EvoBrain: Dynamic Multi-Channel EEG Graph Modeling for Time-Evolving Brain Networks

Rikuto Kotoge[1] **Zheng Chen**[1] **Tasuku Kimura**[1] **Yasuko Matsubara**[1]
Takufumi Yanagisawa[2,3] **Haruhiko Kishima**[2] **Yasushi Sakurai**[1]

[1]SANKEN, The Univerity of Osaka, Japan
[2]Department of Neurosurgery, Graduate School of Medicine, The University of Osaka, Japan
[3]Institute for Advanced Co-Creation Studies, The University of Osaka, Japan

[1]{rikuto88, chenz, tasuku, yasuko, yasushi}@sanken.osaka-u.ac.jp
[2]{tyanagisawa, hkishima}@nsurg.med.osaka-u.ac.jp

Abstract

Dynamic GNNs, which integrate temporal and spatial features in Electroencephalography (EEG) data, have shown great potential in automating seizure detection. However, fully capturing the underlying dynamics necessary to represent brain states, such as seizure and non-seizure, remains a non-trivial task and presents two fundamental challenges. First, most existing dynamic GNN methods are built on temporally fixed static graphs, which fail to reflect the evolving nature of brain connectivity during seizure progression. Second, current efforts to jointly model temporal signals and graph structures and, more importantly, their interactions remain nascent, often resulting in inconsistent performance. To address these challenges, we present the first theoretical analysis of these two problems, demonstrating the effectiveness and necessity of explicit dynamic modeling and time-then-graph dynamic GNN method. Building on these insights, we propose `EvoBrain`, a novel seizure detection model that integrates a two-stream Mamba architecture with a GCN enhanced by Laplacian Positional Encoding, following neurological insights. Moreover, `EvoBrain` incorporates explicitly dynamic graph structures, allowing both nodes and edges to evolve over time. Our contributions include (a) a theoretical analysis proving the expressivity advantage of explicit dynamic modeling and *time-then-graph* over other approaches, (b) a novel and efficient model that significantly improves AUROC by 23% and F1 score by 30%, compared with the dynamic GNN baseline, and (c) broad evaluations of our method on the challenging early seizure prediction task.

1 Introduction

Epileptic seizures are typically considered a network disorder (Burns et al., 2014a). The abnormal connections across brain regions often serve as markers of seizure events (Li et al., 2021b). As such, recent studies have leveraged graph neural networks (GNNs) to model these networks for automating seizure detection (Cai et al., 2023; Ho and Armanfard, 2023). Considering the unfolding time-course of brain dynamics, a recent trend models EEGs as a sequence of time-varying graphs. Temporal models, such as recurrent neural networks (RNNs), integrated with GNNs, known as dynamic GNNs, have been proposed to learn spatiotemporal features in EEGs (Tang et al., 2023). By modeling graphs at fine-grained time steps and their sequential representations, these methods can capture how graphs or brain states evolve over time, further enhancing seizure detection accuracy.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

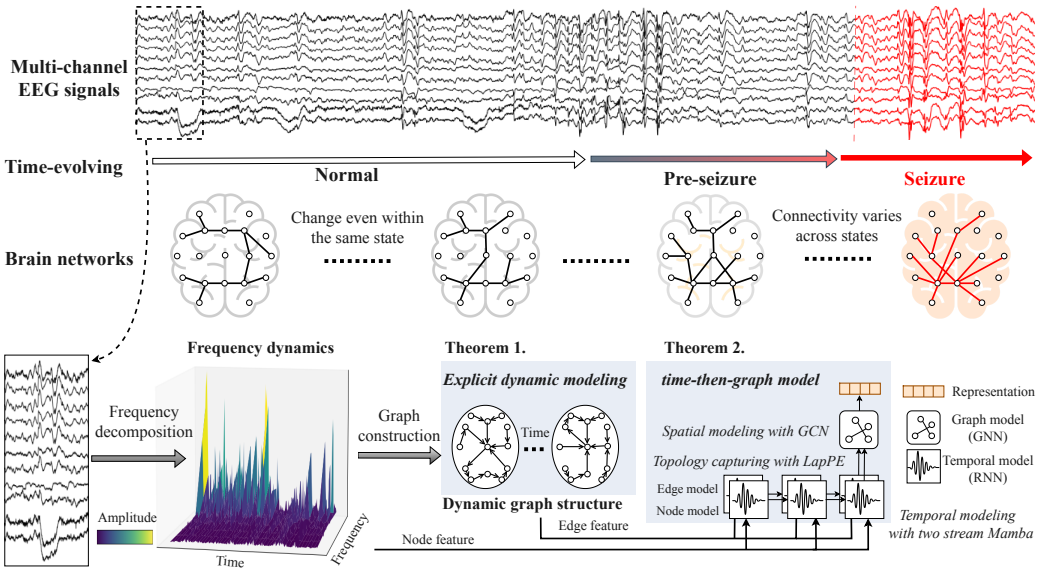


Figure 1: The brain network evolves over time, and changes occurring during seizures and immediately before them in the pre-seizure phase, especially within specific zones, are clinically important. These changes are captured using dynamic graphs derived from multi-channel EEG signals. `EvoBrain` incorporates a explicit dynamic graph modeling and time-then-graph architecture.

However, effectively representing brain dynamics by integrating graph and temporal models remains non-trivial. In this paper, we study this critical problem in accurate seizure detection and prediction by learning dynamic representations from EEGs. In essence, we mainly face two challenges:
(1) *Representing EEG dynamics* (Problem 1). We observe that although many existing methods are labeled as "dynamic," they often employ static graph structures (Ho and Armanfard, 2023; Tang et al., 2022). Typically, these methods construct a predefined graph based on EEG channel correlations from the initial snapshot, which is then fixed and shared across all subsequent EEG snapshots. This implicitly assumes that the spatial structure of brain activity remains constant over time. In contrast, seizures induce evolving patterns of neuronal synchrony and desynchrony across different brain regions (Rolls et al., 2021; Li et al., 2021b; Chen et al., 2025), as shown in Figure 1.
(2) *Effective Spatio-temporal Modeling (Problem 2).* Dynamic GNNs can generally be categorized into *time-then-graph*, *graph-then-time*, and *time-and-graph* architectures, following the taxonomy of Gao and Ribeiro (2022) (A concise illustration is provided in Appendix Figure 7). The *time-then-graph* first model the temporal dynamics and then employ GNNs to learn spatial representations. The *graph-then-time* first applies GNNs to each EEG snapshot independently, and then learns temporal dynamics from the resulting graph features. The *time-and-graph* is a recurrent GNN to capture temporal interactions between EEG snapshots before performing graph learning at each time step. However, the independent GNNs in *graph-then-time* represent information at single time steps without accounting for dynamic interactions between time steps. While recurrent GNNs in *time-and-graph* capture graph interactions, they also rely heavily on the independent initial GNNs. Overall, an effective dynamic GNNs for integrating temporal and graph-based representations to model brain dynamics remains poorly understood.

Novelty and Contributions: We first analyze a theoretical foundation for designing dynamic graph structures and models that effectively represent brain dynamics. Building upon this foundation, we propose `EvoBrain`, which effectively learns **Evo**lving, dynamic characteristics in **Brain** networks for accurate seizure detection and prediction. Overall, we summarize our contributions as:

- **Theoretical Analysis.** To tackle the first challenge, we propose and analyze dynamic graph structure that explicitly incorporate temporal EEG graph modeling (Theorem 1). To tackle the second challenge, we first theoretically analyze different dynamic GNNs approaches from EEG graph perspective (Theorem 2). We discuss the necessarily of dynamic GNNs at node-level, since the node similarity measures are key factors in determining EEG graph construction.

- **Novel Model Design.** We hence propose a novel *time-then-graph* model, `EvoBrain`, which integrates a two-stream Mamba architecture (Gu and Dao, 2024) with a GCN enhanced by Laplacian Positional Encoding (Wang et al., 2022), following neurological insights. `EvoBrain` achieves up to 23% and 30% improvements in AUROC and F1 scores, respectively, compared with the dynamic GNN baseline. Also, `EvoBrain` is 17× faster than the SOTA *time-and-graph* dynamic GNN.
- **Broad Evaluation.** Unlike most seizure detection studies (Eldele et al., 2021; Cai et al., 2023; Ho and Armanfard, 2023), we evaluate the more challenging task of seizure prediction, which aims to identify the preictal state before seizures. This is critical for early intervention in clinical settings, and `EvoBrain` maintains performance, with a 13.8% improvement in AUROC.

2 Problem Setting

2.1 Notations

We define an EEG $\boldsymbol{x}$ with N channels and T snapshots as a graph $\mathcal{G} = (\mathcal{A}, \mathcal{X})$, where $\mathcal{A} = (\mathcal{V}, \mathcal{E})$ and $\mathcal{A} \in \mathbb{R}^{N \times N \times T}$ are the adjacency matrices. $\mathcal{V}$ and $\mathcal{E}$ represent the channels (i.e., nodes) and edges. Notably, most existing work constructs a weighted adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ as fixed across T meaning that all EEG snapshots share the same graph structure and only the node features H are computed iteratively at each snapshot. In this paper, each edge $e_{i,j,t} \in \mathcal{E}$ represents pairwise connectivity between channels v_i and v_j, where $i, j \in N$. The feature vector $x_{i,t} \in \mathbb{R}^d$ captures the electrical activity of i-th channel during the EEG snapshot at time step t. If $e_{i,j,t}$ exists, $\boldsymbol{a}_{i,j,t}$ is a non-zero value. $\boldsymbol{a}_{i,j,t} \in \mathbb{R}$ quantifies the connectivity strength between two channels for each snapshot. To represent temporal EEG graphs, we define the embedding of node v_i at time step t as $h_{i,t}^{node} \in \mathbb{R}^k$, which captures both the spatial connectivity information from the edges and the temporal dynamics from previous node embeddings. The embedding of edge $e_{i,j,t}$, denoted as $h_{i,j,t}^{edge} \in \mathbb{R}^l$, captures the temporal evolution of channel connectivity, reflecting changes in brain networks.

2.2 Problem - Dynamic GNN Expressivity in EEG Modeling

Brain networks in different states can manifest as distinct graph structures, as shown in Figure 1. We treat *expressivity analysis* as a graph isomorphism problem (Xu et al., 2019), where non-isomorphic EEG graphs represent different brain states, enabling the model to effectively distinguish between seizure and non-seizure graphs. We formulate the challenge of representing EEG dynamics as the distinction between implicit and explicit dynamic graph modeling in Problem 1. We further define the challenge of effective spatio-temporal modeling as the investigation of the expressivity of *graph-then-time*, *time-and-graph*, and *time-then-graph* (Gao and Ribeiro, 2022) in dynamic EEG graph analysis in Problem 2.

Definition 1 (Implicit Dynamic Graph Modeling - Static Structure)**.** *This approach fixes a single adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ across all time steps, although each node's feature vector $\boldsymbol{x}_{:,t}$ evolves across t. Formally, $\mathbf{A}_{:,:,t} = \widehat{\mathbf{A}}$, for each $t \in \{1, \ldots, T\}$, where $\widehat{\mathbf{A}}$ is constant for all t. Hence, the spatial connectivity among EEG channels remains unchanged, and only node features capture the dynamic aspects.*

Definition 2 (Explicit Dynamic Graph Modeling - Dynamic Structure)**.** *In contrast to the implicit setting, here both node features $\boldsymbol{x}_{:,t}$ and the adjacency matrices $\{\mathbf{A}_{:,:,t}\}_{t=1}^{T}$ can vary with time. Specifically, $\mathbf{A}_{:,:,t} \in \mathbb{R}^{N \times N}$ at each snapshot t may be computed by a function f: $\mathbf{A}_{:,:,t} = f(\boldsymbol{x}_{:,t}), \quad \forall t \in \{1, \ldots, T\}$, thereby capturing the dynamic evolution of channel connectivity in addition to time-varying nodes.*

Problem 1 (Implicit vs. Explicit Dynamic Graph Modeling)**.** *We consider two approaches for capturing spatial relationships. In the implicit (static) approach, a single adjacency matrix $\mathbf{A}$ remains fixed across all time steps, so only the node features evolve. In the explicit (dynamic) approach, both node features and adjacency matrices $\{\mathbf{A}_{:,:,t}\}_{t=1}^{T}$ can change with t, allowing for time-varying connectivities derived from the EEG data. Our goal is to compare the expressiveness of these two approaches in capturing spatiotemporal dependencies for dynamic EEG graph analysis.*

Definition 3 (Graph-then-time)**.** *This architecture first apply GNNs to learn spatial, graph information at each t independently, followed by the temporal processing (e.g., by RNNs) of the resulting node embeddings. The formal definition is given as:*

$$\boldsymbol{h}_{i,t}^{node} = \mathit{Cell}\Big(\Big[\mathit{GNN}_{in}^{L}\big(\mathbf{X}_{:,t}, \mathbf{A}_{:,:,t}\big)\Big]_i, \boldsymbol{h}_{i,t-1}^{node}\Big). \quad \mathbf{Z} \quad = \boldsymbol{H}_{:,T}^{node}, \quad \forall i \in \mathcal{V}, \tag{1}$$

where $\boldsymbol{h}_{i,t}^{node}(1 \le t \le T)$ denotes the embedding of node $i \in \mathcal{V}$, $\mathit{GNN}_{in}^{L}(\mathbf{X}_{:,t}, \boldsymbol{a}_{:,:,t})$ denotes graph learning on the current snapshot through L layer GNNs. The learned embeddings, $\boldsymbol{h}_{i,t-1}^{node}$, are then passed into the RNN cell to capture the temporal dependencies. The last step output $\boldsymbol{h}_{:,T}$ is considered the final representation $\mathbf{Z}$.

