Continual Learning for fMRI-Based Brain Disorder Diagnosis via Functional Connectivity Matrices Generative Replay

Supplementary Material

Our supplementary materials are organized as follows:

- Detailed Description of the Cross-Site Continual Learning Pipeline
- rs-fMRI Preprocessing
- Details in Hierarchical Contextual Thompson Sampling
- Extended Evaluation and Ablation Studies of FCM-VAE
- Additional Ablation Results

A. Detailed Description of the Cross-Site Continual Learning Pipeline

Figure 1 provides an overview of our cross-site continual learning pipeline. For each clinical site, we formulate a site-specific classification task that maps rs-fMRI data to diagnostic labels. Starting from the raw rs-fMRI scans acquired at site i, we first apply a standardized preprocessing procedure, as described in Section B. The preprocessed volumes are subsequently parcellated into cortical and subcortical regions using the AAL atlas, yielding a fixed set of 116 regions of interest (ROIs). For each ROI, we extract its blood-oxygen-level-dependent (BOLD) time series by averaging the voxelwise signals. Functional connectivity (FC) matrices are then computed as the Pearson correlation between every pair of ROI time series, resulting in a subject-specific 116×116 symmetric FC matrix.

Each FC matrix is interpreted as a weighted brain graph and fed into a graph classifier that outputs a diagnostic prediction. The classification labels distinguish subjects with the target disorder from healthy controls. Across clinical sites, we cast the problem as a continual learning scenario in which each site i defines a task T_i that arrives sequentially.

After completing task T_{i-1}, the trained GCN model from site M_{i-1} is frozen and used as a teacher network. When task T_i arrives with data from a new clinical site, we process the rs-fMRI data through the same pipeline and train a student classifier on the new site's FC graphs.

During this stage, the student model is trained jointly on two types of data: (i) the current site's FC matrices and (ii) replayed samples retrieved from the replay buffer. For each training batch, the frozen teacher network produces supervision signals from the same FC inputs, while the student extracts graph embeddings that pass through its prediction head.

As illustrated in Figure 2, the student model is optimized using three complementary objectives: (i) a graph readout distillation loss that aligns the student's graph-level embeddings with those of the teacher, (ii) a logits distillation loss that enforces agreement between the teacher's and student's predictive distributions, and (iii) a replay loss applied to samples from previous sites. Together, these losses enable the student to retain cross-site knowledge while adapting to newly arriving data, thereby mitigating catastrophic forgetting across tasks.

B. rs-fMRI Preprocessing

All rs-fMRI datasets were preprocessed using a standardized pipeline to ensure consistency across clinical sites, following established practices in previous studies. Preprocessing was performed using the FMRIB Software Library (FSL)[1] and the Statistical Parametric Mapping toolbox (SPM12)[2]. For each subject, the first several volumes were removed to allow for magnetization stabilization. Slice-timing correction was applied to account for inter-slice acquisition delays, followed by rigid-body motion correction to align all volumes to a reference frame and reduce head-motion artifacts. Distortion correction, addressing susceptibility-induced warps using scans with varying acquisition parameters, was conducted with the topup toolbox in FSL using default settings.

The preprocessed fMRI volumes were parcellated into anatomically defined regions using the AAL-116 atlas. For each of the 116 ROIs, we extracted the mean BOLD time series by averaging the voxelwise signals within the region. Functional connectivity (FC) matrices were then computed by calculating the Pearson correlation between the time series of every pair of ROIs, resulting in a 116×116 symmetric FC matrix for each subject.

C. Details in Hierarchical Contextual Thompson Sampling

While a generative replay buffer $\mathcal{R}_i$ enables privacy-preserving rehearsal, using a fixed replay budget is both inefficient and oblivious to distributional shifts across hospitals, ultimately weakening cross-domain generalization. To overcome this, we introduce a Hierarchical Contextual Thompson Sampling mechanism that dynamically allocates replay capacity at two levels: (i) site-level replay quotas adapt to accuracy–forgetting trade-offs; (ii) sample-level quotas enforce diversity and representativeness. Both layers leverage hierarchical contextual features and linear Gaus-

