Attending to Graph Transformers

Gökhan Özbulak

Outline

- PhD Project
- Graph Transformers (GTs)
  - Introduction
  - Background
  - Properties of GT
  - Applications of GT
  - Experimental Study
  - Conclusions
- Discussion

PhD Project

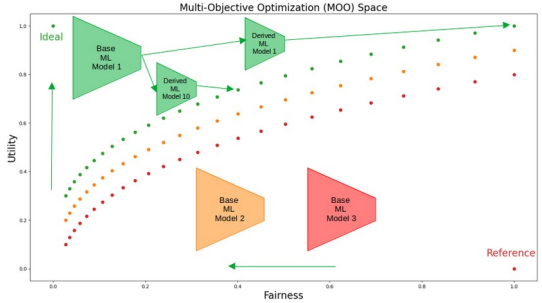
- 1st year PhD Student at EPFL, Research Assistant at Idiap Research Institute (LIDIAP).
- Supervised by Dr. André Anjos and Dr. Jean-Marc Odobez.
- FairMI – Machine Learning Fairness with Applicatin to Medical Images.
- A SNSF funded project in a collaboration with the Federal University of Sao Paulo, Brazil.
- The main purpose :
  - Introducing fairness into medical image analysis so that Machine Learning (ML) could be democratized independently from the sensitive attributes such as race, gender and ethnicity.
  - Some keywords : Machine Learning, Fairness, Medical Image Analysis

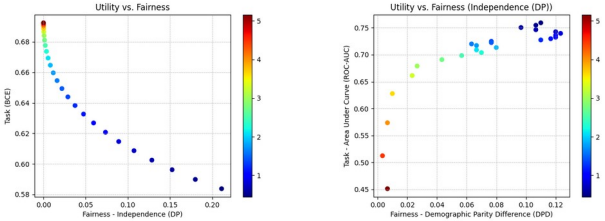
PhD Project

- Three objectives :
  - Adjustable fairness
  - Fairness boundaries
  - Fair mixed human-AI decision support
- Adjustable Fairness
  - ML performance conflicts with fairness in general.
    - A Multi-Objective Optimization (MOO) problem.
  - We may need to select proper combination of the performance with fairness.
    - Based on the necessity of the task.
  - We consider ML tasks in healthcare domain :
    - Glaucome disease : An eye disease that is seen more in Black people.

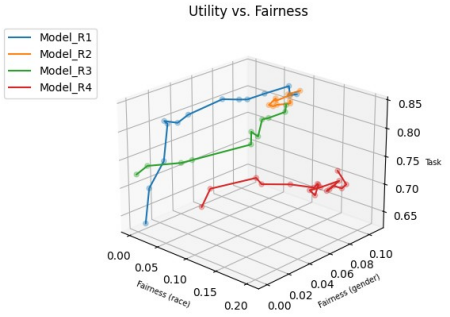
PhD Project

- Utility-Fairness Trade-off





- Hypernetwork [1] based trade-off selection mechanism.
- Base networks generate derived networks.
- Derived networks represent a choice for trade-off :
  - Better performance and worse fairness (D1).
  - Worse performance and better performance (D10).



[1] Ha, D., Dai, A., & Le, Q. V. (2016). Hypernetworks. arXiv preprint arXiv:1609.09106.

Graph Transformers (GTs)

- Introduction
  - GT is a transformer designed to capture graph structure with attention.
    - Aggregating information from all nodes, no local structure bias.
  - GTs have promising results in many problems including molecular property prediction.
  - GTs have some advantages over GNNs for some issues :
    - Over-smoothing : The representative similarities of the nodes, no discrimination.
      - GNNs with many layers
    - Over-squashing : Insufficient capacity to retain the information from other nodes.
      - GNNs capturing large graphs.
  - GTs are combined with structural/positional encodings to keep more information on graph.

Graph Transformers (GTs)

- Background
  - A graph G with a pair (V(G), E(G)) :
    - V(G) : a finite set of nodes
    - E(G) : a set of edges, $E(G) \subseteq \{\{u, v\} \subseteq V \mid u \neq v\}$
    - The neighborhood of $v \in V(G)$, $N(v)=\{u \in V(G) \mid (v, u) \in E(G)\}$.
    - The graphs G and H are isomorphic if there exists an edge-preserving bijection g: $(V(G) \rightarrow V(H))$.
      - G: A-B, A-C, B-C ; H: X-Y, X-Z, Y-Z → G and H are isomorphic.
  - Equivariance
    - f(**T**x) = **T**f(x), T : transformation (rotation, translation etc.), x : input vector
  - Invariance
    - f(**T**x) = f(x), T : transformation (rotation, translation etc.), x : input vector

Graph Transformers (GTs)

- Background
  - A single attention head :

$$\text{Attn}(\mathbf{X}^{(t)}) := \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}$$

where $Q=X^{(t)}W_Q$, $K=X^{(t)}W_K$, $V=X^{(t)}W_V$. We can rewrite $\text{Attn}(X^{(t)})$ as :

$$\text{Attn}(\mathbf{X}^{(t)})_v = \sum_{u \in V(G)} \frac{k_{\exp}(\mathbf{X}_v^{(t)}, \mathbf{X}_u^{(t)})}{\sum_{w \in V(G)} k_{\exp}(\mathbf{X}_v^{(t)}, \mathbf{X}_w^{(t)})} \mathbf{X}_u^{(t)} \mathbf{W}_V$$

for $v \in V(G)$ and :

$$k_{exp}(X^{(t)}_v, X^{(t)}_w) = \exp(X^{(t)}_v W_Q X^{(t)}_w W_K / \sqrt{d_k})$$

  - And, transformer layer updates $X^{(t)}$ as :

$$X^{(t+1)} = \text{FFN}(\text{MultiHead}(X^{(t)}) + X^{(t)})$$

Graph Transformers (GTs)

- Properties of GT
  - GTs may overcome local structural bias by structural/positional encoding.
    - This is also case for GNNs.
    - Structural encoding : Make the GT aware of graph on local, relative, global level.
      - The degree of node (local).
    - Positional encoding : Make the node aware of its relative position to other nodes.
      - The distance between node pairs (relative).
  - GTs support both of non-geometric and geometric features.
    - Geometric features require 3D information of the nodes and edges.
      - The input needs to be equivariant and invariant, a harder case.
      - 3D molecular graphs, 3D bones structure etc.
  - GTs carry information between nodes through attention for message passing.
    - This has a computational cost when the graph is large.

Graph Transformers (GTs)

- Applications of GT
  - GTs are applied on different areas including molecular property prediction.
    - Although it's a new approach and has some drawbacks.
  - [2] proposes a Brain Network Transformer to diagnose the disease from MRI.
    - The adjacency matrix as a structural encoding, better than Laplacian.
  - [3],[4] apply the Equivariant Transformer architecture to predict the quantum mechanical propoerties of molecules.
    - Molecular structure in atom level captured by multi-head attention.
  - [5] uses GTs for the web-scale heterogeneous (heterophilic) graphs.
    - Different projection matrices for attention during capture of node/edge relations.

[2] Xuan Kan, Wei Dai, Hejie Cui, Zilong Zhang, Ying Guo, and Carl Yang. Brain network transformer. ArXiv, 2022.
[3] Yi-Lun Liao and Tess Smidt. Equiformer: Equivariant graph attention transformer for 3D atomistic graphs. In ICLR, 2023.
[4] Philipp Thölke and Gianni De Fabritiis. Equivariant transformers for neural network based molecular potentials. In ICLR, 2022.
[5] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In WWW, pp. 2704–2710, 2020.

