Deep Representations for Time-varying Brain Datasets

Sikun Lin
University of California, Santa Barbara
Santa Barbara, CA, USA
sikun@ucsb.edu

Shuyun Tang
University of California, Santa Barbara
Santa Barbara, CA, USA
shuyun@ucsb.edu

Scott T. Grafton
University of California, Santa Barbara
Santa Barbara, CA, USA
scott.grafton@psych.ucsb.edu

Ambuj K. Singh
University of California, Santa Barbara
Santa Barbara, CA, USA
ambuj@cs.ucsb.edu

ABSTRACT

Finding an appropriate representation of dynamic activities in the brain is crucial for many downstream applications. Due to its highly dynamic nature, temporally averaged fMRI (functional magnetic resonance imaging) can only provide a narrow view of underlying brain activities. Previous works lack the ability to learn and interpret the latent dynamics in brain architectures. This paper builds an efficient graph neural network model that incorporates both region-mapped fMRI sequences and structural connectivities obtained from DWI (diffusion-weighted imaging) as inputs. We find good representations of the latent brain dynamics through learning sample-level adaptive adjacency matrices and performing a novel multi-resolution inner cluster smoothing. We also attribute inputs with integrated gradients, which enables us to infer (1) highly involved brain connections and subnetworks for each task, (2) temporal keyframes of imaging sequences that characterize tasks, and (3) subnetworks that discriminate between individual subjects. This ability to identify critical subnetworks that characterize signal states across heterogeneous tasks and individuals is of great importance to neuroscience and other scientific domains. Extensive experiments and ablation studies demonstrate our proposed method's superiority and efficiency in spatial-temporal graph signal modeling with insightful interpretations of brain dynamics.

CCS CONCEPTS

• **Computing methodologies** → **Learning latent representations**; • **Applied computing** → **Imaging**.

KEYWORDS

fMRI time series, graph neural networks, feature attribution

ACM Reference Format:
Sikun Lin, Shuyun Tang, Scott T. Grafton, and Ambuj K. Singh. 2022. Deep Representations for Time-varying Brain Datasets. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539301

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-1-4503-9385-0/22/08.
https://doi.org/10.1145/3534678.3539301

1 INTRODUCTION

Neuroimaging techniques such as fMRI (functional magnetic resonance imaging) and DWI (diffusion-weighted imaging) provide a window into complex brain processes. Yet, modeling and understanding these signals has always been a challenge. Network neuroscience [1] views the brain as a multiscale networked system and models these signals in their graph representations: nodes represent brain ROIs (regions of interest), and edges represent either structural or functional connections between pairs of regions.

With larger imaging datasets and developments in graph neural networks, recent works leverage variants of graph deep learning, modeling brain signals with data-driven models and getting rid of Gaussian assumptions that typically existed in linear models [15, 38]. These methods are making progress on identifying physiological characteristics and brain disorders: In [9], authors combine grad-CAM [23] and GIN [35] to highlight brain regions that are responsible for gender classification with resting-state fMRI data. Others [16] propose to use regularized pooling with GNN to identify fMRI biomarkers. However, these works use time-averaged fMRI, losing rich dynamics in the temporal domain. They also do not incorporate structural modality that can provide additional connectivity information missing in the functional modality. Another work [18] embeds both topological structures and node signals of fMRI networks into a low-dimensional latent representations for better identification of depression, but it combines nodes' temporal and feature dimensions instead of handling them separately, leading to a suboptimal representation (as discussed in section 3.2). To overcome these issues, we propose ReBraiD (Deep **Re**presentations for Time-varying **Brai**n **D**atasets), a graph neural network model that jointly models dynamic functional signals and structural connectivities, leading to a more comprehensive deep representation of brain dynamics.

To simultaneously encode signals along spatial and temporal dimensions, some works in traffic prediction and activity recognition domains such as Graph WaveNet [34] alternate between TCN (temporal convolution network) [13] and GCN (graph convolutional network) [11]. Others [17, 26] use localized spatial-temporal graph to embed both domains' information in this extended graph. Some proposed methods also incorporate gated recurrent networks for the temporal domain such as [21, 24]. We choose to alternate TCN with GCN layers for ReBraiD, as it is more memory and time-efficient and can support much longer inputs. On top of this design,

we propose novel "sample-level adaptive adjacency matrix learning" and "multi-resolution inner cluster smoothing," both of which learn and refine latent dynamic structures. With the choice of the temporal layer, our model is more efficient than other baselines while having the highest performance.

We perform extensive ablation studies to examine individual components of the model. We also explore the best option when alternating spatial and temporal layers for encoding brain activities. After quantitatively showing the representation ability of our model, we utilize IG (integrated gradients) [27] to identify how brain ROIs participate in various processes. This can lead to better behavioral understanding, discovery of biomarkers, and characterization of individuals or groups. We also make the novel contribution of identifying temporally important frames with graph attribution techniques; this can enable more fine-grained temporal analysis around keyframes when combined with other imaging modalities such as EEG (electroencephalogram). In addition, our subject-level and group-level attribution studies unveil heterogeneities among ROIs, tasks, and individuals.

In summary, the main contributions of our work are as follows:

- We present ReBraid, an efficient graph neural network model that jointly models both structural and dynamic functional brain signals, providing a more comprehensive representation of brain activities when compared to the current fMRI literature.
- Unlike typical spatial-temporal GCNs that learn a universal latent structure, we propose sample-level latent adaptive adjacency matrix learning based on input snippets. This captures the evolving dynamics of a task better.
- We propose multi-resolution inner cluster smoothing, which effectively encodes long-range node relationships while keeping the graph structure, enabling the model to leverage structural and latent adjacency matrices throughout the process. Together with subject SC and sample-level adjacency matrix learning, the inner cluster smoothing learns and refines latent dynamic structures on limited signal data.
- We carry out extensive ablation studies and model comparisons to show ReBraid's superiority in representing brain dynamics. We also leverage integrated gradients to attribute and interpret the importance of both spatial brain ROIs and temporal keyframes, as well as heterogeneities among brain ROIs, tasks, and subjects. These can open up new opportunities for identifying biomarkers for different tasks or diseases and markers for other complex scientific phenomena.

2 METHOD

2.1 Preliminaries

We utilize two brain imaging modalities mapped onto a same coordinate: SC (structural connectivity) from DWI scans, and time-varying fMRI scans. We represent them as a set of L graphs $\mathcal{G}_i = (A_i, X_i)$ with $i \in [1, L]$. $A_i \in \mathbb{R}^{N\times N}$ represents normalized adjacency matrix with an added self-loop: $A_i = \tilde{D}_{\mathrm{SC}_i}^{-\frac{1}{2}} \tilde{\mathrm{SC}}_i \tilde{D}_{\mathrm{SC}_i}^{-\frac{1}{2}}$, $\tilde{\mathrm{SC}}_i = \mathrm{SC}_i + I_N$ and $\tilde{D}_{\mathrm{SC}_i} = \sum_w (\tilde{\mathrm{SC}}_i)_{vw}$ is the diagonal node degree matrix. Graph signal matrix obtained from fMRI scans of the i^{th} sample is represented as $X_i \in \mathbb{R}^{N\times T}$. Here N is the number of nodes, and each node represents a brain region; T is the input signal length on each node. We refine our representation using the task of classifying brain signals $\mathcal{G}_i$ into one of C task classes through learning latent graph structures.

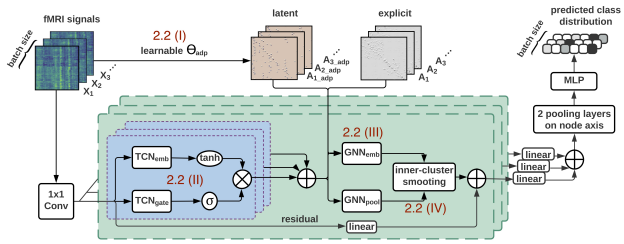


Figure 1: The proposed ReBraiD model for integrating brain structure and dynamics (the architecture shown is for classification). For each batch with batch size B, input X has a dimension of $(B, 1, N, T)$[1], and $A, A_{\mathbf{adp}}$ both have the dimension (B, N, N). The encoder (green part) encodes temporal and spatial information alternately, producing a latent representation in $(B, d_{\mathbf{latent}}, N, 1)$. These embeddings are followed by linear layers for pooling and classification. The final output has a dimension of (B, C).

