FEDDIG: FEDERATED CONTINUAL GRAPH LEARNING WITH DISENTANGLED GENERATIVE REPLAY

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated Continual Graph Learning (FCGL) is essential for real-world applications where graph data is both decentralized across clients and evolving over time. The primary challenge in this setting is catastrophic forgetting without centralizing raw client graphs. This forgetting manifests in two coupled aspects: the erosion of semantic decision boundaries and the degradation of structural relational patterns. Graph representations entangle node-level semantics with topology-dependent message passing, so a shift in either aspect can silently corrupt the other. In the federated continual setting this coupling is amplified: raw graphs are not centralized, precluding a shared raw-graph replay buffer, while heterogeneous topologies fragment old knowledge. Existing methods generally do not explicitly disentangle semantic and structural factors at the representation level, limiting factor-specific retention under drift. We propose FedDiG (Federated Disentangled continual Graph learning), which mitigates forgetting through post-hoc dual-gradient feature masking and generative replay. FedDiG derives complementary semantic and structural channels, trains a client-side generator on the current local graph, and applies server-side distillation that aligns both channels. Extensive experiments on seven datasets show that FedDiG achieves the highest average accuracy and lowest average forgetting among the compared baselines.

1 INTRODUCTION

Graph Neural Networks (GNNs) have emerged as the predominant paradigm for representation learning on relational data, demonstrating strong performance across diverse applications including molecular discovery Stokes et al. (2020), recommendation systems He et al. (2020), and financial risk analysis Dou et al. (2020); Liu et al. (2018). Through iterative aggregation of node features and neighborhood contexts, GNNs produce expressive embeddings that jointly capture attribute semantics and relational inductive biases. Despite their empirical success, real-world deployment of GNNs faces two critical challenges. First, graph data are often siloed across multiple entities (e.g., banks, hospitals), where privacy or governance constraints can restrict raw-data sharing, motivating Federated Graph Learning (FGL) He et al. (2021); Fu et al. (2022); Liu et al. (2025). Second, graph data exhibit inherent non-stationarity: interactions, labels, and behavioral patterns evolve continuously, demanding adaptive learning rather than static training, which motivates Continual Graph Learning (CGL) Liu et al. (2021); Zhou & Cao (2021). However, existing solutions address these challenges in isolation: FGL methods McMahan et al. (2017); Zhang et al. (2021); Baek et al. (2023); Wan et al. (2024) assume stationary distributions, while CGL approaches operate in centralized settings without privacy or communication constraints. This gap motivates **Federated Continual Graph Learning (FCGL)**: collaborative GNN training across decentralized clients without centralizing raw graph data, where data distributions evolve over time. FCGL introduces additional challenges beyond federated optimization, notably catastrophic forgetting under decentralized and evolving graph streams. As models are repeatedly updated with sequential client data, they tend to overfit recent observations while erasing previously learned knowledge. This effect is amplified in federated settings: raw graphs are not centralized in our setting, precluding a shared raw-graph replay buffer, while client heterogeneity fragments old knowledge across local views. Figure 1 illustrates the two resulting failure modes.

(1) Semantic Overlap under Federated Non-Stationarity. Non-stationary client streams gradually distort global decision boundaries and weaken previously learned class-discriminative cues, causing representations of different classes to overlap. **(2) Structural Forgetting under Evolving Graph**