Graph Transformers (GTs)

- Experimental Study
  - Three research questions are asked to focus on different aspects of GTs :
    - Q1 : Structural awareness ?
    - Q2 : Reduced over-smoothing ?
    - Q3 : Reduced over-squashing ?
  - Q1 : Structural awareness ?
    - Does introducing different structural awareness into GTs improve capturing graph properties better ?
    - Three tasks having different level of difficulty are tested on different datasets.
      - Edge detection (easy), Triangle count (medium), Link skip (hard)

Graph Transformers (GTs)

- Experimental Study
  - Q1 : Structural awareness ?
    - Does introducing different structural awareness into GTs improve capturing graph properties better ?
    - Three tasks having different level of difficulty are tested on different datasets.
      - Edge detection (easy), Triangle count (medium), Link skip (hard)

Model	Easy	Medium		Hard
	EDGES	TRIANGLES-SMALL	TRIANGLES-LARGE	CSL
	2-way Accuracy ↑	10-way Accuracy ↑	10-way Accuracy ↑	10-way Accuracy ↑
GIN	**98.11** ±1.78	71.53 ±0.94	33.54 ±0.30	10.00 ±0.00
Transformer	55.84 ±0.32	12.08 ±0.31	10.01 ±0.04	10.00 ±0.00
Transformer (LapPE)	**98.00** ±1.03	78.29 ±0.25	10.64 ±2.94	**100.00** ±0.00
Transformer (RWSE)	**97.11** ±1.73	**99.40** ±0.10	**54.76** ±7.24	**100.00** ±0.00
Graphormer	**97.67** ±0.97	**99.09** ±0.31	42.34 ±6.48	90.00 ±0.00

Graph Transformers (GTs)

- Experimental Study
  - Q2 : Reduced over-smoothing ?
    - Do GTs solve the over-smoothing problem better than GANs ?
    - Two different heterophilic dataset compilation (smaller (6) and larger (5))

Model (PE/SE type)	Actor	Cornell	Texas	Wisconsin	Chameleon	Squirrel
Geom-GCN (Pei et al., 2020)	31.59 ±1.15	60.54 ±3.67	64.51 ±3.66	66.76 ±2.72	**60.00 ±2.81**	38.15 ±0.92
GCN (no PE/SE)	33.92 ±0.63	53.78 ±3.07	65.95 ±3.67	66.67 ±2.63	43.14 ±1.33	30.70 ±1.17
GCN (LapPE)	34.30 ±1.12	56.22 ±2.65	65.95 ±3.67	66.47 ±1.37	43.53 ±1.45	30.80 ±1.38
GCN (RWSE)	33.69 ±1.07	53.78 ±4.09	62.97 ±3.21	69.41 ±2.66	43.84 ±1.68	31.77 ±0.65
GCN (DEG)	33.99 ±0.91	53.51 ±2.65	66.76 ±2.72	67.26 ±1.53	46.36 ±2.07	34.50 ±0.87
$\text{GPS}^{\text{GCN+Transformer}}$ (LapPE)	37.68 ±0.52	66.22 ±3.87	75.41 ±1.46	74.71 ±2.97	48.57 ±1.02	35.58 ±0.58
$\text{GPS}^{\text{GCN+Transformer}}$ (RWSE)	36.95 ±0.65	65.14 ±5.73	73.51 ±2.65	**78.04 ±2.88**	47.57 ±0.90	34.78 ±1.21
$\text{GPS}^{\text{GCN+Transformer}}$ (DEG)	36.91 ±0.56	64.05 ±2.43	73.51 ±3.59	75.49 ±4.23	52.59 ±1.81	42.24 ±1.09
Transformer (LapPE)	**38.43 ±0.87**	**69.46 ±1.73**	**77.84 ±1.08**	76.08 ±1.92	49.69 ±1.11	35.77 ±0.50
Transformer (RWSE)	**38.13 ±0.63**	**70.81 ±2.02**	**77.57 ±1.24**	**80.20 ±2.23**	49.45 ±1.34	35.35 ±0.75
Transformer (DEG)	37.39 ±0.50	**71.89 ±2.48**	**77.30 ±1.32**	**79.80 ±0.90**	56.18 ±0.83	**43.64 ±0.65**
Graphormer (DEG only)	36.91 ±0.85	68.38 ±1.73	76.76 ±1.79	77.06 ±1.97	54.08 ±2.35	**43.20 ±0.82**
Graphormer (DEG, attn. bias)	36.69 ±0.70	68.38 ±1.73	76.22 ±2.36	77.65 ±2.00	53.84 ±2.32	**43.75 ±0.59**

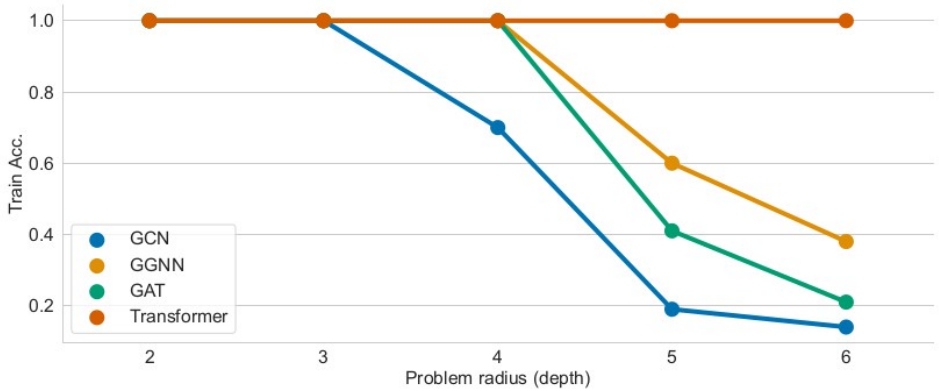
Graph Transformers (GTs)

- Experimental Study
  - Q2 : Reduced over-smoothing ?
    - Do GTs solve the over-smoothing problem better than GANs ?
    - Two different heterophilic dataset compilation (smaller (6) and larger (5))

Model (PE/SE type)	Roman-Empire	Amazon-Ratings	Minesweeper	Tolokers	Questions
GCN Platonov et al. (2023)	73.69 ±0.74	48.70 ±0.63	89.75 ±0.52	83.64 ±0.67	76.09 ±1.27
GAT Platonov et al. (2023)	80.87 ±0.30	49.09 ±0.63	92.01 ±0.68	83.70 ±0.47	77.43 ±1.20
GCN (LapPE)	83.37 ±0.55	44.35 ±0.36	**94.26** ±0.49	84.95 ±0.78	**77.79** ±1.34
GCN (RWSE)	84.84 ±0.55	46.40 ±0.55	**93.84** ±0.48	**85.11** ±0.77	**77.81** ±1.40
GCN (DEG)	84.21 ±0.47	**50.01** ±0.69	94.14 ±0.50	82.51 ±0.83	76.96 ±1.21
GAT (LapPE)	84.80 ±0.46	44.90 ±0.73	93.50 ±0.54	**84.99** ±0.54	76.55 ±0.84
GAT (RWSE)	**86.62** ±0.53	48.58 ±0.41	92.53 ±0.65	**85.02** ±0.67	77.83 ±1.22
GAT (DEG)	85.51 ±0.56	**51.65** ±0.60	93.04 ±0.62	84.22 ±0.81	77.10 ±1.23
$GPS^{GCN+Performer}$ (LapPE)	83.96 ±0.53	48.20 ±0.67	**93.85** ±0.41	84.72 ±0.77	**77.85** ±1.25
$GPS^{GCN+Performer}$ (RWSE)	84.72 ±0.65	48.08 ±0.85	92.88 ±0.50	84.81 ±0.86	76.45 ±1.51
$GPS^{GCN+Performer}$ (DEG)	83.38 ±0.68	48.93 ±0.47	93.60 ±0.47	80.49 ±0.97	74.24 ±1.18
$GPS^{GAT+Performer}$ (LapPE)	**85.93** ±0.52	48.86 ±0.38	92.62 ±0.79	84.62 ±0.54	76.71 ±0.98
$GPS^{GAT+Performer}$ (RWSE)	**87.04** ±0.58	49.92 ±0.68	91.08 ±0.58	84.38 ±0.91	77.14 ±1.49
$GPS^{GAT+Performer}$ (DEG)	85.54 ±0.58	**51.03** ±0.60	91.52 ±0.46	82.45 ±0.89	76.51 ±1.19
$GPS^{GCN+Transformer}$ (LapPE)	OOM	OOM	91.82 ±0.41	83.51 ±0.93	OOM
$GPS^{GCN+Transformer}$ (RWSE)	OOM	OOM	91.17 ±0.51	83.53 ±1.06	OOM
$GPS^{GCN+Transformer}$ (DEG)	OOM	OOM	91.76 ±0.61	80.82 ±0.95	OOM
$GPS^{GAT+Transformer}$ (LapPE)	OOM	OOM	92.29 ±0.61	84.70 ±0.56	OOM
$GPS^{GAT+Transformer}$ (RWSE)	OOM	OOM	90.82 ±0.56	84.01 ±0.96	OOM
$GPS^{GAT+Transformer}$ (DEG)	OOM	OOM	91.58 ±0.56	81.89 ±0.85	OOM

