.7±2.1	71.1±2.6	76.6±1.3
DS^2	74.6±2.7	69.7±3.2	77.5±1.5
ADD-MCI Classification			
Network	**ℓNET**		
	Acc	F1	AUC
DS + Max GraphSAGE	74.2±1.6	28.7±8.1	**75.4±1.2**
DS + Mean GraphSAGE	76.4±2.8	**42.0±8.0**	71.8±2.1
DS + GCN	76.8±3.3	41.1±8.7	70.8±1.9
DS + GIN	70.5±3.6	31.0±9.4	62.2±1.6
DS + 4-Head GAT	75.1±2.0	38.4±5.9	68.8±1.2
DS^2	**76.9±2.6**	40.7±6.3	72.3±1.8

The results from both experimental setups underscore a persistent challenge: while GNNs are designed to exploit graph structure through message aggregation, they struggle to leverage the sparsified Pearson fNETs and inherently sparse ℓNETs to achieve consistent improvements over DeepSet. For Pearson fNETs, thresholding may inadvertently remove critical information, while for ℓNETs, the simplicity of the graph structure may limit GNN expressiveness.

Among the tested GNNs, Max and Mean GraphSAGE showed the most promise, delivering competitive performance across tasks. Specifically, mean aggregation ap-

peared more effective for ADD-MCI classification, while max aggregation performed better for ADD-SCI classification.

Moving forward, the investigation will focus refining GraphSAGE models to achieve performance gains over DeepSet. Max and Mean GraphSAGE will be investigated over the ℓNETs in terms of architecture selection, pooling methods and additional features.

Based on the outcomes of the previous experiments, GraphSAGE emerged as the top-performing and only promising model among the GNN architectures evaluated. GAT is experimented with combination of 1-Layer DeepSet. However, while the DeepSet + GAT architecture showed promise, it may not represent the optimal configuration for achieving the best results. To address this and further investigate the potential of GraphSAGE, a series of experiments will be conducted, focusing on the changing overall architecture similarly to the procedure followed in sNET Experiments:

- Plain GNN Architectures: In this setup, GNN models were directly applied to process input without a preceding DeepSet layer. Different numbers of GNN layers, specifically 1 and 2, were evaluated. These architectures are denoted as GNN^k where GNN refers to Max GraphSAGE, GIN, or 4-Head GAT, and k represents the number of GNN layers. The plain GNN architectures are defined as

$$\mathbf{z} = \sum_{i \in V} GNN^k(\mathbf{X}). \tag{4.32}$$

  Here, GNN^k represents a stack of k GNN layers.
- Adding another layer to DS + GNN architecture: This setup extends the previously explored DS + GNN architecture by adding an additional GNN layer. These configurations are denoted as $DS + \text{GNN}^2$, where DS represents the initial DeepSet layer, and GNN^2 denotes a 2-layer GNN. This experiment aimed to determine whether increasing the depth of the GNN component would enhance performance. However, to mitigate the well-known issue of oversmoothing in

GNNs, experiments were limited to two GNN layers.

Table 4.16. Performance of Plain GNN and GNN^k architectures compared to DS + Max GraphSAGE, DS + GIN and DS +4-Head GAT on ℓNET.

ADD-SCI Classification			
Network	**ℓNET**		
	Acc	F1	AUC
Max GraphSAGE^1	72.6±4.2	67.0±4.6	80.6±2.3
Mean GraphSAGE^1	72.9±3.0	67.0±4.1	75.5±1.7
Max GraphSAGE^2	76.2±2.8	70.2±3.8	**82.9±1.7**
Mean GraphSAGE^2	75.7±2.3	70.0±3.0	78.1±1.3
DS + Max GraphSAGE^2	76.4±3.4	69.9±5.5	79.1±2.7
DS + Mean GraphSAGE^2	75.5±4.0	69.7±5.1	77.4±3.7
DS^2(ℓNET)	74.6±2.7	69.7±3.2	77.5±1.5
DS^2(Pearson)	66.7±2.8	60.8±3.6	69.5±2.3
DS + Max GraphSAGE	**76.9±3.2**	**70.8±4.3**	80.6±1.9
ADD-MCI Classification			
Network	**ℓNET**		
	Acc	F1	AUC
Max GraphSAGE^1	74.8±2.3	43.8±5.9	72.8±1.5
Mean GraphSAGE^1	77.2±1.8	43.9±6.6	71.2±1.0
Max GraphSAGE^2	76.8±1.3	41.1±4.5	**74.7±2.0**
Mean GraphSAGE^2	**77.8±1.7**	**46.3±5.8**	72.4±1.8
DS + Max GraphSAGE^2	73.1±2.5	36.1±8.9	72.6±2.9
DS + Mean GraphSAGE^2	75.3±2.7	36.6±7.1	70.1±3.1
DS^2(ℓNET)	76.9±2.6	40.7±6.3	72.3±1.8
DS^2(Pearson)	72.6±3.1	44.5±5.6	65.1±1.7
DS + Max GraphSAGE	74.2±1.6	28.7±8.1	**75.4±1.2**

Experimental results for both GNN^k and DS + GNN^2 configurations, along with the top-performing DS + GNN architectures from previous experiments and 2-Layer DeepSet, can be found in Table 4.16. The summarized findings are as follows:

- Adding another GNN layer to DS + GNN architectures negatively affected Max GraphSAGE. It may show that Max GraphSAGE extract most of the information within the immediate neighbors.
- For standalone GNN^k models, increasing the number of layers ($k = 2$) improves performance in the ADD-SCI classification task. This contradicts earlier findings, suggesting that encoding node features prior to GNN processing may reduce the need for multi-hop neighbor information in this specific task.
- Mean GraphSAGE^2 ($k = 2$) achieves the best overall performance for ADD-MCI classification, outperforming both DS + GNN and single-layer ($k = 1$) configurations.
- DS + GNN^2 architectures underperform compared to their standalone GNN^2 counterparts in most setups, indicating that the combined encoding and deeper GNN structure negatively impacts performance.
- DS + Max GraphSAGE^1 trained on ℓNET emerges as the best architecture for ADD-SCI classification while Max GraphSAGE^2 outperforms other models for ADD-MCI classification.

Results show that GraphSAGE models hold significant promise for functional network analysis in the progression of ADD. These models demonstrate their ability to extract meaningful features from functional brain networks, leading to some improvements over baseline models. However, the observed advancements are not as pronounced as those achieved in structural network analyses. This disparity underscores the inherent differences between structural and functional networks, as well as the challenges associated with leveraging functional connectivity data for accurate diagnosis. Since the justification for using GNNs has not been established, the remaining experiments, including the pooling method comparisons and model refinement, are omitted.

A notable observation is the variability in the best-performing architectures across different classification tasks. For example, while DS + Max GraphSAGE[1] emerged as the top performer for ADD-SCI classification, Max GraphSAGE[2] demonstrated superior performance in ADD-MCI classification. This variability illustrates the complexity of functional brain network analysis, where task-specific architectures are often required to achieve optimal results. It highlights the need for systematic experimentation to tailor model configurations to the nuances of each classification task.

The use of ℓNET representations provides a significant improvement over correlation-based FNETs, particularly for ADD-SCI classification. ℓNET achieves notable gains in accuracy, F1-Score, and AUC, validating its suitability for graph-based approaches like GNNs. In the case of ADD-MCI classification, ℓNET enhances accuracy and AUC, though it struggles to improve the F1-Score. This indicates that while ℓNET excels at general predictive performance, it may face challenges in handling imbalanced datasets or minority class predictions, which are critical in medical diagnostic tasks.

The sparse nature of ℓNET representations makes them well-suited for GNNs, as they effectively reduce noise and emphasize meaningful connections between brain regions. This opens new opportunities for functional brain network analysis by enabling GNNs to process high-dimensional data more efficiently. However, one key limitation of ℓNET is its reliance on the smoothness assumption, which encodes simultaneous relationships between brain regions but fails to account for lagged or temporal interactions. Such lagged interactions are essential for capturing the dynamic and causal nature of brain activity, particularly in functional connectivity analysis. Addressing this limitation is crucial for advancing the field.

Overall, the findings indicate that while GNN architectures can improve functional brain network analysis for ADD diagnosis, their impact remains less promising than their application to structural brain networks. This may be due to the inherently dynamic and less deterministic nature of functional networks compared to the more static and well-defined structure of anatomical connections.

4.2.3. Multimodal Experiments

In the experiments, two classification tasks are performed: ADD-SCI and ADD-MCI. Performance on these tasks is quantified using accuracy, F1-score, and AUC metrics. Cross-validation is employed to evaluate performance, and each cross-validation experiment is repeated 20 times to mitigate the impact of random seeds. For the ADD-SCI task, 21-fold cross-validation is utilized due to the limited sample size. It is observed that using larger folds reduces the generalizability of models for ADD-SCI classification. The 21-fold cross-validation strikes a balance between improved generalization and runtime efficiency, being twice as fast as leave-one-out cross-validation while yielding comparable results. For ADD-MCI classification tasks, 7-fold cross-validation is applied. Detailed description for both experiments including hyperparameters are given in the Table 4.17.

Table 4.17. Summary of Multi-modal Experimental Configurations for Classification Tasks.

Description	**ADD-SCI**	**ADD-MCI**
Label Distribution	18 ADD/24 SCI	18 ADD/46 SCI
Number of Folds	21	7
Batch Size	6	6
Number of Epochs	100	100
Learning Rate	0.001	0.001
Optimizer	Adam	Adam
Weight Decay	0.0001	0.001

sNETs and fNETs provide distinct yet complementary information about the brain. Leveraging the complementary nature of these modalities can enrich single-modal network analysis. However, two main challenges arise: although both modalities share the same set of nodes, their construction methods and underlying interpretations differ significantly. Additionally, as shown in previous experiments, different architec-

tures are optimal for analyzing structural and functional networks. These differences make it difficult to effectively integrate information from both modalities without losing or diluting their unique contributions.

To address these challenges, we conduct a simple experiment following the step: For each classification task, the best-performing model from one modality is initially used to process both structural and functional networks in a siamese-like setting with shared weights. This ensures a unified approach across both modalities.

Table 4.18. Performance of Siamese GNN architectures for multi-modal analysis compared to unimodal DS + 4-Head GAT, DS + Max GraphSAGE and Max GraphSAGE^2.

ADD-SCI Classification			
Model	Acc	F1	AUC
DS + 4-Head GAT (multi-modal)	**88.6±1.6**	**87.5±1.9**	95.0±1.4
DS + Max GraphSAGE (multi-modal)	88.4±2.3	86.6±2.9	92.7±1.1
DS + 4-Head GAT (sNET)	88.2±1.9	87.1±2.2	**95.2±1.3**
DS + Max GraphSAGE (ℓNET)	76.9±3.2	70.8±4.3	80.6±1.9
ADD-MCI Classification			
Model	Acc	F1	AUC
DS + 4-Head GAT (multi-modal)	76.9±2.4	55.0±5.1	78.8±1.4
Max GraphSAGE^2 (multi-modal)	76.8±1.8	53.3±5.2	79.9±1.1
DS + 4-Head GAT (sNET)	77.1±1.3	**56.2±3.2**	**80.6±1.1**
Max GraphSAGE^2 (ℓNET)	**77.8±1.7**	46.3±5.8	72.4±2.0

For the ADD-SCI classification task, DS + 4-Head GAT and DS + Max GraphSAGE are used within a siamese architecture. For ADD-MCI classification, DS + 4-Head GAT and Mean GraphSAGE^2 are employed in the same siamese framework. Log-scaled sNET and ℓNET are selected to represent the structural and functional modalities, respectively. In this configuration, the models share parameters, processing both modalities in a coupled manner. The experimental results for siamese DS + 4-Head GAT, DS + Max GraphSAGE, and Mean GraphSAGE^2, alongside the single-

modality baselines, are presented in Table 4.18.