[1] https://fsl.fmrib.ox.ac.uk/fsl/docs/

[2] https://www.fil.ion.ucl.ac.uk/spm/software/spm12/

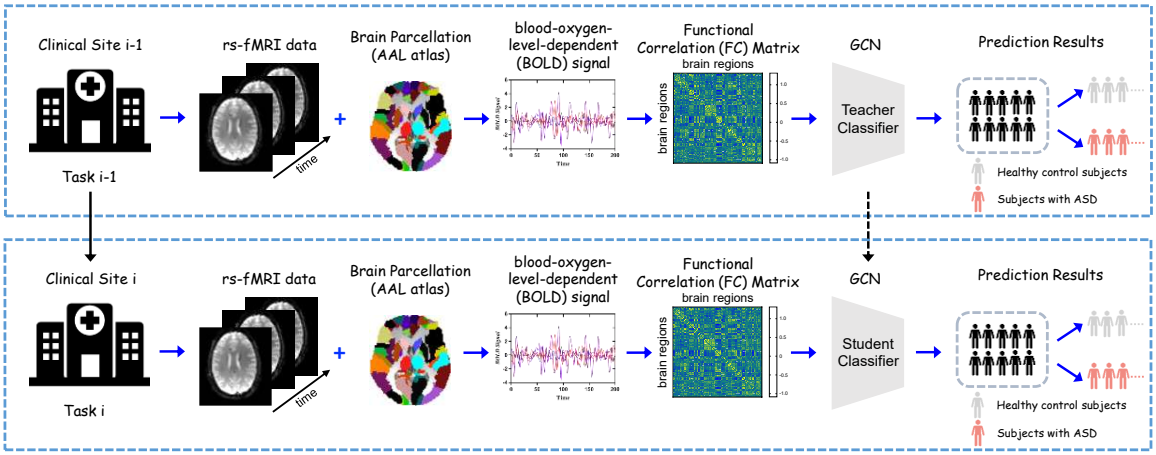

Figure 1. Continual Learning pipeline for transforming raw rs-fMRI data into inputs for the classification task.

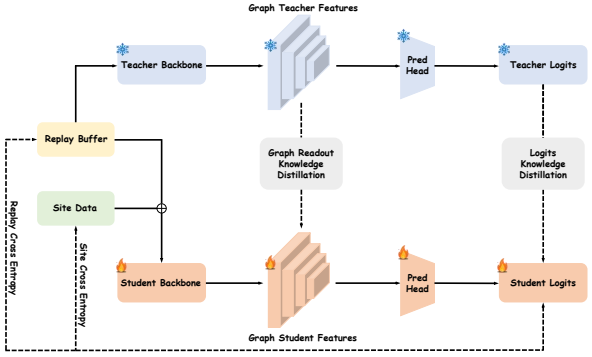

Figure 2. Dual-Level Knowledge Distillation Framework for Continual Graph Learning

sian posteriors for Thompson sampling, enabling online adaptive optimization. We next detail the sampling strategies at both the site and sample levels.

C.1. Details of Site-Level Thompson Sampling Allocation

In cross-hospital continual learning, a fixed replay budget $\mathcal{R}$ restricts each round to a limited number of generated samples. To allocate this budget effectively, we design a site-level adaptive mechanism that jointly accounts for current accuracy and historical forgetting. Concretely, for each site $M_i \in \mathcal{M}$, we define a context vector as:

$$\boldsymbol{\phi}_i = \left[\mathrm{Acc}_i,\ \mathrm{Forget}_i\right]$$

where $\mathrm{Acc}_i = \frac{1}{|\mathcal{D}_i^{\mathrm{test}}|}\sum_{(x_i,y_i)\in\mathcal{D}_i^{\mathrm{test}}} \mathbb{1}\{f_\theta(x_i) = y_i\}$ denotes the current accuracy of f_θ on site M_i, and $\mathbb{1}$ is the indicator function. $\mathrm{Forget}_i = \max(0,\ \mathrm{Acc}_i^{\mathrm{past}} - \mathrm{Acc}_i^{\mathrm{curr}})$ denotes forgetting at site M_i, where $\mathrm{Acc}_i^{\mathrm{past}}$ is the best past accuracy and $\mathrm{Acc}_i^{\mathrm{curr}}$ is the current one.

The expected replay utility is modeled as

$$r_i \ = \ \boldsymbol{\phi}_i^\top \mathbf{w} + \epsilon_i, \qquad \epsilon_i \sim \mathcal{N}(0, \sigma^2), \tag{1}$$

where $\mathbf{w}$ is a latent weight vector whose posterior distribution $p(\mathbf{w}|\mathcal{D})$ encodes uncertainty conditioned on the history $\mathcal{D}$ of previously observed contexts and rewards. At each update round, Thompson Sampling draws a posterior sample $\tilde{\mathbf{w}} \sim p(\mathbf{w}|\mathcal{D})$ and computes the predicted utility $\tilde{r}_i = \boldsymbol{\phi}_i^\top \tilde{\mathbf{w}}$. Meanwhile, we introduce a closed-loop update mechanism to incorporate performance feedback. We define the reward change as

$$r_i^{(t)} \ = \ \sigma\left(\frac{\mathrm{CE}_i^{\mathrm{prev}} - \mathrm{CE}_i^{\mathrm{new}}}{\tau}\right), \tag{2}$$

where $\sigma(\cdot)$ is the sigmoid function, $\mathrm{CE}_h^{\mathrm{prev}}$ and $\mathrm{CE}_h^{\mathrm{new}}$ are the cross-entropy losses of site i before and after the update, and τ is a temperature parameter