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

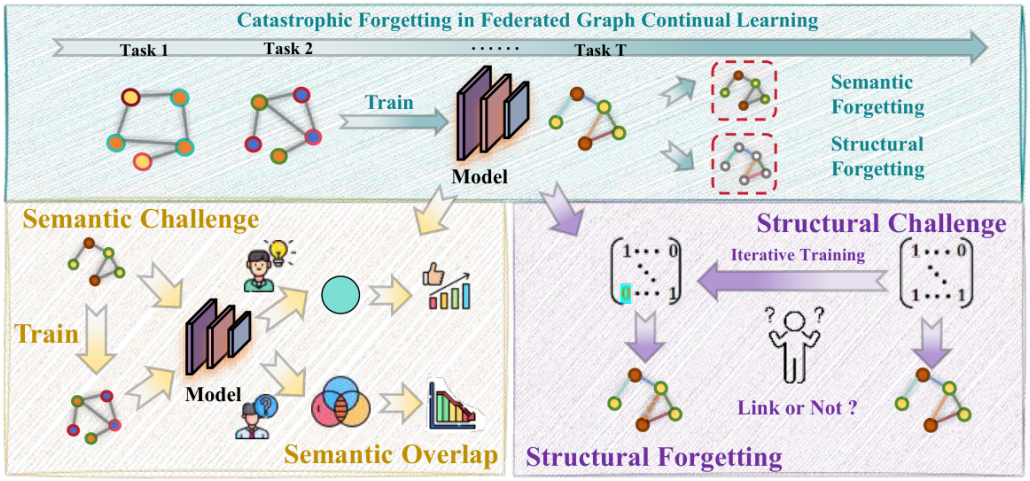

Figure 1: Illustration of catastrophic forgetting in FCGL. The model suffers from two distinct failure modes: **Semantic Overlap**, where class-discriminative decision boundaries are eroded by non-stationary client streams, and **Structural Forgetting**, where evolving topology degrades previously learned relational patterns.

Topology. Topology evolution changes message-passing contexts and weakens previously useful relational dependencies.

To address these two effects jointly, we propose FedDiG, which combines post-hoc dual-gradient channel separation with generative replay. For (1), the generator enforces semantic consistency in the semantic channel, while the server applies semantic-channel distillation to preserve class-level decision knowledge. For (2), the generator uses topology-constrained graph synthesis to retain transferable structural patterns, while the server applies structural-channel distillation to preserve relational reasoning under topology drift. Our main contributions are as follows:

- We identify two distinct failure modes in FCGL, namely semantic overlap and structural forgetting, and show empirical benefits of addressing them separately through complementary representation channels.
- We propose FedDiG, a framework that introduces post-hoc dual-gradient masking to derive complementary semantic and structural channels, a per-task topology-preserving generator, and server-side dual-channel distillation on a disjoint buffer of synthetic subgraphs.
- Extensive experiments show that FedDiG achieves the highest mean AA and the lowest mean AF across seven benchmarks under the unified protocol.

2 Related Work

Continual Graph Learning. Continual Graph Learning (CGL) studies catastrophic forgetting when GNNs are trained on a sequence of tasks. Existing methods broadly fall into regularization- or constraint-based approaches (e.g., EWC Kirkpatrick et al. (2017), GEM Lopez-Paz & Ranzato (2017)), replay-based approaches (e.g., ER-GNN Zhou & Cao (2021), SSM Zhang et al. (2022)), and architecture-based approaches such as expansion or isolation Rusu et al. (2016). Graph-specific variants such as TWP Liu et al. (2021) additionally consider topology preservation, while DSLR Choi et al. (2024) enhances replay diversity and incorporates structure learning to retain graph-specific knowledge across tasks. Despite these advances, most CGL methods optimize a single retention objective and do not explicitly separate semantic and structural factors in node representations. In practice, semantic information (e.g., class-discriminative features and decision boundaries) is tightly coupled with message passing, so shifts in topology can indirectly alter semantic embeddings even if logits are regularized. Meanwhile, structural knowledge corresponds to how a model uses neighbor-

hoods and connectivity patterns during aggregation, which is not well captured by parameter-level constraints alone. This coupling motivates a more explicit treatment of semantics and structure in continual graph learning.

Federated Graph Learning. Federated Graph Learning (FGL) He et al. (2021); Fu et al. (2022); Liu et al. (2025) enables collaborative GNN training across decentralized clients without exchanging raw data. Early frameworks such as FedAvg McMahan et al. (2017) and FedProx Li et al. (2020) aggregate local models to form a global one, while graph-specific methods handle structural challenges via neighbor completion (FedSage+ Zhang et al. (2021)), client clustering (GCFL Xie et al. (2021)), or topology-aware aggregation (FedGTA Li et al. (2023)). More recent work further addresses personalization and heterogeneity through prototype-based alignment Wan et al. (2024); Baek et al. (2023) and federated semantic–structural learning Huang et al. (2023). Most FGL methods, however, assume static data distributions. In drifting graph streams, naive aggregation tends to bias the global model toward recent updates, which can distort decision boundaries and accelerate forgetting of earlier classes. In addition, mechanisms designed for heterogeneity often adapt to current structural patterns without preserving long-horizon aggregation behaviors, leaving structural cues from earlier tasks under-protected.

Federated Continual Learning. Federated Continual Learning (FCL) studies incremental adaptation in federated systems under non-stationary data streams. Representative methods include GLFC Dong et al. (2022), TARGET Zhang et al. (2023), and Lander Tran et al. (2024). GLFC and TARGET mitigate forgetting through exemplar-free semantic distillation, and Lander introduces text-enhanced data-free replay. These methods are effective for semantic retention, but they are not designed to explicitly preserve graph-specific structural dynamics.