The results presented in Table 4.18 indicate that while the multi-modal Siamese model led to a slight improvement in ADD-SCI classification, it did not yield significant benefits for ADD-MCI classification. In fact, the multi-modal Siamese model showed a decrease in performance compared to the unimodal sNET model for ADD-MCI classification. This suggests that treating structure and function as complementary modalities may not always provide the optimal solution for multi-modal analysis, particularly for more complex tasks like ADD-MCI classification. These findings highlight the need for more sophisticated approaches in combining structural and functional information for multi-modal brain network analysis.

4.3. Structure-Function Discrepancy Learning

To evaluate the potential of discrepancy as a biomarker, we primarily focused on the ADD-SCI classification task. This approach was chosen because sfDLN is trained to discern different classes, and selecting ADD and SCI enables capturing the entire ADD spectrum. By leveraging this classification task, we aimed to assess whether the discrepancy measure could effectively distinguish between SCI individuals and those with ADD, thereby providing insights into its viability as a diagnostic tool.

ADD-SCI classification was conducted using 21-fold cross-validation, consistent with previous experiments. Performance was assessed using Accuracy and F1-Score metrics. The AUC metric was not utilized due to the nature of sfDLN's training process, where the output discrepancy scores are not designed to converge to binary values (e.g., 0 or 1). Instead, sfDLN distinguishes distinct mean discrepancy scores for different classes. However, these scores may vary across folds, resulting in less consistency compared to traditional classification approaches, where predictions are independent of fold variations. The training details and model hyperparameters are summarized in Table 4.24.

Table 4.19. Summary of sfDLN Experimental Configurations for Classification Tasks.

Description	**ADD-SCI**
Label Distribution	18 ADD/24 SCI
Number of Folds	21
Batch Size	20
Number of Epochs	100
Learning Rate	0.001
Optimizer	Adam
Weight Decay	0.0001
λ	0.1
m	0.1
d	32

sfDLN is designed to quantify the discrepancy between the structure and function of the brain, which is then leveraged to classify and monitor ADD progression by discerning patients with different clinical labels. To evaluate the ability of sfDLN to serve as a biomarker for representing the entire ADD spectrum, the following experiments were conducted:

- ADD-SCI Experiments : To capture the entire ADD spectrum, the ADD-SCI classification task was primarily performed. Various GNN architectures were evaluated to determine the most effective design for sfDLN in this classification scenario. Prior insights from earlier experiments regarding model configurations and input data were incorporated to guide this process. Once the optimal architecture was determined, the hypothesis that Structure-Function Discrepancy increases with disease progression was tested. This was done by training the sfDLN under a reversed hypothesis, where Structure-Function Discrepancy is assumed to decrease with disease progression, to validate the original assumption. Additionally, the functional network selection was justified by experimenting with two alternative functional network configurations. Finally, experiments were con-

ducted to assess the impact of classifier hyperparameters on sfDLN performance.

- Structure-Function Discrepancy Across the Cognitive Decline Continuum: To further explore sfDLN's potential as a biomarker, a blind assessment of MCI cases in the dataset was conducted. For this experiment, sfDLN was trained exclusively on ADD and SCI instances, and the resulting discrepancy scores for the MCI data were analyzed. The results were compared to ensure that MCI cases fit consistently between ADD and SCI in terms of discrepancy scores, reflecting their intermediate position along the cognitive decline continuum. To refute the objection that the observed MCI discrepancy distribution could result from random selection or uncorrelated labels, two control experiments were performed. In the first, sfDLN was trained on MCI-ADD and tested on SCI; in the second, it was trained on SCI-MCI and tested on ADD.
- Source of the Structure-Function Discrepancy: To provide a direct neurological interpretation for the observed discrepancy, the GNNExplainer approach [**?**, 135] was employed. This method identified the most significant structural and functional network changes that contribute to the discrepancy, offering insights into the specific brain regions and connections driving the differences between Structure-Function Discrepancy scores.

Each branch of the sfDLN network generates a 128-dimensional connectome embedding. The model parameters listed in Table 4.24 were selected to facilitate faster training convergence. The final output of sfDLN is a bounded scalar discrepancy score, γ, for each subject. This score serves a dual purpose: it is used for classification and to evaluate structure-function coherence across the disease spectrum, effectively acting as a disease severity measure.

For training and testing the sfDLN network, ADD and SCI groups were designated as positive (p) and negative (n) cases, respectively. The separability of these groups based on their γ scores was evaluated using a k-NN classifier with $k = 7$, where the value of k was empirically determined.

To evaluate the potential of the discrepancy score as a biomarker, we conducted cross-validation experiments on ADD-SCI classification using sfDLN. The mean and standard deviation of model accuracies and F1 scores across these experiments are reported in Table 4.20. To determine the optimal backbone for sfDLN, different architectures were tested. Additionally, comparisons were made between sfDLN and both uni-modal and multi-modal models discussed in the previous section, highlighting sfDLN's superior performance for diagnostic purposes.

Table 4.20. Performance of sfDLN with different models as backbone.

ADD-SCI Classification		
Model	Acc	F1
DS + Max GraphSAGE sfDLN	89.5±2.4	87.5±2.7
DS + Mean GraphSAGE sfDLN	90.5±1.8	88.9±1.9
DS + ChebyNet sfDLN	90.5±1.5	88.6±1.8
DS + GCN sfDLN	91.7±1.2	90.1±1.4
DS + GIN sfDLN	88.6±1.7	85.9±2.2
DS + 4-Head GAT sfDLN	89.8±2.1	87.7±2.6
DS + 4-Head GAT sNET	88.2±1.9	87.1±2.2
DS + 4-Head GAT multi-modal	88.6±1.6	87.5±1.9

Key observations from the table are :

- The DS + GCN sfDLN model outperforms all others in terms of both accuracy and F1 score. This indicates that the GCN backbone is the most effective for ADD-SCI classification within the sfDLN framework. Although GCN did not show superior performance in uni-modal tasks, it perfoms well in extracting multi-modal representation.
- Both DS + ChebyNet sfDLN and DS + Mean GraphSAGE sfDLN perform competitively, demonstrating that these models also capture relevant information in brain network data.

- Other models, DS + Max GraphSAGE, DS + GIN and DS + 4-Head GAT performs slightly worse than the others for sfDLN, showing that mean aggregation might be more effective capturing the multi-modal relationship.
- Most sfDLN variants outperforms the both uni-modal and multi-modal models, showing that the structure-function relationship might be a key to improve multi-modal and uni-modal methods in ADD-SCI classification.

We concluded that the DS + GCN architecture is the most suitable backbone for sfDLN. Consequently, all subsequent experiments will be conducted using the DS + GCN sfDLN architecture.

Table 4.21. Comparison of sfDLN trained with increasing vs decreasing discrepancy.

Model	**Accuracy**	**F1 Score**
DS + GCN sfDLN-I	88.5±2.5	86.4±2.9
DS + GCN sfDLN	**91.7±1.2**	**90.1±1.4**

We conducted additional experiments with sfDLN using modified loss functions to test our hypothesis that the structure-function discrepancy increases with cognitive decline. Specifically, we replaced the mean separation term in the loss function (Equation 3.45) with $\max(0, m-(\mu^n-\mu^p))$. This modification resulted in a variant, sfDLN-I, trained under the assumption that SCI patients exhibit higher structure-function discrepancy scores than ADD patients. As shown in Table 4.21, sfDLN outperforms sfDLN-I, validating our original hypothesis that the structure-function discrepancy widens with cognitive decline.

Additionally, we justified the selection of ℓNET by replacing it with Pearson fNET and training sfDLN with the sNET-Pearson fNET pair. As evident from Table 4.22, sfDLN performs poorly when using Pearson fNET, demonstrating the superiority of ℓNET. This difference in performance is likely due to the sparse structure of ℓNET,

which makes it easier for the sfDLN model to quantify the similarity between sNET and ℓNET.

Table 4.22. Performance of sfDLN trained with sNET and Pearson fNET.

Model	**Accuracy**	**F1 Score**
DS + GCN sfDLN (sNET + Pearson fNET)	71.7±3.1	63.0±5.0
DS + GCN sfDLN (sNET + ℓNET)	**91.7±1.2**	**90.1±1.4**

Lastly, we investigated the impact of the k-NN classifier by conducting experiments with varying k values, as presented in Table 4.23. Table 4.23 demonstrates that $k = 7$ is the optimal choice for the k-NN classifier. However, the overall results show minimal variation across different k values, highlighting the robustness of the produced bounded structure-function discrepancy scores.

Table 4.23. Performance of sfDLN with varying k parameters.

Model	**Accuracy**	**F1 Score**
DS + GCN sfDLN (k=5)	91.4±1.1	89.8±1.3
DS + GCN sfDLN (k=7)	**91.7±1.2**	**90.1±1.4**
DS + GCN sfDLN (k=9)	90.9±1.6	89.4±1.8

Previous experiments demonstrated that the structure-function discrepancy score (γ) outperforms both uni-modal and multi-modal ADD-SCI classifiers, showcasing its diagnostic potential. However, to establish its utility for disease staging, a more thorough evaluation is required.

To explore its potential for monitoring disease progression, we conducted the following experiments:

- Blind Assesment of MCI Cases: We trained sfDLN using all ADD and SCI subjects, and the trained model was then used to generate structure-function discrepancy scores for MCI subjects. The results, presented in Figure 4.2, indicate that the structure-function discrepancy scores of MCI subjects fall between those of ADD and SCI subjects. This suggests that the structure-function discrepancy score can provide a consistent representation of cognitive decline progression toward ADD.
- Control Experiments: Although the observation in Figure 4.2 is noteworthy, one might argue that connectomes without labels that are unrelated to ADD could still be distributed between the ADD and SCI classes. To address this concern, we conducted two additional experiments. First, we trained sfDLN on ADD-MCI data and tested it on SCI data. Second, we trained sfDLN on MCI-SCI data and tested it on ADD data. The results of these experiments are shown in Figure 4.3 and Figure 4.4.

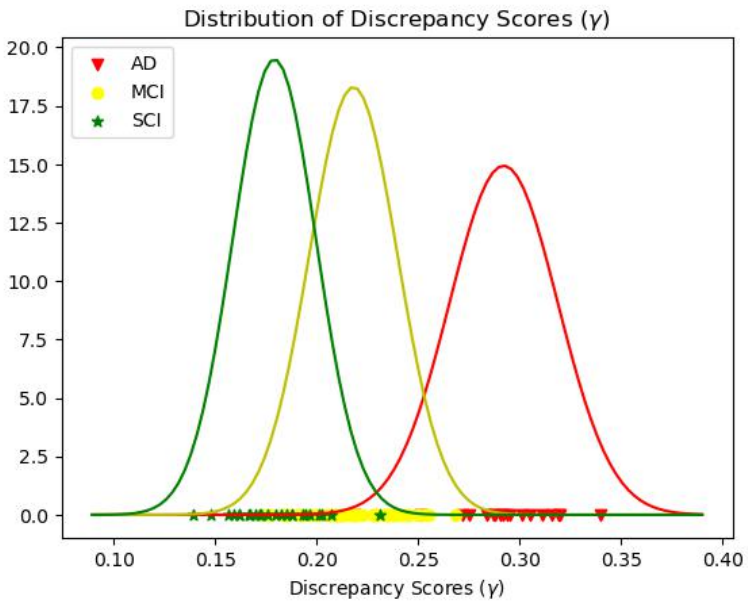

Figure 4.2. Distributions of γ-scores for ADD, MCI, and SCI groups, computed by sfDLN trained over ADD-SCI only.