Definition 4 (Time-and-graph)**.** *This architecture alternately processes time and graph components, applying GNNs to each EEG snapshot, as formally defined by:*

$$\boldsymbol{h}_{i,t}^{node} = \mathit{Cell}\Big(\Big[\mathit{GNN}_{in}^{L}\big(\boldsymbol{X}_{:,t}, \mathbf{A}_{:,:,t}\big)\Big]_i, \Big[\mathit{GNN}_{rc}^{L}\big(\boldsymbol{H}_{:,t-1}, \mathbf{A}_{:,:,t}\big)\Big]_i\Big), \quad \mathbf{Z} \quad = \boldsymbol{H}_{:,T}^{node}, \quad \forall i \in \mathcal{V}, \tag{2}$$

where we initialize $H_{i,0} = 0$. GNN_{in}^{L} encodes each $\boldsymbol{x}_{:,t}$ while GNN_{rc}^{L} encodes representations from historical snapshots $\boldsymbol{H}_{:,t-1}$, and RNN cell embeds evolution of those graph representations.

Definition 5 (Time-then-graph)**.** *This architecture first models the evolution of node and edge attributes over time and then applies a GNN to the resulting static graph for final representation:*

$$\begin{aligned} \mathbf{h}_{\mathbf{i}}^{node} &= \mathit{RNN}^{node}\big(\boldsymbol{X}_{i,\le T}\big), \quad \forall i \in \mathcal{V}, \quad \mathbf{h}_{\mathbf{i,j}}^{edge} \quad = \mathit{RNN}^{edge}\big(\boldsymbol{a}_{i,j,\le T}\big), \quad \forall (i,j) \in \mathcal{E}, \\ \mathbf{Z} &= \mathit{GNN}^{L}\big(\mathbf{H}^{node}, \mathbf{H}^{edge}\big). \end{aligned} \tag{3}$$

time-then-graph represents the evolution of $\boldsymbol{h}^{node}$ and $\boldsymbol{h}^{edge}$ using two sequential models RNN^{node} and RNN^{edge}, resulting in a new (static) graph, which is then encoded by a GNN^{L}.

Problem 2 (Expressive Dynamic EEG Graph Architecture)**.** *Determine which of these three architectures exhibits the highest expressiveness for dynamic EEG graph modeling, and characterize their relative representational power.*

3 Theoretical Analysis for Dynamic EEG Graphs

In this section, we aim to provide a theoretical analysis of the two problems. To this end, we employ 1-Weisfeiler-Lehman (1-WL) GNNs, a standard tool for analyzing graph isomorphism, as detailed in Appendix B. In the context of dynamic EEG analysis, an expressive model should be able to distinguish between different brain states by identifying non-isomorphic graphs.

Remark. For Theorem 2, we follow the general GNN taxonomy proposed by Gao and Ribeiro (2022), but extend the analysis to EEGs, specifically focusing on node-level expressivity in dynamic EEG graphs. Their analysis partially targets edge- or structure-level representations using unattributed graphs, but it does not explicitly consider node features. In contrast, node features are essential in EEG analysis, as EEG graph construction typically relies on pairwise similarity between channels (Ho and Armanfard, 2023; Tang et al., 2022). To this end, we provide formal theorems and proofs that consistently incorporate node features throughout the expressivity analysis. We also discuss the necessity of jointly modeling both node and edge representations from the perspective of EEG graphs in Appendix C.1.

Theorem 1. *[Implicit $\precneqq$ Explicit Dynamic Graph Modeling.] Explicit dynamic modeling (dynamic adjacency matrices) is strictly more expressive than implicit dynamic modeling (static graph structures) in capturing spatiotemporal dependencies in EEG signals.*

Proof. Let $\mathcal{F}_{\text{implicit}}$ and $\mathcal{F}_{\text{explicit}}$ denote the function classes expressible by implicit and explicit dynamic models, respectively. Since an explicit model can replicate any implicit model by ignoring time variations in adjacency, it follows that $\mathcal{F}_{\text{implicit}} \subseteq \mathcal{F}_{\text{explicit}}$. To show strict separation, we construct two temporal EEG graphs that share identical node features but differ in adjacency at a single time step. An implicit model compresses adjacency into a static representation, potentially making these graphs indistinguishable, while an explicit model processes time-varying adjacency and can distinguish them. Thus, $\mathcal{F}_{\text{implicit}} \neq \mathcal{F}_{\text{explicit}}$, proving $\mathcal{F}_{\text{implicit}} \subset \mathcal{F}_{\text{explicit}}$. The full proof is provided in Appendix C.2. □

Lemma 1. *[graph-then-time $\precneqq$ time-and-graph] time-and-graph is strictly more expressive than graph-then-time representation family on $\mathcal{T}_{n,T,\mathcal{H}eta}$ as long as we use 1-WL GNNs.*

Proof. By Definition 3, $\boldsymbol{h}_{i,t-1}$ is passed without this additional GNN (i.e., $\text{GNN}_{\text{rc}}^{L}(\cdot)$) to learn interactions between EEG snapshots. This results in a simpler form of temporal representation compared to *time-and-graph*: $\boldsymbol{h}_{i,t-1} \subseteq \left[\text{GNN}_{\text{rc}}^{L}\left(\mathbf{H}_{:,t-1}, \mathbf{A}_{:,:,t}\right)\right]_i$. *graph-then-time* is a strict subset of *time-and-graph* in terms of expressiveness. □

Lemma 2. *[time-and-graph $\precneqq$ time-then-graph] time-then-graph is strictly more expressive than time-and-graph representation family on $\mathbb{T}_{n,T,\theta}$, as time-then-graph outputs different representations, while time-and-graph does not.*

In Appendix C.3, we prove Lemma 2 using both node and edge representation perspectives (based on Lemma 4 in Appendix C.1) to hold Theorem 2. Notably, we provide a synthetic EEG task where any *time-and-graph* representation fails, while a *time-then-graph* approach succeeds. *time-then-graph* learns node and edge features across time steps to capture temporal dependencies. This is done by encoding the temporal adjacency matrices $\mathbf{A}_{:,:,\leq t}$ and node features $\mathbf{X}_{:,\leq t}$ together, enabling the model to distinguish between graphs with distinct temporal structures. However, *time-and-graph* handles each time step independently, leading to identical representations across time.

Theorem 2. *[Temporal EEG Graph Expressivity] Based on Lemmas 1 and 2, we conclude that graph-then-time is strictly less expressive than time-and-graph, and time-and-graph is strictly less expressive than time-then-graph on $\mathbb{T}_{n,T,\theta}$, when the graph representation is a 1-WL GNN:*

$$\textit{graph-then-time} \precneqq_{\mathbb{T}_{n,T,\theta}} \textit{time-and-graph} \quad \precneqq_{\mathbb{T}_{n,T,\theta}} \textit{time-then-graph}. \tag{4}$$

We confirm this conclusion in our experiments of both seizure detection and early prediction tasks.

4 Proposed Method - EvoBrain

Based on the above analysis, this section presents our `EvoBrain`, which is built on top of explicit dynamic modeling and *time-then-graph* architecture to represent temporal EEG structures.

4.1 Explicit Dynamic Brain Graph Structure

Based on Theorem 1, we propose to construct EEG graphs for each snapshot instead of constructing a single static graph from the entire EEG recording. We first segment an EEG epoch into short time durations (i.e., snapshots) at regular intervals and compute channel correlations to construct a sequence of graph structures. Specifically, for the t-th snapshot, we define the edge weight $a_{i,j,t}$ as the weighted adjacency matrix $\mathcal{A}$, computed as the absolute value of the normalized cross-correlation between nodes v_i and v_j. To prevent information redundancy and create sparse graphs, we rank the correlations among neighboring nodes and retain only the edges with the top-τ highest correlations.

$$a_{i,j,t} = |x_{i,t} * x_{j,t}|, \quad \text{if } v_j \in \mathcal{N}(v_i), \text{ else } 0,$$

where $x_{i,:,t}$ and $x_{j,:,t}$ represent v_i and v_j channels of t-th EEG snapshot. $*$ denotes the normalized cross-correlation operation. $\mathcal{N}(v_i)$ denotes the set of top-τ neighbors of v_i with higher correlations. After computing this for T snapshots, we obtain a sequence of directed, weighted EEG graphs $\mathbf{G}$ to represent brain networks at different time points. In other words, the dynamic nature of the EEG is captured by the evolving structure of these graphs over time.

4.2 Dynamic GNN in Time-Then-Graph Model

Based on Theorem 2, we propose a novel *time-then-graph* model, where two-stream Mamba Gu and Dao (2024) learn the temporal evolution of node and edge attributes independently, followed by Graph Convolutional Networks (GCN) to capture spatial dependencies across electrodes in a static graph. This method effectively captures the temporal and spatial dynamics inherent in EEG data for seizure detection and prediction. Notably, the model input is not the raw EEG signals but their **frequency** representation. Here, clinical seizure analysis aims to identify specific frequency oscillations and waveforms, such as spikes (Khan et al., 2018). To effectively capture such features,

we apply a short-term Fourier transform (STFT) to the EEG signal, retaining the log amplitudes of the non-negative frequency components, following prior studies (Covert et al., 2019; Asif et al., 2020; Tang et al., 2022). The EEG frequency snapshots are then normalized using z-normalization across the training set. Consequently, an EEG frequency representation with a sequence of snapshots is formulated as $\mathcal{X} \in \mathbb{R}^{N \times T \times d}$, serving N node initialization and dynamic graph construction.

4.2.1 Temporal Modeling with Mamba

We introduce a two-stream Mamba framework, with separate processes for node and edge attributes to capture their temporal dynamics. Given a dynamic EEG graph $\mathcal{G}$, Mamba can be interpreted as a linear RNN with selective state updates, making it suitable for modeling brain dynamics that involve both short-term and long-term memory processes. In the brain, short-term memory maintains transient information over brief intervals, while long-term memory spans extended timescales, enabling the integration of past experiences into current processing.

For a traditional RNN processing sequence $\{\mathbf{x}_t\}_{t=1}^{T}$, the hidden state update is: $\mathbf{h}_t = \sigma(\mathbf{W}\mathbf{h}_{t-1} + \mathbf{U}\mathbf{x}_t)$, where the weight matrices $\mathbf{W}$ and $\mathbf{U}$ representing synaptic connectivity between neurons are fixed across time t. However, the synaptic weights are constantly changing, controlled in part by chemicals such as neuromodulators (Citri and Malenka, 2008). This limits the model's ability to evolve information over long sequences (Costacurta et al., 2024). In contrast, Mamba can be viewed as a linear RNN with input-dependent parameters for each element $\mathbf{x^e}$ (node feature $\mathbf{x_{i,t}}$ or weighted adjacency matrix $\mathbf{a_{i,j,t}}$):

$$\begin{aligned} \Delta_t^e &= \tau_\Delta(f_\Delta^e(\mathbf{x}_t^e)), \quad \mathbf{B}_t^e = f_B^e(\mathbf{x}_t^e), \quad \mathbf{C}_t^e = f_C^e(\mathbf{x}_t^e), \\ \mathbf{h}_t^e &= \underbrace{(1 - \Delta_t^e \cdot \mathbf{D})}_{\text{selective forgetting}} \mathbf{h}_{t-1} + \underbrace{\Delta_t^e \cdot \mathbf{B}_t^e}_{\text{selective update}} \mathbf{x}_t^e, \quad \mathbf{y}_t^e = \mathbf{C}_t^e \mathbf{h}_t^e, \end{aligned} \tag{5}$$

where f_* are learnable projections for important frequency bands or edge connectivity. The hidden state $\mathbf{h}_t^e$ encodes the evolving neural activity, serving as a substrate for both transient (working) memory and more persistent (long-term) memory traces. The softplus activation τ_Δ guarantees that the gating variable Δ_t^e remains positive, thereby modulating the trade-off between retaining previous neural states and incorporating new inputs. Specifically, the forgetting term $(1 - \Delta_t^e \cdot \mathbf{D})$ implements a selective mechanism analogous to **synaptic decay or inhibitory processes** that diminish outdated or irrelevant information. Conversely, the update term $\Delta_t^e \cdot \mathbf{B}_t^e$ mirrors **neuromodulatory gating** (e,g, via dopamine signaling) that selectively reinforces and integrates salient new information. The projection $\mathbf{C}_t^e$ translates the internal neural state into observable outputs. The final hidden states $\mathbf{h}_i^{\text{node}} = \mathbf{y}_{i,T}^{\text{node}}$ and $\mathbf{h}_{ij}^{\text{edge}} = \mathbf{y}_{ij,T}^{\text{edge}}$ capture the temporal evolution of nodes and edges.