To maintain a Bayesian estimate of the latent weight vector $\mathbf{w}$, each site keeps sufficient statistics A_i and b_i, updated at each step as:

$$\begin{aligned} A_i^{(t+1)} &\leftarrow \gamma A_i^{(t)} + (1-\gamma)\,\boldsymbol{\phi}_i^{(t)}\boldsymbol{\phi}_i^{(t)\top}, \\ b_i^{(t+1)} &\leftarrow \gamma b_i^{(t)} + (1-\gamma)\,\boldsymbol{\phi}_i^{(t)} r_i^{(t)}, \end{aligned} \tag{3}$$

where γ is a forgetting factor. We initialize $A_i^{(0)} = \lambda I$ and $b_i^{(0)} = 0$ for numerical stability.

The posterior mean and covariance of the Thompson sampling model are approximated as

$$\mu_i^{(t)} = \left(A_i^{(t)}\right)^{-1} b_i^{(t)}, \qquad \Sigma_i^{(t)} = \sigma^2 \left(A_i^{(t)}\right)^{-1}. \tag{4}$$

A Thompson sample $\mathbf{w}_i^{(t)} \sim \mathcal{N}\left(\mu_i^{(t)},\ \Sigma_i^{(t)}\right)$ yields the predicted utility

$$\hat{r}_i^{(t)} \ = \ \boldsymbol{\phi}_i^{(t)\top} \mathbf{w}_i^{(t)}. \tag{5}$$

These predicted utilities are then standardized using z-score normalization and converted into allocation weights

$$w_i^{(t)} = \frac{\exp(\hat{r}_i^{(t)})}{\sum_j \exp(\hat{r}_j^{(t)})}, \tag{6}$$

from which the final replay allocation is obtained via

$$k_i^{(t)} \;=\; K\, w_i^{(t)}. \tag{7}$$

After each update, the replay buffer draws $k_i^{(t)}$ samples from each site's generative memory, the model is updated, and new rewards and contexts are computed, forming a closed-loop adaptive replay strategy.

C.2. Sample-Level Contextual Thompson Sampling

Given the site-specific replay quota k_i, we further identify the most beneficial samples within each replay buffer $\mathcal{R}_i$. To model sample utility, we introduce a sample-level context vector that captures predictive uncertainty and geometric consistency. For a candidate sample u from site M_i, the context is defined as

$$\boldsymbol{\psi}_u = \big[\mathrm{margin}_u,\ \mathrm{closeness}_u\big]. \tag{8}$$

The first component, margin_u, measures predictive uncertainty. Let $p = \mathrm{softmax}(h_\theta(u))$ denote the model output probabilities. The margin is computed as

$$\mathrm{margin}_u = \max_c p_c - \max_{c' \neq c} p_{c'}, \tag{9}$$

where a smaller value implies greater ambiguity. Theoretically, low-margin samples generally lie closer to the decision boundary and thus provide higher learning value.

The second component, $\mathrm{closeness}_u$, quantifies how well u aligns with its site-specific data manifold. Let

$$z_u^{(t)} = \rho\Big(E_\theta^{(t)}(u)\Big)$$

denote the latent representation at update step t. The closeness score is defined as

$$\mathrm{closeness}_u = 1- \tanh\left(\sqrt{\big(z_u^{(t)} - \boldsymbol{\mu}_{y_u}^{i,(t)}\big)^\top \big(\boldsymbol{\Sigma}_{y_u}^{i,(t)}\big)^{-1} \big(z_u^{(t)} - \boldsymbol{\mu}_{y_u}^{i,(t)}\big)}\right), \tag{10}$$

where $\boldsymbol{\mu}_{y_u}^{i,(t)}$ and $\boldsymbol{\Sigma}_{y_u}^{i,(t)}$ are the prototype mean and covariance for class y_u at site M_i, updated through exponential moving averages over the replay-augmented dataset $\tilde{\mathcal{D}}_i^{(t)}$. This Mahalanobis-based criterion encourages selecting samples that best preserve site structure while respecting privacy constraints.