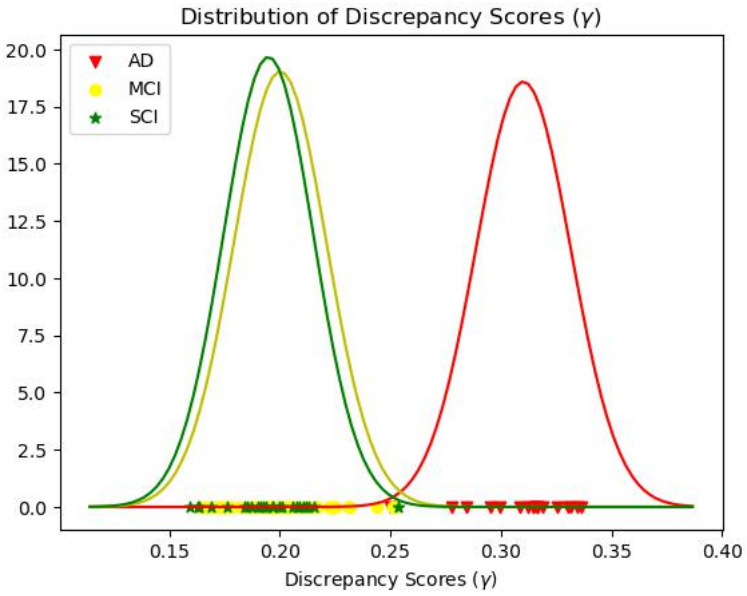

Figure 4.3. Distributions of γ-scores for ADD, MCI, and SCI groups, computed by sfDLN trained over ADD-MCI only.

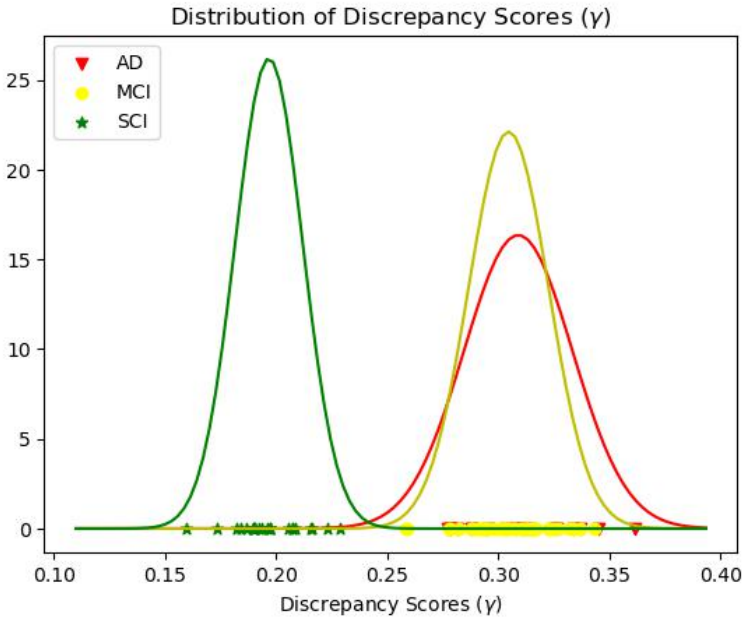

Figure 4.4. Distributions of γ-scores for ADD, MCI, and SCI groups, computed by sfDLN trained over MCI-SCI only.

Figure 4.3 and Figure 4.4 illustrate that in both control experiments, the discrepancy scores of the test group remained distant from the central region of the spectrum defined by the training set. In the first experiment, the test SCI subjects' scores were positioned furthest from the ADD-end of the spectrum, which was limited by the absence of ADD examples during training—this is the best performance achievable by sfDLN under these conditions. In the second experiment, ADD test scores were similarly found to be positioned farthest from the SCI-end of the available spectrum. Consequently, in both experiments, the test scores overlapped with those of the MCI group. This outcome aligns with the known limitations of deep learning models in extrapolating beyond the range of the training data [136]. The fact that sfDLN maps out-of-spectrum test data to the closest point within the peripheral region of the training spectrum further supports our hypothesis. Together, the observations from Figures 4.2 reinforce the idea that sfDLN has the potential to function as a biomarker for staging and severity assessment in the progression of ADD, provided it is trained on a comprehensive spectrum of data.

Although the results presented in previous experiments provide compelling empirical support for our central hypothesis, increasing structure-function discrepancy over the cognitive decline, the current architecture does not directly link the observed discrepancy to specific neurological features. To address this limitation, we employed the GNNExplainer approach [135], which enabled us to identify the key changes in sNETs and fNETs contributing to the discrepancy. Given the structural ($\mathbf{A}_s$, $\mathbf{X}_s$) and functional connectomes ($\mathbf{A}_f$, $\mathbf{X}_f$), the GNNExplainer utilizes a learnable $N \times N$ *importance* map, $\mathbf{M}$, with $m_{ij} \in [0, 1]$, to construct new importance-weighted connectomes, by applying $\hat{\mathbf{A}}_s = \mathbf{M} \odot \mathbf{A}_s$ and $\hat{\mathbf{A}}_f = \mathbf{M} \odot \mathbf{A}_f$. The corresponding sfDLN outputs $y = sfDLN(\mathbf{A}_s, \mathbf{X}_s, \mathbf{A}_f, \mathbf{X}_f)$ and $\hat{y} = sfDLN(\hat{\mathbf{A}}_s, \mathbf{X}_s, \hat{\mathbf{A}}_f, \mathbf{X}_f)$ are then used to compute the following GNNExplainer loss for minimization

$$J_{GNNexp} = (y - \hat{y})^2 + \nu \sum_{i,j} \mathbf{M}_{ij} + ent(\mathbf{M}) \ . \tag{4.33}$$

The first term in Equation 4.33 encourages minimal change in the sfDLN output when the mask $\mathbf{M}$ is applied. The second term is an L1 penalty, encouraging a sparse mask

with a regularization parameter $\nu = 0.01$. The final term is an entropy function that enhances contrast in the mask weights by driving small values toward 0 and large values toward 1.

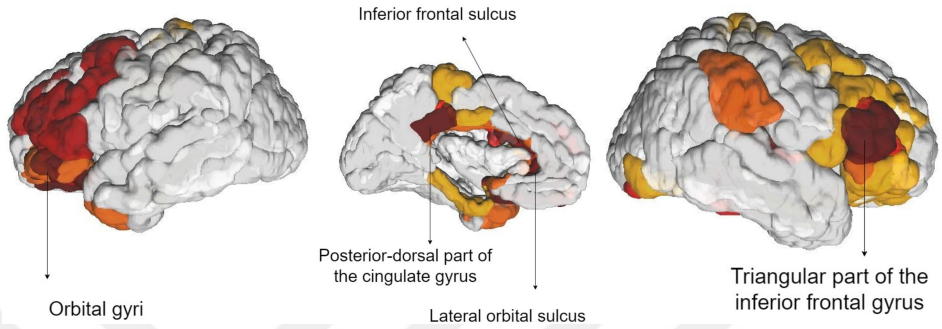

Figure 4.5. All cortical regions associated with significant connections contributing to discrepancy scoring, as identified by the GNNExplainer.

To identify the source of the observed discrepancy, we used pretained DS + GCN sfDLN on all ADD and SCI patients to generate y and $\hat{y}$. We then determined the importance map $\breve{\mathbf{M}}$ that minimizes the loss in Equation 4.33 through gradient descent. The most significant connections contributing to the structure-function discrepancy were identified from the symmetrized edge-relevance map ($\breve{\mathbf{M}} + \breve{\mathbf{M}}^T$), selecting those with a magnitude at least 3 standard deviations greater than the mean across all edges. We also calculated a nodal importance score for each node, which was defined as the sum of the weights of the selected edges connected to the node. Figure 4.5 shows all nodes with a non-zero importance score, highlighting 5 nodes with a positive deviation from the mean node score exceeding 3σ. These nodes include the "left dorsal posterior cingulate cortex" (dPCC), "left orbital gyri" (OG), "left inferior frontal sulcus" (IFS), "left lateral orbital sulcus" (LOS), and "right triangular part of the inferior frontal gyrus" (IFG).

4.4. Disentangled Attention Graph Neural Network

To evaluate the potential of the DAGNN model, we focused on a more challenging task: the ADD-MCI classification. While sfDLN demonstrated strong performance in the ADD-SCI classification with little room for improvement, it showed limited impact on ADD-MCI classification. This provided an opportunity for DAGNN to demonstrate its strengths, particularly in addressing the more nuanced ADD-MCI classification task. For this classification task, we focused on sNETs.

The ADD-MCI classification was conducted using 7-fold cross-validation, consistent with previous experimental setups. Cross-validation experiments are repeated 20 times to eliminate the impact of random seeds. Performance was evaluated using Accuracy, F1-Score, and AUC metrics. Training details and model hyperparameters are summarized in Table 4.14.

Table 4.24. Summary of DAGNN Experimental Configurations for Classification Tasks.

Description	**ADD-MCI**
Label Distribution	18 ADD/46 MCI
Number of Folds	8
Batch Size	6
Number of Epochs	100
Learning Rate	0.001
Optimizer	Adam
Weight Decay	0.001
λ	0.1
m	1.5

DAGNN is designed to enhance attention-based GNNs by fully leveraging the multi-head attention mechanism, with each head explicitly disentangled to focus on

different brain regions. This disentanglement aims to improve performance and provide inherent explainability without requiring additional techniques like those employed in sfDLN. To evaluate the benefits of disentangled attention mechanisms, the following experiments were conducted:

- ADD-MCI Experiments: To assess the full potential of DAGNN, the ADD-MCI classification task was chosen as the primary evaluation. The impact of disentanglement was investigated by comparing the same attention-based GNN models with and without disentanglement. To further demonstrate robustness, experiments were conducted using two types of attention-based GNNs. Additionally, the effect of the number of attention heads on model performance, both with and without disentanglement, was systematically examined. The results were used to identify the best architecture for DAGNN.
- DAGNN Characteristics: The outputs of attention heads in DAGNN were compared to those of conventional attention-based GNNs to highlight the enhancements introduced by disentanglement. Furthermore, a neurological interpretation was attempted by identifying the brain regions most prominently highlighted by DAGNN, providing insights into its explainability.

Table 4.25. Performance of DAGNN compared with baselines on Binary and Log-Scaled sNETs.

ADD-MCI Classification						
Network	**Binary sNET**			**Log-Scaled sNET**		
	Acc	F1	AUC	Acc	F1	AUC
DS + 4-Head GAT	77.4±2.1	56.5±3.9	80.1±1.4	77.1±1.3	56.2±3.2	80.6±1.1
DS + 4-Head TransformerGAT	75.3±2.3	52.5±4.9	78.7±1.9	76.2±2.2	53.0±3.4	79.2±2.0
DAGNN(DS + 4-Head GAT)	80.4±2.0	60.1±3.8	85.3±3.5	79.8±2.4	58.5±4.0	84.8±2.2
DAGNN(DS + 4-Head TransformerGAT)	79.2±1.4	59.3±3.0	84.6±2.2	80.1±2.2	57.0±3.0	82.9±1.7

To ensure a fair comparison, we followed the same setup as in the sNET experiments, where each DS + GNN generates 128-dimensional embeddings. First, we used DS + 4-Head GAT and DS + 4-Head TransformerGAT as baselines. We then compared the disentangled versions to these baselines to demonstrate the improvement achieved through disentanglement. Both binary sNET and Log-scaled sNETs were tested to evaluate the robustness of the DAGNN.

The results, shown in Table 4.25, indicate that DAGNN outperforms the baselines across all three metrics. This highlights that disentanglement not only enhances GAT but also significantly improves the stability and generalization capability of any attention-based GNN, further reinforcing its effectiveness in different settings. Based on these findings, we proceeded with experiments using binary sNETs, as they showed superior performance in the ADD-MCI classification task.