4.2.2 Spatial Modeling with GCN and LapPE

We adapt Graph Convolutional Network (GCN) and Laplacian Positional Encoding (LapPE) (Wang et al., 2022) for efficiently modeling spatial dependencies in EEG signals.

Capturing Brain Network Topology. Recent studies of neuroscience reveal that particular functions are closely associated with specific brain regions (e.g., Neocortex or Broca's area) (Hawkins and Ahmad, 2016; Rubinov and Sporns, 2010). However, due to the fundamental computation of GNNs, nodes that are structurally equivalent under graph automorphism receive identical representations, effectively losing their individual identities (Wang et al., 2022). In order to reflect the spatial specificity of the brain within spatial modeling, we introduce Laplacian Positional Encoding (LapPE). Given edge feature $\mathbf{H}^{edge}$ the normalized Laplacian $\mathbf{L}$ is defined as: $\mathbf{L} = \mathbf{I} - (\mathbf{D}^{-1/2}\mathbf{A'}\mathbf{D}^{-1/2}) = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top, \quad \mathbf{A'} = \tau_{\text{edge}}(f_{edge}(\mathbf{H}^{edge}))$, where τ_{edge} is the softplus function, f_{edge} is the learnable projection, $\mathbf{A'}$ is a weighted adjacency matrix $\mathbf{A'}$, and $\mathbf{D}$ is a degree matrix, diagonal with $\mathbf{D} = \{\sum_j \mathbf{a'}_{i,j}\}_{i=1}^{N}$. $\mathbf{I} \in \mathbb{R}^{N \times N}$ is the identity matrix, and $\mathbf{D}^{-1/2}$ is the element-wise inverse square root of the degree matrix $\mathbf{D}$. Performing an eigendecomposition on $\mathbf{L}$ yields. The columns of $\mathbf{U} \in \mathbb{R}^{N \times N}$ are the eigenvectors of $\mathbf{L}$, and $\mathbf{\Lambda}$ is a diagonal matrix containing the corresponding eigenvalues.

To capture the global geometry of the network, we select the first K eigenvectors corresponding to the smallest eigenvalues. For node i, its Laplacian-based positional encoding $\mathbf{p}_i \in \mathbb{R}^K$ is given by $\mathbf{p}_i = [\mathbf{u}_1[i], \mathbf{u}_2[i], \ldots, \mathbf{u}_K[i]]^\top, \quad \mathbf{x}_i^{node} = [\mathbf{h}_i^{\text{node}}; \mathbf{p}_i]$, where $\mathbf{u}_k[i]$ denotes the i-th component of the k-th eigenvector and $[;]$ indicates vector concatenation.

Modeling EEG Spatial Dynamics. We then adapt a GCN to learn spatial dependencies. These graph embeddings capture the temporal evolution of EEG snapshots, with each snapshot reflecting the brain state at that particular time. Each layer of the GCN updates the node embeddings by aggregating information from neighboring nodes and edges. The node embeddings are updated as follows:

$\mathbf{h}_i^{(l+1)} = \sigma\left((\mathbf{D}^{-1/2}\mathbf{A}'\mathbf{D}^{-1/2})\mathbf{h}_j^{(l)}\mathbf{\Theta}^{(l)}\right)$, where $\mathbf{h}_i^{(l)}$ is the embedding of node i at layer l and $\mathbf{h}_i^{(0)} = \mathbf{x}_i^{node}$. $\mathbf{\Theta}^{(l)}$ is the learnable weight matrix at layer l, and σ is an activation function (e.g., ReLU). Afterward, we apply max pooling over the node embeddings, i.e., $\mathbf{Z} = \mathbf{h}_i^{(L)}$, followed by a fully connected layer and softmax activation for seizure detection and prediction tasks.

5 Experiments and Results

5.1 Experimental Setup

Tasks. In this study, we focus on two tasks: seizure detection and seizure early prediction.

- **Seizure detection** is framed as a binary classification problem, where the goal is to distinguish between seizure and non-seizure EEG segments, termed epochs. This task serves as the foundation for automated seizure monitoring systems.
- **Seizure early prediction** is the more challenging and clinically urgent task. It aims to predict the onset of an epileptic seizure before it occurs. Researchers typically frame this task as a classification problem (Burrello et al., 2020; Batista et al., 2024), where the goal is to distinguish between pre-ictal EEG epochs and the normal state. Accurate classification enables timely patient warnings or preemptive interventions, such as electrical stimulation, to prevent or mitigate seizures. The seizure prediction task is defined as a classification problem between inter-ictal (normal) and pre-ictal states (Burrello et al., 2020). However, there is no clear clinical definition regarding its onset or duration of pre-ictal state (Lopes et al., 2023). So it is defined as a fixed duration before the seizure occurrence (Batista et al., 2024). This duration is chosen to account for the time required for stimulation by implanted devices (Cook et al., 2013) and to allow for seizure preparation. In this study, we define the pre-ictal state as one minute, providing adequate time for effective electrical stimulation to mitigate seizures or minimal preparation. Data labeled as seizures were discarded, and a five-minute buffer zone around the boundary data was excluded from the analysis. The remaining data were used as the normal state.

Datasets. We used the Temple University Hospital EEG Seizure dataset v1.5.2 (**TUSZ**) (Shah et al., 2018) to evaluate `EvoBrain`. Data description can be found in Appendix F. TUSZ is the largest public EEG seizure database, containing 5,612 EEG recordings with 3,050 annotated seizures. Each recording consists of 19 EEG channels. Additionally, we used the smaller CHB-MIT dataset, which consists of 844 hours of 22-channel scalp EEG data from 22 patients, including 163 recorded seizure episodes. For the TUSZ dataset, we followed the official data split, in which a subset of patients is designated for testing. Similarly, for the CHB-MIT dataset, we used randomly selected 15% of the patient's data for testing. Hyperparameters and augmentation strategies were set following prior studies and can be found in Appendix D. *Preprocessing:* For the early prediction task, we defined the *one-minute period* before a seizure as the preictal phase, implying the ability to predict seizures up to one minute in advance.

Baselines. We selected four dynamic GNNs studies as baselines: EvolveGCN-O (Pareja et al., 2020), which follows the *graph-then-time* architecture, and a *time-and-graph* work, DCRNN (Tang et al., 2022). *time-then-graph* approach, GRAPHS4MER (Tang et al., 2023) and GRU-GCN Gao and Ribeiro (2022) are *time-then-graph* architectures. We included EEG foundation models, BIOT (Yang et al., 2023a), LaBraM (Jiang et al., 2024), and EEGPT (Wang et al., 2024). LaBraM and EEGPT were fine-tuned from the publicly available base model checkpoint. We also evaluated LSTM (Hochreiter and Schmidhuber, 1997) and CNN-LSTM (Ahmedt-Aristizabal et al., 2020) as referenced in Tang et al. (2022) and conventional methods, Support Vector Machine (SVM) and Random Forest.

Metrics. We used the Area Under the Receiver Operating Characteristic Curve (AUROC) and F1 score as evaluation metrics. AUROC considers various threshold scenarios, providing an overall measure of the model's ability to distinguish between classes. The F1 score focuses on selecting the

Table 1: Performance comparison of TUSZ dataset on seizure detection and prediction for 12s and 60s. The **best** and second best results are highlighted.

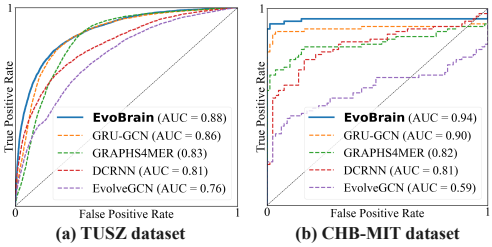
	Detection				Prediction			
	12s		60s		12s		60s	
Method	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1
SVM	0.765 ±0.004	0.369 ±0.007	0.720 ±0.017	0.390 ±0.019	0.566 ±0.016	0.320 ±0.037	0.561 ±0.025	0.312 ±0.029
Random Forest	0.778 ±0.004	0.354 ±0.005	0.739 ±0.031	0.386 ±0.038	0.566 ±0.032	0.344 ±0.037	0.550 ±0.037	0.330 ±0.037
LSTM	0.794 ±0.006	0.381 ±0.019	0.720 ±0.014	0.392 ±0.012	0.572 ±0.024	0.353 ±0.011	0.559 ±0.026	0.393 ±0.025
CNN-LSTM	0.754 ±0.009	0.356 ±0.008	0.680 ±0.007	0.331 ±0.016	0.621 ±0.018	0.389 ±0.010	0.528 ±0.020	0.316 ±0.028
BIOT	0.726 ±0.016	0.320 ±0.018	0.651 ±0.024	0.280 ±0.013	0.576 ±0.019	0.425 ±0.016	0.574 ±0.006	0.388 ±0.008
LaBraM	0.825 ±0.003	0.472 ±0.013	0.793 ±0.002	0.469 ±0.010	0.661 ±0.003	**0.482 ±0.010**	**0.669 ±0.014**	**0.413 ±0.020**
EEGPT	0.803 ±0.005	0.415 ±0.011	0.743 ±0.003	0.406 ±0.012	0.672 ±0.005	0.465 ±0.012	0.610 ±0.018	0.396 ±0.022
EvolveGCN	0.757 ±0.004	0.343 ±0.012	0.670 ±0.017	0.340 ±0.015	0.622 ±0.006	0.437 ±0.010	0.531 ±0.020	0.344 ±0.019
DCRNN	0.817 ±0.008	0.415 ±0.039	0.808 ±0.014	0.435 ±0.019	0.634 ±0.021	0.401 ±0.024	0.601 ±0.031	0.397 ±0.029
GRAPHS4MER	0.833 ±0.005	0.413 ±0.017	0.778 ±0.021	0.439 ±0.012	0.632 ±0.021	0.438 ±0.022	0.638 ±0.025	0.355 ±0.031
GRU-GCN	0.856 ±0.009	0.505 ±0.009	0.822 ±0.013	0.438 ±0.014	0.659 ±0.020	0.453 ±0.012	0.601 ±0.028	0.392 ±0.017
`EvoBrain` **(Ours)**	**0.877 ±0.005**	**0.539 ±0.009**	**0.865 ±0.009**	**0.483 ±0.006**	**0.675 ±0.015**	0.470 ±0.003	0.651 ±0.023	0.401 ±0.023

best threshold by balancing precision and recall. All results are averaged over five runs with different random seeds. Experimental details are provided in Appendix E.

5.2 Results

Main Results. Table 1 presents a performance comparison of seizure detection and prediction for the TUSZ dataset using various models over 12-second and 60-second windows. `EvoBrain` consistently outperforms dynamic GNN baselines and achieves competitive performance to the large-scale foundation model, LaBraM, while using **30 times fewer parameters**. For the seizure detection in a 12-second window data, `EvoBrain` shows a 14% increase in AUROC and 12% increase in F1 score compared with LaBraM. For the 60-second window data, `EvoBrain` shows a 23% increase in AUROC and 30% ($0.670 \rightarrow 0.865$) increase in F1 score ($0.340 \rightarrow 0.483$) compared with EvolveGCN. All *time-then-graph* models demonstrate relatively high performance, supporting our theoretical analysis and conclusion of Theorem 2 in Section 3.