2.2 Model

ReBraiD takes (A, X) as inputs and outputs task class predictions. The overall model structure is shown in fig. 1. For the i^{th} sample $X_i \in \mathbb{R}^{N\times 1\times T}$, the initial 1×1 convolution layer increases its hidden feature dimension to d_{h1}, outputting (N, d_{h1}, T). The encoder then encodes temporal and spatial information alternately, and generates a hidden representation of size $(N, d_{h2}, 1)$. The encoder is followed by two linear layers to perform pooling on node embeddings and two MLP layers for classification. Cross entropy is used as the loss function: $L_{CE} = -\sum_i y_i \log \hat{y}_i$, where $y_i \in \mathbb{R}^C$ is the one-hot vector of ground truth task labels and $\hat{y}_i \in \mathbb{R}^C$ is the model's predicted distribution. We now explain the different components of the model.

(I) Learning sample-level latent graph structures. Structural scans serve as our graph adjacency matrices. However, they remain fixed across temporal frames and across tasks. In contrast, FC (functional connectivities) are highly dynamic, resulting in different connection patterns across both time and tasks. To better capture dynamic graph structures, we learn an adaptive adjacency matrix from each input graph signal. Unlike other works such as [34] that use a universal latent graph structure, our model does not assume that all samples share the same latent graph. Instead, our goal is to give each sample a unique latent structure that can reflect its own signaling pattern. This implies that the latent adjacency matrix cannot be directly treated as a learnable parameter as a part of the model. To solve this, we minimize the assumption down to a shared projection Θ_{adp} that projects each input sequence into an embedding space and use this embedding to generate the latent graph structure. Projection Θ_{adp} can be learned in an end-to-end manner. The generated adaptive adjacency matrix for the i^{th} sample can be written as follows (Softmax is applied column-wise):

$$A_{i\_\mathrm{adp}} = \mathrm{Softmax}\left(\mathrm{ReLU}\left(\left(X_i\Theta_{\mathrm{adp}}\right)\left(X_i\Theta_{\mathrm{adp}}\right)^\top\right)\right), \Theta_{\mathrm{adp}} \in \mathbb{R}^{T\times h_{\mathrm{adp}}} \quad (1)$$

[1] Axis order follows PyTorch conventions. Dimension at the second index is the expanded feature dimension.

(II) Gated TCN (Temporal Convolutional Network). To encode signal dynamics, we use the gating mechanism as in [19] in our temporal layers:

$$H^{(l+1)} = \tanh\left(\text{TCN}_{\text{emb}}(H^{(l)})\right) \odot \sigma\left(\text{TCN}_{\text{gate}}(H^{(l)})\right), \tag{2}$$

where $H^{(l)} \in \mathbb{R}^{N\times d\times t}$ is one sample's activation matrix of the l^{th} layer, $\odot$ denotes the Hadamard product, and σ is the Sigmoid function. In contrast to TCNs that are generally used in sequence to sequence models that consist of dilated Conv1d and causal padding along the temporal dimension ([31]), we simply apply Conv1d with kernel = 2 and stride = 2 as our TCN_{emb} and TCN_{gate} to embed temporal information. The reason is twofold: first, for a sequence to sequence model with a length-T output, y_τ should only depend on $x_{t\le\tau}$ to avoid information leakage and causal convolution can ensure this. In contrast, our model's task is classification, and the goal of our encoder along the temporal dimension is to embed signal information into the feature axis while reducing the temporal dimension to 1. The receptive field of this single temporal point (with multiple feature channels) is meant to be the entire input sequence. Essentially, our TCN is the same as the last output node of a *kernel-two causal TCN* whose dilation increases by two at each layer (fig. 8). Second, from a practical perspective, directly using strided non-causal TCN works the same as taking the last node of dilated causal TCNs, as discussed above, while simplifying the model structure and reducing training time to less than a quarter.

(III) Graph Network layer. In our model, every set of l temporal layers (appendix B.1 studies the best l to choose) is followed by a spatial layer to encode signals with the graph structure. Building temporal and spatial layers alternately helps spatial modules to learn embeddings at different temporal scales, and this generates better results than placing spatial layers after all the temporal ones.

To encode spatial information, [11] uses first-order approximation of spectral filters to form the layer-wise propagation rule of a GCN layer: $H^{(l+1)} = \text{GCN}(H^{(l)}) = f(AH^{(l)}W^{(l)})$. It can be understood as spatially aggregating information among neighboring nodes to form new node embeddings. In the original setting without temporal signals, $H^{(l)} \in \mathbb{R}^{N\times d}$ is the activation matrix of l^{th} layer, $A \in \mathbb{R}^{N\times N}$ denotes the normalized adjacency matrix with self-connections as discussed in section 2.1, $W^{(l)} \in \mathbb{R}^{d\times d'}$ is learnable model parameters, and f is a nonlinear activation function of choice. Parameters d and d' are the number of feature channels.

We view a GCN layer as a local smoothing operation followed by an MLP, and simplify stacking K layers to $A^K H$ as in [33]. In ReBraiD, every graph network layer aggregates information from each node's K-hop neighborhoods based on both brain structural connectivity and the latent adaptive adjacency matrix: thus, we have both ${A_i}^K H^{(l)} W_K$ and ${A_{i\_\text{adp}}}^K H^{(l)} W_{K\_\text{adp}}$ for input $H^{(l)}$. We also gather different levels (from 0 to K) of neighbor information with concatenation. In other words, one graph convolution layer here corresponds to a small module that is equivalent to K simple GCN layers with residual connections. We can write our layer as:

$$\begin{aligned} H^{(l+1)} &= \text{GNN}^{(l)}\left(H^{(l)}\right) \\ &= \text{MLP}\left[\text{Concat}_{k=1}^{K}\left(H^{(l)}, \text{ReLU}({A_i}^k H^{(l)}), \text{ReLU}({A_{i\_\text{adp}}}^k H^{(l)})\right)\right] \end{aligned} \tag{3}$$

Note that in eq. (3), $A_i \in \mathbb{R}^{N\times N}$ and $H^{(l)} \in \mathbb{R}^{N\times d\times t}$, and as a result their product $\in \mathbb{R}^{N\times d\times t}$. Outputs of different $\text{GNN}^{(l)}$ layers are parameterized and then skip connected with a summation. Since the temporal lengths of these outputs are different because of TCNs, max-pooling is used before each summation to make the lengths identical.

(IV) Multi-resolution inner cluster smoothing. While GNN layers can effectively pass information between neighboring nodes, long-range relationships among brain regions that neither appear in SC nor learned by latent A_{adp} can be better captured using soft assignments, similar to DiffPool[36]. To generate the soft assignment tensor $S^{(l)}$ that assigns N nodes into c clusters (c chosen manually), we use $\text{GNN}_{pool}^{(l)}$ that obeys the same propagation rule as in eq. (3), followed by Softmax along c. This assignment is applied to $Z^{(l)}$, the output of $\text{GNN}_{emb}^{(l)}$ which carries out the spatial embedding for the l^{th} layer input $H^{(l)}$, producing clustered representation $\tilde{H}^{(l)}$:

$$\begin{aligned} S^{(l)} &= \text{Softmax}\left(\text{GNN}_{pool}^{(l)}\left(H^{(\ell)}\right), 1\right) \in \mathbb{R}^{N\times c\times t} \\ Z^{(l)} &= \text{GNN}_{emb}^{(l)}\left(H^{(l)}\right) \in \mathbb{R}^{N\times d\times t} \\ \tilde{H}^{(l)} &= {S^{(l)}}^\top Z^{(l)} \in \mathbb{R}^{c\times d\times t} \end{aligned} \tag{4}$$

The additional temporal dimension allows nodes to be assigned to heterogeneous clusters at different frames. We find that using coarsened $A_i^{(l+1)} = {S^{(l)}}^\top A_i^{(l)} S^{(l)} \in \mathbb{R}^{c\times c}$ as the graph adjacency matrix leads to worse performance compared to using SC-generated A_i and learned A_{i_adp} (comparison in section 3.1). In addition, if the number of nodes is changed, residual connections coming from the beginning of temporal-spatial blocks can not be used, impacting the overall performance. To continue using A_i and A_{i_adp} as graph adjacency matrices and to allow residual connections, we reverse-assign $\tilde{H}^{(l)}$ with assignment tensor obtained from applying Softmax on ${S^{(l)}}^\top$ along N, so that the number of nodes is kept unchanged:

$$\begin{aligned} \tilde{S}^{(l)} &= \text{Softmax}\left({S^{(l)}}^\top, 1\right) \in \mathbb{R}^{c\times N\times t} \\ H^{(\ell+1)} &= {\tilde{S}^{(l)}}^\top \tilde{H}^{(l)} \in \mathbb{R}^{N\times d\times t} \end{aligned} \tag{5}$$

In fact, eqs. (4) and (5) perform signal smoothing on nodes within each soft-assigned cluster. With the bottleneck $c < N$, the model is forced to pick up latent community structures. This inner cluster smoothing is carried out at multiple spatial resolutions: as the spatial receptive field increases with more graph layers, we decrease cluster number c for the assignment operation. As these GNN layers alternate with TCN layers, the inner cluster smoothing also learns the community information across multiple temporal scales.