Graph Transformers (GTs)

- Experimental Study
  - Q3 : Reduced over-squashing ?
    - Do GTs solve the over-squashing problem better than GANs ?
    - One synthetic dataset for the NeighborsMatch problem (d=[2,6])



Graph Transformers (GTs)

- Conclusions
  - GTs have some advantages over GNNs for over-smoothing and over-squashing problems.
    - Integrating structural bias mechanisms (structural/positional encodings) into GNNs also makes them robust against these problems.
  - GTs have some computational complexity issues especially on larger graphs with deeper layers/depths (3 datasets in Q2 related experiments).
  - Heterophilic graphs are still hard to be captured by GT/GNN.
  - Generalizability may be achieved sometimes, not always.
    - They generalized poorly on large datasets such as Triangles-Large in Q1 related experiments.

Graph Transformers (GTs)

- Thanks for listening, Q/A ?

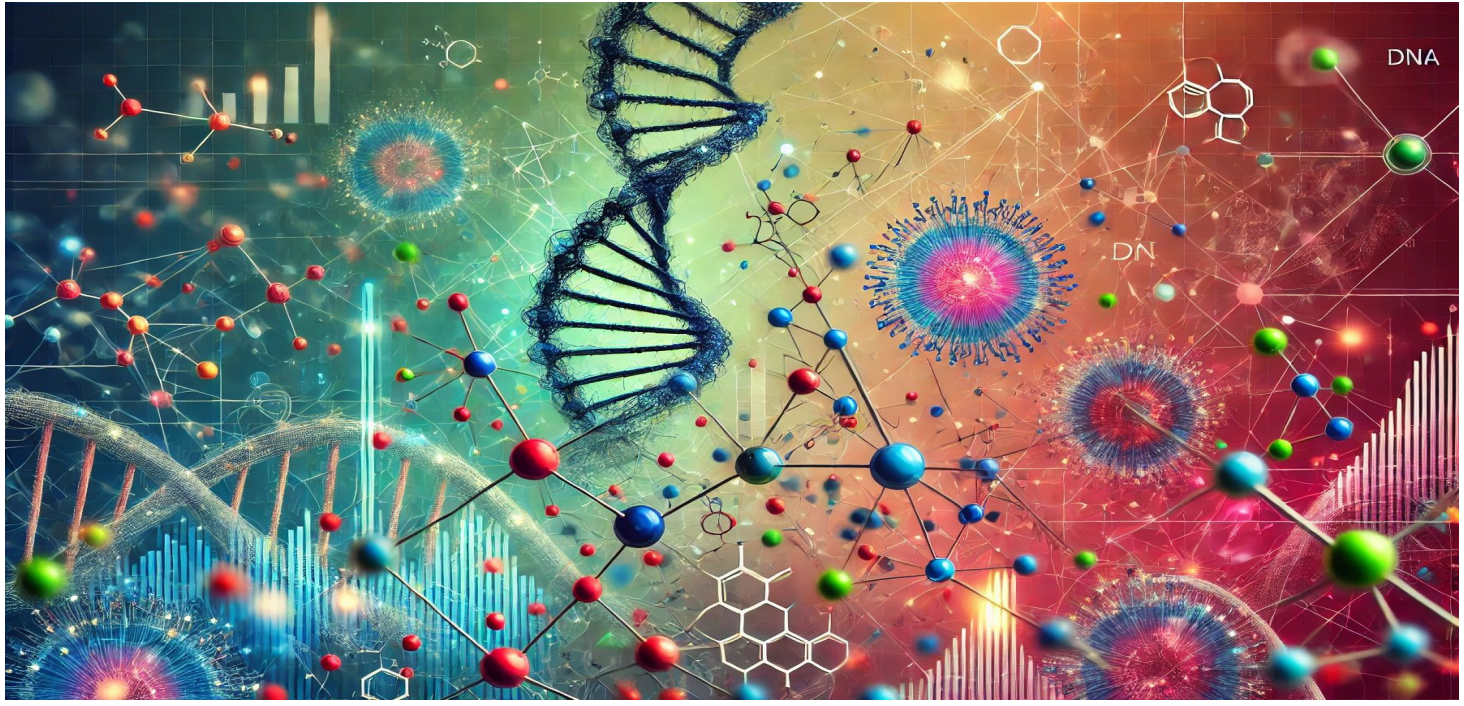
Discussions

- Do we need to switch to GTs now ?
  - There are some drawbacks such as computational complexity and additional mechanism need (structural bias etc.).
  - A comprehensive study by Google [6] showed that ViTs (Vision Transformers) are as good as traditional CNNs such as ResNet for image classification tasks.
    - Not better than CNNs. This may also be case for GT vs. GNN.

[6] Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, Andreas Veit, Proceedings of the IEEE/CVF International Conference on Computer Vision (2021)

Week 5: Graph Transformers (Application Aspects)

Presented by Hossein Mirzaei



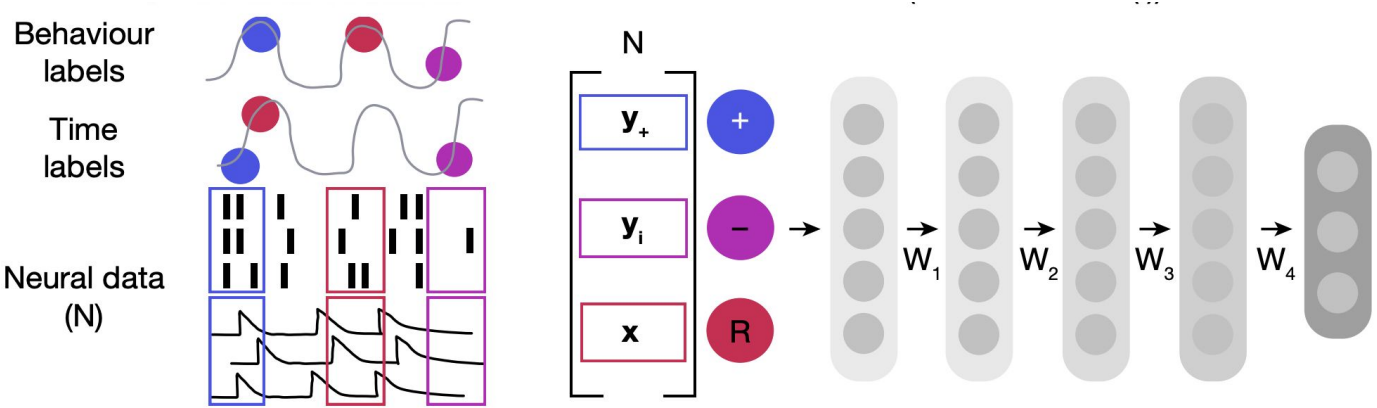
Previous Research as a Master's Student

Trustworthy machine learning, with a focus on the vision domain:

- Adversarially robust deep neural networks.
- Out-of-distribution detection.
- Backdoor attacks and defenses.