To incorporate feedback and maintain Bayesian estimates over sample utilities, each site maintains sufficient statistics C_i and $\boldsymbol{d}_i$, which capture second-order correlations among sample contexts and their rewards. After selecting a provisional sample subset $\mathcal{S}_i^{(t)}$, the site-level reward $r_i^{(t)}$ is evenly assigned to its samples, and the statistics are updated using a forgetting factor γ:

$$\begin{aligned} C_i^{(t)} &\leftarrow \gamma C_i^{(t-1)} + (1-\gamma) \sum_{u \in \mathcal{S}_i^{(t)}} \boldsymbol{\psi}_u^{(t)} \boldsymbol{\psi}_u^{(t)\top}, \\ \boldsymbol{d}_i^{(t)} &\leftarrow \gamma \boldsymbol{d}_i^{(t-1)} + (1-\gamma) \sum_{u \in \mathcal{S}_i^{(t)}} \boldsymbol{\psi}_u^{(t)} \frac{r_i^{(t)}}{|\mathcal{S}_i^{(t)}|}. \end{aligned} \tag{11}$$

We initialize $C_i^{(0)} = \lambda I$ and $\boldsymbol{d}_i^{(0)} = 0$ for numerical stability. The posterior mean and covariance are then estimated as

$$\boldsymbol{\mu}_i^{(t)} = (C_i^{(t)})^{-1} \boldsymbol{d}_i^{(t)}, \quad \boldsymbol{\Sigma}_i^{(t)} = \sigma^2 (C_i^{(t)})^{-1},$$

where σ^2 denotes the reward variance. Thompson sampling is applied by drawing

$$\mathbf{v}_i^{(t)} \sim \mathcal{N}(\boldsymbol{\mu}_i^{(t)}, \boldsymbol{\Sigma}_i^{(t)}),$$

and scoring each candidate u via

$$s_u^{(t)} = \mathbf{v}_i^{(t)\top} \boldsymbol{\psi}_u^{(t)}. \tag{12}$$

To select a final replay subset, we combine value-based ranking and diversity promotion. Candidates in $\tilde{\mathcal{D}}_i^{(t)}$ are first ranked by $s_u^{(t)}$, and the top $2k_i$ form a shortlist $\mathcal{C}_i^{(t)}$. The final k_i samples are chosen using a greedy farthest-first strategy in the representation space:

$$\mathcal{S}_i^{(t)} = \arg \max_{S \subseteq \mathcal{C}_i^{(t)},\, |S|=k_i} \min_{\substack{G,G' \in S \\ G \neq G'}} \left\| \rho(E_\theta(G)) - \rho(E_\theta(G')) \right\|_2. \tag{13}$$

This ensures that replayed samples are both high-value and distributionally diverse.

D. Extended Evaluation and Ablation Studies of FCM-VAE

To better understand the behavior of different FC-graph generators, we first visualize their outputs on the ASD dataset, as shown in Figure 3. For a representative negative and positive subject, we compare the original FC matrix with those produced by ReGate [26], BrainNetGAN [22], GR-SPD-GAN [38] and our proposed FCM-VAE. Existing methods roughly recover the coarse modular structure =, but tend to over-smooth fine-grained connectivity patterns or introduce spurious high-magnitude edges, leading to visually distorted or overly homogenized graphs. In contrast, FCM-VAE yields FC matrices that closely follow the original both in global organization and local edge patterns, preserving salient functional modules while suppressing artifacts.