2.3 Attribution with IG (Integrated Gradients)

As one approach to model interpretability, *attribution* assigns credits to each part of the input, assessing how important they are to the final predictions. [32] gives an extensive comparison between different graph attribution approaches, in which IG [27] is top-performing and can be applied to trained models without any alterations of the model structure. IG also has other desirable properties, such as implementation invariance that other gradient methods lack. It is also more rigorous and accurate than obtaining explanations from attention weights or pooling matrices that span multiple feature channels. Intuitively, IG calculates how real inputs contribute differently compared to a selected baseline; it does so by aggregating model gradients at linearly interpolated inputs between the real and baseline inputs.

In order to apply IG, we calculate attributions at each point of both input $A \in \mathbb{R}^{N\times N}$ and $X \in \mathbb{R}^{N\times T}$ for each sample:

$$\text{Attr}_{\mathcal{G}_{vw}} = (\mathcal{G}_{vw} - \mathcal{G}'_{vw}) \times \sum_{m=1}^{M} \frac{\partial F\left(\mathcal{G}_{\text{Intrpl}}\right)}{\partial \mathcal{G}_{\text{Intrpl}_{vw}}} \times \frac{1}{M}, \qquad (6)$$
$$\mathcal{G} = (A, X), \quad \mathcal{G}_{\text{Intrpl}} = \mathcal{G}' + \frac{m}{M} \times (\mathcal{G} - \mathcal{G}')$$

$F(\mathcal{G})$ here represents our signal classification model, M is the step number when making Riemann approximation of the path integral, and $\mathcal{G}'$ is the baselines of $\mathcal{G}$ (see section 3.3 for more details). Note that eq. (6) calculates the attribution of one edge or one node on one sample. The process is repeated for every input point, so attributions $\text{Attr}_A, \text{Attr}_X$ have identical dimensions as inputs A, X. To obtain the brain region importance of a task, we aggregate attributions across multiple samples of that task.

3 EXPERIMENTS

We use fMRI signals from the CRASH dataset [12] for our experiments. The model classifies input fMRI into six tasks: resting state, VWM (visual working memory task), DYN (dynamic attention task), MOD (math task), DOT (dot-probe task), and PVT (psychomotor vigilance task). We preprocess 4D voxel-level fMRI images into graph signals $\mathcal{G} = (A, X)$ by averaging voxel activities into regional signals with the 200-ROI cortical parcellation (voxel to region mapping) specified by [22]. We also standardize signals for each region and discard scan sessions with obvious abnormal spikes that may be caused by head movement, etc. DWI scans are mapped into the same MNI152 coordinate and processed into adjacency matrices with the same parcellation as fMRI. Our processed data contains 1940 scan sessions from 56 subjects. Session length varies from 265 frames to 828 frames (see table 1 for details). TR (Repetition Time) is 0.91s.

The 1940 scan sessions from CRASH are separated into training, validation, and test sets with a ratio of 0.7-0.15-0.15 (subject-wise split does not lead to any noticeable difference). Each split receives a proportional number of samples for each class. Hyperparameters including dropout rate, learning rate, and weight decay are selected using grid search based on validation loss. All results reported in this section are obtained from the test set. For each scan session, we use a stride-10 sliding window to generate input sequences (in the following experiments $T \in \{8, 16, 32, 64, 128, 256\}$) and feed them to the model. To encode temporal and spatial information alternately, we find stacking two TCN layers per one GNN layer leads to better performance most times (see appendix B.1 (I)). We tested $h_{\text{adp}} = 2, 5, 10$ in eq. (1) for our experiments, and 5 appears to be the best; so we use this value for all the following experiments. $K = 1, 2, 3$ in eq. (3) were tested on a few settings, and K = 2, 3 have a similar performance, both outperforming K = 1. Since smaller values of K have smaller computation needs, we use $K = 2$ for all experiment settings, meaning each GNN layer aggregates information from 2-hop neighbors based on the provided adjacency matrices. We evaluate our model with weighted F1 as the metric in order to account for the imbalance in the number of samples in each task. Our models are written in PyTorch, trained with Google Colab GPU runtimes, and 30 epochs are run for each experiment setting. Code is publicly available [2].

[2] https://github.com/sklin93/ReBraiD

Table 1: fMRI scan details for six tasks.

Tasks	Rest	VWM	DYN	DOT	MOD	PVT	(Total)
Valid sessions	209	514	767	155	138	157	1940
Frames / Scan	321	300	265	798	828	680	—

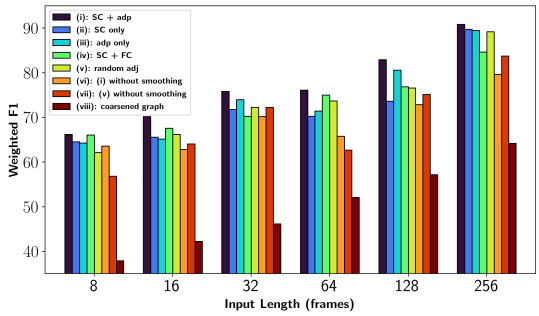


Figure 2: Ablation studies on different input length (please see table 3 in appendix for numerical values of weighted F1 under each setting).

3.1 Model components

Ablation studies on graph adjacency matrices. For each input sample $\mathcal{G}_i$, we test different options to provide graph adjacency matrices to the GNN layer. They include (i) our proposed method: using both adaptive adjacency matrix A_{i_adp} and SC-induced A_i, (ii) only using A_i, (iii) only using A_{i_adp}, (iv) replacing A_{i_adp} in setting i with A_{i_FC} derived from functional connectivity, and (v) only using random graph adjacency matrices with the same level of sparsity as real A's. The results under different settings are reported in fig. 2 (and table 3 in appendix for numerical values).

From the results of setting (ii) plotted in fig. 2, we see that removing the adaptive adjacency matrix impacts the performance differently at different input lengths: the gap peaks for signals of length 64–128, and becomes smaller for either shorter or longer sequences. This could suggest the existence of more distinct latent states of brain signals of this length that structural connectivities cannot capture. On the other hand, removing SC (setting (iii)) seems to have a more constant impact on the model performance, with shorter inputs more likely to see a slightly larger drop. In general, only using A_{adp} leads to a smaller performance drop than only using SC, indicating the effectiveness of A_{adp} in capturing useful latent graph structures. More detailed studies below show that A_{adp} learns distinct representations not captured by A.

As mentioned in section 2, our motivation behind creating sample-level adaptive adjacency matrices is FC's highly dynamic nature. Therefore, for setting (iv), we test directly using adjacency matrices A_{i_FC} obtained from FC instead of the learned A_{i_adp}. In particular, $A_{i\_\text{FC}} = \tilde{D}_{\text{FC}_i}^{-\frac{1}{2}} \tilde{\text{FC}}_i \tilde{D}_{\text{FC}_i}^{-\frac{1}{2}} \in \mathbb{R}^{200\times 200}$, where $(\text{FC}_i)_{vw} = \text{corr}((X_i)_v, (X_i)_w)$, $\tilde{\text{FC}}_i = \text{FC}_i + I_N$ and $\tilde{D}_{\text{FC}_i} = \sum_w (\tilde{\text{FC}}_i)_{vw}$. Fig. 2 shows A_{i_FC} constantly underperforms A_{i_adp}, except for being really close for length-8 inputs. Larger performance gaps are observed for longer inputs, where $\text{Corr}((X_i)_v, (X_i)_w)$ struggles to capture the changing dynamics in the inputs. This demonstrates that our input-based latent A_{i_adp} has better representation power than input-based FC. We

also notice batch correlation coefficients calculation for A_{i_FC} results in a slower training speed than computing A_{i_adp}.

An interesting result comes from setting (v), where we use randomly generated Erdős-Rényi graphs with the edge creation probability the same as averaged edge existence probability of A's. Its performance is similar to or even better than settings (ii) and (iii). We examine this further in section 3.3.