Current Research as a PhD Student in NeuroAI

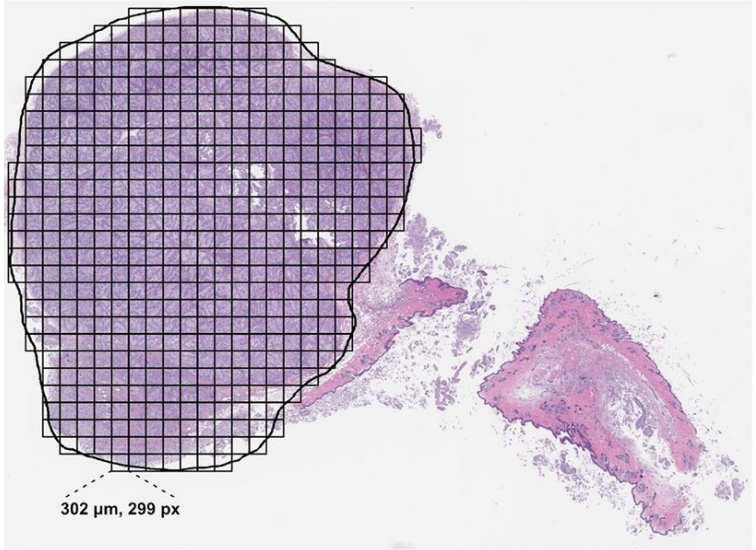
- Focusing on time series and vision data.
- Learning multi-dimensional representations within the neuroscience domain.
- Extracting robust features from brain neural activity.
- Modeling and predicting brain-vision encoding.



A Graph-Transformer for Whole Slide Image Classification

Yi Zheng, Rushin H. Gindra , Emily J. Green, Eric J. Burks , Margrit Betke, Jennifer E. Beane , Vijaya B. Kolachalama

Background



What Are Whole Slide Images (WSIs)?

- Digital high-resolution scans of histopathological slides.
- Used for computational pathology and disease diagnosis.

Challenges of WSIs

- Gigabyte-sized images: Complex to analyze.
- Requires techniques that capture both local (patch) and global (WSI) information.

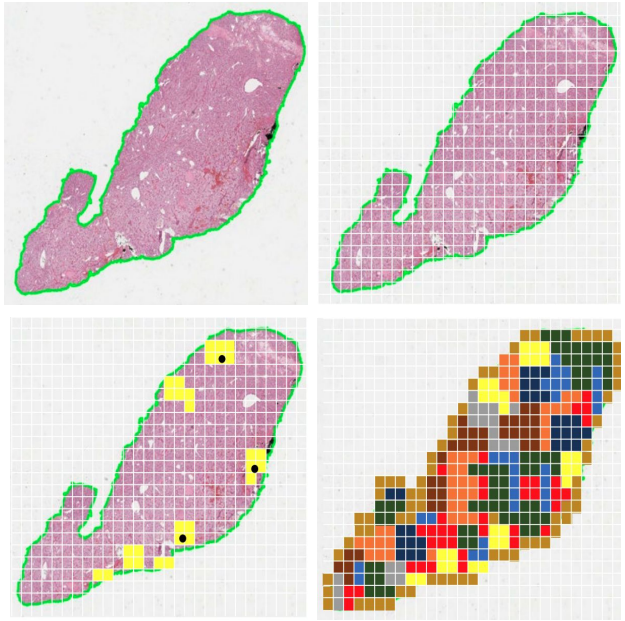
Background

Traditional Analysis Methods

- Patch-based methods: Divide WSI into smaller patches.
- Limitations: Ignores WSI-level context, assumes all patches are independent.

WSI Applications

- Disease classification (e.g., cancer grading).
- Identifying the presence or absence of a tumor on an WSI.
- Tissue segmentation, mutation prediction, etc.



Problem Setup

Task

- Classify whole slide images into categories like Normal vs. Cancerous.
- Critical for accurate diagnosis and treatment planning.

Focus on Lung Cancer

- **Normal Tissue** vs. **Lung Adenocarcinoma (LUAD)** vs. **Lung Squamous Cell Carcinoma (LSCC)**.
- Different cancers require distinct treatments, making accurate classification essential.

Key Challenges in Classification

- Large image sizes: Millions of pixels to process.
- Label noise: Assuming every patch has the same label as the entire WSI.
- Need for models that can handle both patch-level and WSI-level information.

Limitations of Traditional Patch-Based Methods

- Patches treated independently, losing spatial relationships between regions.
- Unable to capture global tumor architecture.

Motivation

Why Use Graphs?

- Represent WSI as a graph where:
  - Nodes = Patches of the image.
  - Edges = Spatial relationships between patches.
- Preserves the **spatial structure** of WSIs.

Why Use Transformers?

- Vision Transformers excel in modeling **long-range dependencies**.
- Capture global WSI context efficiently, improving classification accuracy.

GTP Workflow

- **Feature Extraction**: Use contrastive learning to get patch features.
- **Graph Construction**: Connect patches based on adjacency.
- **Transformer Processing**: Predict WSI-level labels (Normal, LUAD, LSCC).

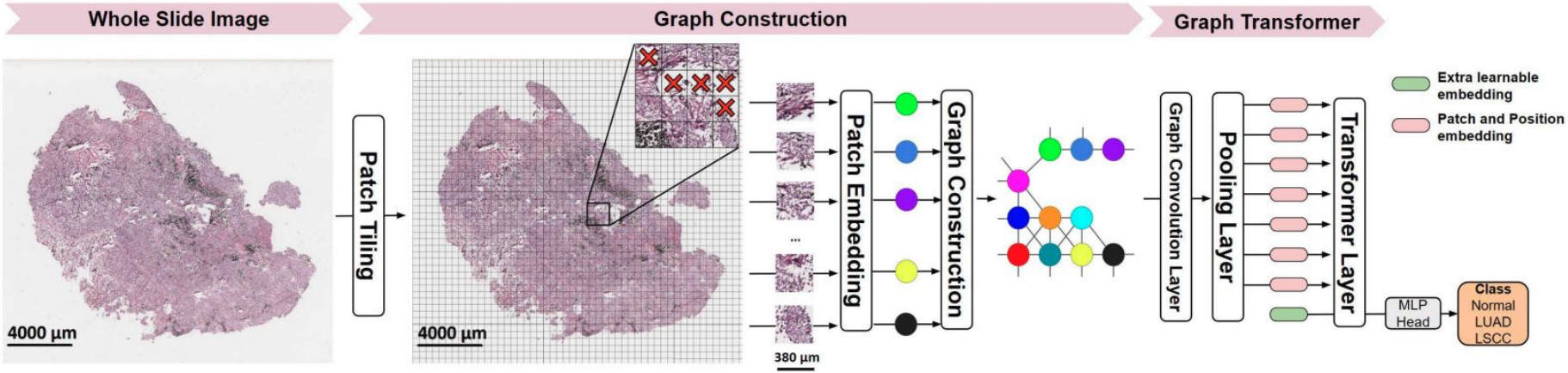
Proposed Method

Key Contributions

1. **Novel Graph-Transformer Framework (GTP)**
   - Integrates **Graph Neural Networks (GNN)** and **Vision Transformers (ViT)** for whole slide image (WSI) classification.
   - Effectively captures both **local patch-level** and **global WSI-level** information.

2. **GraphCAM: Class Activation Mapping**
   - Introduces a new **GraphCAM** technique for generating interpretable **saliency maps** on WSIs.
   - Enables visualization of regions highly associated with predicted class labels (Normal, LUAD, LSCC).