Federated Continual Graph Learning (FCGL) extends FCL to graph data, where clients receive sequential tasks under evolving semantics and topology drift. POWER Zhu et al. (2025) combines experience-node replay with prototype-based server alignment, but retains real nodes in the local GNN objective. MOTION Wan et al. (2025) targets topology preservation at task transitions: clients apply topology-preserving coarsening, and the server performs topology-sensitive aggregation. FedFST Guo et al. (2026) instead addresses spectral catastrophic forgetting, restoring high-frequency inconsistency knowledge and transferring low-frequency semantic consistency. These methods do not explicitly partition representation dimensions into semantic and structural channels. FedDiG addresses this gap by training a per-task generator on the current local graph, storing the resulting synthetic subgraphs for rehearsal, and explicitly disentangling semantic and structural knowledge.

3 METHODOLOGY

3.1 PRELIMINARIES AND PROBLEM FORMULATION

We consider FCGL with K clients and T sequential tasks. At task t, client k observes $G_{k,t} = (V_{k,t}, E_{k,t}, Y_{k,t})$ with $E_{k,t} = \{(u,v) \in E_k : u, v \in V_{k,t}\}$; inter-task edges are excluded from training and evaluation (Sec. 4.1). The global GNN f_θ minimizes

$$\min_{\theta} \sum_{t'=1}^{t} \mathbb{E}_{(G,y)\sim\mathcal{D}_{t'}} \left[\mathcal{L}_{\mathrm{ce}}(f_\theta(G), y)\right], \tag{1}$$

without retaining raw past graphs for later training. The core challenge is dual: models must preserve *semantic invariants* (decision boundaries) and *structural dependencies* (message-passing patterns). FedDiG addresses this via a three-stage disentangled framework, summarized in Figure 2.

3.2 DUAL-GRADIENT FEATURE DISENTANGLEMENT

Local GNN training uses only $\mathcal{L}_{\mathrm{ce}}$. After current-task aggregation we freeze f_θ, extract $H = f_\theta^{\mathrm{hid}}(G_{k,t}) \in \mathbb{R}^{N\times D}$, and score each hidden dimension with two post-hoc sensitivities. Neither score updates θ. Semantic importance is the mean CE-gradient magnitude

$$I_d^{\mathrm{sem}} = \frac{1}{N} \sum_{i=1}^{N} \left|\partial \mathcal{L}_{\mathrm{ce}} / \partial H_{i,d}\right|. \tag{2}$$

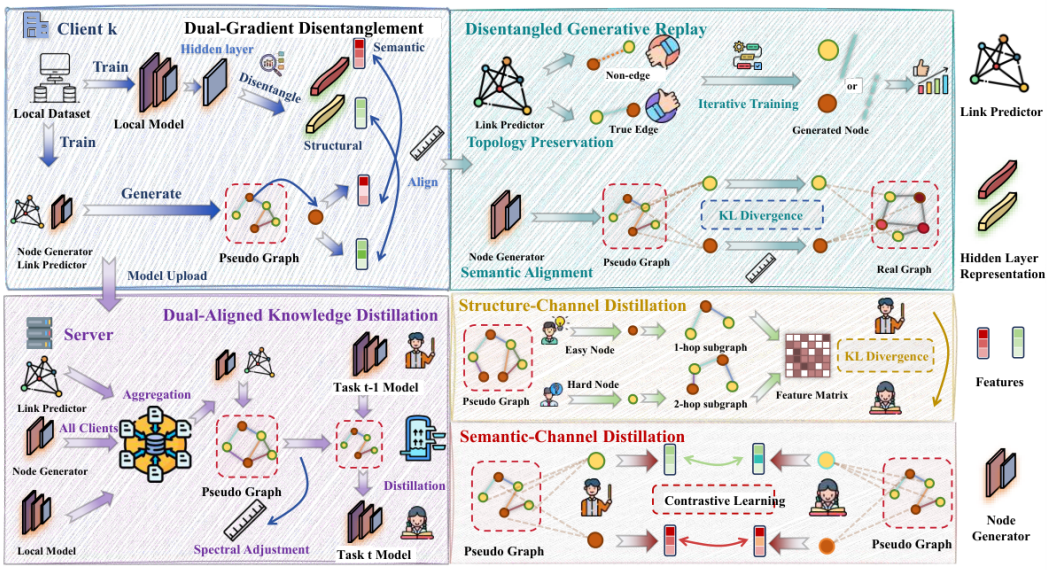

Figure 2: Overview of FedDiG. Post-hoc dual-gradient masking splits H into H_S/H_R; each task trains a generator and link predictor on the current graph; the server stores the calibrated synthetic subgraph and distills semantic and structural channels on the disjoint historical union.