Latent adaptive adjacency matrix A_{adp}. The above results demonstrate latent A_{adp} can complement the task- and temporal-fixed A. We now show that the learned A_{i_adp} is sparse for each sample, has evident task-based patterns, and provides new information beyond A_i. The sparsity of A_{i_adp} can be seen from fig. 11a in appendix: each input only gets a few important columns (information-providing nodes in GNN). These columns vary from one sample to another, indicating A_{adp}'s ability to adapt to changing inputs within the same task. However, when we look into inputs generated by consecutive sliding windows (not shuffled) from the same scan session as in fig. 11b, we can see the latent structures change smoothly. In addition, when we aggregate samples inside each task, noticeable task-based patterns emerge (fig. 11c). These patterns are different from Attr_A in fig. 4, suggesting that A_{adp} embeds dynamics not captured by A.

Quantitatively, A_{i_adp} entry values range between (0, 1) because of the Softmax, and only around 2% of entries in A_{i_adp} have values larger than 0.05. As a reference, the largest entry value is larger than 0.99. A similar sparsity pattern is found when using synthetic data on the same model, indicating that the sparsity is more due to the model than the underlying biology. Given how A_{i_adp} is used in GNN layers, each column of it represents a signal-originating node during message passing. We hypothesize that the model learns the most effective *hubs* that pass information to their neighbors. A related idea is information bottleneck [29]: deep learning essentially compresses the inputs as much as possible while retaining the mutual information between inputs and outputs. In a sense, A_{i_adp} represents the compressed hubs for a given input signal. We also note that this sparsity emerges even without any additional constraints. In fact, adding L_1 constraints on A_{adp} does not change the model performance or the A_{i_adp} sparsity level. We hypothesize that the naturally trained A_{i_adp} is sparse enough, and further sparsification is unnecessary.

We visualize the projected inputs $X_i\Theta_{\text{adp}}$ in fig. 11d, which clearly shows the task, node and subject heterogeneities. Different tasks have varied representations in the latent space for the same node, but DOT, MOD, PVT has similar embedding patterns across individuals and most nodes. Indeed, when looking at the confusion matrix across models (fig. 12 in appendix), the misclassifications mostly cluster between these three tasks, indicating their natural similarity. We want to note here that adding a learnable bias to $X\Theta_{\text{adp}}$ does not separate the task embeddings further, nor does it improve overall performance. Subjects also exhibit heterogeneity: the same pair of nodes during the same task can have different embedding distances, thus graph edge weights, for each individual.

Multi-resolution inner cluster smoothing. To verify the capability of inner cluster smoothing operation in capturing latent graph dynamics, we test the following settings: (vi) using our proposed model and inputs, except removing paralleled GNN_{pool} and inner cluster smoothing module; (vii) previous setting (v) but remove GNN_{pool} and inner cluster smoothing module; (viii) keep GNN_{pool}, but using coarsened graph instead of smoothing (essentially performing DiffPool with an added temporal dimension). In this last setting, we hierarchically pool and reduce the graph to a single node, and we keep the total number of GNN layers the same as our other settings. Values of soft-assigned cluster number c are chosen to be halved per smoothing module (e.g., $N/2, N/4, \cdots$) for our experiments. Different choices of c affect the model convergence rate but only have a minor impact on the final performance (see appendix B.1 (II)). Results are reported in fig. 2 (and table 3 in appendix). Apart from these three settings, we also test adding pooling regularization terms (described in appendix A.2) into the loss function, but they do not lead to much of a difference.

Table 2: Model comparisons with length-256 inputs.

Model	Weighted F1	Training time (s / epoch)
GCN [11]	42.84	713
GAT V2 [2]	50.36	1142
GConvGRU [24]	56.05	9886
GraphSAGE [8]	61.87	1048
Graph Transformer [25]	66.11	1890
MVTS Transformer [37]	88.16	**39**
ReBraiD (proposed: TCN + GNN)	**90.85**	298
ReBraiD (TCN only)	71.98	119
ReBraiD (TCN + CNN)	75.79	124

The above results demonstrate that both setting (vi) and (vii) outperforms (viii) by a large margin, indicating the importance of keeping the original node number when representing brain signals. In addition, all three settings underperform our proposed method. They are also mostly worse than changing graph adjacency matrices as in settings (ii)–(v): this shows the inner cluster smoothing module has a more significant impact in learning latent graph dynamics. We also find using adaptive adjacency matrices and inner cluster smoothing can stabilize training, making the model less prone to over-fitting and achieving close-to-best performance over a larger range of hyperparameters (see fig. 10).

3.2 Model Comparisons

Since we adopt a network view to studying the brain, where brain regions are treated as graph nodes, we source our baselines from graph models. To do so, we examined all models in PyTorch Geometric (PyG) [3] and its temporal extension (PyG-T) [4] as they contain the most up-to-date and well-organized open-source graph neural network model implementations. In particular, we compare our model with the vanilla GCN from [11], Chebyshev Graph Convolutional Gated Recurrent Unit (GConvGRU) from [24], GraphSAGE from [8], GAT V2 from [2] and Graph Transformer as in [25]. Baseline models are constructed similar to ours: each has four graph encoding layers taking in both signals and adjacency matrices, followed by two linear layers along the node axis and two linear layers for the final classification. We train baseline models with the same input,

[3] https://pytorch-geometric.readthedocs.io/
[4] https://pytorch-geometric-temporal.readthedocs.io/

loss, optimizer, and epoch settings (all models are well-converged). Grid search is used to optimize the rest of the hyperparameters. We compare weighted F1 and training time per epoch in table 2; we also plot our model and Graph Transformer's confusion matrices in fig. 12.

Our model shows significant performance gains and requires less training time than graph baselines. We believe the most critical reason is that the models in PyG treat temporal signals as feature vectors instead of placing them into a separate temporal dimension. Without sequence modeling on the temporal dimension, even the state-of-the-art graph attention models (GAT-v2 and graph Transformer) cannot perform well. In addition, almost all models in PyG-T assume one common graph for the inputs (application scenarios are traffic network forecasting, link predictions, etc.), whereas we need to feed different SC for every sample. Out of them, we were able to choose one model (GConvGRU) that supports different adjacency matrices, but it didn't give a satisfactory result. Our proposed ideas of sample-level adaptive adjacency matrix learning and multi-resolution inner cluster smoothing help capture latent brain dynamics and improve the performance. The higher model performance here reflects a better encoding ability of brain signals, which can benefit different downstream tasks such as disease and trait prediction.

In addition to graph baselines, we also tested the state-of-the-art model for multivariate time series classification (MVTS Transformer [37]), which has comparable performance to ours. This stresses the critical role of temporal modeling when dealing with dynamic signals, so we tested our model without GNN layers. We experiment both removing GNN layers altogether and replacing them with 1×1 CNN layers: both outperform graph models that focus on the spatial modeling aspect. Although these results demonstrate that temporal modeling is crucial, adding graph modeling that includes signals' spatial relationships as proposed can further improve the performance. Since the MVTS Transformer model has projections to generate queries, keys, and values from the input sequence, it can also implicitly learn spatial relationships between variables (*nodes*). On the other hand, explicitly adding graph components allows the model to utilize prior structures (e.g., SC). The attribution of graph models can also provide better interpretability of brain networks, such as identifying critical region connections.

3.3 Interpretation with IG

This section studies the contributions of different brain ROIs and subnetworks defined by their functionalities. For the subnetwork definition, we choose to use the 17 networks specified in [28], which has a mapping from our previous 200-ROI parcellation[5]. To select baseline inputs, we follow the general principle for attribution methods: when the model takes in a baseline input, it should produce a near-zero prediction, and Softmax(outputs) should give each class about the same probability in a classification model. All-zero baselines A' and X' can roughly achieve this for our model, so we choose them as our baseline inputs. Step number M is set to 30. The IG computation is done on 900 inputs for each task to get an overall distribution.

[5] https://github.com/ThomasYeoLab/CBIG/blob/master/stable_projects/brain_parcellation

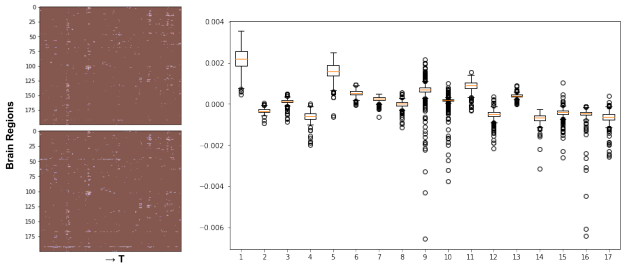


Figure 3: (a) Temporal importance sanity check of IG results on two pieces of inputs with a large overlap period. Attribution maps are offset aligned. (b) Attr_X distributions across 17 brain subnetworks (defined as in [28]) for VWM.