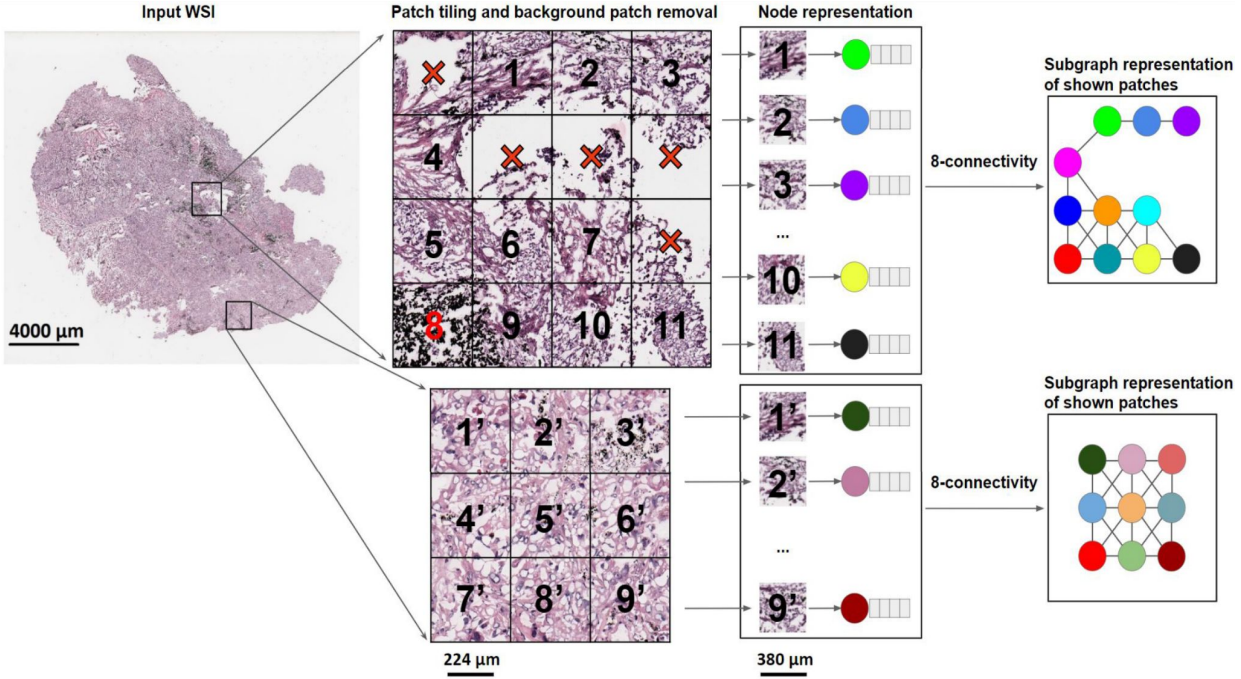
Proposed Method



Key Components of the Method

- **Contrastive Learning-Based Feature Extractor**: Generates robust patch features without manual labels.
- **Graph Neural Network (GNN)**: Captures spatial relationships between patches.
- **Vision Transformer (ViT)**: Models global context and long-range dependencies.

Proposed Method

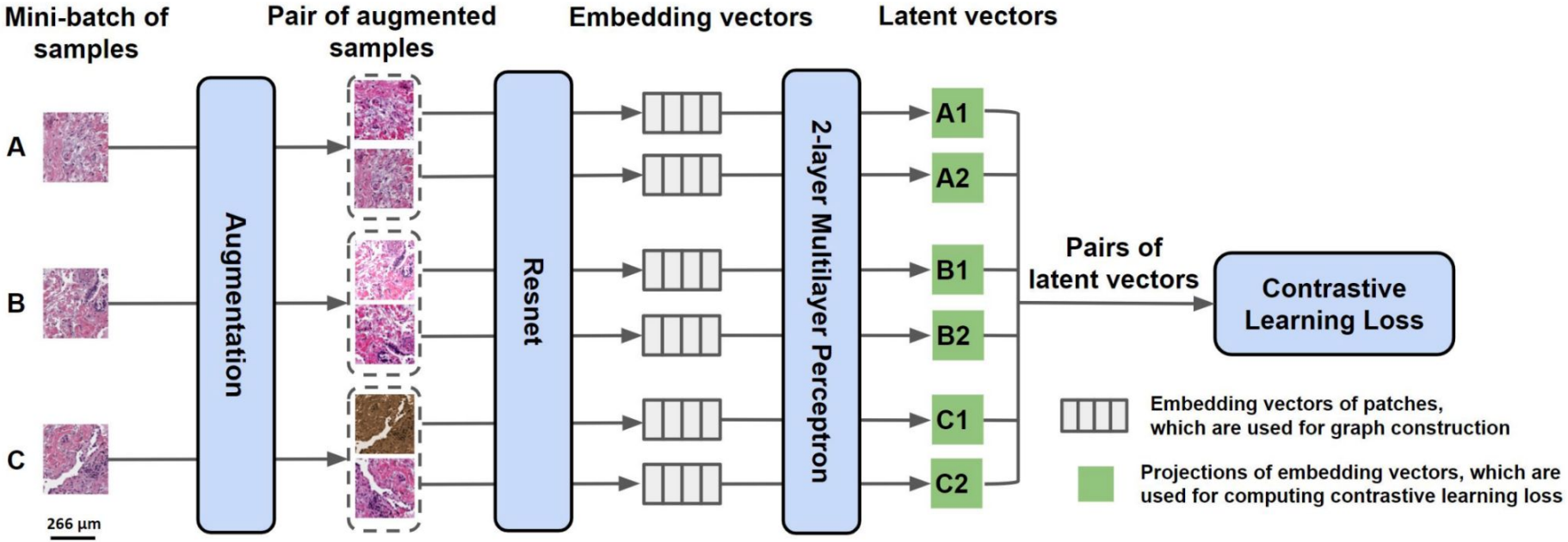


(b) Node representation and connectivity information.

Step-by-Step Workflow

- **1. Patch Extraction**: WSI divided into smaller image patches.
- **2. Contrastive Feature Extraction**: Feature vectors for each patch are extracted via contrastive learning.
- **3. Graph Construction**: Patches are nodes, edges represent spatial adjacency.
- **4. Graph Convolutional Layer**: Aggregates information from neighboring patches.
- **5. Transformer Layer**: Captures global WSI context for accurate classification.

Proposed Method



(c) Feature generation and contrastive learning.

Training a Feature Extractor on WSIs Using a Common Contrastive Learning Objective Function

Results

PERFORMANCE METRICS FOR THE 3-LABEL (NORMAL VS. LUAD VS. LSCC) CLASSIFICATION TASK. MEAN PERFORMANCE METRICS ARE REPORTED ALONG WITH THE CORRESPONDING VALUES OF STANDARD DEVIATION IN PARENTHESES

(a) Precision, Recall/Sensitivity, and Specificity (Percentage (%) values are reported).

Method	Data	Precision			Recall/Sensitivity			Specificity		
		Normal	LUAD	LSCC	Normal	LUAD	LSCC	Normal	LUAD	LSCC
TransMIL [27]	CPTAC	89.7(1.8)	81.0(3.1)	87.1(1.7)	90.4(1.9)	81.2(2.0)	85.9(3.7)	94.4(1.1)	90.8(1.9)	93.7(1.2)
	TCGA	76.6(5.1)	64.3(4.1)	80.4(1.7)	87.3(3.6)	75.8(5.3)	55.6(7.7)	90.6(2.9)	73.0(5.6)	92.4(1.9)
AttPool [15]	CPTAC	88.4(2.7)	77.1(2.7)	80.6(3.3)	85.9(3.3)	78.0(4.7)	81.6(1.8)	94.0(1.5)	88.9(2.1)	90.1(2.2)
	TCGA	89.1(4.1)	69.9(3.9)	81.4(3.0)	87.8(2.7)	79.4(2.4)	71.3(4.0)	**94.9(2.0)**	82.5(2.9)	91.4(1.7)
GTP* (only GCN)	CPTAC	82.9(6.5)	81.6(6.5)	86.5(4.7)	93.6(5.0)	74.4(6.7)	80.3(4.8)	89.3(5.3)	91.6(3.9)	93.6(2.7)
	TCGA	72.5(9.0)	69.1(3.2)	82.0(9.8)	90.7(8.3)	57.7(9.7)	69.9(9.5)	82.4(9.9)	85.7(8.4)	90.7(6.1)
GTP	CPTAC	**93.2(3.0)**	**88.4(3.9)**	**87.8(3.0)**	**95.9(2.2)**	**83.9(4.5)**	**89.2(4.0)**	**96.2(1.7)**	**94.7(1.9)**	**93.8(1.7)**
	TCGA	**89.2(2.8)**	**74.4(2.7)**	**84.4(0.7)**	**92.6(2.7)**	**79.8(1.9)**	**75.2(1.6)**	94.7(1.6)	**86.0(2.3)**	**92.7(0.4)**