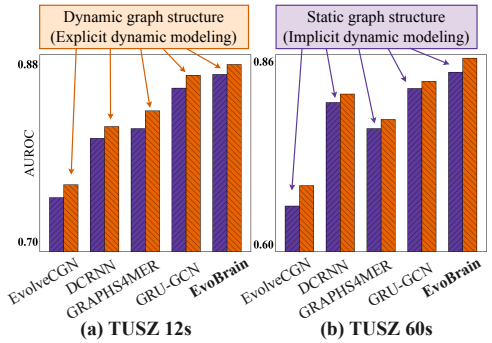
Figure 2 shows the ROC curves results comparing `EvoBrain` with other dynamic GNN models on seizure detection task. In subfigure (a), for the TUSZ dataset, `EvoBrain` achieves an AUC of 0.88, outperforming DCRNN (0.81) and EvolveGCN (0.76). Our ROC curve is positioned higher, indicating a stronger ability to differentiate between seizure and non-seizure events. In subfigures (b) for the CHB-MIT dataset, `EvoBrain` achieves an AUC of 0.94, significantly higher than the 0.81 and 0.59 of *time-and-graph* and *graph-then-time* approaches, respectively. The results show the effectiveness of *time-then-graph* for identifying seizures.



(a) TUSZ dataset **(b) CHB-MIT dataset**

Figure 2: ROC curve results for the 12-second seizure detection task on two datasets.

Dynamic Graph Structure Evaluation. Figure 3 shows the effectiveness of our proposed dynamic graph structure (i.e,. explicit dynamic modeling) compared to the static graph structure (i.e., implicit dynamic modeling) commonly used in existing work (Ho and Armanfard, 2023; Tang et al., 2022). The purple bar shows the performance of the static graph structure, while the orange bar represents the results when the static graph is replaced with our dynamic graph structures. As seen, the improvements are **not limited** to our *time-then-graph* method but also enhance the performance of all dynamic GNNs approaches. The figure highlights the effectiveness and necessity of dynamic graphs in capturing brain dynamics. The results imply that



(a) TUSZ 12s **(b) TUSZ 60s**

Figure 3: Comparison of the proposed dynamic graph structure and the static structure.

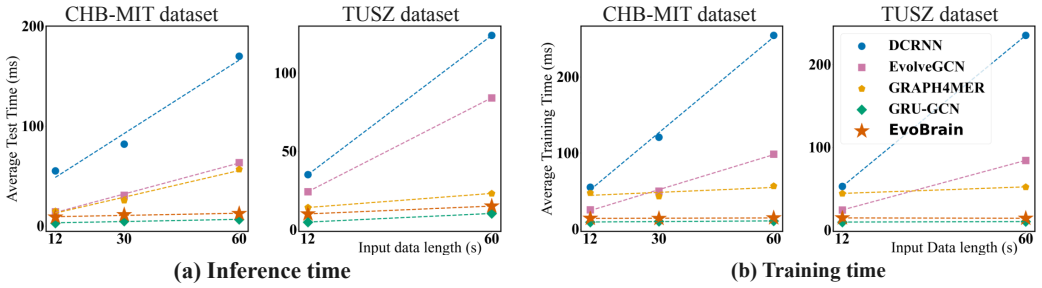


Figure 4: (a) Inference and (b) training time vs. input data length on CHB-MIT and TUSZ datasets. Our model achieves up to **17x faster** training times and **14x faster** inference times than its competitors, demonstrating scalability.

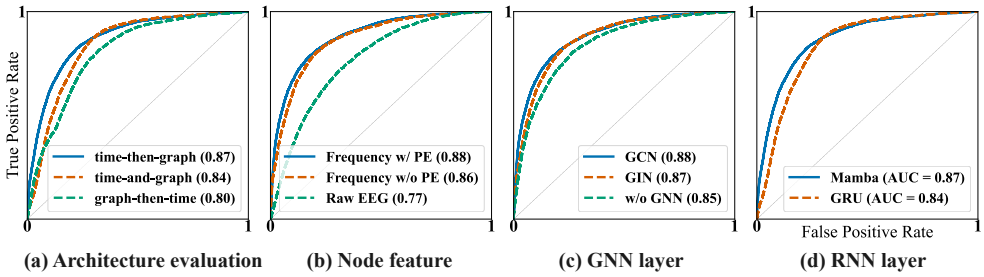


Figure 5: (a) Architecture evaluation using same GNN and RNN layers. (b) Results using raw EEG instead of frequency-domain features. (c) GNN and (d) RNN layer evaluation.

modeling temporal dynamics in EEGs should incorporate various channel connectivity or structural information, supporting our theoretical analysis and conclusion of Theorem 1 in Section 3.

Computational Efficiency. To assess the computational efficiency of our method, we measured the training time and inference time. Dynamic GNNs require computation time that scales with the length of the input data (details are provided in Appendix D). Figure 4 (a) illustrates the average training time per step for dynamic GNNs with a batch size of 1 across various input lengths. In practice, while the RNN component operates very quickly, the GNN processing accounts for most of the computation time. Since our method performs GNN processing only once for each data sample, it is up to 17$\times$ faster training time and more than 14$\times$ faster inference time than DCRNN. Thus, our approach is not only superior in performance but also fast in computational efficiency.

Ablation Study. Figure 5 (a) shows the results of different architectures applied to the same GRU and GCN layers using the 12-second TUSZ dataset on the seizure detection task. Consistent with Theorem 1, the time-then-graph architecture achieved the best performance. Figure 5 (b) shows the results when the FFT processing or LapPE was removed. Figure 5 (c) illustrates that replacing the GCN with GIN yields nearly identical performance, whereas removing the GNN and using only the RNN leads to a performance drop. Figure 5 (d) shows results of replacing Mamba with GRU layers on the seizure detection task using the 60-second TUSZ dataset. The results demonstrates that Mamba provides performance benefits, especially for longer input sequences.

5.3 Clinical Analysis.

We show a case analysis of our constructed dynamic graphs from a neuroscience perspective. Figure 6 shows the top-10 edges with the strongest connections in the learned dynamic graphs , where the yellow color represents the strength of the connections. These edges are selected based on the highest values, indicating the most significant relationships captured by the model. In Figure 6 (a), a sample unrelated to a seizure shows weak, sparse connections spread across various regions over an extended period. Figure 6 (b) shows a pre-seizure sample, where some connections gradually strengthen. Figure 6 (c) shows the result of a focal seizure, a type of seizure that originates in a specific area, with sustained strong connections only in a specific region. In Figure 6 (d), a generalized seizure is

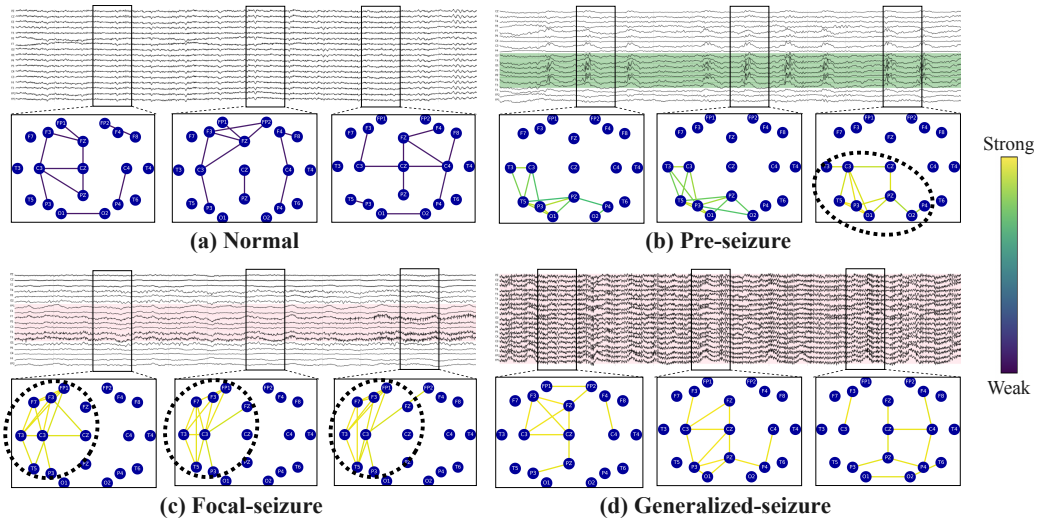


Figure 6: Learned graph structure visualizations. The yellow color of the edges indicates the strength of the connections. In (a) Normal state, the dark shows weak connections. In (b) Pre-seizure state, the connections in specific regions strengthen over time. In (c) Focal seizure, which occurs only in a specific area of the brain, strong connections are consistently present in a particular region. In (d) Generalized seizure, strong connections are observed across the entire brain.

illustrated, characterized by strong connections across the entire brain.
Successful surgical and neuromodulatory treatments critically depend on accurate localization of the seizure onset zone (SOZ) (Li et al., 2021a). Even the most experienced clinicians are challenged because there is no clinically validated biomarker of SOZ. Prior studies have shown that abnormal connections across several channels may constitute a more effective marker of the SOZ (Scharfman, 2007; Burns et al., 2014b; Li et al., 2018). Our dynamic graph structures align with neuroscientific observations, successfully visualizing these abnormal connections and their changes. This offers promising potential for application in surgical planning and treatment strategies.

6 Conclusion

In this work, we propose a dynamic multi-channel EEG graph modeling approach, `EvoBrain`. By adopting a *time-then-graph* strategy together with explicit dynamic modeling, `EvoBrain` captures the evolving nature of brain networks. Our theoretical analysis establishes the expressivity advantages of both explicit dynamic graphs and the *time-then-graph* paradigm over alternatives, providing a principled foundation for the design choices. `EvoBrain` yields substantial empirical gains, including improvements of 23% in AUROC and 30% in F1 over a strong dynamic GNN baseline, and strong performance on early seizure prediction. Case clinical analyses further highlight potential clinical utility by revealing network changes consistent with seizure progression. Taken together, these results suggest `EvoBrain` is a practical and theoretically grounded step toward reliable, clinically meaningful EEG-based seizure analysis.

Limitations and Social Impacts. In seizure prediction, pre-ictal patterns are typically weaker and more spatially diffuse. While EvoBrain achieves the best performance among lightweight GNN-based models, LaBraM benefits from more parameters and pretraining on large-scale multiple EEG corpora, which enhances its ability to generalize under limited and noisy pre-seizure data conditions. While we focus our evaluation on the seizure task, generalization to other tasks remains a challenge of future work. Regarding potential negative societal impacts, we recognize key risks such as bias and system malfunction. Specifically, models trained on EEG data from specific demographic groups may exhibit biased performance when applied to broader populations, potentially leading to unequal diagnostic accuracy. Additionally, miscalibrated early seizure prediction could lead to false alarms, causing unnecessary interventions or patient distress. These challenges highlight promising directions for future work, where incorporating fairness assessments, demographic audits, and human-in-the-loop strategies can lead to more robust and trustworthy EEG-based systems.

Acknowledgments

The authors would like to sincerely thank the anonymous reviewers for their valuable comments and helpful suggestions. This work was supported by JST BOOST, Japan Grant Number JPMJBS2402, "Program for Leading Graduate Schools" of The University of Osaka, JSPS KAKENHI Grant-in-Aid for Scientific Research Number JP24K20778, JST CREST JPMJCR23M3, JST START JPMJST2553, JST CREST JPMJCR20C6, the Japan Science and Technology Agency (JST) AIP Acceleration Research (JPMJCR24U2, TY), Japan Agency for Medical Research and Development (AMED, JP19dm0307103, HK, JP24wm0625207, TY), the Japan Science and Technology Agency (JST) Moonshot R &D-MILLENNIA Program (JPMJMS2012, TY), AIP Acceleration Research (JPMJCR24U2, TY), JST K Program JPMJKP25Y6, JST COI-NEXT JPMJPF2009, JST COI-NEXT JPMJPF2115,Future Social Value Co-Creation Project - The University of Osaka.

References

David Ahmedt-Aristizabal, Tharindu Fernando, Simon Denman, Lars Petersson, Matthew J. Aburn, and Clinton Fookes. 2020. Neural Memory Networks for Seizure Type Classification. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. 569–575.

Shiva Asadzadeh, Tohid Yousefi Rezaii, Soosan Beheshti, and Saeed Meshgini. 2022. Accurate emotion recognition using Bayesian model based EEG sources as dynamic graph convolutional neural network nodes. *Scientific Reports* 12, 1 (2022), 10282.

Umar Asif, Subhrajit Roy, Jianbin Tang, and Stefan Harrer. 2020. SeizureNet: Multi-Spectral Deep Feature Learning for Seizure Type Classification. In *Machine Learning in Clinical Neuroimaging and Radiogenomics in Neuro-oncology*. 77–87.

Joana Batista, Mauro Pinto, Mariana Tavares, Fábio Lopes, Ana Oliveira, and César Teixeira. 2024. EEG epilepsy seizure prediction: the post-processing stage as a chronology. *Scientific Reports* (2024).