The extracted high-attribution regions and connections should be reproducible across different initializations to be used for downstream tasks. Since the overall problem is non-convex, we empirically test and confirm the attribution reproducibility with two randomly initialized models before proceeding to the following analyses. In addition, [32] demonstrates IG's consistency (reproducibility among a range of hyperparameters) and faithfulness (more accurate attribution can be obtained with better performing models). Since our model has higher performance with longer inputs, we compute IG attributions of a model trained on length-256 input signals in this section.

Temporal importance. On the single input level, we can attribute which parts of the inputs in $\mathcal{G}_i$ are more critical in predicting the target class by looking into $(\text{Attr}_X)_i$. This attribution map not only shows which brain regions contribute more but also reveals the important signal frames. One critical drawback of fMRI imaging is its low temporal resolution, but if we know which part is more important, we can turn to more temporally fine-grained signals such as EEG to see if there are any special activities during that time. To confirm that the attributions we get are valid and consistent, we perform a sanity check of IG results on two overlapped inputs with an offset τ: the first input is obtained from window $[t_0, t_0 + T]$ and the second is obtained from window $[t_0 + \tau, t_0 + \tau + T]$. Offset aligned results are shown in fig. 3a, in which the attributions agree with each other quite well.

Spatial importance. We examine the connection importance between brain ROIs by looking at Attr_A. In particular, columns in Attr_A with higher average values are sender ROIs of high-contributing connections, which is what matters in the GNN operation. We also explore why using random graph adjacency matrices (setting (v) in section 3.1) can produce a similar result for length-256 inputs compared to using both SC-induced A_i and A_{i_adp} (setting (i)). By examining Attr_A under both settings (fig. 4), we see that the column averages of Attr_A under these two settings are similar for almost all tasks, meaning the model can learn the important signal sending regions relatively well even without explicit structures. We credit this ability primarily to multi-resolution inner cluster smoothing, as the performance drops notably without it (setting (vii)). However, using ground truth SC not only gives us higher performance for shorter inputs but also provides the opportunity to interpret brain region connections better. We can directly use

task-averaged Attr_A as the weighted adjacency matrix to plot edges between brain ROIs, just as in fig. 5. Important brain regions obtained from Attr_A mostly comply with the previous literature (see appendix B.3 for details).

In addition to Attr_A, Attr_X can also provide insights on spatial importance when the attribution maps are aggregated along the temporal dimension. But it does so from another perspective: based on how the model takes in the inputs, larger Attr_A implies critical *structural connections* between brain regions, meaning that information passing between those regions is deemed essential in classifying task states. In contrast, larger Attr_X reveals regions or subnetworks that are sources of the important *signals*: it does not matter if the signal activities propagate from one region to another. Instead, the signals themselves are crucial for differentiating between task states. We notice that signal-important ROIs are not necessarily the same as connection-important ROIs: top-ranked subnetworks for resting state are DefaultA and DefaultB by Attr_A, and VisCent and DorsAttnA by Attr_X; although they do coincide with each other for tasks like VMN. This disparity is reflected in fig. 5 as edge and node differences. Another observation is that DYN and PVT have similar Attr_A patterns; both have a high attribution on connections originating from visual, control, and somatomotor systems. But when looking at Attr_X, DYN and PVT are extreme opposites. For example, PVT has a very high Attr_X for a few ROIs in LH_SomMotA, DorsAttnA_TempOcc, and RH_VisCent_ExStr, while DYN has very low Attr_X for them. This suggests that the model uses these ROIs' activities to distinguish between the two tasks. Therefore, the attributions are not absolute but relative to what they are compared against. As a result, when identifying biomarkers with attribution, it is crucial to have *contrasts*—for example, different tasks, different disease states, etc.

In fig. 3b, we plot the distribution of time-averaged and subnetwork-averaged (mapping 200 ROIs into 17 subnetworks) Attr_X during the VWM task. We can see the clear dominance of VisCent, DorsAttnA, and ContA subnetworks (numbered as 1, 5, 11), indicating signals from these regions are useful for the model to decide if the input is from the VWM task. More informative than the rankings is the distribution itself: even though VisCent, DorsAttnA, and ContA ranked top 3 for both resting state and VWM for signal attributions, their relative importance and attribution distribution variances are drastically different. In a sense, the distribution can act as a task fingerprint based on brain signal states.

Group, session, and region heterogeneity. Average variances of attributions are very different across tasks, especially those of Attr_X: VWM and DYN have much smaller attribution variances compared to other tasks. This can be caused by either task dynamics when certain tasks have more phase transitions and brain status changes, or/and group heterogeneity when individuals carry out specific tasks more differently than the others. We investigate this by examining three subjects that have multiple scan sessions for every task.

We report the following findings: (1) Even only aggregating attributions over a single subject's sessions, attribution variances of the other four tasks are still larger than VWM and DYN. And these variance values are comparable to that of aggregating over many subjects. This means the large variances are not mainly due to group heterogeneity; rather, some tasks have more states than others. (2) There is still group heterogeneity apart from different task dynamics, and the group heterogeneity is also more evident for tasks with more dynamics (high attribution variances). We can see from fig. 6 that attributions for VMM are much more concentrated and universal across subjects than that of MOD. (3) Flexibility of different subnetworks varies: subnetworks with small distribution IQR (interquartile range) of the same subject's different sessions are also more consistent across subjects. One example is that subnetwork 18 during the MOD task has both higher within-subject IQR and more significant across-subject differences than subnetwork 19. This indicates that for a particular task, some subnetworks are more individual and flexible (may activate differently across time), while others are more collective and fixed. In summary, we can find both critical regions that a particular task must rely on and regions that can characterize individual differences during tasks.

3.4 Simulation study

To validate the results of our interpretations, we perform simulation studies with known ground truth. All graphs are generated with SBM (stochastic block model) using the same community structure (200 nodes, 10 communities), but each graph has its own adjacency matrix. This generation process mimics brain structures in that samples share similar community structures but have distinct structural connectivities. Fig 7a shows a typical adjacency matrix of a synthetic graph. All adjacency matrices are binary. Time-series on each node are then generated with code adapted from pytorch-gnn repository [6]. In particular, the value at each time step of each node is a small temporal Gaussian random noise plus signals from neighbors' (a small spatial Gaussian noise is added to the adjacency matrix) previous step.

Simulation (I) We create two classes for this simulation. In class one, only the first three communities (nodes 1–60) generate small temporal noises, and other nodes are only affected by neighbors. In class two, only the last three communities (nodes 141–200) generate small temporal noises, and other nodes are only affected by neighbors. We visualize the task aggregated Attr_X and A_{adp} and in figs. 7b and 7c. The signals are characterized well in Attr_X. For the generated series, signals are more important in node 1–60 for class 1 and 141–200 for class 2: A_{adp} finds this pattern and helps propagate signals in these regions better. We notice that Attr_A is mostly random, with no apparent patterns. This is consistent with the graph signal generation: when aggregating information from neighbors, all connected edges are weighted the same (binary); thus, the connections do not affect generated signals. We perform the following study to understand the opposite effect.

Simulation (II) We again create two classes for the simulation: in class one, connections from nodes 61–100 are strengthened; in class two, connections from nodes 101–140 are strengthened. The weights of strengthened edges are increased from 1 to 5 during signal generation. However, the model still takes in binary adjacency matrices as inputs (processed as mentioned in section 2.1 before feeding to the model). We visualize the task aggregated A_{adp} and Attr_A in fig. 7d. This time the connection differences are reflected in Attr_A. Signals in node 61–100 for class 1 or 101–140 for class 2 are less important because stronger connections can send

[6] https://github.com/alelab-upenn/graph-neural-networks

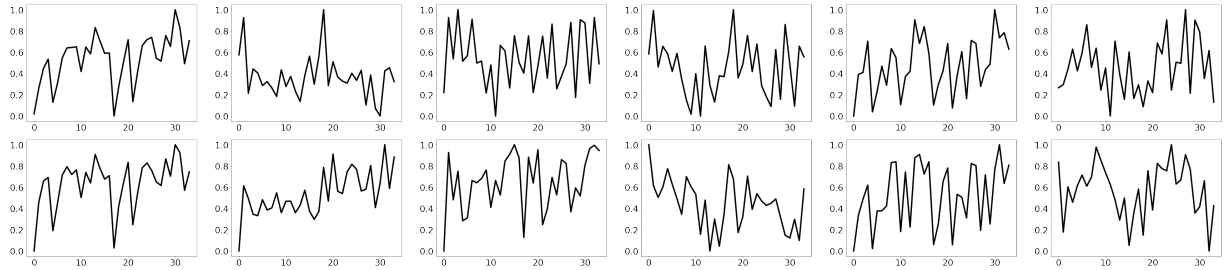


Figure 4: Column averages of task-averaged Attr_A (mapped into 34 subnetworks defined by the 17-network parcellation with left, right hemispheres). Top row is obtained from real SC induced A and bottom rows is obtained from random SC induced A_{rand}. Attributions are normalized to $[0, 1]$. Tasks are: Rest, VWM, DYN, DOT, MOD, PVT from left to right.