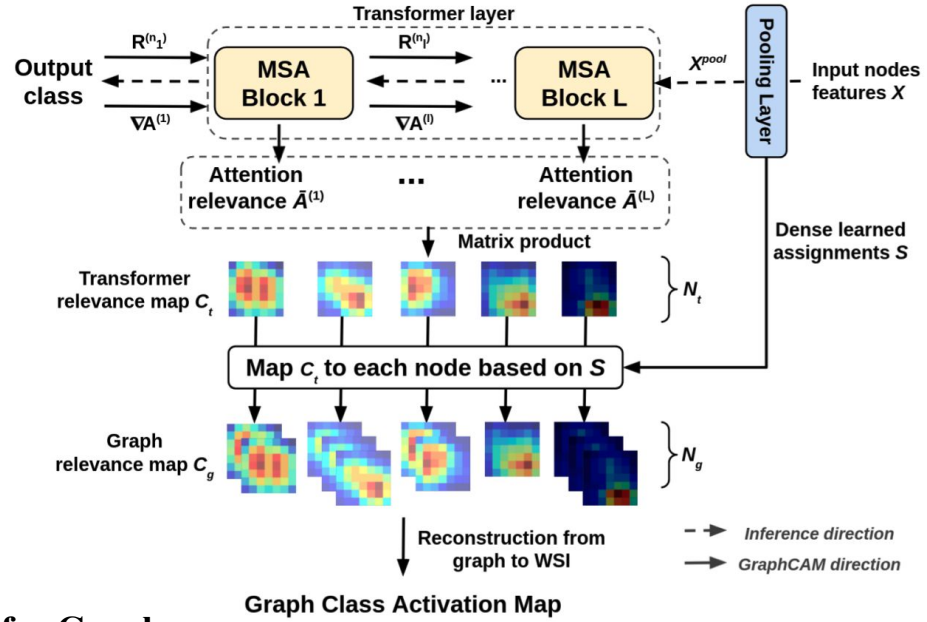
(b) Accuracy and AUC (Percentage (%) values are reported).

Method	Data	Accuracy	AUC
TransMIL [27]	CPTAC	85.9(0.7)	96.1(0.3)
	TCGA	71.6(2.3)	88.0(0.7)
AttPool [15]	CPTAC	81.9(2.1)	92.5(1.6)
	TCGA	79.3(2.3)	91.3(1.1)
GTP* (only GCN)	CPTAC	83.0(2.7)	95.2(1.2)
	TCGA	72.4(4.9)	86.6(3.8)
GTP	CPTAC	**91.2(2.5)**	**97.7(0.9)**
	TCGA	**82.3(1.0)**	**92.8(0.3)**

(c) DeLong's algorithm for comparing the AUC values between GTP and other methods. $\log_{10}(0.05) = -1.301$.

Method	Data	$\log_{10}$(p-value)
TransMIL [27]	CPTAC	−1.578(0.853)
	TCGA	−5.627(2.263)
AttPool [15]	CPTAC	−2.305(1.250)
	TCGA	−2.068(1.339)
GTP* (only GCN)	CPTAC	−1.759(1.129)
	TCGA	−5.373(3.146)

GraphCAM



Class Activation Mapping for Graphs

- GraphCAM is designed to generate **saliency maps** on **graph-structured data**, highlighting regions of the WSI most relevant to the classification decision.

Propagation of Relevance

- GraphCAM propagates **relevance scores** from the output class prediction back through the transformer and graph layers.
- Uses the attention maps from the transformer layer to understand which nodes (patches) are most important for the classification.

Discussion

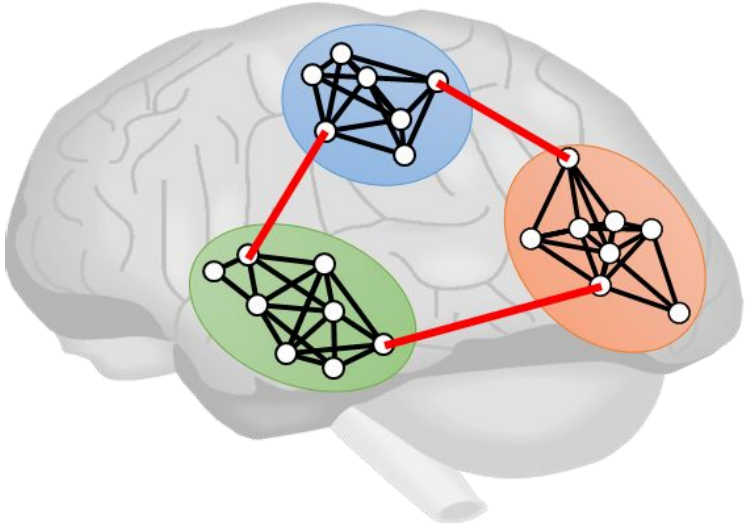
Why Graph-Transformer is a Great Fit for This Work

- Captures both local patch-level and global WSI-level information
- Vision Transformer effectively models long-range dependencies
- Flexibly handles variable patch sizes and complex WSI structures
- Enhances interpretability with GraphCAM
- Demonstrates strong generalization across multiple datasets

Brain Network Transformer

Xuan Kan, Wei Dai, Hejie Cui, Zilong Zhang, Ying Guo, Carl Yang

Background

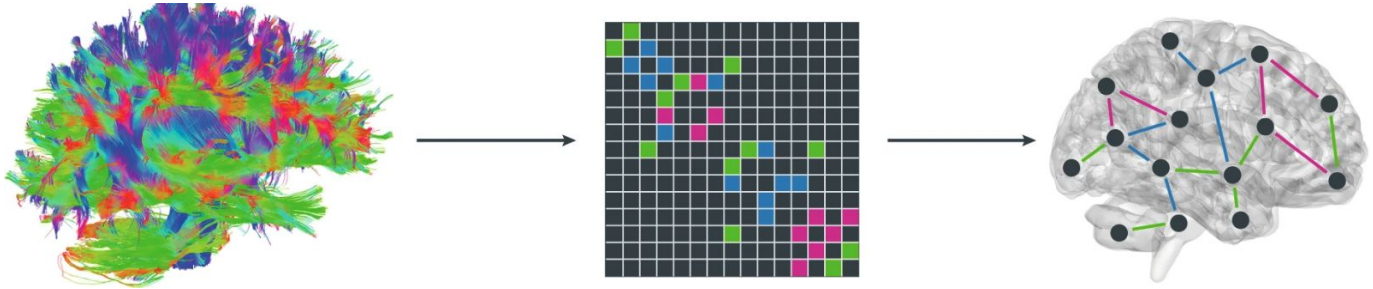


Understanding Brain Networks

1. **What Are Brain Networks?**
   - Brain networks are maps of how different parts of the brain, called **Regions of Interest (ROIs)**, communicate with each other.
   - In these networks, **nodes** represent ROIs, and **edges** show the strength of the connections between them.

2. **How Do We Build Brain Networks?**
   - Brain activity is captured using **Functional Magnetic Resonance Imaging (fMRI)**.
   - fMRI measures changes in blood oxygen levels (**BOLD signals**), which indicate the connections between different brain regions.