Structural importance is not the complement of I^{sem}. On frozen H we train a small label-free probe g_ω (a two-layer MLP) that reconstructs adjacency from observed edges $\mathcal{E}^+_{\text{str}}$ and sampled non-edges $\mathcal{E}^-_{\text{str}}$; only ω is trained, because g_ω operates on frozen hidden representations solely for channel attribution, whereas LP_ψ operates as the topology prior for synthetic graph generation:

$$q_{ij} = \sigma\big(g_\omega([h_i \| h_j])\big), \qquad \mathcal{L}_{\text{str}} = -\frac{1}{|\mathcal{E}^+_{\text{str}}|} \sum_{(i,j)\in\mathcal{E}^+_{\text{str}}} \log q_{ij} - \frac{1}{|\mathcal{E}^-_{\text{str}}|} \sum_{(i,j)\in\mathcal{E}^-_{\text{str}}} \log(1-q_{ij}). \tag{3}$$

Here σ is the sigmoid; later KL terms use $\mathrm{softmax}$. After freezing g_ω,

$$I^{\text{str}}_d = \frac{1}{N}\sum_{i=1}^{N} \left|\partial \mathcal{L}_{\text{str}}/\partial H_{i,d}\right|, \qquad R_d = \log(I^{\text{sem}}_d + \epsilon_0) - \log(I^{\text{str}}_d + \epsilon_0). \tag{4}$$

A mask M keeps the top-r coordinates of R_d as $H_S = H \odot M$ and the rest as $H_R = H \odot (1-M)$. Class labels affect only I^{sem} through cross-entropy; I^{str} is computed from observed edges and sampled non-edges rather than same-label pairs, so structural attribution does not assume homophily. Before distillation the server recomputes the same scores from teacher f_{θ^-} on $\widetilde{G}^{1:t-1}$. H_S is aligned by $\mathcal{L}_{\text{align}}/\mathcal{L}_{\text{contrast}}$ and H_R by $\mathcal{L}_{\text{sim}}$.

3.3 Disentangled Generative Replay (Client Side)

Each task trains a fresh generator G_{ϕ_t} and link predictor LP_{ψ_t} on $G_{k,t}$ only. Clients upload $(\phi_t, \psi_t, R^{*(k)}_{\text{hf}})$; the server aggregates them, materializes $\widetilde{G}_t$, and stores $\mathcal{B}_t = \{\widetilde{G}_1, \ldots, \widetilde{G}_t\}$. Distillation uses the disjoint union $\widetilde{G}^{1:t-1} = \bigsqcup_{t'<t} \widetilde{G}_{t'}$, so old relational patterns are replayed without generator forgetting.

Generative Schema & Topology Inference. The generator takes a noise vector $z \sim \mathcal{N}(0, I)$ and a class embedding $e_c = \text{Emb}(c)$ as input. It outputs normalized node features $\tilde{x}$:

$$\tilde{h} = G_\phi([z \| e_c]), \qquad \tilde{x} = \frac{\tilde{h}}{\|\tilde{h}\|_2}. \tag{5}$$

Instead of directly outputting edges, we employ a learnable Link Predictor (LP) parameterized by ψ, trained on $G_{k,t}$ with binary cross-entropy over observed edges and an equal number of sampled non-edges. The edge probability p_{ij} between generated nodes $\tilde{x}_i$ and $\tilde{x}_j$ is computed via a non-linear MLP:

$$p_{ij} = \sigma\big(W_2 \cdot \mathrm{ReLU}(W_1[\tilde{x}_i \| \tilde{x}_j] + b_1) + b_2\big). \tag{6}$$

We symmetrize and threshold the probabilities to obtain the binary adjacency matrix $\widetilde{A}$:

$$P_{ij} = \frac{p_{ij} + p_{ji}}{2}, \qquad \widetilde{A}_{ij} = \mathbf{1}[P_{ij} > \tau_{\mathrm{lp}}]. \tag{7}$$

Disentangled Optimization Objectives. We optimize the generator with two key objectives in addition to classification on generated samples.