Karla Burelo, Georgia Ramantani, Giacomo Indiveri, and Johannes Sarnthein. 2022. A neuromorphic spiking neural network detects epileptic high frequency oscillations in the scalp EEG. *Scientific Reports* (2022), 1798.

Samuel P. Burns, Sabato Santaniello, Robert B. Yaffe, Christophe C. Jouny, Nathan E. Crone, Gregory K. Bergey, William S. Anderson, and Sridevi V. Sarma. 2014a. Network dynamics of the brain and influence of the epileptic seizure onset zone. *Proceedings of the National Academy of Sciences* (2014), E5321–E5330.

Samuel P. Burns, Sabato Santaniello, Robert B. Yaffe, Christophe C. Jouny, Nathan E. Crone, Gregory K. Bergey, William S. Anderson, and Sridevi V. Sarma. 2014b. Network dynamics of the brain and influence of the epileptic seizure onset zone. *Proceedings of the National Academy of Sciences* 49 (2014), E5321–E5330.

Alessio Burrello, Kaspar Schindler, Luca Benini, and Abbas Rahimi. 2020. Hyperdimensional Computing With Local Binary Patterns: One-Shot Learning of Seizure Onset and Identification of Ictogenic Brain Regions Using Short-Time iEEG Recordings. *IEEE Transactions on Biomedical Engineering* (2020), 601–613.

Donghong Cai, Junru Chen, Yang Yang, Teng Liu, and Yafeng Li. 2023. MBrain: A Multi-Channel Self-Supervised Learning Framework for Brain Signals. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 130–141.

Junru Chen, Yang Yang, Tao Yu, Yingying Fan, Xiaolong Mo, and Carl Yang. 2022. BrainNet: Epileptic Wave Detection from SEEG with Hierarchical Graph Diffusion Learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*. 2741–2751.

Zheng Chen, Yasuko Matsubara, Yasushi Sakurai, and Jimeng Sun. 2025. Long-Term EEG Partitioning for Seizure Onset Detection. *Proceedings of the AAAI Conference on Artificial Intelligence* (2025), 14221–14229.

Zheng Chen, Lingwei Zhu, Haohui Jia, and Takashi Matsubara. 2023. A Two-View EEG Representation for Brain Cognition by Composite Temporal-Spatial Contrastive Learning. In *SDM*. 334–342.

Ami Citri and Robert C Malenka. 2008. Synaptic plasticity: multiple forms, functions, and mechanisms. *Neuropsychopharmacology* (2008), 18–41.

Mark J Cook, Terence J O'Brien, Samuel F Berkovic, Michael Murphy, Andrew Morokoff, Gavin Fabinyi, Wendyl D'Souza, Raju Yerra, John Archer, Lucas Litewka, et al. 2013. Prediction of seizure likelihood with a long-term, implanted seizure advisory system in patients with drug-resistant epilepsy: a first-in-man study. *The Lancet Neurology* 12, 6 (2013), 563–571.

Filippo Costa, Eline Schaft, Geertjan Huiskamp, Erik Aarnoutse, Maryse Klooster, Niklaus Krayenbühl, Georgia Ramantani, Maeike Zijlmans, Giacomo Indiveri, and Johannes Sarnthein. 2024. Robust compression and detection of epileptiform patterns in ECoG using a real-time spiking neural network hardware framework. *Nature Communications* (2024).

Julia Costacurta, Shaunak Bhandarkar, David Zoltowski, and Scott Linderman. 2024. Structured flexibility in recurrent neural networks via neuromodulation. In *Advances in Neural Information Processing Systems*. 1954–1972.

Ian C. Covert, Balu Krishnan, Imad Najm, Jiening Zhan, Matthew Shore, John Hixson, and Ming Jack Po. 2019. Temporal Graph Convolutional Networks for Automatic Seizure Detection. In *Proceedings of the 4th Machine Learning for Healthcare Conference*. 160–180.

Andac Demir, Toshiaki Koike-Akino, Ye Wang, and Deniz Erdoğmuş. 2022. EEG-GAT: Graph Attention Networks for Classification of Electroencephalogram (EEG) Signals. In *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. 30–35.

Andac Demir, Toshiaki Koike-Akino, Ye Wang, Masaki Haruna, and Deniz Erdogmus. 2021. EEG-GNN: Graph neural networks for classification of electroencephalogram (EEG) signals. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. 1061–1067.

Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. 2021. Time-Series Representation Learning via Temporal and Contextual Contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. 2352–2359.

Kosuke Fukumori, Noboru Yoshida, Hidenori Sugano, Madoka Nakajima, and Toshihisa Tanaka. 2022a. Epileptic Spike Detection by Recurrent Neural Networks with Self-Attention Mechanism. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 1406–1410.

Kosuke Fukumori, Noboru Yoshida, Hidenori Sugano, Madoka Nakajima, and Toshihisa Tanaka. 2022b. Epileptic Spike Detection Using Neural Networks With Linear-Phase Convolutions. *IEEE Journal of Biomedical and Health Informatics* (2022), 1045–1056.

Jianfei Gao and Bruno Ribeiro. 2022. On the Equivalence Between Temporal and Static Equivariant Graph Representations. In *Proceedings of the 39th International Conference on Machine Learning*. 7052–7076.

Albert Gu and Tri Dao. 2024. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. *arXiv preprint arXiv:2312.00752* (2024). arXiv:2312.00752

Jeff Hawkins and Subutai Ahmad. 2016. Why neurons have thousands of synapses, a theory of sequence memory in neocortex. *Frontiers in neural circuits* (2016), 174222.

Thi Kieu Khanh Ho and Narges Armanfard. 2023. Self-Supervised Learning for Anomalous Channel Detection in EEG Graphs: Application to Seizure Analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 7866–7874.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 8 (1997), 1735–1780.

Yimin Hou, Shuyue Jia, Xiangmin Lun, Shu Zhang, Tao Chen, Fang Wang, and Jinglei Lv. 2022. Deep feature mining via the attention-based bidirectional long short term memory graph convolutional neural network for human motor imagery recognition. *Frontiers in Bioengineering and Biotechnology* 9 (2022), 706229.

Weibang Jiang, Liming Zhao, and Bao liang Lu. 2024. Large Brain Model for Learning Generic Representations with Tremendous EEG Data in BCI. In *The Twelfth International Conference on Learning Representations*.

Haidar Khan, Lara Marcuse, Madeline Fields, Kalina Swann, and Bülent Yener. 2018. Focal Onset Seizure Prediction Using Convolutional Networks. *IEEE Transactions on Biomedical Engineering* (2018), 2109–2118.

Diederik Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations* (2014).

Dominik Klepl, Fei He, Min Wu, Daniel J. Blackburn, and Ptolemaios Sarrigiannis. 2022. EEG-Based Graph Neural Network Classification of Alzheimer's Disease: An Empirical Evaluation of Functional Connectivity Methods. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* (2022), 2651–2660.

Rikuto Kotoge, Zheng Chen, Tasuku Kimura, Yasuko Matsubara, Takufumi Yanagisawa, Haruhiko Kishima, and Yasushi Sakurai. 2024. Splitsee: A splittable self-supervised framework for single-channel eeg representation learning. In *2024 IEEE International Conference on Data Mining (ICDM)*. 741–746.

Adam Li, Bhaskar Chennuri, Sandya Subramanian, Robert Yaffe, Steve Gliske, William Stacey, Robert Norton, Austin Jordan, Kareem A Zaghloul, Sara K Inati, et al. 2018. Using network analysis to localize the epileptogenic zone from invasive EEG recordings in intractable focal epilepsy. *Network Neuroscience* (2018), 218–240.

Adam Li, Chester Huynh, Zachary Fitzgerald, Iahn Cajigas, Damian Brusko, Jonathan Jagid, Angel Claudio, Andres Kanner, Jennifer Hopp, Stephanie Chen, Jennifer Haagensen, Emily Johnson, William Anderson, Nathan Crone, Sara Inati, Kareem Zaghloul, Juan Bulacio, Jorge Gonzalez-Martinez, and Sridevi Sarma. 2021b. Neural fragility as an EEG marker of the seizure onset zone. *Nature Neuroscience* (2021), 1–10.

Adam Li, Chester Huynh, Zachary Fitzgerald, Iahn Cajigas, Damian Brusko, Jonathan Jagid, Angel O Claudio, Andres M Kanner, Jennifer Hopp, Stephanie Chen, et al. 2021a. Neural fragility as an EEG marker of the seizure onset zone. *Nature neuroscience* (2021), 1465–1474.

Yang Li, Yu Liu, Yu-Zhu Guo, Xiao-Feng Liao, Bin Hu, and Tao Yu. 2022b. Spatio-Temporal-Spectral Hierarchical Graph Convolutional Network With Semisupervised Active Learning for Patient-Specific Seizure Prediction. *IEEE Transactions on Cybernetics* (2022), 12189–12204.

Zhengdao Li, Kai Hwang, Keqin Li, Jie Wu, and Tongkai Ji. 2022a. Graph-generative neural network for EEG-based epileptic seizure detection via discovery of dynamic brain functional connectivity. *Scientific reports* 12, 1 (2022), 18998.

Meng Liu, Yue Liu, KE LIANG, Wenxuan Tu, Siwei Wang, sihang zhou, and Xinwang Liu. 2024. Deep Temporal Graph Clustering. In *The Twelfth International Conference on Learning Representations*.

Fábio Lopes, Adriana Leal, Mauro F Pinto, António Dourado, Andreas Schulze-Bonhage, Matthias Dümpelmann, and César Teixeira. 2023. Removing artefacts and periodically retraining improve performance of neural network-based seizure prediction models. *Scientific Reports* 13, 1 (2023), 5918.

Navid Mohammadi Foumani, Geoffrey Mackellar, Soheila Ghane, Saad Irtza, Nam Nguyen, and Mahsa Salehi. 2024. EEG2Rep: Enhancing Self-supervised EEG Representation Through Informative Masked Inputs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 5544–5555.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. 2020. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence* (2020), 5363–5370.

Khansa Rasheed, Junaid Qadir, Terence J. O'Brien, Levin Kuhlmann, and Adeel Razi. 2021. A Generative Model to Synthesize EEG Data for Epileptic Seizure Prediction. *IEEE TNSRE* (2021), 2322–2332.

Scott Rich, Axel Hutt, Frances Skinner, Taufik Valiante, and Jérémie Lefebvre. 2020. Neurostimulation stabilizes spiking neural networks by disrupting seizure-like oscillatory transitions. *Scientific reports* (09 2020).

Edmund T. Rolls, Wei Cheng, and Jianfeng Feng. 2021. Brain dynamics: Synchronous peaks, functional connectivity, and its temporal variability. *Human Brain Mapping* (2021), 2790–2801.

Mikail Rubinov and Olaf Sporns. 2010. Complex network measures of brain connectivity: uses and interpretations. *Neuroimage* (2010), 1059–1069.

Khaled Saab, Jared Dunnmon, Christopher Ré, Daniel Rubin, and Christopher Lee-Messer. 2020. Weak supervision as an efficient approach for automated seizure detection in electroencephalography. *npj Digital Medicine* 3, 1 (2020), 1–12.

Helen E Scharfman. 2007. The neurobiology of epilepsy. *Current neurology and neuroscience reports* (2007), 348–354.

Vinit Shah, Eva Von Weltin, Silvia Lopez, James Riley McHugh, Lillian Veloso, Meysam Golmohammadi, Iyad Obeid, and Joseph Picone. 2018. The temple university hospital seizure detection corpus. *Frontiers in neuroinformatics* 12 (2018), 83.

Biao Sun, Han Zhang, Zexu Wu, Yunyan Zhang, and Ting Li. 2021. Adaptive Spatiotemporal Graph Convolutional Networks for Motor Imagery Classification. *IEEE Signal Processing Letters* (2021), 219–223.

Siyi Tang, Jared Dunnmon, Khaled Kamal Saab, Xuan Zhang, Qianying Huang, Florian Dubost, Daniel Rubin, and Christopher Lee-Messer. 2022. Self-Supervised Graph Neural Networks for Improved Electroencephalographic Seizure Analysis. In *International Conference on Learning Representations*.

Siyi Tang, Jared A Dunnmon, Qu Liangqiong, Khaled K Saab, Tina Baykaner, Christopher Lee-Messer, and Daniel L Rubin. 2023. Modeling Multivariate Biosignals With Graph Neural Networks and Structured State Space Models. In *Proceedings of the Conference on Health, Inference, and Learning*. 50–71.

Yuxing Tian, Yiyan Qi, and Fan Guo. 2024. FreeDyG: Frequency Enhanced Continuous-Time Dynamic Graph Model for Link Prediction. In *The Twelfth International Conference on Learning Representations*.