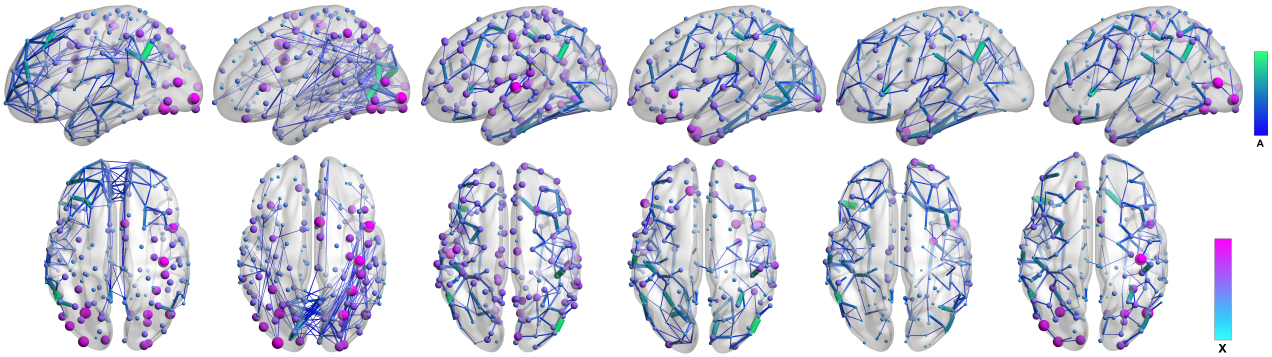


Figure 5: ROI attributions from Attr_A and Attr_X. (Task order is the same as fig. 4). Edge color and width are based on task-averaged $\text{Attr}_A \in \mathbb{R}^{200\times200}$, and node color and size are based on task and temporal-averaged $\text{Attr}_X \in \mathbb{R}^{200}$. For visualization, only edges with highest attributions are shown (the resulting sparsity reduces to 0.009 from 0.196).

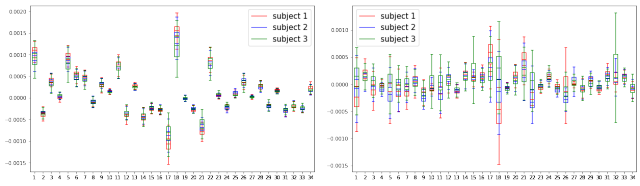
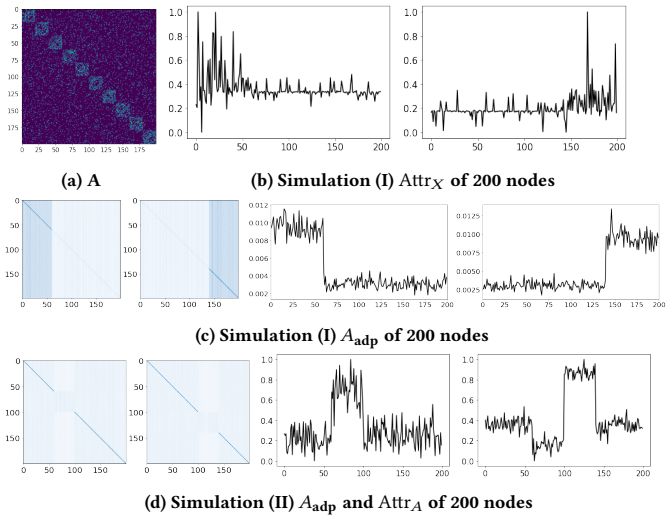


Figure 6: 34 subnetworks' Attr_X distributions of 3 subjects performing the VWM task (left) and the MOD task (right). Outliers that go beyond $[Q1 - 1.5\,\text{IQR}, Q3 + 1.5\,\text{IQR}]$ are omitted. VWM has a much smaller average attribution variance than MOD.

these signals out: this results in smaller values for corresponding columns in A_{adp}. Combined with the previous simulation results, this suggests that strong signal sending regions or regions with weak connections that are over-reflected in the graph adjacency matrix tend to have higher A_{adp} values. In other words, A_{adp} complements both signals and connections to encode latent dynamics, while attributions obtained from IG are better at interpreting the modalities separately.

4 CONCLUSIONS

This paper proposes ReBraiD, a high-performing and efficient graph neural network model that embeds both structural and dynamic functional signals for a more comprehensive representation of brain



(a) A

(b) Simulation (I) Attr_X of 200 nodes

(c) Simulation (I) A_{adp} of 200 nodes

(d) Simulation (II) A_{adp} and Attr_A of 200 nodes

Figure 7: (a) A typical adjacency matrix for simulated graph signals. (b) Task averaged Attr_X of simulation (I). Attribution values are normalized. (c) Task averaged A_{adp} of simulation (I) and its entry averages per column. (d) Task averaged A_{adp} and task averaged Attr_A of simulation (II). Attribution values are normalized.

dynamics. To better capture latent structures, we propose sample-level adjacency matrix learning and multi-resolution inner cluster

smoothing. Apart from quantitative results showing ReBraiD's superiority in representing brain activities, we also leverage integrated gradients to attribute and interpret the importance of both spatial brain regions and temporal keyframes. The attribution also reveals heterogeneities among brain regions (or subnetworks), tasks, and individuals. These findings can potentially reveal new neural basis, biomarkers of tasks or brain disorders when combined with behavioral metrics. They can also enable more fine-grained temporal analysis around keyframes when combined with other imaging techniques and extend to different scientific domains with sample (subject) heterogeneity.

ACKNOWLEDGMENTS

This project was partially supported by funding from the National Science Foundation under grant IIS-1817046.