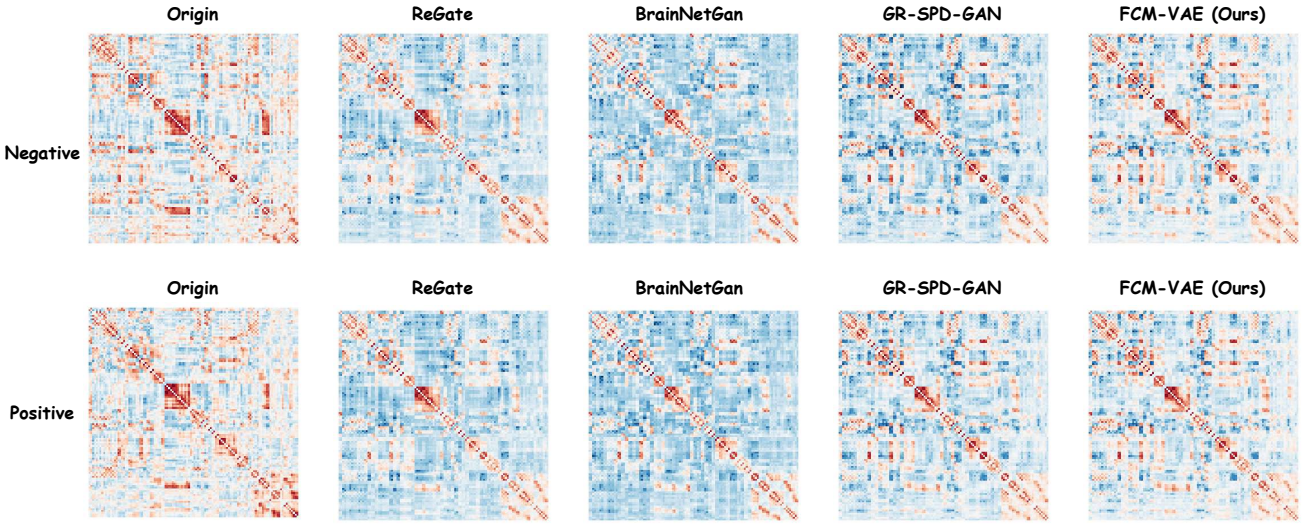

Figure 3. Visualization of functional connectivity (FC) matrices generated by different FC-graph generation methods on the ASD dataset. Columns correspond to different generation method, while rows show negative and positive samples.

To further validate the robustness of the synthetic FC matrices produced by FCM-VAE, we extend the data augmentation experiment to a broader set of graph classifiers. While the main paper reports results using a GCN backbone, Table 1 evaluates five widely used architectures under different FC generation strategies. Below, we briefly summarize the key modeling characteristics of each classifier:

- **GAE** [18]: A graph autoencoder that employs a encoder to map nodes into latent embeddings and an inner-product decoder to reconstruct the graph. It was originally proposed for unsupervised node representation learning and link prediction; in our setting, we use its encoder followed by global pooling as a graph-level classifier.
- **GAT** [42]: A graph attention network where node representations are aggregated using learned attention coefficients, enabling the model to weigh neighbor contributions adaptively based on edge importance.
- **GCN** [19]: The standard graph convolutional network that propagates information through normalized message passing.
- **GraphSAGE** [13]: An inductive graph representation learning framework that learns node embeddings by sampling and aggregating information from local neighborhoods using parametric aggregators.
- **GraphTransformer** [10]: A transformer-based architecture extended to graphs via positional encodings and attention over structural distances.

The main architectural and optimization hyperparameters of these classifiers are summarized in Table 2. All models are trained for 100 epochs under identical data splits and optimization settings to ensure a fair comparison.

For each classifier, we fix the model and vary the FC generation method among five options: no augmentation, ReGate [26], BrainNetGAN [22], GR-SPD-GAN [38], and our FCM-VAE. Performance is reported using accuracy (ACC), precision (PRE), recall (REC), and F1 score on the ASD, BSNIP, and MDD datasets. Across all classifiers and datasets, FCM-VAE consistently achieves either the best or second-best performance, reflecting its ability to generate FC graphs. Notably, even when paired with simple backbones, FCM-VAE offers substantial improvements over alternative generative models, while GR-SPD-GAN typically ranks second overall. These results reinforce that the advantages of FCM-VAE are not tied to a specific classifier architecture. This demonstrates that incorporating global topological structure and local information into the generative process enhances the quality of the synthesized FC graphs.

We further evaluate the contribution of each component in FCM-VAE by removing them individually. As shown in Table 4, the absence of any module degrades performance across all datasets, validating their complementary roles. In particular, removing the node feature encoder causes the most significant accuracy drop, emphasizing the importance of incorporating node features with local connectivity patterns and global geometric features. Both the Local Adjacency Encoding (LAE) and Spectral Positional Encoding (SGE) also contribute to preserving local and global structural information, respectively, confirming that their joint design enables more reliable FC graph generation.

Table 1. FC classification results with data augmentation using different graph generation methods. Each row fixes a classifier and varies the generation model to assess the quality of synthetic FC graphs. Columns report accuracy (ACC), precision (PRE), recall (REC), and F1 score on the ASD, BSNIP, and MDD datasets. Bold numbers denote the best performance, while underlined numbers indicate the second-best results.