Table 4.26. Performance of GAT with and without disentanglement with varying head numbers.

ADD-MCI Classification						
Network	**without Disentanglement**			**w/ Disentanglement**		
	Acc	F1	AUC	Acc	F1	AUC
DS + 4-Head GAT	77.4±2.1	56.5±3.9	80.1±1.4	80.4±2.0	60.1±3.8	85.3±3.5
DS + 8-Head GAT	75.5±2.5	54.7±5.5	79.2±2.1	80.8±1.9	60.5±4.9	84.6±2.0
DS + 12-Head GAT	76.2±2.3	53.6±7.8	79.0±1.7	79.8±2.4	59.6±5.0	83.7±2.4
DS + 16-Head GAT	74.4±2.7	51.0±6.4	77.2±1.9	79.4±2.5	58.7±5.6	83.6±2.7

To demonstrate the robustness of DAGNN, we conducted additional experiments with varying numbers of attention heads. Each head was set to output a 32-dimensional embedding, and the final representation was obtained by concatenating the outputs of all heads. While increasing the number of attention heads generally leads to overfitting due to the increased complexity relative to the limited dataset size, as shown in Table 4.26 and 4.27 , the DAGNN versions of both GAT and TransformerGAT were less

affected by this issue compared to the models that did not use disentanglement. This phenomenon illustrates that disentangling the attention heads enhances the models' capacity, making them more resilient to overfitting.

Table 4.27. Performance of TransformerGAT with and without disentanglement with varying head numbers.

ADD-MCI Classification						
Network	**without Disentanglement**			**w/ Disentanglement**		
	Acc	F1	AUC	Acc	F1	AUC
DS + 4-Head TransformerGAT	75.3±2.3	52.5±4.9	78.7±1.9	79.2±1.4	59.3±3.0	84.6±2.2
DS + 8-Head TransformerGAT	72.3±3.2	47.0±5.1	76.7±3.3	79.2±1.0	57.2±3.2	83.9±2.2
DS + 12-Head TransformerGAT	73.2±2.2	47.0±3.7	77.6±2.1	78.7±3.0	55.4±2.9	84.0±1.8
DS + 16-Head TransformerGAT	71.7±2.4	42.4±5.5	77.7±2.2	77.9±2.5	55.0±7.2	82.0±5.7

We also compared DAGNN with other disentanglement-based methods, including FactorGCN [137] and DisenGCN [138], as shown in Table 4.28. FactorGCN generates factor graphs from the input graph, where these factor graphs are constructed using a multi-head attention mechanism. It assigns arbitrary labels to the factor graphs and assumes that these factor graphs can be distinguished by a classifier. Each factor graph is then processed with a GCN. To make a fair comparison between DAGNN and FactorGCN, we adopted the classifier-wise disentanglement approach from FactorGCN and replaced our disentanglement loss in Equation 3.52 with a classifier-based loss

$$\mathcal{L}_{factor} = L_{cls}(j, GNN_c(\hat{\mathbf{A}}^{b,j})), \tag{4.34}$$

where GNN_c is a GNN encoder based classifier, and j denotes the label of factor graph $\hat{\mathbf{A}}^{b,j}$. This variant will be referred to as Factor*. On the other hand, DisenGCN aims to disentangle latent node representations by partitioning neighbors using a routing

mechanism, where specific features correspond to specific paths. Unlike FactorGCN, DisenGCN does not use an attention mechanism and employs disentanglement at the feature level.

Table 4.28. Comparison of DAGNN with other disentangled-based methods.

ADD-MCI Classification			
Model	Acc	F1	AUC
DAGNN (DS + 4-Head GAT)	**80.4±2.0**	**60.1±3.8**	**85.3±3.5**
Factor* (DS + 4-Head GAT)	79.3±3.0	59.1±4.1	83.0±3.2
DisenGCN	67.5±4.1	44.8±6.7	69.9±3.5

Table 4.28 shows that DAGNN outperforms other disentanglement-based methods. DisenGCN performs poorly compared to the others, while Factor* shows competitive results. The primary difference between DAGNN and Factor* is that DAGNN directly performs disentanglement, whereas Factor* assumes that attention coefficients from different heads can be distinguished in the latent space. This indirect approach may lead to less effective disentanglement compared to the direct approach of DAGNN, as even minor changes in the coefficients could result in larger differences in the latent space. Additionally, the L1 distance used in DAGNN further enhances the attention coefficients by encouraging sparsity, which may improve the model's performance.

We focused on the latent representations generated by DAGNN for a more in-depth analysis. Specifically, we compared the $\mathbf{H}^k$ representations from different heads in DAGNN with those from a model without disentanglement, which we refer to as AGNN. For this analysis, we used DS + 4-Head GAT trained on the entire ADD-MCI dataset and examined the resulting latent representations. The results of this analysis are presented in Figure 4.6.

The superior performance of DAGNN can be attributed to its ability to generate decorrelated latent representations, localized at the mesoscale. As shown in Figure 4.6,

DAGNN learns more distinct latent representations compared to AGNN, as evidenced by the reduced correlation values between the off-diagonal elements.

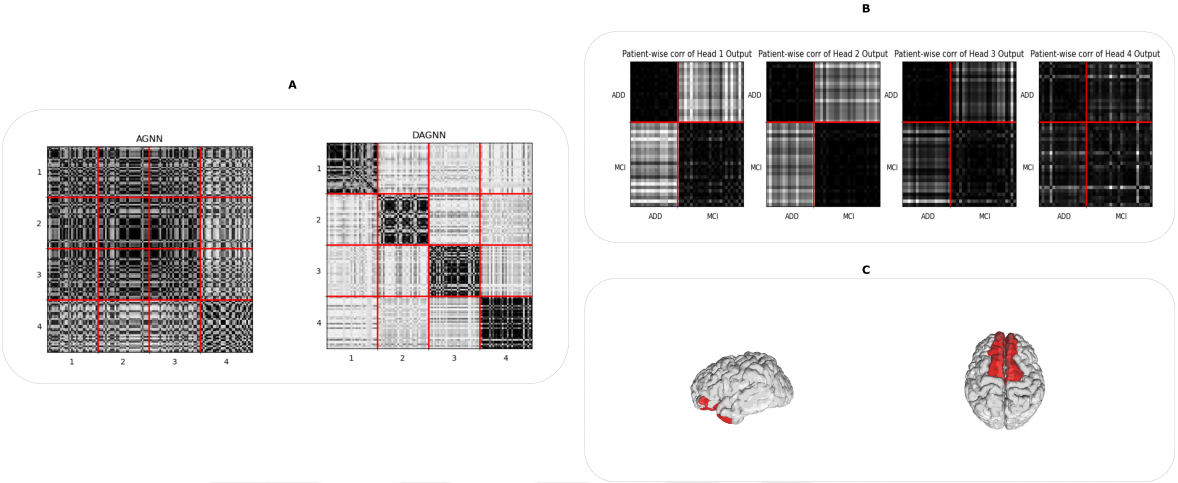

Figure 4.6. DAGNN Analysis.(A) Correlation matrices of features among latent representations generated by AGNN and DAGNN. The numerical annotations on the figures shows that which head contributes to which part of the representations. (B) Correlation of latent representations between patients for each head output is shown. (C) Highlighted brain regions by Head 1 (left) and Head 2 (right) are given.

Additionally, we explored which attention heads are most significant in distinguishing patients. To do this, we computed the correlation coefficient matrix across all patients for each $\mathbf{H}^p$, the GAT output for patient p. As shown in Figure 4.6, attention heads 1 and 2 exhibit lower inter-class correlations, suggesting that the representations they generate are highly discriminatory. To investigate which brain regions these heads highlight, we calculated the mean attention map, denoted as $\ddot{\mathbf{A}}^k$, by averaging $\hat{\mathbf{A}}^{p,k}$ across all patients. We then treated the row sum of $\ddot{\mathbf{A}}^k$ as the importance score for brain regions. For both heads, we identified two nodes with positive deviations from the mean node score exceeding 3 standard deviations, as shown in Figure 2. These highlighted regions were colored in red on the brain images. Our analysis revealed that head 1 focuses on the left Orbital gyri and left Temporal pole, while head 2 highlights the left and right Superior frontal gyri. Notably, three of these regions (left

Orbital gyri, left Superior frontal gyrus, and right Superior frontal gyrus) are part of the DMN [139], which is known to be impacted by ADD progression [140].

5. DISCUSSION

This thesis investigates the potential of GNNs for ADD diagnosis, particularly using brain networks as the input. To address gaps in the existing literature, three primary objectives were set, each focusing on aspects of GNN applications that have not been fully explored or resolved. For each of these objectives, we proposed novel methodologies and strategies to enhance our understanding of the capabilities of GNNs in ADD-related tasks.

All three objectives in this thesis focused on ADD-MCI, ADD-SCI, or both classifications. MCI-SCI classification is omitted in this thesis due to its increasing complexity and being the least studied classification in the current literature.

The first objective was to systematically evaluate various GNN models for their applicability to structural and functional brain network analysis in the context of ADD. This evaluation was driven by the aim to showcase the superiority of GNNs in analyzing complex brain network data. The key novelty of GNNs lies in their message-passing scheme, where node representations are iteratively updated through the aggregation of neighboring nodes. This mechanism of local message propagation allows GNNs to capture relationships and dependencies that are inherently present in graph-structured data, making them highly effective for graph-based tasks across various domains.

In the context of brain networks, the advantage of GNNs over traditional methods lies in their ability to model the intricate interactions between different regions of the brain, which are critical for understanding disease progression. To validate this, we compared GNN-based models with baseline methods that did not employ the message-passing mechanism. Our results demonstrated that GNNs offer significant improvements in the analysis of sNETs for ADD diagnosis. Specifically, we observed a marked increase in classification performance when GNN models were applied to sNETs, highlighting the potential of GNNs to better capture the underlying patterns

of connectivity that characterize ADD.

The results for fNETs were somewhat disappointing. While GNNs demonstrated promising performance on sNETs, they did not show significant improvements over the baseline when applied to fNETs. This suggests that the message-passing mechanism inherent in GNNs may not effectively capture the dynamic and temporal nature of functional connectivity. Therefore, alternative models or additional modifications may be needed to fully leverage the potential of functional networks in ADD diagnosis. These findings provide important insights into the strengths and limitations of GNNs in ADD-related tasks, paving the way for further research aimed at refining GNN-based methods for analyzing functional brain networks.

Among the various GNN models evaluated, GAT demonstrated particularly strong performance across both tasks in sNET analysis. This highlights the potential of the attention mechanism in brain network analysis, as it allows the model to focus on the most relevant connections within the brain's network. Not only did GAT improve classification performance, but it also offered the ability to generate meaningful explanations by examining the attention coefficients computed across different attention heads. This interpretability feature is an important advantage, especially in the context of medical diagnoses like ADD.

Furthermore, we demonstrated that using connectivity information as both node features and the underlying graph structure for GNNs yielded satisfactory results in sNET analysis. Enhancing this approach by encoding the connections with a MLP-based encoder further boosted the GNN's performance on sNET data, underscoring the value of incorporating such encoding techniques for more robust feature representation.

Additionally, multi-modal analysis was explored in this thesis, based on the widely accepted assumption that structure and function provide complementary information. However, our results challenge this assumption. Specifically, we found that the multi-modal model did not lead to a significant improvement over the uni-modal sNET-based

model. This suggests that structure and function may require a more nuanced approach for joint analysis.