Guangyu Wang, Wenchao Liu, Yuhong He, Cong Xu, Lin Ma, and Haifeng Li. 2024. Eegpt: Pretrained transformer for universal and reliable representation of eeg signals. *Advances in Neural Information Processing Systems* 37 (2024), 39249–39280.

Haorui Wang, Haoteng Yin, Muhan Zhang, and Pan Li. 2022. Equivariant and Stable Positional Encoding for More Powerful Graph Neural Networks. In *International Conference on Learning Representations*.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *International Conference on Learning Representations*.

Chaoqi Yang, M Westover, and Jimeng Sun. 2023a. BIOT: Biosignal Transformer for Cross-data Learning in the Wild. In *Advances in Neural Information Processing Systems*, A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.). 78240–78260.

Chaoqi Yang, M Westover, and Jimeng Sun. 2023b. BIOT: Biosignal Transformer for Cross-data Learning in the Wild. In *Advances in Neural Information Processing Systems*. 78240–78260.

Ke Yi, Yansen Wang, Kan Ren, and Dongsheng Li. 2023. Learning Topology-Agnostic EEG Representations with Geometry-Aware Modeling. In *Advances in Neural Information Processing Systems*. 53875–53891.

Daoze Zhang, Zhizhang Yuan, YANG YANG, Junru Chen, Jingjing Wang, and Yafeng Li. 2023. Brant: Foundation Model for Intracranial Neural Signal. In *Advances in Neural Information Processing Systems*. 26304–26321.

Siwei Zhang, Xi Chen, Yun Xiong, Xixi Wu, Yao Zhang, Yongrui Fu, Yinglong Zhao, and Jiawei Zhang. 2024a. Towards Adaptive Neighborhood for Advancing Temporal Interaction Graph Modeling *(KDD '24)*. 4290–4301.

Zuozhen Zhang, Junzhong Ji, and Jinduo Liu. 2024b. MetaRLEC: Meta-Reinforcement Learning for Discovery of Brain Effective Connectivity. *Proceedings of the AAAI Conference on Artificial Intelligence* (2024), 10261–10269.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly state the claims made, including the contributions made in the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We point out the limitations of the work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.

- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All theorems are numbered, and we refer to the formal proofs in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: This paper has instructions on how to reproduce the main experimental results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed

instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We release source code. Datasets are available for download from their respective sources.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specify the experimental details.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The results are accompanied by standard deviations.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper provides information on the computer resources.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses both potential positive and negative societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the original paper of existing assets.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We release our source code.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

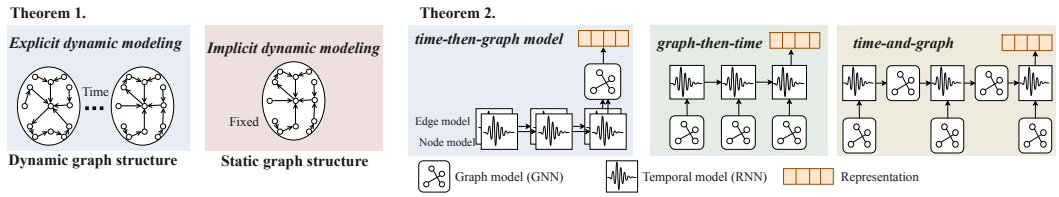
Answer: [NA]

Justification: LLM was used only for writing, editing, and formatting purposes.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix



(a) Two graph structures **(b) Three dynamic graph architectures**

Figure 7: We compare (a) explicit dynamic modeling with implicit dynamic modeling and (b) time-then-graph with graph-then-time and graph-and-time architectures.

A Related Work

Automated Seizure Analysis. The automated detection or prediction of seizures has been a long-standing challenge (Zhang et al., 2024b; Kotoge et al., 2024). Deep learning has shown great achievements in automating EEG feature extraction and detection, using convolutional neural networks (CNNs) (Fukumori et al., 2022b; Ahmedt-Aristizabal et al., 2020; Asif et al., 2020; Saab et al., 2020), RNN-based models (Fukumori et al., 2022a; Ahmedt-Aristizabal et al., 2020; Rasheed et al., 2021), Transformers (Eldele et al., 2021; Yang et al., 2023b; Jiang et al., 2024; Yi et al., 2023), and brain-inspired models (Rich et al., 2020; Burelo et al., 2022; Chen et al., 2023; Costa et al., 2024).

Spatial Relationships in EEG Networks. A seizure is fundamentally a network disease, and detection typically relies on the ability to determine abnormalities in EEG channels (Burns et al., 2014a; Li et al., 2021b; Rolls et al., 2021). Many multi-channel methods have been proposed for capturing spatial information in channels (Jiang et al., 2024; Yi et al., 2023; Zhang et al., 2023; Mohammadi Foumani et al., 2024). Among them, recent studies have proposed GNNs to capture further the non-Euclidean structure of EEG electrodes and the connectivity in brain networks (Covert et al., 2019; Sun et al., 2021; Li et al., 2022b; Chen et al., 2022; Klepl et al., 2022; Demir et al., 2022). These methods form EEGs as graphs, embedding each channel into the nodes and learning spatial graph representations (Demir et al., 2021; Ho and Armanfard, 2023). However, they do not explicitly model temporal relationships, relying instead on convolutional filters or conventional linear projections for node embeddings.

Dynamic GNNs for EEG Modeling. Dynamic GNN is effective in learning temporal graph dynamics, achieving promising results in tasks such as dynamic link prediction (Tian et al., 2024), node classification (Zhang et al., 2024a), and graph clustering (Liu et al., 2024). Recently, some studies have focused on dynamic GNNs aimed to enhance temporal and graph representations for EEG-based seizure modeling. Tang et al. (2022) proposes a *time-and-graph* model , which uses frequency features from FFT as node features and applies GNN and RNN processing simultaneously for each sliding window. Cai et al. (2023) adopts a *graph-then-time* model that combines GCN and RNN for seizure detection. However, these studies construct static graphs with fixed structures across temporal learning. Hou et al. (2022) propose *time-then-graph* model, BiLSTM-GCNNet, which uses RNNs to construct static graph from node feature. GRAPHS4MER (Tang et al., 2023) is also proposed as a *time-then-graph* method. This work learns dynamic graphs using an internal graph structure learning model. Similarly, both Asadzadeh et al. (2022) and Li et al. (2022a) internally learn dynamic graph structures. However, its input for graph structure learning is still based on the static Euclid distance or similarity of the entire data sample (e.g., 12 or 60 seconds) rather than individual snapshots. Our work differs by defining dynamic graph structures that more effectively capture the temporal evolution of brain connectivity in EEGs.

B Graph Isomorphism and 1-WL Test

Graph isomorphism refers to the problem of determining whether two graphs are structurally identical, meaning there exists a one-to-one correspondence between their nodes and edges. This is a crucial challenge in graph classification tasks, where the goal is to assign labels to entire graphs based

on their structures. A model that can effectively differentiate non-isomorphic graphs is said to have high expressiveness, which is essential for accurate classification. In many cases, graph classification models like GNNs rely on graph isomorphism tests to ensure that structurally distinct graphs receive different embeddings, which improves the model's ability to classify graphs correctly.

1-Weisfeiler-Lehman (1-WL) test is a widely used graph isomorphism test that forms the foundation of many GNNs. In the 1-WL framework, each node's representation is iteratively updated by aggregating information from its neighboring nodes, followed by a hashing process to capture the structural patterns of the graph. GNNs leveraging this concept, such as Graph Convolutional Networks (GCNs) and Graph Attention Networks (GATs), essentially perform a similar neighborhood aggregation, making them as expressive as the 1-WL test in distinguishing non-isomorphic graphs (Xu et al., 2019). Modern GNN architectures adhere to this paradigm, making the 1-WL a standard baseline for GNN expressivity. In our work, we also use 1-WL-based GNNs, leveraging their proven expressiveness for dynamic brain graph modeling.

C Proofs of Expressivity Analysis

C.1 Expressiveness with Node and Edge Representations

Lemma 3. *[Necessity of Node Representations]* ***Edges alone*** *(Gao and Ribeiro, 2022)* ***are insufficient*** *to uniquely distinguish certain temporal EEG graphs. Specifically, there exist pairs of temporal EEG graphs that have identical edge features across all time steps but different node features, making them indistinguishable based solely on edge representations. Therefore,* ***incorporating node representations is necessary*** *to achieve full expressiveness in EEG graph classification tasks.*

Proof. Given $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$, with the same sets of $\mathcal{V}$ and $\mathcal{E}$, for $\forall\ t$, the edge features satisfy: $a^{(1)}_{i,j,t} = a^{(2)}_{i,j,t} \quad \forall (v_i, v_j) \in \mathcal{E}, \forall t \in \{1, 2, \ldots, T\}$. However, suppose there exists at least one node $v_k \in \mathcal{V}$ and one time step t' such that: $\boldsymbol{x}^{(1)}_{k,t'} \neq \boldsymbol{x}^{(2)}_{k,t'}$. Since $a^{(1)}_{i,j,t} = a^{(2)}_{i,j,t}$, any GNN architecture that relies solely on edge features will produce identical embeddings for $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$, $\forall t$. □

Lemma 4. *[Expressiveness with Node and Edge Representations] When both node and edge representations are incorporated, a GNN can uniquely distinguish any pair of temporal EEG graphs that differ in either node features or edge features at any time step, provided the GNN is sufficiently expressive (e.g., 1-WL GNN).*

Proof. Given $\mathcal{G}^{(1)} = (\mathcal{A}^{(1)}, \mathcal{X}^{(1)})$ and $\mathcal{G}^{(2)} = (\mathcal{A}^{(2)}, \mathcal{X}^{(2)})$, suppose they differ in at least one node feature or edge feature at some time step t. An expressive GNN can produce different embeddings for these graphs by capturing the differences in node and/or edge features. Specifically:

1. If $\mathbf{X}^{(1)}_{:,t} \neq \mathbf{X}^{(2)}_{:,t}$ for some t, then the node embeddings $h^{(1)}_{i,t}$ and $h^{(2)}_{i,t}$ will differ for at least one node v_i.
2. If $\mathbf{a^{(1)}_{i,j,t}} \neq \mathbf{a^{(2)}_{i,j,t}}$ for some edge (v_i, v_j) and some t, then the edge embeddings $h^{(1)}_{ij,t}$ and $h^{(2)}_{ij,t}$ will differ for that edge.

Since the GNN aggregates information from both node and edge embeddings, any difference in either will propagate through the network, resulting in distinct final representations $\mathbf{Z}^{(1)}$ and $\mathbf{Z}^{(2)}$. Thus, the GNN can uniquely distinguish between $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$. □

C.2 Implicit and Explicit Dynamic Modeling

Theorem 1. *[Implicit $\precneqq$ Explicit Dynamic Graph Modeling.] Explicit dynamic modeling (dynamic adjacency matrices) is strictly more expressive than implicit dynamic modeling (static graph structures) in capturing spatiotemporal dependencies in EEG signals.*

Proof. Let $\mathcal{F}_{\text{implicit}}$ be the family of functions (e.g., representations or classifiers) expressible by an *implicit* dynamic graph model that compresses all adjacency snapshots $\{\mathbf{A}_{:,:,t}\}_{t=1}^{T}$ into a single

adjacency matrix $\widehat{\mathbf{A}}$, and let $\mathcal{F}_{\text{explicit}}$ be the family of functions expressible by an *explicit* dynamic graph model that uses the full time-varying adjacency $\{\mathbf{A}_{:,:,t}\}_{t=1}^{T}$. We prove

$$\mathcal{F}_{\text{implicit}} \subseteq \mathcal{F}_{\text{explicit}} \quad \text{and} \quad \mathcal{F}_{\text{implicit}} \neq \mathcal{F}_{\text{explicit}},$$

i.e.,

$$\mathcal{F}_{\text{implicit}} \subset ; \mathcal{F}_{\text{explicit}}.$$

(1) Subset Relationship. It can be interpreted as any implicit dynamic model aggregates the set of adjacency matrices $\{\mathbf{A}_{:,:,t}\}_{t=1}^{T}$ into a single, static $\widehat{\mathbf{A}}$ by some function $f(\cdot)$ (e.g., averaging, summation, thresholding):

$$\widehat{\mathbf{A}} = f\big(\{\mathbf{A}_{:,:,t}\}_{t=1}^{T}\big).$$

Such a model uses $\widehat{\mathbf{A}}$ for all time steps, disregarding changes in edges across t. In contrast, an explicit dynamic model directly processes the full sequence $\{\mathbf{A}_{:,:,t}\}_{t=1}^{T}$. To see that $\mathcal{F}_{\text{implicit}}$ is contained in $\mathcal{F}_{\text{explicit}}$, observe that any function $f_{\text{imp}} \in \mathcal{F}_{\text{implicit}}$ can be mimicked by an explicit model that simply *ignores* the time-specific variability in adjacency and substitutes $f(\{\mathbf{A}_{:,:,t}\}) = \widehat{\mathbf{A}}$ as if it were the adjacency for every t. Hence,

$$\forall\, f_{\text{imp}} \in \mathcal{F}_{\text{implicit}}, \quad f_{\text{imp}} \in \mathcal{F}_{\text{explicit}},$$

implying

$$\mathcal{F}_{\text{implicit}} \subseteq \mathcal{F}_{\text{explicit}}.$$

(2) Strict Separation by Counterexample. To show that $\mathcal{F}_{\text{implicit}} \neq \mathcal{F}_{\text{explicit}}$, we construct two temporal EEG graphs, $\mathcal{G}^{(1)} = \{\mathbf{X}_{:,t}, \mathbf{A}^{(1)}_{:,:,t}\}_{t=1}^{T}$ and $\mathcal{G}^{(2)} = \{\mathbf{X}_{:,t}, \mathbf{A}^{(2)}_{:,:,t}\}_{t=1}^{T}$, such that