Classifier	Gen. Method	ASD				BSNIP				MDD			
		ACC	PRE	REC	F1	ACC	PRE	REC	F1	ACC	PRE	REC	F1
GAE	No Augmentation	0.726	**0.753**	0.669	0.683	0.662	**0.779**	0.414	0.421	**0.767**	**0.733**	0.795	0.731
	ReGate	0.711	0.728	0.715	0.689	0.630	0.542	0.421	0.434	0.722	0.668	0.761	0.688
	BrainNetGan	0.723	0.743	0.701	0.696	0.660	0.727	**0.440**	**0.455**	0.749	0.669	0.816	0.726
	GR-SPD-GAN	0.704	0.651	**0.846**	0.734	0.663	0.723	0.377	0.434	0.747	0.712	0.768	0.710
	FCM-VAE (Ours)	**0.760**	0.728	0.831	**0.772**	**0.671**	0.737	0.376	0.433	0.765	0.648	**0.885**	**0.745**
GAT	No Augmentation	0.713	0.668	**0.823**	0.735	0.652	0.698	0.397	0.408	0.683	0.582	**0.763**	**0.653**
	ReGate	0.683	0.687	0.734	0.674	0.624	0.622	0.315	0.330	0.686	0.589	0.746	0.649
	BrainNetGan	0.721	0.722	0.716	0.709	0.643	0.684	**0.444**	0.442	0.683	0.596	0.699	0.635
	GR-SPD-GAN	0.762	**0.768**	0.735	0.747	0.679	**0.800**	0.406	0.441	0.717	0.679	0.702	0.600
	FCM-VAE (Ours)	**0.776**	0.767	0.786	**0.770**	**0.690**	0.765	0.403	**0.493**	**0.724**	**0.694**	0.697	0.608
GCN	No Augmentation	0.762	0.748	0.772	0.758	0.688	0.733	0.456	0.502	0.756	**0.733**	0.699	0.700
	ReGate	0.724	0.688	0.806	0.740	0.639	0.618	0.318	0.361	0.724	0.701	0.725	0.676
	BrainNetGan	0.758	0.757	0.752	0.747	0.686	0.747	0.458	0.488	0.748	0.707	0.745	0.712
	GR-SPD-GAN	0.780	0.761	**0.812**	**0.780**	0.703	**0.785**	0.452	0.513	0.767	0.666	**0.866**	**0.745**
	FCM-VAE (Ours)	**0.783**	**0.776**	0.775	0.771	**0.727**	0.727	**0.547**	**0.597**	**0.787**	**0.733**	0.784	0.735
GraphSage	No Augmentation	**0.740**	0.720	0.784	0.736	0.656	0.621	0.425	0.449	0.751	0.712	0.707	0.671
	ReGate	0.708	**0.727**	0.758	0.725	0.623	0.542	**0.460**	**0.467**	0.725	0.681	0.720	0.657
	BrainNetGan	0.695	0.706	0.652	0.658	0.648	0.634	0.365	0.432	0.716	0.678	0.685	0.667
	GR-SPD-GAN	0.729	0.716	0.753	0.729	0.658	0.644	0.365	0.455	0.744	**0.811**	0.594	0.612
	FCM-VAE (Ours)	**0.740**	0.722	**0.795**	**0.745**	**0.666**	**0.702**	0.357	0.452	**0.763**	0.681	**0.796**	**0.730**
GraphTransformer	No Augmentation	0.694	0.663	**0.781**	0.709	0.643	**0.689**	0.393	0.426	0.692	0.605	0.700	0.636
	Regate	0.679	0.653	**0.781**	0.700	0.612	0.602	0.260	0.281	0.680	**0.671**	0.647	0.571
	BrainNetGan	0.692	0.666	0.766	0.705	0.632	0.567	**0.528**	**0.524**	0.702	0.628	0.809	0.678
	GR-SPD-GAN	**0.725**	0.708	0.756	**0.720**	**0.667**	0.642	0.480	0.507	0.719	0.609	**0.859**	**0.710**
	FCM-VAE (Ours)	0.711	**0.716**	0.684	0.696	0.664	0.658	0.444	0.459	**0.723**	0.627	0.837	0.709

Table 2. Hyperparameter summary for the graph classifiers used in our experiments.

Model	Layers	Hidden Dim	Dropout	LR
GCN	4	[128,128,128,128]	0.3	1×10^{-3}
GAT	3	[128,128,128]	0.4	5×10^{-4}
GraphSAGE	4	[128,128,128,128]	0.3	1×10^{-3}
GraphTransformer	3	[128,128,128]	0.3	5×10^{-4}
GAE	3	[128,128,128]	0.2	1×10^{-3}

Table 3. Grid search ranges for the three hyperparameters λ_1, λ_2, and λ_3.

Coefficient	Search space
λ_1	$\{0.20, 0.25, 0.30, 0.35, 0.40\}$
λ_2	$\{0.20, 0.25, 0.30, 0.35, 0.40\}$
λ_3	$\{0.10, 0.15, 0.20, 0.25, 0.30\}$

E. Additional Ablation Results

E.1. Component Contributions in FORGE over DER++

DER++ is a particularly strong and widely adopted continual-learning baseline, making it a rigorous reference point for evaluating FORGE. To clarify what specifically drives FORGE improvements over DER++, we design a controlled ablation that incrementally introduces our key components on top of the DER++ pipeline.