Background



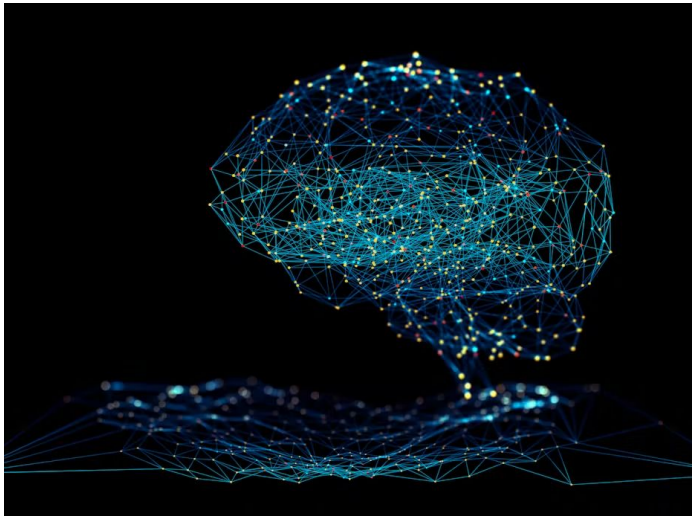
Why Study Brain Networks?

1. **Understanding Brain Function**
   - Brain networks reveal how different regions of the brain work together.
   - Helps us explore key processes like memory, learning, and decision-making.

2. **Diagnosing Mental and Neurological Disorders**
   - Brain networks provide insights into conditions like **autism**, **Alzheimer's**, and **schizophrenia**.
   - They help researchers detect abnormalities and predict disease progression.

3. **Improving Treatment and Interventions**
   - By understanding brain networks, we can develop better treatments for mental health conditions.
   - Helps in designing personalized therapies based on how specific brain regions are affected.

Motivation



Challenges in Brain Network Analysis

1. **Complexity of Brain Networks**
   - Brain networks typically have hundreds of ROIs and up to **160,000 edges**.
   - Handling the vast number of connections is computationally challenging.

2. **Fixed Size and Dense Connectivity**
   - Every node (ROI) connects to every other node.
   - Common graph models struggle with scalability and accuracy for brain networks.

3. **Need for Specialized Models**
   - Traditional graph models fail to capture **functional modules** of brain regions effectively.
   - Transformer models show promise but need adaptation for the unique structure of brain networks.

Modeling Brain Networks as Graphs

- **Nodes** represent Regions of Interest (ROIs) in the brain, as defined by an atlas.
- **Edges** are calculated based on pairwise correlations between the BOLD (blood-oxygen-level-dependent) signals, which are measured using **fMRI**. These signals reflect brain activity and allow us to map the connections between different regions.

Motivation

Problems with Existing Graph Models

Graph Neural Networks (GNNs)

- Commonly used for graph-based data but mainly focus on **local** node connections.
- Struggle with large, fully connected brain networks where **all nodes are important**.

Common Graph Transformers

- Originally designed for general graphs, not brain networks.
- Use **eigenvalues/eigenvectors** for positional encoding, which are costly and unnecessary for brain networks.

Motivation

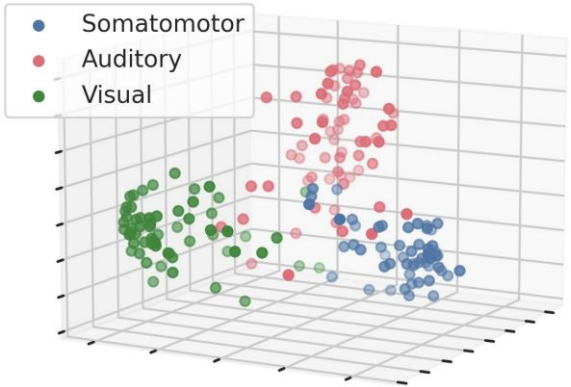
Problems with Existing Graph Models

Scalability Issues

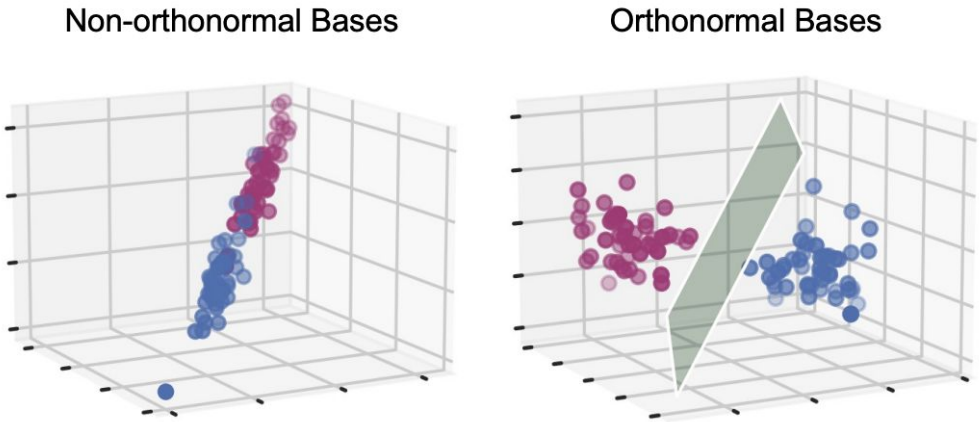
- Brain networks have **more nodes and edges** than typical graph domains, such as molecular graphs.
- Existing graph transformer models are inefficient when applied to large-scale brain networks.

on eigenvalues and eigenvectors redundant. The third challenge is scalability. Typically, the numbers of nodes and edges in molecule graphs are less than 50 and 2500, respectively. However, for brain networks, the node number is generally around 100 to 400, while the edge number can be up to 160,000. Therefore, operations like the generation of all edge features in existing graph transformer models can be time-consuming, if not infeasible.

Orthonormal Clustering Readout (OCREAD)

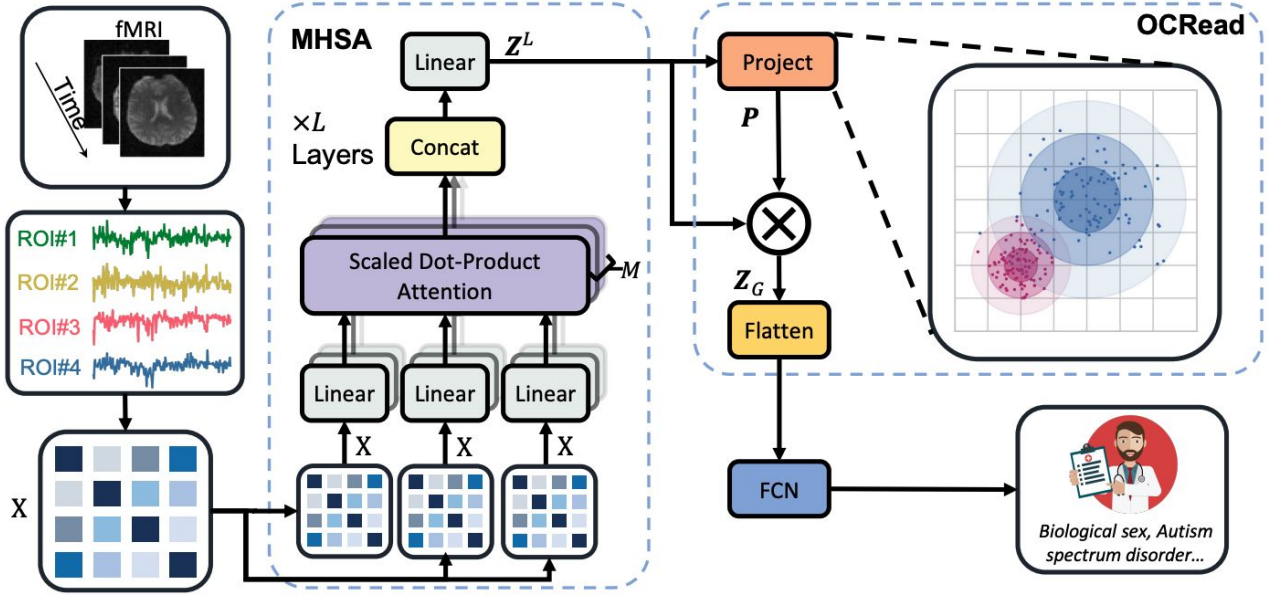


(a) Node features projected to a 3D space with PCA. Colors indicate functional modules.