REFERENCES

[1] Danielle S Bassett and Olaf Sporns. 2017. Network neuroscience. *Nature Neuroscience* 20, 3 (2017), 353–364.
[2] Shaked Brody, Uri Alon, and Eran Yahav. 2021. How Attentive are Graph Attention Networks? arXiv:2105.14491 [cs.LG]
[3] Joshua M Carlson, Felix Beacher, Karen S Reinke, Reza Habib, Eddie Harmon-Jones, Lilianne R Mujica-Parodi, and Greg Hajcak. 2012. Nonconscious attention bias to threat is correlated with anterior cingulate cortex gray matter volume: a voxel-based morphometry result and replication. *Neuroimage* 59, 2 (2012), 1713–1718.
[4] Joshua M Carlson, Jiook Cha, and Lilianne R Mujica-Parodi. 2013. Functional and structural amygdala–anterior cingulate connectivity correlates with attentional bias to masked fearful faces. *Cortex* 49, 9 (2013), 2595–2600.
[5] Sean PA Drummond, Amanda Bischoff-Grethe, David F Dinges, Liat Ayalon, Sara C Mednick, and MJ Meloy. 2005. The neural basis of the psychomotor vigilance task. *Sleep* 28, 9 (2005), 1059–1068.
[6] Roland M Friedrich and Angela D Friederici. 2013. Mathematical logic in the human brain: semantics. *PLoS One* 8, 1 (2013), e53699.
[7] Roland H Grabner, Gernot Reishofer, Karl Koschutnig, and Franz Ebner. 2011. Brain correlates of mathematical competence in processing mathematical representations. *Frontiers in Human Neuroscience* 5 (2011), 130.
[8] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 1025–1035.
[9] Byung-Hoon Kim and Jong Chul Ye. 2020. Understanding Graph Isomorphism Network for rs-fMRI Functional Connectivity Analysis. *Frontiers in Neuroscience* 14 (2020), 630. https://doi.org/10.3389/fnins.2020.00630
[10] Jangjin Kim, Edward A Wasserman, Leyre Castro, and John H Freeman. 2016. Anterior cingulate cortex inactivation impairs rodent visual selective attention and prospective memory. *Behavioral Neuroscience* 130, 1 (2016), 75.
[11] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.
[12] Nina Lauharatanahirun, Kanika Bansal, Steven M Thurman, Jean M Vettel, Barry Giesbrecht, Scott Grafton, James C Elliott, Erin Flynn-Evans, Emily Falk, and Javier O Garcia. 2020. Flexibility of brain regions during working memory curtails cognitive consequences to lack of sleep. *arXiv preprint arXiv:2009.07233* (2020).
[13] Colin Lea, René Vidal, Austin Reiter, and Gregory D. Hager. 2016. Temporal Convolutional Networks: A Unified Approach to Action Segmentation. In *Computer Vision – ECCV 2016 Workshops*, Gang Hua and Hervé Jégou (Eds.). Springer International Publishing, Cham, 47–54.
[14] Robert Leech and David J Sharp. 2014. The role of the posterior cingulate cortex in cognition and disease. *Brain* 137, 1 (2014), 12–32.
[15] Lingge Li, Dustin Pluta, Babak Shahbaba, Norbert Fortin, Hernando Ombao, and Pierre Baldi. 2019. Modeling dynamic functional connectivity with latent factor Gaussian processes. *Advances in Neural Information Processing Systems* 32 (2019), 8263–8273.
[16] Xiaoxiao Li, Yuan Zhou, Nicha C. Dvornek, Muhan Zhang, Juntang Zhuang, Pamela Ventola, and James S. Duncan. 2020. Pooling Regularized Graph Neural Network for fMRI Biomarker Analysis. *Medical Image Computing and Computer-assisted Intervention (MICCAI)* 12267 (2020), 625–635.
[17] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. 2020. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 143–152.
[18] Fuad Noman, Chee-Ming Ting, Hakmook Kang, Raphael C. W. Phan, Brian D. Boyd, Warren D. Taylor, and Hernando Ombao. 2021. Graph Autoencoders for Embedding Learning in Brain Networks and Major Depressive Disorder Identification. arXiv:2107.12838 [q-bio.NC]
[19] Aäron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. 2016. Conditional image generation with PixelCNN decoders. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*. 4797–4805.
[20] Marcus E Raichle. 2015. The brain's default mode network. *Annual Review of Neuroscience* 38 (2015), 433–447.
[21] Luana Ruiz, Fernando Gama, and Alejandro Ribeiro. 2020. Gated graph recurrent neural networks. *IEEE Transactions on Signal Processing* 68 (2020), 6303–6318.
[22] Alexander Schaefer, Ru Kong, Evan M Gordon, Timothy O Laumann, Xi-Nian Zuo, Avram J Holmes, Simon B Eickhoff, and BT Thomas Yeo. 2018. Local-global parcellation of the human cerebral cortex from intrinsic functional connectivity MRI. *Cerebral Cortex* 28, 9 (2018), 3095–3114.
[23] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*. 618–626.
[24] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. 2018. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing*. Springer, 362–373.
[25] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. 2021. Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification. arXiv:2009.03509 [cs.LG]
[26] Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. 2020. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 914–921.
[27] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*. PMLR, 3319–3328.
[28] BT Thomas Yeo, Fenna M Krienen, Jorge Sepulcre, Mert R Sabuncu, Danial Lashkari, Marisa Hollinshead, Joshua L Roffman, Jordan W Smoller, Lilla Zöllei, Jonathan R Polimeni, et al. 2011. The organization of the human cerebral cortex estimated by intrinsic functional connectivity. *Journal of Neurophysiology* 106, 3 (2011), 1125–1165.
[29] Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*. IEEE, 1–5.
[30] J Jay Todd and René Marois. 2004. Capacity limit of visual short-term memory in human posterior parietal cortex. *Nature* 428, 6984 (2004), 751–754.
[31] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. WaveNet: A Generative Model for Raw Audio. In *Proc. 9th ISCA Workshop on Speech Synthesis Workshop (SSW 9)*. 125.
[32] Alexander B Wiltschko, Benjamin Sanchez-Lengeling, Brian Lee, Emily Reif, Jennifer Wei, Kevin James McCloskey, Lucy Colwell, Wesley Qian, and Yiliu Wang. 2020. Evaluating Attribution for Graph Neural Networks. *Google Research* (2020).
[33] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International Conference on Machine Learning*. PMLR, 6861–6871.
[34] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph wavenet for deep spatial-temporal graph modeling. *International Joint Conferences on Artificial Intelligence (IJCAI)* (2019).
[35] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How Powerful are Graph Neural Networks?. In *International Conference on Learning Representations*.
[36] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L Hamilton, and Jure Leskovec. 2018. Hierarchical graph representation learning with differentiable pooling. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 4805–4815.
[37] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. 2021. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2114–2124.
[38] Gemeng Zhang, Biao Cai, Aiying Zhang, Julia M Stephen, Tony W Wilson, Vince D Calhoun, and Yu-Ping Wang. 2019. Estimating dynamic functional brain connectivity with a sparse hidden Markov model. *IEEE Transactions on Medical Imaging* 39, 2 (2019), 488–498.

A MODELS

A.1 Choice of temporal layers

Fig. 8 explains the choice of TCN layers.

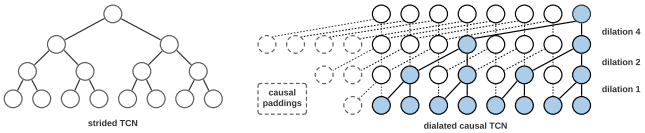


Figure 8: Comparison of strided non-causal TCN (left) and dilated causal TCN (right). For a causal TCN, the causal aspect is achieved through padding (kernel_size − 1) × dilation number of zeros to the layer's input. The resulting y always has the same length as input x, in which y_τ only depends on inputs $x_{t\leq\tau}$. We can view strided non-causal TCN as the rightmost node of a dilated causal TCN.

A.2 Regularization terms for soft-assignment

For each soft assignment matrix $S \in \mathbb{R}^{N\times c\times t}$ in eq. (4), we test three regularization terms:

- Similar to DiffPool, to ensure a more clearly defined node assignment, namely each node is only assigned to few clusters (the closer to one the better), we minimize the entropy of single node assignments: $L_{E_1} = \frac{1}{c}\sum_{i=1}^{c} H(S_i)$.
- To ensure a representation separation among nodes, meaning the assignment should not assign all the nodes a same way, we maximize the entropy of node assignment *patterns* across all nodes: $L_{E_2} = -\frac{1}{c}\sum_{i=1}^{c} H(\sum_{j=1}^{n} S_{ij})$.
- To make the assignment along temporal axis smoother, we penalize assignment variances within a small time window $[\hat{t}, \hat{t}+\tau]$: $L_T = \frac{1}{t-\tau}\sum_{\hat{t}=0}^{t-\tau} \sigma(S_{[\hat{t},\hat{t}+\tau]})$, where σ represents standard deviation.

Together with cross entropy classification loss L_{CE}, the final loss function of the model becomes:

$$L_{reg} = \alpha_1 L_{CE} + \alpha_2 L_{E_1} + \alpha_3 L_{E_2} + \alpha_4 L_T, \quad \sum_i \alpha_i = 1 \qquad (7)$$

B EXPERIMENTS

B.1 Ablation studies

Numerical values of fig. 2 are reported in table 3. Training time ranges from 51 seconds / epoch for length-8 inputs to 298 seconds / epoch for length-256 inputs. Models converges to a relatively stable loss level within 20 epochs.

Table 3: Weighted F1 of ablation study settings.

Input length (frames)	8	16	32	64	128	256
(i): SC + adp	**66.19**	**70.18**	**75.87**	**76.14**	**82.91**	**90.85**
(ii): SC only	64.54	65.58	71.79	70.31	73.63	89.79
(iii): adp only	64.32	65.20	74.01	71.42	80.63	89.46
(iv): SC + FC	66.10	67.58	70.26	75.02	76.91	84.68
(v): random adj	62.17	66.25	72.30	73.72	76.58	89.22
(vi): (i) without smoothing	63.57	62.82	70.19	65.82	72.91	79.65
(vii): (v) without smoothing	56.88	64.08	72.27	62.72	75.16	83.75
(viii): coarsened graph	37.92	42.23	46.18	52.12	57.17	64.25

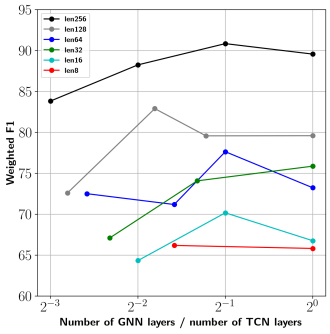


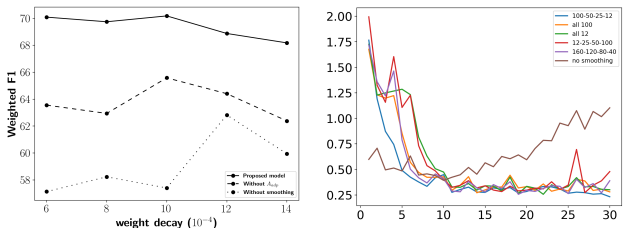
Figure 9: Choosing number of GNN to TCN layer ratio for different input lengths. In most cases, two TCN layers per GNN layer results in the best model performance in terms of F1.