Concretely, starting from DER++ with real sample replay (Real), we replace only the replay source with our generative replay (Gen.) while keeping all other settings fixed. Then, with generative replay held constant, we swap only the sampling strategy (HCTS) or only the distillation objective (Dual-KD). As summarized in Table 5, it shows that HCTS yields the strongest performance, indicating that replay selection is a key driver of the gains, likely because fMRI data quality strongly affects both stability and forgetting. Meanwhile, Dual-KD maintain competitive AAA while achieving lower FOR, suggesting that distillation primarily contributes to reducing forgetting.

Table 4. **Ablation study of FCM-VAE.** Each row fixes a classifier backbone and varies the ablation setting. Columns report accuracy (ACC), precision (PRE), recall (REC), and F1 score on the ASD, BRI, and MDD datasets. Full Framework denotes the complete FCM-VAE model.

Classifier	Ablation	ASD				BSNIP				MDD			
		ACC	PRE	REC	F1	ACC	PRE	REC	F1	ACC	PRE	REC	F1
GAE	**Full Framework**	0.760	0.728	0.831	0.772	0.671	0.737	0.376	0.433	0.765	0.648	0.885	0.745
	w/o Node Feature	0.723	0.701	0.769	0.731	0.656	0.732	0.368	0.414	0.710	0.694	0.632	0.582
	w/o LAE	0.744	0.714	0.813	0.755	0.664	0.799	0.345	0.390	0.753	0.688	0.776	0.709
	w/o SGE	0.734	0.782	0.658	0.706	0.653	0.661	0.379	0.411	0.737	0.618	0.796	0.687
GAT	**Full Framework**	0.776	0.767	0.786	0.770	0.690	0.765	0.403	0.493	0.724	0.694	0.697	0.608
	w/o Node Feature	0.737	0.761	0.688	0.719	0.658	0.764	0.395	0.420	0.706	0.712	0.580	0.563
	w/o LAE	0.743	0.752	0.692	0.720	0.688	0.684	0.459	0.528	0.720	0.688	0.657	0.585
	w/o SGE	0.735	0.728	0.741	0.730	0.681	0.739	0.448	0.501	0.697	0.688	0.608	0.563
GCN	**Full Framework**	0.783	0.776	0.775	0.771	0.727	0.727	0.547	0.597	0.787	0.733	0.784	0.735
	w/o Node Feature	0.746	0.762	0.694	0.712	0.693	0.747	0.429	0.489	0.761	0.638	0.887	0.739
	w/o LAE	0.762	0.740	0.775	0.754	0.701	0.733	0.439	0.517	0.763	0.705	0.773	0.718
	w/o SGE	0.760	0.730	0.786	0.755	0.687	0.635	0.517	0.550	0.745	0.672	0.801	0.720
GraphSAGE	**Full Framework**	0.740	0.722	0.795	0.745	0.666	0.702	0.357	0.452	0.763	0.681	0.796	0.730
	w/o Node Feature	0.721	0.727	0.721	0.701	0.644	0.599	0.411	0.474	0.741	0.688	0.713	0.685
	w/o LAE	0.738	0.722	0.795	0.745	0.664	0.653	0.394	0.483	0.735	0.694	0.725	0.698
	w/o SGE	0.721	0.726	0.708	0.704	0.645	0.704	0.369	0.390	0.742	0.711	0.711	0.694
GraphTransformer	**Full Framework**	0.711	0.716	0.684	0.696	0.664	0.658	0.444	0.459	0.723	0.627	0.837	0.709
	w/o Node Feature	0.693	0.685	0.700	0.686	0.651	0.591	0.398	0.426	0.698	0.657	0.713	0.659
	w/o LAE	0.708	0.694	0.725	0.702	0.653	0.591	0.484	0.502	0.702	0.662	0.730	0.669
	w/o SGE	0.690	0.696	0.672	0.668	0.649	0.602	0.435	0.489	0.698	0.674	0.607	0.619

Method	Replay	Sampling	Distillation	AAA↑	FOR↓
DER++-Real	Real	Reservoir Sampling	LKD	0.669	0.187
DER++-Gen.	Gen.	Reservoir Sampling	LKD	0.671	0.183
DER++-Gen. + HCTS	Gen.	HCTS	LKD	0.685	0.141
DER++-Gen. + Dual-KD	Gen.	Reservoir Sampling	Dual-KD	0.677	0.160
FORGE	Gen.	HCTS	Dual-KD	**0.730**	**0.116**

Table 5. Ablation variants by replay, sampling, and distillation.