The first objective centered on analyzing brain networks using a GNN-based approach. Information related to ADD can be integrated into GNN to build and train models that enhance structural, functional, or multi-modal analyses. Our second objective was to identify a biomarker capable of diagnosing ADD while also monitoring its progression. This goal was achieved by combining an ADD-specific hypothesis with GNN-based models. Specifically, we proposed modeling the relationship between brain structure and function using a GNN architecture, and we utilized this relationship as a biomarker for ADD. We referred to this structure-function relationship as the structure-function discrepancy score, which quantifies the mismatch between the brain's structure and function with a bounded score. Proposed model is called as sfDLN.

The results demonstrate that the sfDLN is more effective in capturing the multi-modal relationship compared to traditional multi-modal models that treat structure and function as complementary for ADD-SCI classification. The sfDLN is trained under the hypothesis that the structure-function discrepancy increases with cognitive decline. To validate this hypothesis, we conducted a comparison with an alternative sfDLN model trained on a reversed hypothesis, which assumes that the structure-function discrepancy is inversely proportional to cognitive decline. The findings confirmed that this reversed hypothesis does not hold, as the model based on it yielded lower classification accuracy than both the original sfDLN and the conventional multi-modal model. These results reinforce the validity of our hypothesis and highlight the sfDLN's capability in modeling structure-function relationships for cognitive decline analysis.

Interestingly, GCN model came up with the best performing one with sfDLN model. This observation highlights the context-dependent nature of model performance. While Max GraphSAGE and GAT excelled in uni-modal sNET analysis, the shift to multi-modal analysis underscores that different tasks require different ap-

proaches. In this context, mean aggregation-based GNNs, such as GCN, ChebyNet, and Mean GraphSAGE, outperformed other models, suggesting that the smoothing effect inheent in these methods is more effective for modeling the relationship between structure and function.

To train the sfDLN, we used log-scaled sNET and ℓNET to represent structural and functional networks, respectively, based on findings from earlier experiments. The ℓNET has been shown to be more effective than correlation-based fNETs, and its sparsity makes it a good candidate for working alongside the sparse sNETs. To validate this approach, we trained the sfDLN using Pearson fNET instead of ℓNET. This resulted in a dramatic decrease in performance, demonstrating that ℓNET is more suitable for both GNN processing and pairing with sNETs.

The structure-function discrepancy score also demonstrates potential for monitoring ADD progression. When the sfDLN, trained on ADD-SCI subjects, is applied to MCI subjects, their predicted scores are distributed between those of ADD and SCI subjects. This suggests that the structure-function discrepancy score aligns consistently with the stages of ADD progression. To further validate this finding, we conducted control experiments to ensure that this behavior is not due to chance or random factors, thereby strengthening the reliability of the score as a biomarker for ADD staging.

Classification performance of a model is not sufficient on its own. The sfDLN model offers some level of explainability by proposing the hypothesis that an increasing discrepancy score correlates with cognitive decline. However, it still exhibits black-box characteristics. To better understand the progression of ADD, it is crucial to identify which brain regions contribute to the discrepancy score. To address this, we employed an explanation-based framework on sfDLN and highlighted the sources of discrepancy. The highlighted brain networks were interpreted and found to be neurologically plausible in collaboration with our neurologist partners.

sfDLN provides a robust model for classifying ADD-SCI and monitoring disease progression. However, it does not adequately address the more complex classification of ADD-MCI. To address this challenge, we pursued our third and final objective by proposing a novel GNN model aimed at improving performance in ADD-MCI classification. This model emphasizes the identification of distinctive subnetworks associated with ADD. To accomplish this, we introduced a disentanglement-based approach, offering an enhancement over conventional attention-based GNNs.

Based on the findings from earlier GNN experiments, the GAT demonstrates strong performance in classifying ADD-MCI within sNET analysis. In addition to its effectiveness, GAT has the capability to identify subnetworks through its multi-head mechanism. Each head generates attention coefficients, which can be interpreted as subnetworks associated with ADD. This feature of GAT enables a more detailed mesoscale analysis of ADD.

We observed a significant limitation within the GAT. Although each attention head is initialized independently, they tend to converge to a state where they produce nearly identical attention coefficients. This results in only one subnetwork being highlighted across all heads, causing the GAT to potentially overlook important aspects of the graph. To address this issue, we proposed a novel disentanglement loss that maximizes the distance between the attention coefficients across different heads. By combining this loss with GAT, or any other attention-based GNN, we introduce the DAGNN model. This approach ensures that each head focuses on a different part of the graph, increasing the model's capacity and enabling it to identify multiple important subnetworks simultaneously.

We experimented with both log-scaled and binary sNETs, and initial results demonstrated that DAGNN consistently outperformed conventional attention-based GNNs. To further investigate the robustness of DAGNN, we conducted additional experiments using varying numbers of attention heads across two standard attention-based GNNs: GAT and TransformerGAT. The experimental results revealed that

DAGNN maintained superior performance across these configurations, highlighting its effectiveness in diverse scenarios. Moreover, DAGNN exhibited greater resilience to overfitting compared to conventional attention-based GNNs as the number of attention heads increased. This robustness can be attributed to DAGNN's disentanglement mechanism, which enhances its capacity by ensuring that each attention head learns distinct and complementary features.

DAGNN not only disentangles attention coefficients across heads but also generates decorrelated features across them, unlike conventional attention-based GNNs, which produce correlated features. This dual disentanglement—both coefficient-wise and feature-wise—enhances DAGNN's capacity to model complex relationships effectively. As a result, DAGNN proves to be a more robust and efficient model.

Furthermore, by analyzing the attention coefficients produced by DAGNN across different heads, we identified specific brain regions associated with ADD progression. These findings provide valuable insights into the neurological basis of ADD, offering both interpretability and potential clinical relevance.

6. CONCLUSION

This thesis explored the potential of GNNs for ADD diagnosis using brain networks. The study aimed to address gaps in the existing literature by proposing novel methodologies to improve GNN applications in ADD-related tasks. The first objective was to evaluate GNN models for their effectiveness in analyzing structural and functional brain networks. Results showed that GNNs, particularly GAT, performed well on sNETs, but the approach was less effective for functional networks fNETs, highlighting the need for more specialized models to capture the dynamic nature of functional connectivity. Nevertheless, a novel functional network, ℓNET is proposed as a better alternative for conventional fNETs.

The second objective focused on utilizing GNNs to propose a new biomarker for diagnosing and monitoring ADD progression. The structure-function discrepancy score, modeled through the sfDLN, showed promise in tracking cognitive decline by quantifying mismatches between brain structure and function. This score was validated through experiments that demonstrated its superiority in ADD-SCI classification and correlation with the various stages of ADD, which supports its clinical relevance and potential as a staging biomarker. Additionally, specific brain regions responsible for the observed discrepancies are identified to provide interpretability.

The third objective focused on improving the ADD-MCI classification by introducing the DAGNN. By overcoming the limitations of traditional attention-based GNNs, DAGNN enhanced the model's capacity to identify multiple distinct subnetworks within the brain. This innovative approach outperformed other models in terms of performance and robustness, demonstrating DAGNN's resilience to overfitting and its ability to capture complex brain connectivity patterns. Furthermore, the model's ability to generate disease-related subnetworks may deepen our understanding of ADD.

In conclusion, this thesis demonstrates the effectiveness of GNNs in analyzing brain networks for ADD diagnosis and understanding the ADD progression . By introducing novel approaches, such as the structure-function discrepancy score and the DAGNN model, the research advances our understanding of ADD and offers new avenues for improving diagnostic accuracy. These findings provide a foundation for future work that could refine these models, explore additional modalities, and expand their clinical applicability in larger and more diverse datasets.

Future work should concentrate on refining and optimizing the proposed models, especially by addressing their limitations in capturing the dynamic nature of functional brain networks. Furthermore, incorporating larger and more diverse datasets, including data from multiple centers, could improve the generalizability and robustness of the models. Lastly, exploring a GNN-based model, which has shown strong performance in the MCI-SCI classification, would be a valuable direction for future research.

REFERENCES

1. "2023 Alzheimer's disease facts and figures", https://alz-journals.onlinelibrary.wiley.com/doi/abs/10.1002/alz.13016, accessed on October 29, 2024.

2. Mosconi, L., V. Berti, L. Glodzik, A. Pupi, S. D. Santi and M. J. de Leon, "Pre-Clinical Detection of Alzheimer's Disease Using FDG-PET, with or without Amyloid Imaging", *Journal of Alzheimer's Disease*, Vol. 20, No. 3, pp. 843–854, 2010.

3. Mirra, S. S., A. Heyman, D. McKeel, S. M. Sumi, B. J. Crain, L. M. Brownlee, F. S. Vogel, J. P. Hughes, G. van Belle and L. Berg, "The Consortium to Establish a Registry for Alzheimer's Disease (CERAD). Part II. Standardization of the neuropathologic assessment of Alzheimer's disease", *Neurology*, Vol. 41, No. 4, pp. 479–486, 1991.

4. Sporns, O., "The Human Connectome: A Complex Network", *Annals of the New York Academy of Sciences*, Vol. 1224, pp. 109–125, 2011.

5. Destrieux, C., B. Fischl, A. Dale and E. Halgren, "Automatic Parcellation of Human Cortical Gyri and Sulci Using Standard Anatomical Nomenclature", *NeuroImage*, Vol. 53, No. 1, pp. 1–15, 2010.

6. Schaefer, A., R. Kong, E. M. Gordon, T. O. Laumann, X.-N. Zuo, A. J. Holmes, S. B. Eickhoff and B. T. T. Yeo, "Local-Global Parcellation of the Human Cerebral Cortex from Intrinsic Functional Connectivity MRI", *Cerebral Cortex*, Vol. 28, No. 9, pp. 3095–3114, 2018.

7. Sporns, O., G. Tononi and R. Kötter, "The Human Connectome: A Structural Description of the Human Brain", *PLoS Computational Biology*, Vol. 1, No. 4, p.

e42, 2005.

8. Sanz-Arigita, E. J., M. M. Schoonheim, J. S. Damoiseaux, S. A. R. B. Rombouts, E. Maris, F. Barkhof, P. Scheltens and C. J. Stam, "Loss of 'Small-World' Networks in Alzheimer's Disease: Graph Analysis of fMRI Resting-State Functional Connectivity", *PLOS ONE*, Vol. 5, No. 11, pp. 1–14, 2010.

9. Supekar, K., V. Menon, D. Rubin, M. Musen and M. D. Greicius, "Network Analysis of Intrinsic Functional Brain Connectivity in Alzheimer's Disease", *PLOS Computational Biology*, Vol. 4, No. 6, pp. 1–11, 2008.

10. Zhao, X., Y. Liu, X. Wang, B. Liu, Q. Xi, Q. Guo, H. Jiang, T. Jiang and P. Wang, "Disrupted Small-world Brain Networks in Moderate Alzheimer's Disease: A Resting-state fMRI Study", *PLoS One*, Vol. 7, No. 3, p. e33540, 2012.

11. Sun, Y., Q. Yin, R. Fang, X. Yan, Y. Wang, A. Bezerianos, H. Tang, F. Miao and J. Sun, "Disrupted Functional Brain Connectivity and Its Association to Structural Connectivity in Amnestic Mild Cognitive Impairment and Alzheimer's Disease", *PLOS ONE*, Vol. 9, No. 5, pp. 1–14, 2014.

12. Çiftçi, K., "Minimum Spanning Tree Reflects the Alterations of the Default Mode Network During Alzheimer's Disease", *Annals of Biomedical Engineering*, Vol. 39, No. 5, pp. 1493–1504, 2011.

13. Dai, Z., C. Yan, K. Li, Z. Wang, J. Wang, M. Cao, Q. Lin, N. Shu, M. Xia, Y. Bi and Y. He, "Identifying and Mapping Connectivity Patterns of Brain Network Hubs in Alzheimer's Disease", *Cerebral Cortex*, Vol. 25, No. 10, pp. 3723–3742, 2014.