(I) Number of GNN layers. The total number of temporal layers depends on the input signal length since each strided TCN layer reduces the temporal length by a factor of two: if the input length is 2^i, there need to be i temporal layers. *But is alternating every TCN with GNN the best strategy, or do we only need to follow one GNN after a few TCNs*? We study this question with different input lengths.

Model weighted F1 are plotted in fig. 9 for all possible GNN to total TCN ratios (e.g. length-256 inputs requires 8 TCN layers. The possible ratios are $\frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1$ since we can insert one GNN per 8, 4, 2, 1 TCN layers). The figure shows alternating every layer rarely yields the highest performance and the best ratio lies around one GNN per two TCN layers for our dataset. We repeat the experiment for $K = 1, 3$ (in eq. (3)) to rule out the possibility that this result is related to how many neighbors one GNN layer can reach; we find they have roughly the same pattern as the $K = 2$ case. We hypothesize that a lower GNN to TCN ratio does not capture enough spatial context, while higher ones might be overfitting. We leave exploring the relationship between this ratio and the number of nodes N to a future study.

The best GNN to TCN ratio also depends on whether model incorporates latent adjacency matrices or not: without A_{adp}, length-128 signals achieves its relative best (among all ratios) when having one GNN per two TCNs, but it only needs one GNN per three TCNs if using A_{adp}. This shows learning latent structures A_{adp} not only improves overall model accuracy but can also reduce model parameters, thus complexity, in achieving better results.

(II) Effects of soft-assignment cluster numbers. During our experiments, we find that as long as the smoothing module is used, the final performance will be close to each other, only the convergence rates are different. Fig. 10b shows how validation loss converges with different c (cluster number) or when there is no smoothing module. From it, we can observe that halving the numbers (100-50-25-12) is the most helpful setting, and we use it for our other experiments; decreasing the numbers (160-120-80-40) or all larger numbers (all 100) works better than increasing the numbers (12-25-50-100) or all smaller numbers (all 12). With the inner cluster smoothing module, all cluster number settings converge to around 0.23 at their smallest when trained for 30 epochs; their test weighted F1 range from 89.47 (model with 12-25-50-100) to 90.85 (model with 100-50-25-12).



(a) (b)

Figure 10: (a) adding inner cluster smoothing or input-dependent adaptive adjacency matrix makes the model more stable across various learning rates (results shown are from length-16 inputs). (b) Validation loss v.s. training epochs. Input length is 256, and four smoothing modules are used. Legends are the soft-assignment cluster numbers of the four smoothing modules. Our other experiments use decreasing cluster numbers that halved at each module, corresponding to the 100-50-25-12 choice here.

On the contrary, if no smoothing module is used, the model overfits easily, and the validation loss can only reach about 0.4 before going up (with the best set of learning rate and weight decay parameters found with grid search). Understandably, the model is prone to overfitting given the complexity of GNN and the relatively small dataset size. However, our added inner cluster smoothing module effectively counters the effect and further brings the loss down in a stable manner.

B.2 Model comparisons

We plot confusion matrices of ReBraiD, the model from ablation study setting (viii), and the best performing graph baseline in fig. 12. Misclassification pairs clustered at the first three tasks (resting, VWM, DYN) and the latter three (DOT, MOD, PVT). Shown confusion matrices are from models trained on length-256 inputs. We note that these misclassification pairs may differ for models trained on other input lengths (like 128-frame, etc.).

B.3 Attributions

Many discriminatory regions obtained from Attr_A are consistent with existing literature:

Resting state: The top attributed ROIs belong to the default mode network, which is regarded salient during the resting state [20].

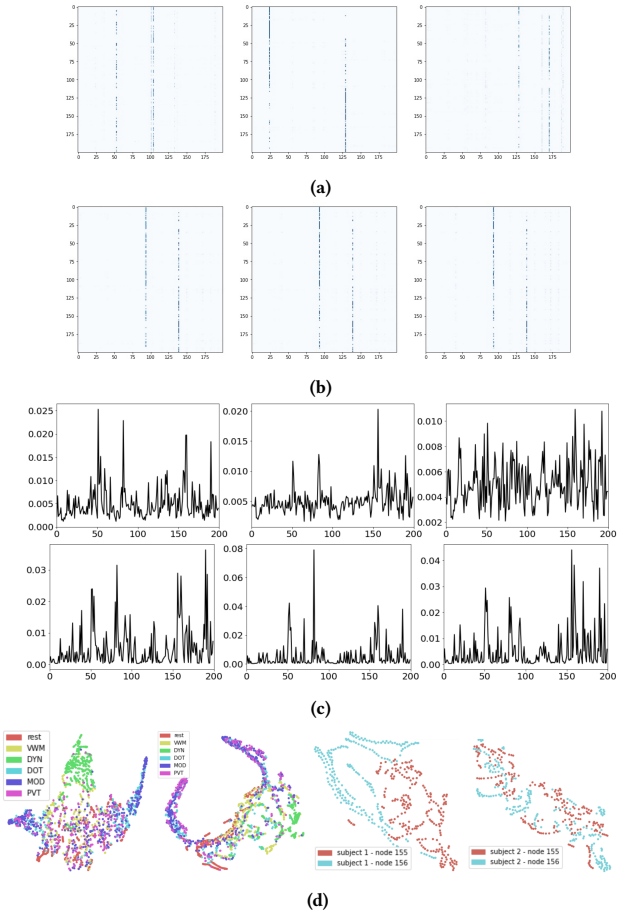
VWM: The dominant attributions are from visual regions and posterior parietal regions, which complies with [30].

DYN: Attributions from our model suggest regions along cingulate gyrus (defaultA-SalValAttnB-ContA-ContC-defaultC), as well as peripheral visual and somatomotor regions. Literature suggests anterior cingulate cortex (ACC) to be active [10] and posterior cingulate cortex (PCC) to be inactive [14] during visual attention tasks. This means both regions provide discriminative information about the DYN states, which is what our attribution method votes for.

DOT: Important ROIs from our analysis are located in control networks, in particular both ACC and PCC, as well as in the peripheral visual system. In the literature, dorsal and rostral regions of the ACC are proved to be involved with dot-probe performance [3, 4].

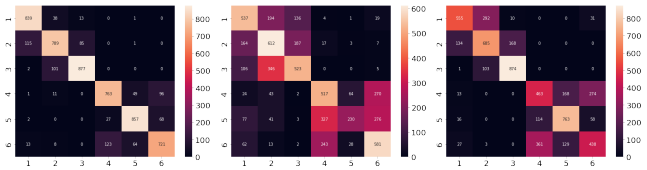
MOD: Our important ROIs are mostly in temporal-parietal regions and default mode network (anatomically frontoparietal), and literature suggests similar regions: parietal [7] and prefrontal [6].

PVT: Our top attributed ROIs belong to control networks, attention networks, and somatomotor regions. This is similar to [5], where both attention and motor systems are considered important.



(a) (b) (c) (d)

Figure 11: Learned latent adaptive adjacency matrices. (a) A_{i_adp} of 3 randomly sampled inputs during the DOT task. (b) A_{i_adp} of 3 consecutive inputs from a same session during the DOT task. (c) column averages of task-averaged A_{adp} for resting state, VWM, DYN, DOT, MOD, PVT. (d) left two: t-SNE of $X^{(\text{node-2, 156})}\Theta_{\text{adp}}$ in six tasks of one subject; right two: t-SNE of $X^{(\text{node-155, 156})}\Theta_{\text{adp}}$ during the resting state of two subjects (multiple sessions are aggregated).



(a) (b) (c)

Figure 12: Confusion matrices of: (a) ReBraiD (our proposed model), (b) model with coarsened graph (setting (viii)), (c) Graph Transformer (best graph baseline). Tasks are 1-Rest, 2-VWM, 3-DYN, 4-DOT, 5-MOD, 6-PVT.