(b) Orthonormal bases can make indistinguishable nodes in non-orthonormal bases easily distinguishable.

Method



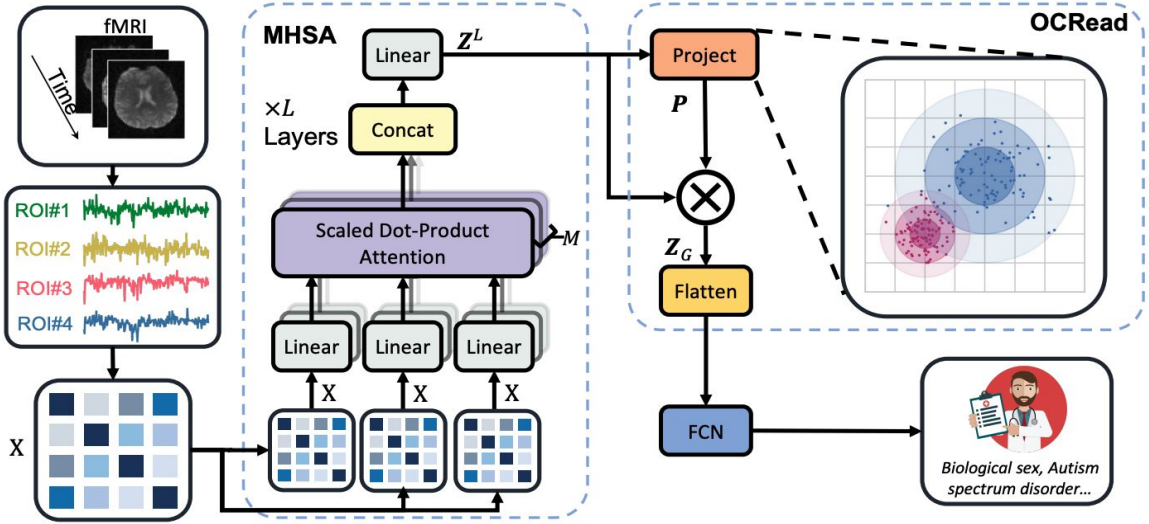
Input: Brain Networks

- Brain networks are modeled as **graphs** where **nodes** are Regions of Interest (ROIs) and **edges** represent connections between regions, derived from **fMRI** signals.

Multi-Head Self-Attention (MHSA)

- The model uses **Multi-Head Self-Attention** to focus on **all regions** in the brain at once.
- This allows the model to learn important patterns across all brain regions, handling the complexity of fully connected graphs.

Method



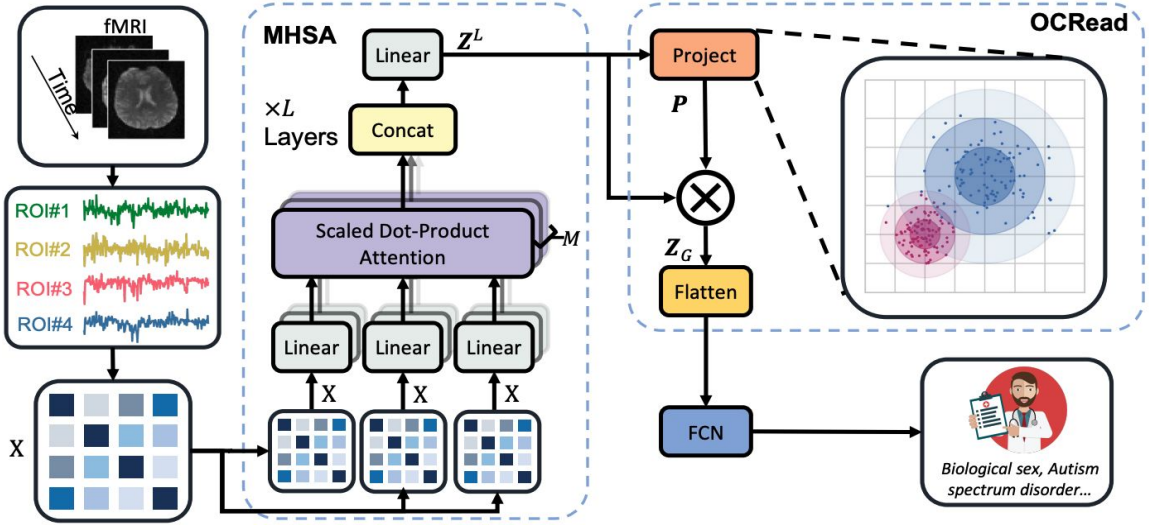
Connection Profiles as Node Features

- Instead of using complex embeddings, each node (ROI) is represented by a **connection profile**.
- The connection profile captures how each brain region is connected to others, serving as simple yet powerful node features.

Orthonormal Clustering Readout (OCREAD)

- Brain regions often act together in groups (functional modules). OCREAD assigns brain regions to these **soft clusters** based on their connections.
- The **orthonormal projection** makes sure that clusters are well-separated, improving accuracy.
- OCREAD summarizes the whole brain network into meaningful, cluster-aware representations, which are then used to make predictions.

Method



Output: Predictions

- The model outputs predictions about the brain's state, such as classifying whether someone has autism or identifying other brain-related outcomes.

Results

Table 1: Performance comparison with different baselines (%). The performance gains of BRAIN-NETTF over the baselines have passed the t-test with p-value<0.03.

Type	Method	Dataset: ABIDE				Dataset: ABCD			
		AUROC	Accuracy	Sensitivity	Specificity	AUROC	Accuracy	Sensitivity	Specificity
Graph Transformer	SAN	71.3±2.1	65.3±2.9	55.4±9.2	68.3±7.5	90.1±1.2	81.0±1.3	84.9±3.5	77.5±4.1
	Graphormer	63.5±3.7	60.8±2.7	**78.7±22.3**	36.7±23.5	89.0±1.4	80.2±1.3	81.8±11.6	82.4±7.4
	VanillaTF	76.4±1.2	65.2±1.2	66.4±11.4	71.1±12.0	94.3±0.7	85.9±1.4	87.7±2.4	82.6±3.9
Fixed Network	BrainGNN	62.4±3.5	59.4±2.3	36.7±24.0	70.7±19.3	OOM	OOM	OOM	OOM
	BrainGB	69.7±3.3	63.6±1.9	63.7±8.3	60.4±10.1	91.9±0.3	83.1±0.5	84.6±4.3	81.5±3.9
	BrainNetCNN	74.9±2.4	67.8±2.7	63.8±9.7	71.0±10.2	93.5±0.3	85.7±0.8	87.9±3.4	83.0±4.4
Learnable Network	FBNETGNN	75.6±1.2	68.0±1.4	64.7±8.7	62.4±9.2	94.5±0.7	87.2±1.2	87.0±2.5	86.7±2.8
	BrainNetGNN	55.3±1.9	51.2±5.4	67.7±37.5	33.9±34.2	75.3±5.2	67.5±4.7	67.7±5.7	68.0±6.5
	DGM	52.7±3.8	60.7±12.6	53.8±41.2	51.1±40.9	76.8±19.0	68.6±8.1	40.5±29.7	**95.6±4.2**
Ours	BRAINNETTF	**80.2±1.0**	**71.0±1.2**	72.5±5.2	**69.3±6.5**	**96.2±0.3**	**88.4±0.4**	**89.4±2.6**	88.4±1.5


Thank
You