14. Ren, H., J. Zhu, X. Su, S. Chen, S. Zeng, X. Lan, L.-Y. Zou, M. E. Sughrue and Y. Guo, "Application of Structural and Functional Connectome Mismatch for Classification and Individualized Therapy in Alzheimer Disease", *Frontiers in*

Public Health, Vol. 8, p. 584430, 2020.

15. Lo, C. Y., P. N. Wang, K. H. Chou, J. Wang and Y. He, "Diffusion Tensor Tractography Reveals Abnormal Topological Organization in Structural Cortical Networks in Alzheimer's Disease", *The Journal of Neuroscience*, Vol. 30, No. 50, pp. 16876–85, 12 2010.

16. John, M., T. Ikuta and J. Ferbinteanu, "Graph Analysis of Structural Brain Networks in Alzheimer's Disease: Beyond Small World Properties", *Brain Structure & Function*, Vol. 222, No. 2, pp. 923–942, 2017.

17. Bullmore, E. and O. Sporns, "Complex Brain Networks: Graph Theoretical Analysis of Structural and Functional Systems", *Nature Reviews Neuroscience*, Vol. 10, No. 3, pp. 186–198, 2009.

18. Chen, G., H.-Y. Zhang, C. Xie, G. Chen, Z.-J. Zhang, G.-J. Teng and S.-J. Li, "Modular Reorganization of Brain Resting State Networks and Its Independent Validation in Alzheimer's Disease Patients", *Frontiers in Human Neuroscience*, Vol. 7, p. 456, 2013.

19. Durusoy, G., Z. Yıldırım, D. Y. Dal, C. Ulasoglu-Yildiz, E. Kurt, G. Bayır, E. Özacar, E. Özarslan, A. Demirtaş-Tatlıdede, B. Bilgiç, T. Demiralp, H. Gurvit, A. Kabakçıoğlu and B. Acar, "B-Tensor: Brain Connectome Tensor Factorization for Alzheimer's Disease", *IEEE Journal of Biomedical and Health Informatics*, Vol. 25, No. 5, pp. 1591–1600, 2021.

20. Jie, B., D. Zhang, C.-Y. Wee and D. Shen, "Topological Graph Kernel on Multiple Thresholded Functional Connectivity Networks for Mild Cognitive Impairment Classification", *Human Brain Mapping*, Vol. 35, No. 7, p. 2876–2897, 2014.

21. Bronstein, M. M., A. Jacques, P. Vandergheynst, T. Kipf, M. Welling and J. Bruna, "Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and

Gauges", ArXiv:2104.13478 [cs], 2024.

22. Xing, X., Q. Li, H. Wei, M. Zhang, Y. Zhan, X. S. Zhou, Z. Xue and F. Shi, "DS-GCNs: Connectome Classification Using Dynamic Spectral Graph Convolution Networks with Assistant Task Training", *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 639–646, Shenzen, China, 2019.

23. Parisot, S., S. I. Ktena, E. Ferrante, M. Lee, R. Guerrero, B. Glocker and D. Rueckert, "Disease Prediction Using Graph Convolutional Networks: Application to Autism Spectrum Disorder and Alzheimer's Disease", *Medical Image Analysis*, Vol. 48, pp. 117–130, 2018.

24. Kong, Y., W. Wang, X. Liu, S. Gao, Z. Hou, C. Xie, Z. Zhang and Y. Yuan, "Multi-Connectivity Representation Learning Network for Major Depressive Disorder Diagnosis", *IEEE Transactions on Medical Imaging*, Vol. 42, No. 10, pp. 3012–3024, 2023.

25. Kazi, A., S. Shekarforoush, S. A. Krishna, H. Burwinkel, G. Vivar, K. Kortüm, S.-A. Ahmadi, S. Albarqouni and N. Navab, "InceptionGCN: Receptive Field Aware Graph Convolutional Network for Disease Prediction", *Information Processing in Medical Imaging*, pp. 73–85, Hong Kong, China, 2019.

26. Ghorbani, M., A. Kazi, M. S. Baghshah, H. R. Rabiee and N. Navab, "RA-GCN: Graph Convolutional Network for Disease Prediction Problems with Imbalanced Data", *Medical Image Analysis*, Vol. 75, p. 102272, 2022.

27. Lei, B., Y. Zhu, S. Yu, H. Hu, Y. Xu, G. Yue, T. Wang, C. Zhao, S. Chen, P. Yang, X. Song, X. Xiao and S. Wang, "Multi-scale Enhanced Graph Convolutional Network for Mild Cognitive Impairment Detection", *Pattern Recognition*, Vol. 134, p. 109106, 2023.

28. Qiu, Y., S. Yu, Y. Zhou, D. Liu, X. Song, T. Wang and B. Lei, "Multi-channel Sparse Graph Transformer Network for Early Alzheimer's Disease Identification", *IEEE 18th International Symposium on Biomedical Imaging*, pp. 1794–1797, Nice, France, 2021.

29. Song, X., A. Frangi, X. Xiao and S. Wang, "Integrating Similarity Awareness and Adaptive Calibration in Graph Convolution Network to Predict Disease", *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 121–130, 2020.

30. Jiang, H., P. Cao, M. Xu, J. Yang and O. Zaiane, "Hi-GCN: A Hierarchical Graph Convolution Network for Graph Embedding Learning of Brain Network and Brain Disorders Prediction", *Computers in Biology and Medicine*, Vol. 127, p. 104096, 2020.

31. Chen, H., F. Zhuang, L. Xiao, L. Ma, H. Liu, R. Zhang, H. Jiang and Q. He, "AMA-GCN: Adaptive Multi-layer Aggregation Graph Convolutional Network for Disease Prediction", *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp. 2202–2208, 2021.

32. Pan, J., H. Lin, Y. Dong, Y. Wang and Y. Ji, "MAMF-GCN: Multi-scale Adaptive Multi-Channel Fusion Deep Graph Convolutional Network for Predicting Mental Disorder", *Computers in Biology and Medicine*, Vol. 148, p. 105823, 2022.

33. Huang, Y. and A. C. S. Chung, "Edge-Variational Graph Convolutional Networks for Uncertainty-Aware Disease Prediction", *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 562–572, 2020.

34. Zhang, H., R. Song, L. Wang, L. Zhang, D. Wang, C. Wang and W. Zhang, "Classification of Brain Disorders in rs-fMRI via Local-to-Global Graph Neural Networks", *IEEE Transactions on Medical Imaging*, Vol. 42, No. 2, pp. 444–455, 2023.

35. Hu, Y., J. Wang, H. Zhu, J. Li and J. Shi, "Cost-Sensitive Weighted Contrastive Learning Based on Graph Convolutional Networks for Imbalanced Alzheimer's Disease Staging", *IEEE Transactions on Medical Imaging*, Vol. 43, No. 9, pp. 3126–3136, 2024.

36. Yang, F., H. Wang, S. Wei, G. Sun, Y. Chen and L. Tao, "Multi-model Adaptive Fusion-based Graph Network for Alzheimer's Disease Prediction", *Computers in Biology and Medicine*, Vol. 153, p. 106518, 2023.

37. Guan, Z., J. Yu, Z. Shi, X. Liu, R. Yu, T. Lai, C. Yang and H. Li, "Dynamic Graph Transformer Network via Dual-View Connectivity for Autism Spectrum Disorder Identification", *Computers in Biology and Medicine*, Vol. 174, p. 108415, 2024.

38. Ktena, S. I., S. Parisot, E. Ferrante, M. Rajchl, M. Lee, B. Glocker and D. Rueckert, "Metric Learning with Spectral Graph Convolutions on Brain Connectivity Networks", *NeuroImage*, Vol. 169, pp. 431–441, 2018.

39. Yao, D. and J. Sui, "A Mutual Multi-Scale Triplet Graph Convolutional Network for Classification of Brain Disorders Using Functional or Structural Connectivity", *IEEE Transactions on Biomedical Engineering*, Vol. 67, No. 9, pp. 2565–2575, 2020.

40. Gadgil, S., Q. Zhao, A. Pfefferbaum, E. V. Sullivan, E. Adeli and K. M. Pohl, "Spatio-Temporal Graph Convolution for Resting-State fMRI Analysis", *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 1–9, 2020.

41. Cui, W., J. Du, M. Sun, S. Zhu, S. Zhao, Z. Peng, L. Tan and Y. Li, "Dynamic Multi-Site Graph Convolutional Network for Autism Spectrum Disorder Identification", *Computers in Biology and Medicine*, Vol. 163, p. 106749, 2023.

42. Zhang, Y., Y. Wang, Y. Zhang, Y. Zhang, Y. Zhang and Y. Zhang, "A Cascaded

Multi-modality Analysis in Mild Cognitive Impairment", *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 1–9, Shenzhen, China, 2019.

43. Kim, B.-H., J. C. Ye and J.-J. Kim, "Learning Dynamic Graph Representation of Brain Connectome with Spatio-Temporal Attention", *Proceedings of the 35th Conference on Neural Information Processing Systems*, pp. 4314–4327, 2021.

44. Azevedo, T., A. Campbell, R. Romero-Garcia, L. Passamonti, R. A. Bethlehem, P. Liò and N. Toschi, "A Deep Graph Neural Network Architecture for Modelling Spatio-temporal Dynamics in Resting-state Functional MRI Data", *Medical Image Analysis*, Vol. 79, p. 102471, 2022.

45. Chen, D. and L. Zhang, "FE-STGNN: Spatio-Temporal Graph Neural Network with Functional and Effective Connectivity Fusion for MCI Diagnosis", *Medical Image Computing and Computer-Assisted Intervention*, pp. 67–76, Vancouver, Canada, 2023.

46. Liu, L., G. Wen, P. Cao, T. Hong, J. Yang, X. Zhang and O. R. Zaiane, "BrainTGL: A Dynamic Graph Representation Learning Model for Brain Network Analysis", *Computers in Biology and Medicine*, Vol. 153, p. 106521, 2023.

47. Ma, Y., W. Cui, J. Liu, Y. Guo, H. Chen and Y. Li, "A Multi-Graph Cross-Attention-Based Region-Aware Feature Fusion Network Using Multi-Template for Brain Disorder Diagnosis", *IEEE Transactions on Medical Imaging*, Vol. 43, No. 3, pp. 1045–1059, 2024.

48. Sivgin, I., H. A. Bedel, Şaban Öztürk and T. Çukur, "A Plug-In Graph Neural Network to Boost Temporal Sensitivity in fMRI Analysis", *IEEE Journal of Biomedical and Health Informatics*, Vol. 27, No. 5, pp. 2023–2033, 2023.

49. D'Souza, N. S., M. B. Nebel, D. Crocetti, J. Robinson, S. Mostofsky and

A. Venkataraman, "M-GCN: A Multimodal Graph Convolutional Network to Integrate Functional and Structural Connectomics Data to Predict Multidimensional Phenotypic Characterizations", *Proceedings of the Fourth Conference on Medical Imaging with Deep Learning*, pp. 119–130, PMLR, 2021.

50. Liu, J., G. Ma, F. Jiang, C.-T. Lu, P. S. Yu and A. B. Ragin, "Community-preserving Graph Convolutions for Structural and Functional Joint Embedding of Brain Networks", *Proceedings of the 2019 IEEE International Conference on Big Data*, pp. 1163–1168, Los Angeles, USA, 2019.

51. Zhang, Y., Y. Zhang, Y. Wang, Y. Zhang, Y. Wang and Y. Zhang, "Deep Fusion of Brain Structure-Function in Mild Cognitive Impairment", *IEEE Transactions on Medical Imaging*, Vol. 40, No. 11, pp. 3140–3149, 2021.

52. Huang, J., L. Zhou, L. Wang and D. Zhang, "Attention-Diffusion-Bilinear Neural Network for Brain Network Analysis", *IEEE Transactions on Medical Imaging*, Vol. 39, No. 7, pp. 2541–2552, 2020.