E.2. Hyperparameter Analysis on regularization weight coefficients

We study the sensitivity of our continual learning objective by varying the three weights $\lambda_1, \lambda_2, \lambda_3$ in:

$$\mathbb{E}_{(G,y)\sim\mathcal{D}_{t_c}^{\text{train}}}\big[\ell\big(y, f_\theta(G)\big)\big] + \lambda_1 \sum_{t=1}^{t_c-1} \mathbb{E}_{(G,y)\sim\mathcal{R}_t}\big[\ell\big(y, f_\theta(G)\big)\big] + \lambda_2 \sum_{t=1}^{t_c-1} \mathbb{E}_{(G,u)\sim\mathcal{R}_t}\Big[\|h_\theta(G) - u\|_2^2\Big] + \lambda_3 \sum_{t=1}^{t_c-1} \mathbb{E}_{(G,r)\sim\mathcal{R}_t}\big\|\rho\big(E_\theta(G)\big) - r\big\|_2^2, \tag{14}$$

We perform a grid search over a predefined set of values for these coefficients, with the search ranges informed by parameter choices commonly used in related work [2] as summarized in Table 3. Across all evaluated configurations, the best-performing setting differs slightly across datasets. Specifically, the optimal coefficients for ASD correspond to $\lambda_1 = 0.20$, $\lambda_2 = 0.35$, and $\lambda_3 = 0.15$. For the BSNIP dataset, the best configuration is $\lambda_1 = 0.25$, $\lambda_2 = 0.30$, and $\lambda_3 = 0.15$, while for the MDD dataset the optimal setting becomes $\lambda_1 = 0.30$, $\lambda_2 = 0.30$, and $\lambda_3 = 0.20$.

As illustrated in Figure 4, the overall performance remains stable as the hyperparameters are varied on the ASD dataset. This demonstrates that our framework does not rely on finely tuned loss weights and that the model consistently maintains strong accuracy and low forgetting under a wide range of $(\lambda_1, \lambda_2, \lambda_3)$ configurations. Such behavior highlights the inherent robustness and adaptability of our method, indicating that strong performance is retained even under large variations in the coefficients.

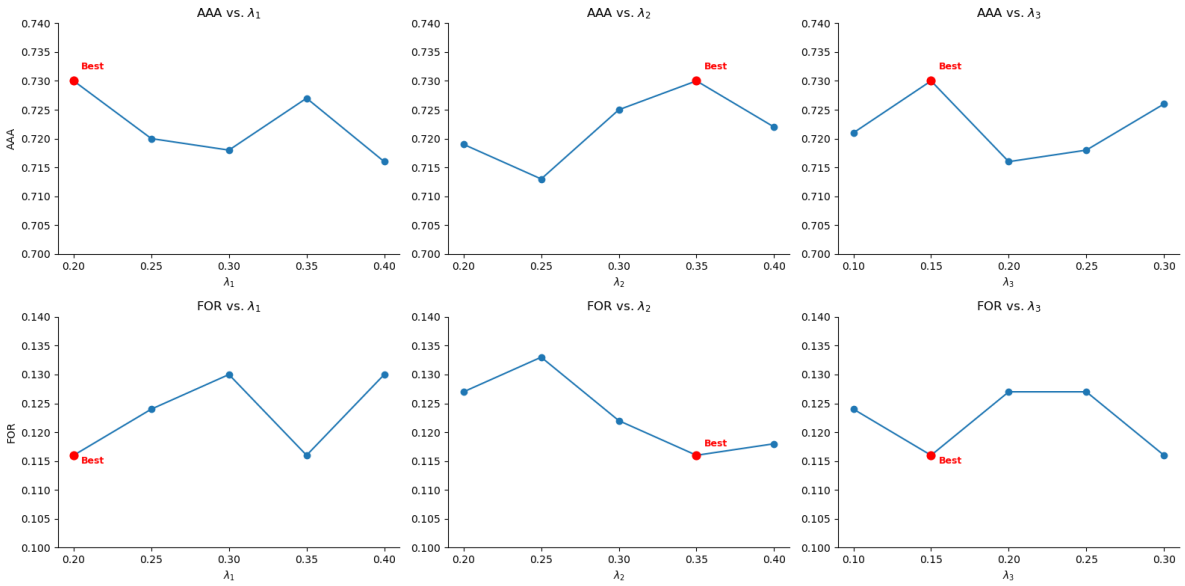

Figure 4. Sensitivity analysis of λ_1, λ_2, and λ_3 on the ASD dataset. For each hyperparameter, we vary its value while keeping the other two coefficients fixed at their default settings.