1. Their node features match at every time step:

$$\mathbf{X}^{(1)}_{:,t} = \mathbf{X}^{(2)}_{:,t} \quad \forall\, t \in \{1, \ldots, T\}.$$

2. Their adjacency differs *only* at one time step t':

$$\mathbf{A}^{(1)}_{:,:,t'} \neq \mathbf{A}^{(2)}_{:,:,t'}, \quad \text{and} \quad \mathbf{A}^{(1)}_{:,:,t} = \mathbf{A}^{(2)}_{:,:,t} \quad \forall\, t \neq t'.$$

Under *implicit* dynamic modeling, the function $f(\{\mathbf{A}_{:,:,t}\}_{t=1}^{T})$ may yield the same compressed adjacency $\widehat{\mathbf{A}}$ for both $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$. Hence, any implicit model would treat the two graphs *identically*, failing to separate them because the single $\widehat{\mathbf{A}}$ cannot preserve the difference at t'.

In contrast, the *explicit* dynamic model processes $\{\mathbf{A}^{(k)}_{:,:,t}\}_{t=1}^{T}$ without discarding the time-step-specific variations. Consequently, it *does* see $\mathbf{A}^{(1)}_{:,:,t'} \neq \mathbf{A}^{(2)}_{:,:,t'}$, and can therefore distinguish $\mathcal{G}^{(1)}$ from $\mathcal{G}^{(2)}$. Thus, there exist inputs in the domain of temporal EEG graphs that are *indistinguishable* by any implicit dynamic model but *distinguishable* by an explicit dynamic model. This proves

$$\mathcal{F}_{\text{implicit}} \neq \mathcal{F}_{\text{explicit}},$$

and in view of the subset argument, we conclude

$$\mathcal{F}_{\text{implicit}} \subset \mathcal{F}_{\text{explicit}},$$

explicit dynamic modeling is strictly more expressive than implicit dynamic modeling. □

C.3 *time-and-graph* and *time-then-graph*

Lemma 2. *[time-and-graph $\precneqq$ time-then-graph] time-then-graph is strictly more expressive than time-and-graph representation family on $\mathbb{T}_{n,T,\theta}$, as time-then-graph outputs different representations, while time-and-graph does not.*

Gao and Ribeiro (2022) prove that a *time-then-graph* representation that outputs the same embeddings as an arbitrary *time-and-graph* representation. Thus, *time-then-graph* is as expressive as *time-and-graph*. To prove Lemma 2 we also provide a EEG graph classification task where any *time-and-graph* representation will fail while a *time-then-graph* would work, which then, added to the previous result, proves that *time-then-graph* is strictly more expressive than *time-and-graph*.

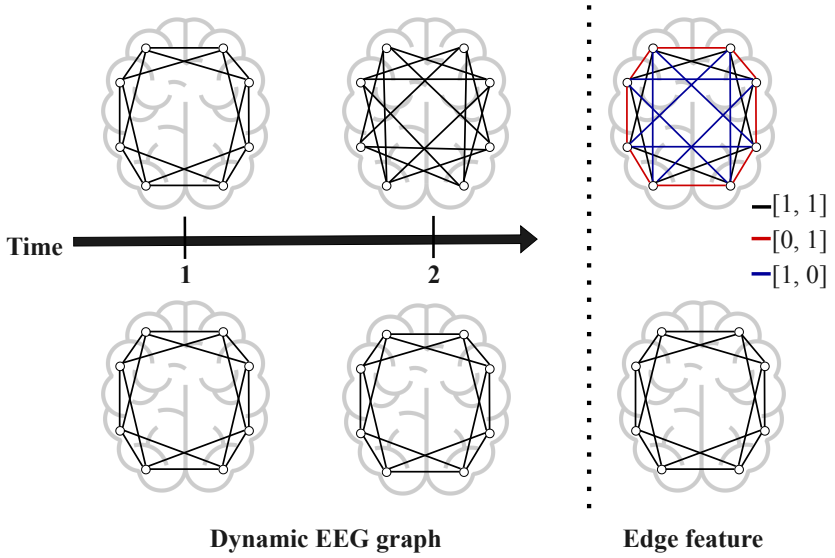


Figure 8: **A synthetic EEG task where only *time-then-graph* is expressive.** The top and bottom 2-time temporal graphs on the left side have snapshots of different structure at time t_2 (denote by $\mathcal{C}_{8,2}$ and $\mathcal{C}_{8,1}$). The top and bottom temporal graphs on the left show different dynamic-graph structures. The right side shows their aggregated versions, where edge attributes indicate whether they existed (1) or not (0) over time, using different colors. The goal is to distinguish the structural differences between the top and bottom graphs. *time-and-graph* has the same node representation neighbors in both snapshots, indistinguishable. *time-then-graph* aggregates the dynamic graphs into different node representations and succeeds in distinguishing them.

Proof. We now propose a synthetic EEG task, whose temporal graph is illustrated in Figure 8. The goal is to differentiate the topologies between two 2-step temporal graphs. Each snapshot is a static EEG graph with 8 attributed nodes, denoted as $\mathcal{C}_{8,1}$ and $\mathcal{C}_{8,2}$.

Two temporal graphs differ in their second time step t_2. If the graphs have the same features, any 1-WL GNN will output the same representations for both $\mathcal{C}_{8,1}$ and $\mathcal{C}_{8,2}$. We use $\mathbf{a}^{(\text{top})}$ to represent the adjacency matrix of dynamics in the top left of Figure 8, and $\mathbf{a}^{(\text{btm})}$ for dynamics in the bottom left of Figure 8. Note that $\mathbf{X}^{(\text{top})} = \mathbf{X}^{(\text{btm})}$ since the temporal graph has the same features.

Hence, for a *time-and-graph* representation:

$$\begin{aligned}
&\text{GNN}_{\text{in}}^{L}\big(\mathbf{X}_{:,1}^{(\text{top})}, \mathbf{A}_{:,:,1}^{(\text{top})}\big) = \text{GNN}_{\text{in}}^{L}\big(\mathbf{X}_{:,2}^{(\text{top})}, \mathbf{A}_{:,:,2}^{(\text{top})}\big) \\
= \text{GNN}_{\text{in}}^{L}\big(\mathbf{X}_{:,1}^{(\text{btm})}, \mathbf{A}_{:,:,1}^{(\text{btm})}\big) &= \text{GNN}_{\text{in}}^{L}\big(\mathbf{X}_{:,2}^{(\text{btm})}, \mathbf{A}_{:,:,2}^{(\text{btm})}\big), \\
\text{GNN}_{\text{rc}}^{L}\big(\mathbf{H}_{:,0}^{(\text{top})}, \mathbf{A}_{:,:,1}^{(\text{top})}\big) &= \text{GNN}_{\text{rc}}^{L}\big(\mathbf{H}_{:,0}^{(\text{btm})}, \mathbf{A}_{:,:,1}^{(\text{btm})}\big), \\
\text{GNN}_{\text{rc}}^{L}\big(\mathbf{H}_{:,1}^{(\text{top})}, \mathbf{A}_{:,:,2}^{(\text{top})}\big) &= \text{GNN}_{\text{rc}}^{L}\big(\mathbf{H}_{:,1}^{(\text{btm})}, \mathbf{A}_{:,:,2}^{(\text{btm})}\big).
\end{aligned}$$

Then, when we apply Equation (2) at the first time step, for the top graph:

$$\mathbf{h}_{\mathbf{i},\mathbf{1}}^{(\text{top})} = \text{Cell}\Big(\Big[\text{GNN}_{\text{in}}^{L}\big(\mathbf{X}_{:,1}^{(\text{top})}, \mathbf{A}_{:,:,1}^{(\text{top})}\big)\Big]_{i}, \Big[\text{GNN}_{\text{rc}}^{L}\big(\mathbf{h}_{:,\mathbf{0}}^{(\text{top})}, \mathbf{A}_{:,:,\mathbf{1}}^{(\text{top})}\big)\Big]_{\mathbf{i}}\Big),$$

for the bottom graph:

$$\mathbf{h}_{\mathbf{i},\mathbf{1}}^{(\text{btm})} = \text{Cell}\Big(\Big[\text{GNN}_{\text{in}}^{L}\big(\mathbf{X}_{:,1}^{(\text{btm})}, \mathbf{A}_{:,:,1}^{(\text{btm})}\big)\Big]_{i}, \Big[\text{GNN}_{\text{rc}}^{L}\big(\mathbf{h}_{:,\mathbf{0}}^{(\text{btm})}, \mathbf{A}_{:,:,\mathbf{1}}^{(\text{btm})}\big)\Big]_{\mathbf{i}}\Big).$$

Since $\mathbf{X}^{(\text{top})} = \mathbf{X}^{(\text{btm})}$, $\mathbf{A}_{:,:,1}^{(\text{top})} = \mathbf{A}_{:,:,1}^{(\text{btm})}$, and $\mathbf{h}_{:,\mathbf{0}}^{(\text{top})} = \mathbf{h}_{:,\mathbf{0}}^{(\text{btm})}$,
we have: $\mathbf{h}_{\mathbf{i},\mathbf{1}}^{(\text{top})} = \mathbf{h}_{\mathbf{i},\mathbf{1}}^{(\text{btm})}$.

For the second time step:

$$\mathbf{h}_{\mathbf{i},\mathbf{2}}^{(\text{top})} = \text{Cell}\Big(\Big[\text{GNN}_{\text{in}}^{L}\big(\mathbf{X}_{:,2}^{(\text{top})}, \mathbf{A}_{:,:,2}^{(\text{top})}\big)\Big]_{i}, \Big[\text{GNN}_{\text{rc}}^{L}\big(\mathbf{h}_{:,\mathbf{1}}^{(\text{top})}, \mathbf{A}_{:,:,\mathbf{2}}^{(\text{top})}\big)\Big]_{\mathbf{i}}\Big),$$
$$\mathbf{h}_{\mathbf{i},\mathbf{2}}^{(\text{btm})} = \text{Cell}\Big(\Big[\text{GNN}_{\text{in}}^{L}\big(\mathbf{X}_{:,2}^{(\text{btm})}, \mathbf{A}_{:,:,2}^{(\text{btm})}\big)\Big]_{i}, \Big[\text{GNN}_{\text{rc}}^{L}\big(\mathbf{h}_{:,\mathbf{1}}^{(\text{btm})}, \mathbf{A}_{:,:,\mathbf{2}}^{(\text{btm})}\big)\Big]_{\mathbf{i}}\Big).$$

Despite $\mathbf{A}_{:,:,2}^{(\text{top})} \neq \mathbf{A}_{:,:,2}^{(\text{btm})}$, the 1-WL GNN will output the same representations $\mathbf{h}_{\mathbf{i},\mathbf{2}}^{(\text{top})} = \mathbf{h}_{\mathbf{i},\mathbf{2}}^{(\text{btm})}$ for both $\mathcal{C}_{8,1}$ and $\mathcal{C}_{8,2}$.

Therefore:

$$\mathbf{Z}^{(\text{top})} = \mathbf{h}_{\mathbf{i},\mathbf{2}}^{(\text{top})} = \mathbf{h}_{\mathbf{i},\mathbf{2}}^{(\text{btm})} = \mathbf{Z}^{(\text{btm})}.$$

Thus, time-and-graph will output the same final representation $\mathbf{Z}^{(\text{top})} = \mathbf{Z}^{(\text{btm})}$ for two different temporal graphs in Figure 8.