53. Li, Y., Q. Wei, E. Adeli, K. M. Pohl and Q. Zhao, "Joint Graph Convolution for Analyzing Brain Structural and Functional Connectome", *Medical Image Computing and Computer-Assisted Intervention*, pp. 231–240, Singapore, 2022.

54. Xia, J., Y. H. Chan, D. Girish and J. C. Rajapakse, "IMG-GCN: Interpretable Modularity-Guided Structure-Function Interactions Learning for Brain Cognition and Disorder Analysis", *Medical Image Computing and Computer-Assisted Intervention*, pp. 470–480, Marakkesh, Morocco, 2024.

55. Amodeo, C., I. Fortel, O. Ajilore, L. Zhan, A. Leow and T. Tulabandhula, "Unified Embeddings of Structural and Functional Connectome via a Function-Constrained Structural Graph Variational Auto-Encoder", *Medical Image Computing and Computer-Assisted Intervention*, pp. 439–449, Singapore, 2022.

56. Liu, J., Y. Wang, Y. Zhang, Y. Wang and Y. Zhang, "An Enhanced Multi-Modal Brain Graph Network for Classifying Neuropsychiatric Disorders", *IEEE Transactions on Medical Imaging*, Vol. 41, No. 11, pp. 3140–3149, 2022.

57. Yang, Y., C. Ye, X. Guo, T. Wu, Y. Xiang and T. Ma, "Mapping Multi-Modal Brain Connectome for Brain Disorder Diagnosis via Cross-Modal Mutual Learning", *IEEE Transactions on Medical Imaging*, Vol. 43, No. 1, pp. 108–121, 2024.

58. Zhang, S., X. Chen, X. Shen, B. Ren, Z. Yu, H. Yang, X. Jiang, D. Shen, Y. Zhou and X.-Y. Zhang, "A-GCL: Adversarial Graph Contrastive Learning for fMRI Analysis to Diagnose Neurodevelopmental Disorders", *Medical Image Analysis*, Vol. 90, p. 102932, 2023.

59. Wen, G., P. Cao, L. Liu, J. Yang, X. Zhang, F. Wang and O. R. Zaiane, "Graph Self-Supervised Learning with Application to Brain Networks Analysis", *IEEE Journal of Biomedical and Health Informatics*, Vol. 27, No. 5, pp. 1791–1801, 2023.

60. Huang, L., X. Ye, M. Yang, L. Pan and S. H. Zheng, "MNC-Net: Multi-task Graph Structure Learning Based on Node Clustering for Early Parkinson's Disease Diagnosis", *Computers in Biology and Medicine*, Vol. 152, p. 106308, 2023.

61. Wen, G., P. Cao, H. Bao, W. Yang, T. Zheng and O. R. Zaiane, "MVS-GCN: A Prior Brain Structure Learning-Guided Multi-View Graph Convolution Network for Autism Spectrum Disorder Diagnosis", *Computers in Biology and Medicine*, Vol. 142, p. 105239, 2022.

62. Chen, D., M. Liu, Z. Shen, X. Zhao, Q. Wang and L. Zhang, "Learnable Subdivision Graph Neural Network for Functional Brain Network Analysis and Interpretable Cognitive Disorder Diagnosis", *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 41–50, Vancouver, Canada, 2023.

63. Liu, M., H. Zhang, F. Shi and D. Shen, "Hierarchical Graph Convolutional Network Built by Multiscale Atlases for Brain Disorder Diagnosis Using Functional Connectivity", *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 34, No. 5, pp. 2271–2283, 2023.

64. Zhou, Z., Q. Wang, X. An, S. Chen, Y. Sun, G. Wang and G. Yan, "A Novel Graph Neural Network Method for Alzheimer's Disease Classification", *Computers in Biology and Medicine*, Vol. 180, p. 108869, 2024.

65. Li, N., J. Xiao, N. Mao, D. Cheng and X. Li, "Joint Learning of Multi-Level Dynamic Brain Networks for Autism Spectrum Disorder Diagnosis", *Computers in Biology and Medicine*, Vol. 171, p. 108054, 2024.

66. Yu, R., C. Pan, X. Fei, M. Chen and D. Shen, "Multi-Graph Attention Networks With Bilinear Convolution for Diagnosis of Schizophrenia", *IEEE Journal of Biomedical and Health Informatics*, Vol. 27, No. 3, pp. 1443–1454, 2023.

67. Zhang, Y. and H. Huang, "New Graph-Blind Convolutional Network for Brain Connectome Data Analysis", *Information Processing in Medical Imaging: 26th International Conference*, pp. 3–14, Hong Kong, China, 2019.

68. Yang, C., P. Wang, Y. Wu, S. Li, S. Lu and J. Zhang, "Autism Spectrum Disorder Diagnosis Using Graph Attention Network Based on Spatial-Constrained Sparse Functional Brain Networks", *Computers in Biology and Medicine*, Vol. 137, p. 104781, 2021.

69. Li, Y., J. Liu, Y. Jiang, Y. Liu and B. Lei, "Virtual Adversarial Training-Based Deep Feature Aggregation Network From Dynamic Effective Connectivity for MCI Identification", *IEEE Transactions on Medical Imaging*, Vol. 41, No. 1, pp. 237–251, 2022.

70. Xia, Z., H. Wang, T. Zhou, Z. Jiao and J. Lu, " Customized Relationship Graph

Neural Network for Brain Disorder Identification ", *Medical Image Computing and Computer Assisted Intervention*, pp. 109–118, Marakkesh, Morocco, 2024.

71. Wang, Y., H. Long, T. Bo and J. Zheng, "Residual Graph Transformer for Autism Spectrum Disorder Prediction", *Computer Methods and Programs in Biomedicine*, Vol. 247, p. 108065, 2024.

72. Kim, B.-H. and J. C. Ye, "Understanding Graph Isomorphism Network for rs-fMRI Functional Connectivity Analysis", *Frontiers in Neuroscience*, Vol. 14, p. 630, 2020.

73. Choi, I., G. Wu and W. H. Kim, "How Much to Aggregate: Learning Adaptive Node-Wise Scales on Graphs for Brain Networks", *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 2095–2105, Signapore, 2022.

74. Zhao, K., B. Duka, H. Xie, D. J. Oathes, V. Calhoun and Y. Zhang, "A Dynamic Graph Convolutional Neural Network Framework Reveals New Insights into Connectome Dysfunctions in ADHD", *NeuroImage*, Vol. 246, p. 118774, 2022.

75. Park, J. and M. K. Chung, "Convolving Directed Graph Edges via Hodge Laplacian for Brain Network Analysis", *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 789–799, Vancouver, Canada, 2023.

76. Huang, J., M. K. Chung and A. Qiu, "Heterogeneous Graph Convolutional Neural Network via Hodge-Laplacian for Brain Functional Data", *International Conference on Information Processing in Medical Imaging*, pp. 278–290, Bariloche, Argentina, 2023.

77. Qu, G., A. Orlichenko, J. Wang, G. Zhang, L. Xiao, K. Zhang, T. W. Wilson, J. M. Stephen, V. D. Calhoun and Y.-P. Wang, "Interpretable Cognitive Ability

Prediction: A Comprehensive Gated Graph Transformer Framework for Analyzing Functional Brain Networks", *IEEE Transactions on Medical Imaging*, Vol. 43, No. 4, pp. 1568–1578, 2024.

78. Jia, Y., Z. He, Z. Peng and Y. Yuan, "Hierarchical Graph Learning with Small-World Brain Connectomes for Cognitive Prediction", *Proceedings of the 27th International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 306–316, Marakkesh, Morocco, 2024.

79. Hwang, Y., S. Hwang, G. Wu and W. H. Kim, "Multi-order Simplex-Based Graph Neural Network for Brain Network Analysis", *Medical Image Computing and Computer-Assisted Intervention*, pp. 532–541, Marakkesh, Morocco, 2024.

80. Xu, J., Q. Bian, X. Li, A. Zhang, Y. Ke, M. Qiao, W. Zhang, W. K. J. Sim and B. Gulyás, "Contrastive Graph Pooling for Explainable Classification of Brain Networks", *IEEE Transactions on Medical Imaging*, Vol. 43, No. 9, pp. 3292–3305, 2024.

81. Huang, J., N. Chen and A. Qiu, "Topological Cycle Graph Attention Network for Brain Functional Connectivity", *Proceedings of the 27th International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 532–541, Marakkesh, Morocco, 2024.

82. Li, X., N. C. Dvornek, Y. Zhou, J. Zhuang, P. Ventola and J. S. Duncan, "Graph Neural Network for Interpreting Task-fMRI Biomarkers", *Medical Image Computing and Computer-Assisted Intervention*, pp. 349–357, Shenzen, China, 2019.

83. Cui, H., W. Dai, Y. Zhu, X. Li, L. He and C. Yang, "Interpretable Graph Neural Networks for Connectome-Based Brain Disorder Analysis", *Medical Image Computing and Computer-Assisted Intervention*, pp. 324–334, Singapore, 2022.

84. Li, X., Y. Zhou, N. Dvornek, M. Zhang, S. Gao, J. Zhuang, D. Scheinost, L. Staib,

P. Ventola and J. S. Duncan, "BrainGNN: Interpretable Brain Graph Neural Network for fMRI Analysis", *Medical Image Analysis*, Vol. 71, p. 102062, 2021.

85. Zheng, K., S. Yu, L. Chen, L. Dang and B. Chen, "BPI-GNN: Interpretable Brain Network-Based Psychiatric Diagnosis and Subtyping", *NeuroImage*, Vol. 273, p. 120594, 2024.

86. Bechler-Speicher, M., I. Amos, R. Gilad-Bachrach and A. Globerson, "Graph Neural Networks Use Graphs When They Shouldn't", *Proceedings of the 41st International Conference on Machine Learning*, pp. 3284 – 3304, Vienna, Austria, 2024.

87. Cui, H., W. Dai, Y. Zhu, X. Kan, A. A. C. Gu, J. Lukemire, L. Zhan, L. He, Y. Guo and C. Yang, "BrainGB: A Benchmark for Brain Network Analysis With Graph Neural Networks", *IEEE Transactions on Medical Imaging*, Vol. 42, No. 2, pp. 493–506, 2023.

88. Said, A., R. G. Bayrak, T. Derr, M. Shabbir, D. Moyer, C. Chang and X. Koutsoukos, "NeuroGraph: Benchmarks for Graph Machine Learning in Brain Connectomics", *Proceedings of the 37th Conference on Neural Information Processing Systems*, pp. 1–12, New Orleans, USA, 2023.

89. Wang, Z., Z. Dai, G. Gong, C. Zhou and Y. He, "Understanding Structural-functional Relationships in the Human Brain: a Large-scale Network Perspective", *PLoS One*, Vol. 21, No. 3, pp. 290–305, 2014.

90. Dai, Z., Q. Lin, T. Li, X. Wang, H. Yuan, X. Yu, Y. He and H. Wang, "Disrupted Structural and Functional Brain Networks in Alzheimer's Disease", *Neurobiology of Aging*, Vol. 75, pp. 71–82, 2019.

91. Gilmer, J., S. S. Schoenholz, P. F. Riley, O. Vinyals and G. E. Dahl, "Neural Message Passing for Quantum Chemistry", *Proceedings of the 34th International*

Conference on Machine Learning, p. 1263–1272, Sydney, Australia, 2017.

92. Krizhevsky, A., I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", *Communications of the ACM*, Vol. 60, No. 6, pp. 84–90, 2012.

93. He, K., X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Boston, USA, 2015.

94. Atwood, J. and D. Towsley, "Diffusion-Convolutional Neural Networks", *Proceedings of the 30th International Conference on Neural Information Processing Systems*, p. 2001–2009, Barcelona, Spain, 2016.

95. Hamilton, W. L., R. Ying and J. Leskovec, "Inductive Representation Learning on Large Graphs", ArXiv:1706.02216.

96. Chung, F. R., *Spectral Graph Theory*, American Mathematical Society, Providence, RI, 1997.

97. Bruna, J., W. Zaremba, A. Szlam and Y. Lecun, "Spectral Networks and Locally Connected Networks on Graphs", *International Conference on Learning Representations*, Banff, Canada, 2014.

98. Defferrard, M., X. Bresson and P. Vandergheynst, "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering", *Proceedings of the 30th Int. Conference on Neural Information Processing Systems*, pp. 3844–3852, Barcelona, Spain, 2016.

99. Shuman, D. I., S. K. Narang, P. Frossard, A. Ortega and P. Vandergheynst, "The Emerging Field of Signal Processing on Graphs: Extending High-dimensional Data Analysis to Networks and Other Irregular Domains", *IEEE Signal Processing Magazine*, Vol. 30, No. 3, pp. 83–98, 2013.

100. Kipf, T. N. and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks", *International Conference on Learning Representations*, Toulon, France, 2017.

101. Wu, F., A. H. de Souza Jr., T. Zhang, C. Fifty, T. Yu and K. Q. Weinberger, "Simplifying Graph Convolutional Networks", *Proceedings of the 36th International Conference on Machine Learning*, pp. 6861–6871, California, USA, 2019.

102. Levie, R., F. Monti, X. Bresson and M. M. Bronstein, "CayleyNets: Graph Convolutional Neural Networks With Complex Rational Spectral Filters", *IEEE Transactions on Signal Processing*, Vol. 67, No. 1, pp. 97–109, 2019.

103. He, M., Z. Wei, Z. Huang and H. Xu, "BernNet: Learning Arbitrary Graph Spectral Filters via Bernstein Approximation", *Proceedings of the 35th International Conference on Neural Information Processing Systems*, pp. 14239 – 14251, 2021.

104. Li, R., S. Wang, F. Zhu and J. Huang, "Adaptive Graph Convolutional Neural Networks", *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, pp. 3546 – 3553, New Orleans, Louisiana, USA, 2018.

105. Shervashidze, J., P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn and K. M. Borgwardt, "Weisfeiler-Lehman Graph Kernels", *Journal of Machine Learning Research*, Vol. 12, pp. 2539–2561, 2011.

106. Xu, K., W. Hu, J. Leskovec and S. Jegelka, "How Powerful are Graph Neural Networks?", *International Conference on Learning Representations*, New Orleans, USA, 2019.

107. Corso, G., L. Cavalleri, D. Beaini, P. Liò and P. Veličković, "Principal Neighbourhood Aggregation for Graph Nets", *Proceedings of the 34th International*

Conference on Neural Information Processing Systems, pp. 13260 – 13271, 2020.

108. Bouritsas, G., F. Frasca, S. Zafeiriou and M. M. Bronstein, "Improving Graph Neural Network Expressivity via Subgraph Isomorphism Counting", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 45, No. 1, pp. 657–668, 2023.

109. Sato, R., M. Yamada and H. Kashima, "Random Features Strengthen Graph Neural Networks", *Proceedings of the 2021 SIAM International Conference on Data Mining*, pp. 1–9, Aukland, New Zealand, 2021.

110. Maron, H., H. Ben-Hamu, N. Shamir and Y. Lipman, "Invariant and Equivariant Graph Networks", *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

111. Morris, C., M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan and M. Grohe, "Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks", *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, pp. 4602 – 4609, Hawaii, USA, 2019.

112. Maron, H., H. Ben-Hamu, H. Serviansky and Y. Lipman, "Provably Powerful Graph Networks", *Advances in Neural Information Processing Systems*, pp. 2153–2164, Vancouver, Canada, 2019.

113. Bevilacqua, B., F. Frasca, D. Lim, B. Srinivasan, C. Cai, G. Balamurugan, M. M. Bronstein and H. Maron, "Equivariant Subgraph Aggregation Networks", *Proceedings of the 10th International Conference on Learning Representations*, 2022.

114. Bahdanau, D., K. Cho and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate", *Proceedings of the 3rd International Conference*

on Learning Representations, California, USA, 2015.

115. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, "Attention Is All You Need", *31st Conference on Neural Information Processing Systems*, pp. 5998–6008, California, USA, 2017.

116. Cheng, J., L. Dong and M. Lapata, "Long Short-Term Memory-Networks for Machine Reading", J. Su, K. Duh and X. Carreras (Editors), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 551–561, Austin, Texas, 2016.

117. Lin, Z., M. Feng, C. N. dos Santos, M. Jeong, B. Xiang and B. Zhou, "A Structured Self-Attentive Sentence Embedding", *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

118. Dosovitskiy, A., L. Neumann, T. K. Mueller, D. J. Rezende, A. Athalye, D. V. Foster, J. Carreira, A. Dosovitskiy, L. Neumann, T. K. Mueller and D. J. Rezende, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", *Proceedings of the 9th International Conference on Learning Representations*, 2021.

119. Veličković, P., G. Cucurull, A. Casanova, A. Romero, P. Liò and Y. Bengio, "Graph Attention Networks", *International Conference on Learning Representations*, Vancouver, Canada, 2018.

120. Shi, Y., Z. Huang, S. Feng, H. Zhong, W. Wang and Y. Sun, "Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification", *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp. 1548–1554, 2021.

121. Dwivedi, V. P. and X. Bresson, "A Generalization of Transformer Networks to Graphs", *AAAI Workshop on Deep Learning on Graphs: Methods and Applica-*

tions, Washington, USA, 2021.

122. Brody, S., U. Alon and E. Yahav, "How Attentive are Graph Attention Networks?", *International Conference on Learning Representations*, 2022.

123. Rampášek, L., M. Galkin, V. P. Dwivedi, A. T. Luu, G. Wolf and D. Beaini, "Recipe for a General, Powerful, Scalable Graph Transformer", *Advances in Neural Information Processing Systems*, Vol. 35, pp. 14501 – 14515, 2022.

124. Kreuzer, D., D. Beaini, W. L. Hamilton, V. Létourneau and P. Tossou, "Rethinking Graph Transformers with Spectral Attention", *35th International Conference on Neural Information Processing Systems*, pp. 21618 – 21629, 2021.

125. Gamgam, G., Z. Yıldırım, A. Kabakçıoğlu, H. Gurvit, T. Demiralp and B. Acar, "Siamese Graph Convolutional Network Quantifies Increasing Structure-function Discrepancy Over the Cognitive Decline Continuum", *Computer Methods and Programs in Biomedicine*, Vol. 254, p. 108290, 2024.

126. Gamgam, G., A. Kabakcioglu, D. Yüksel Dal and B. Acar, " Disentangled Attention Graph Neural Network for Alzheimer's Disease Diagnosis ", *Medical Image Computing and Computer Assisted Intervention*, pp. 219 – 228, Marakkesh, Morocco, 2024.

127. Tench, C., P. Morgan, M. Wilson and L. Blumhardt, "White Matter Mapping Using Diffusion Tensor MRI", *Magnetic Resonance Imaging*, Vol. 18, No. 9, pp. 1261–1267, 2000.

128. Dong, X., D. Thanou, P. Frossard and P. Vandergheynst, "Learning Laplacian Matrix in Smooth Graph Signal Representations", *IEEE Transactions on Signal Processing*, Vol. 64, No. 23, pp. 6160–6173, 2016.

129. Kalofolias, V., "How to Learn a Graph from Smooth Signals", *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pp.

920–929, Cadiz,Spain, 2016.

130. Albert, M. S., S. T. DeKosky, D. Dickson, B. Dubois, H. H. Feldman, N. C. Fox, A. Gamst, D. M. Holtzman, W. J. Jagust, R. C. Petersen, P. J. Snyder, M. C. Carrillo, B. Thies and C. H. Phelps, "The Diagnosis of Mild Cognitive Impairment Due to Alzheimer's Disease: Recommendations from the National Institute on Aging-Alzheimer's Association Workgroups on Diagnostic Guidelines for Alzheimer's Disease", *Alzheimer's & Dementia*, Vol. 7, No. 3, pp. 270–279, 2011.

131. McKhann, G. M., D. S. Knopman, H. Chertkow, B. T. Hyman, C. R. Jack Jr., C. H. Kawas, W. E. Klunk, W. J. Koroshetz, J. J. Manly, R. Mayeux, R. C. Mohs, J. C. Morris, M. N. Rossor, P. Scheltens, M. C. Carrillo, B. Thies, S. Weintraub and C. H. Phelps, "The Diagnosis of Dementia due to Alzheimer's Disease: Recommendations from the National Institute on Aging-Alzheimer's Association Workgroups on Diagnostic Guidelines for Alzheimer's Disease", *Alzheimer's & Dementia*, Vol. 7, No. 3, pp. 263–269, 2011.

132. Sperling, R. A., P. S. Aisen, L. A. Beckett, D. A. Bennett, S. Craft, A. M. Fagan, T. Iwatsubo, C. R. Jack Jr., J. Kaye, T. J. Montine, D. C. Park, E. M. Reiman, C. C. Rowe, E. Siemers, Y. Stern, K. Yaffe, M. C. Carrillo, B. Thies, M. Morrison-Bogorad, M. V. Wagster and C. H. Phelps, "Toward Defining the Preclinical Stages of Alzheimer's Disease: Recommendations from the National Institute on Aging-Alzheimer's Association Workgroups on Diagnostic Guidelines for Alzheimer's Disease", *Alzheimer's & Dementia*, Vol. 7, No. 3, pp. 280–292, 2011.

133. Zaheer, M., S. K. Kamath, A. S. L., S. R. Kumar and X. Bresson, "Deep Sets", *Neural Information Processing Systems*, pp. 3394 – 3404, California, USA, 2017.

134. Alon, U. and E. Yahav, "On the Bottleneck of Graph Neural Networks and its Practical Implications", *International Conference on Learning Representations*,

2021.

135. Ying, Z., D. Bourgeois, J. You, M. Zitnik and J. Leskovec, "GNNExplainer: Generating Explanations for Graph Neural Networks", *Advances in Neural Information Processing Systems*, pp. 9244 – 9255, Vancouver, Canada, 2019.

136. Xu, K., M. Zhang, J. Li, S. S. Du, K.-I. Kawarabayashi and S. Jegelka, "How Neural Networks Extrapolate: From Feedforward to Graph Neural Networks", *International Conference on Learning Representations*, 2021.

137. Yang, Y., Z. Feng, M. Song and X. Wang, "Factorizable Graph Convolutional Networks", *Advances in Neural Information Processing Systems*, pp. 20286 – 20296, 2020.

138. Ma, J., P. Cui, K. Kuang, X. Wang and W. Zhu, "Disentangled Graph Convolutional Networks", *Proceedings of the 36th International Conference on Machine Learning*, pp. 4212–4221, California, USA, 2019.

139. Thomas Yeo, B. T., F. M. Krienen, J. Sepulcre, M. R. Sabuncu, D. Lashkari, M. Hollinshead, J. L. Roffman, J. W. Smoller, L. Zöllei, J. R. Polimeni, B. Fischl, H. Liu and R. L. Buckner, "The Organization of the Human Cerebral Cortex Estimated by Intrinsic Functional Connectivity", *Journal of Neurophysiology*, Vol. 106, No. 3, pp. 1125–1165, 2011.

140. Mevel, K., G. Chételat, F. Eustache and B. Desgranges, "The Default Mode Network in Healthy Aging and Alzheimers Disease", *International Journal of Alzheimer's Disease*, Vol. 2011, No. 1, p. 535816, 2011.