For the time-then-graph representation, we apply Equation (3). First, for the node representations:

$$\mathbf{h}_{\mathbf{i}}^{\text{node(top)}} = \text{RNN}^{\text{node}}\big(\mathbf{X}_{i,\leq 2}^{(\text{top})}\big) = \text{RNN}^{\text{node}}\big([\mathbf{x}_{\mathbf{i},\mathbf{1}}^{(\text{top})}, \mathbf{x}_{\mathbf{i},\mathbf{2}}^{(\text{top})}]\big),$$
$$\mathbf{h}_{\mathbf{i}}^{\text{node(btm)}} = \text{RNN}^{\text{node}}\big(\mathbf{X}_{i,\leq 2}^{(\text{btm})}\big) = \text{RNN}^{\text{node}}\big([\mathbf{x}_{\mathbf{i},\mathbf{1}}^{(\text{btm})}, \mathbf{x}_{\mathbf{i},\mathbf{2}}^{(\text{btm})}]\big).$$

Since $\mathbf{X}^{(\text{top})} = \mathbf{X}^{(\text{btm})}$, we have $\mathbf{h}_{\mathbf{i}}^{\text{node(top)}} = \mathbf{h}_{\mathbf{i}}^{\text{node(btm)}}$ for all nodes i.

Now, for the edge representations:

$$\mathbf{h}_{\mathbf{i},\mathbf{j}}^{\text{edge(top)}} = \text{RNN}^{\text{edge}}\big(\mathbf{A}_{i,j,\leq 2}^{(\text{top})}\big) = \text{RNN}^{\text{edge}}\big([\mathbf{a}_{\mathbf{i},\mathbf{j},\mathbf{1}}^{(\text{top})}, \mathbf{a}_{\mathbf{i},\mathbf{j},\mathbf{2}}^{(\text{top})}]\big),$$
$$\mathbf{h}_{\mathbf{i},\mathbf{j}}^{\text{edge(btm)}} = \text{RNN}^{\text{edge}}\big(\mathbf{A}_{i,j,\leq 2}^{(\text{btm})}\big) = \text{RNN}^{\text{edge}}\big([\mathbf{a}_{\mathbf{i},\mathbf{j},\mathbf{1}}^{(\text{btm})}, \mathbf{a}_{\mathbf{i},\mathbf{j},\mathbf{2}}^{(\text{btm})}]\big).$$

Here, $\mathbf{a}_{\mathbf{i},\mathbf{j},\leq\mathbf{2}}^{(\text{top})} \neq \mathbf{a}_{\mathbf{i},\mathbf{j},\leq\mathbf{2}}^{(\text{btm})}$ for some (i, j) pairs, because the graph structures differ at t_2. Therefore, $\mathbf{h}_{\mathbf{i},\mathbf{j}}^{\text{edge(top)}} \neq \mathbf{h}_{\mathbf{i},\mathbf{j}}^{\text{edge(btm)}}$ for these pairs. Finally, we apply the GNN:

$$\mathbf{Z}^{(\text{top})} = \text{GNN}^{L}\big(\mathbf{H}^{\text{node(top)}}, \mathbf{H}^{\text{edge(top)}}\big),$$
$$\mathbf{Z}^{(\text{btm})} = \text{GNN}^{L}\big(\mathbf{H}^{\text{node(btm)}}, \mathbf{H}^{\text{edge(btm)}}\big).$$

Since $\mathbf{h}^{\text{edge(top)}} \neq \mathbf{h}^{\text{edge(btm)}}$, and 1-WL GNNs can distinguish graphs with different edge attributes, we have:

$$\mathbf{Z}^{(\text{top})} \neq \mathbf{Z}^{(\text{btm})}. \tag{6}$$

Thus, time-then-graph outputs different final representations $\mathbf{Z}^{(\text{top})} \neq \mathbf{Z}^{(\text{btm})}$ for the two temporal graphs in Figure 8, successfully distinguishing them.

Finally, we conclude:

1. The *time-then-graph* is at least as expressive as the *time-and-graph*;
2. The *time-then-graph* can distinguish temporal graphs not distinguishable by *time-and-graph*.

Thus, *time-then-graph* is strictly more expressive than *time-and-graph*. More precisely,

$$\textit{time-and-graph} \precneqq_{\mathbb{T}_{n,T,\theta}} \textit{time-then-graph},$$

concluding our proof.

□

D Computational Complexity

In this section, we provide the computational complexities of the three architectures: *graph-then-time*, *time-and-graph*, and *time-then-graph*. We demonstrate that the *time-then-graph* architecture has the lowest computational complexity among them.

Let T be the number of time steps, V be the number of nodes, E_t be the number of edges at time t, $\sum_t E_t$ be the total number of edges across all time steps, E_{agg} be the number of edges in the aggregated graph (i.e., the union of all edges across time steps), and d be the dimension of the node and edge representations.

D.1 *graph-then-time* Architecture

In the *graph-then-time* architecture, at each time step t, a GNN is applied to the snapshot graph $(\mathbf{X}_{:,t}, \mathbf{a}_{:,:,\mathbf{t}})$ to capture spatial relationships. Subsequently, an RNN processes the node embeddings over time to capture temporal dependencies.

The computational complexity per time step t is dominated by:

$$\mathcal{O}\left(Vd^2 + E_t d\right),$$

where Vd^2 accounts for node-wise transformations (e.g., linear layers), and $E_t d$ accounts for message passing over edges.

Over all time steps, the total complexity for the GNN computations is:

$$\mathcal{O}\left(TVd^2 + \sum_{t=1}^{T} E_t d\right).$$

The RNN processes the node embeddings over time with complexity:

$$\mathcal{O}\left(VTd^2\right).$$

Therefore, the overall computational complexity of the *graph-then-time* architecture is:

$$\mathcal{O}\left(VTd^2 + \sum_{t=1}^{T} E_t d\right). \tag{7}$$

D.2 *time-and-graph* Architecture

In the *time-and-graph* architecture, temporal dependencies are integrated into the GNN computations. At each time step t, two GNNs are applied:

- $\text{GNN}_{\text{in}}^{L}$ processes the current snapshot inputs $(\mathbf{X}_{:,t}, \mathbf{a}_{:,:,\mathbf{t}})$.
- $\text{GNN}_{\text{rc}}^{L}$ processes the representations from the previous time step $(\mathbf{h}_{:,\mathbf{t-1}}, \mathbf{a}_{:,:,\mathbf{t}})$.

The computational complexity per time step t is:

$$\mathcal{O}\left(Vd^2 + E_t d^2\right),$$

due to the node-wise transformations and edge-wise message passing with updated representations.

Over all time steps, the total complexity for the GNN computations is:

$$\mathcal{O}\left(TVd^2 + \sum_{t=1}^{T} E_t d^2\right).$$

The RNN (or any recurrent unit) further processes the node embeddings with complexity:

$$\mathcal{O}\left(VTd^2\right).$$

Therefore, the overall computational complexity of the *time-and-graph* architecture is:

$$\mathcal{O}\left(VTd^2 + \sum_{t=1}^{T} E_t d^2\right). \tag{8}$$

D.3 *time-then-graph* Architecture

In the *time-then-graph* architecture, temporal evolutions of node and edge attributes are modeled first using sequence models (e.g., RNNs). A GNN is then applied to the resulting static graph with aggregated temporal information.

The computational complexities are as follows:

Node Sequence Modeling. For each node $i \in \mathcal{V}$, an RNN processes its temporal features $\mathbf{X}_{i,\leq T}$:

$$\mathcal{O}\left(VTd^2\right).$$

Edge Sequence Modeling. For each edge $(i, j) \in \mathcal{E}_{\text{agg}}$, an RNN processes its temporal adjacency features $\mathbf{a}_{\mathbf{i},\mathbf{j},\leq \mathbf{T}}$:

$$\mathcal{O}\left(E_{\text{agg}} T d^2\right).$$

GNN over Aggregated Graph. A GNN is applied once to the static graph with updated node and edge representations:

$$\mathcal{O}\left(Vd^2 + E_{\text{agg}} d^2\right).$$

Therefore, the overall computational complexity of the *time-then-graph* architecture is:

$$\mathcal{O}\left(\left(V + E_{\text{agg}}\right) T d^2\right). \tag{9}$$

D.4 Comparison of Complexities

To compare the computational complexities, we consider the dominant terms in Equations (7), (8), and (9).

- **graph-then-time**:

$$\mathcal{O}\left(VTd^2 + \sum_{t=1}^{T} E_t d\right).$$

- **time-and-graph**:

$$\mathcal{O}\left(VTd^2 + \sum_{t=1}^{T} E_t d^2\right).$$

- **time-then-graph**:

$$\mathcal{O}\left(\left(V + E_{\text{agg}}\right) T d^2\right).$$

By comparing these computational complexities, the **time-then-graph** method is superior under the aggregated number of edges E_{agg} is smaller than the total sum of edges over all time steps, i.e., $E_{\text{agg}} \ll \sum_{t=1}^{T} E_t$.

E Experimental details and implementation

Model training. Training for all models was accomplished using the Adam optimizer (Kingma and Ba, 2014) in PyTorch on NVIDIA A6000 GPU and Xeon Gold 6258R CPU.

Data augmentation. During the training process, we applied the following data augmentation techniques, following prior studies (Tang et al., 2022; Eldele et al., 2021): randomly scaling the amplitude of the raw EEG signals by a factor between 0.8 and 1.2.

Implementation details. We used binary cross-entropy as the loss function to train all models. The models were trained for 100 epochs with an initial learning rate of 1e-4. To enhance efficiency and sparsity, we set $\tau = 3$ and the top-3 neighbors' edges were kept for each node. The dropout probability was 0 (i.e., no dropout). `EvoBrain` has two Mamba consisting of two stacked layers and a two-layer GCN with 64 hidden units, resulting in 114,794 trainable parameters. We release GitHub repository (`https://github.com/Kotoge/EvoBrain`).

Implementation of baselines. For baselines, DCRNN (Tang et al., 2022), EvolveGCN (Pareja et al., 2020), and LSTM (Hochreiter and Schmidhuber, 1997), we used the number of RNN and GNN layers and hidden units in our `EvoBrain`. For BIOT, we use the same model architecture described in Yang et al. (2023a), i.e., four Transformer layers with eight attention heads and 256-dimensional embedding. For LaBraM, we used the official checkpoint provided by Jiang et al. (2024). For CNN-LSTM, we use the same model architecture described in Ahmedt-Aristizabal et al. (2020), i.e., two stacked convolutional layers (32 3×3 kernels), one max-pooling layer (2×2), one fully-connected layer (output neuron = 512), two stacked LSTM layers (hidden size = 128), and one fully connected layer. Table 2 shows a comparison of trainable parameters, with our `EvoBrain` achieving the best performance using the relatively few parameters.

Table 2: Comparison of trainable parameters.

	`EvoBrain`	GRU-GCN	DCRNN	EvolveGCN	EEGPT	LaBraM	BIOT	CNN-LSTM	LSTM
Trainable Parameters	183,834	114,794	280,769	200,301	51,221,121	5,803,137	3,187,201	5,976,033	536,641

F Data description

Tabel 3 shows the details of TUSZ dataset.

Table 3: Number of EEG data samples and patients in the train, validation, and test sets on TUSZ dataset. Train, validation, and test sets consist of distinct patients.

Task	EEG Input Length (Secs)	Train Set		Validation Set		Test Set	
		EEG samples % (Pre-) Seizure	Patients	EEG samples % (Pre-) Seizure	Patients	EEG samples % (Pre-) Seizure	Patients
Seizure Detection	60-s	38,613 (9.3%)	530	5,503 (11.4%)	61	8,848 (14.7%)	45
	12-s	196,646 (6.9%)	531	28,057 (8.7%)	61	44,959 (10.9%)	45
Seizure Prediction	60-s	7,550 (9.9%)	530	999 (12.0%)	61	1,277 (24.4%)	45
	12-s	40,716 (12.8%)	531	5,439 (16.0%)	61	6,956 (27.6%)	45

G Memory Efficiency

Figure 4 shows memory efficiency comparisons across dynamic GNNs in both training and inference settings with a batch size of 1. While DCRNN and EvolveGCN exhibit the lowest memory usage, EvoBrain achieves over 10× faster computation compared to these baselines, making it a compelling choice for time-critical applications. Among the three time-then-graph models, EvoBrain demonstrates the best balance between memory consumption and computational speed, with significantly lower memory requirements than GRAPHS4MER and comparable usage to GRU-GCN.

Table 4: GPU memory consumption during training and inference. EvoBrain offers competitive memory efficiency compared to baselines.

Model	**Training (MB)**	**Inference (MB)**
EvoBrain	51.35	46.64
GRU-GCN	54.61	52.09
GRAPHS4MER	369.46	93.02
DCRNN	21.10	20.54
EvolveGCN	22.06	20.07