Semantic Alignment ($\mathcal{L}_{\mathrm{align}}$). This term enforces prototype consistency only in the semantic subspace. With θ frozen, we encode a generated graph $\widetilde{G}_{k,t} = (\widetilde{X}, \widetilde{A}, \widetilde{Y})$ with the same GNN used on $G_{k,t}$, then apply the local mask M_k:

$$\begin{aligned} H^{\mathrm{syn}} &= f_\theta^{\mathrm{hid}}(\widetilde{G}_{k,t}), \qquad H_S^{\mathrm{syn}} = H^{\mathrm{syn}} \odot M_k, \\ p_{c,S}^{\mathrm{real}} &= \mathrm{mean}\{h_{i,S} : y_i = c\}, \qquad p_{c,S}^{\mathrm{syn}} = \mathrm{mean}\{h_{i,S}^{\mathrm{syn}} : \tilde{y}_i = c\}, \\ \mathcal{L}_{\mathrm{align}} &= \sum_{c \in \mathcal{C}_{k,t}} \mathrm{KL}\Big(\mathrm{softmax}(p_{c,S}^{\mathrm{real}}) \,\Big\|\, \mathrm{softmax}(p_{c,S}^{\mathrm{syn}})\Big), \end{aligned} \tag{8}$$

where $\mathcal{C}_{k,t}$ is the set of classes observed on $G_{k,t}$. $\mathcal{L}_{\mathrm{ce}}^{\mathrm{gen}}$ is cross-entropy on $\mathrm{Clf}(H_S^{\mathrm{syn}})$ with θ frozen.

Topology Preservation ($\mathcal{L}_{\mathrm{topo}}$). With ψ frozen, $\mathcal{L}_{\mathrm{topo}}$ transfers the LP prior—learned from real edges vs. non-edges on $G_{k,t}$—by BCE on generated pairs ranked by p_{ij}: the top-m form $\mathcal{E}^+$ and the bottom-m form $\mathcal{E}^-$. Pairs are not selected by class equality, so the objective does not impose a same-label-implies-edge bias on heterophilic graphs:

$$\mathcal{L}_{\mathrm{topo}} = -\mathbb{E}_{(i,j)\in\mathcal{E}^+}[\log p_{ij}] - \mathbb{E}_{(i,j)\in\mathcal{E}^-}\big[\log(1 - p_{ij})\big]. \tag{9}$$

Total Objective.

$$\mathcal{L}_{\mathrm{gen}} = \lambda_{\mathrm{cls}}\mathcal{L}_{\mathrm{ce}}^{\mathrm{gen}} + \lambda_{\mathrm{align}}\mathcal{L}_{\mathrm{align}} + \lambda_{\mathrm{topo}}\mathcal{L}_{\mathrm{topo}}. \tag{10}$$

3.4 Dual-Aligned Knowledge Distillation (Server Side)

Global Parameter Aggregation. Before synthesis, the server aggregates the current-task generator, link predictor, and roughness target from K active clients. Let $n_k = |V_{k,t}|$ and $N_{\mathrm{total}} = \sum_{k=1}^K n_k$. Then

$$\phi_t^{(g)} = \sum_{k=1}^K \frac{n_k}{N_{\mathrm{total}}}\phi_t^{(k)}, \quad \psi_t^{(g)} = \sum_{k=1}^K \frac{n_k}{N_{\mathrm{total}}}\psi_t^{(k)}, \quad R_{\mathrm{hf}}^* = \sum_{k=1}^K \frac{n_k}{N_{\mathrm{total}}}R_{\mathrm{hf}}^{*(k)}. \tag{11}$$

G_ϕ maps (z, e_c) to input-space features and does not contain M_k; the mask is a frozen readout in $\mathcal{L}_{\mathrm{align}}$ only and is not uploaded. We therefore average (ϕ, ψ) even when local M_k differ. A client missing class c still contributes the shared trunk; its unused class embedding is left unchanged. Distillation later recomputes one server mask from f_{θ^-} on $\widetilde{G}^{1:t-1}$.

Per-task Synthesis & Spectral Alignment. Using $G_{\phi_t^{(g)}}$ and $\mathrm{LP}_{\psi_t^{(g)}}$, the server synthesizes $\widetilde{G}_t = (\widetilde{X}_t, \widetilde{A}_t, \widetilde{Y}_t)$, appends it to $\mathcal{B}_t$, and—for $t > 1$—updates f_θ on the disjoint historical union $\widetilde{G}^{1:t-1}$. We calibrate $\widetilde{G}_t$ with a high-frequency roughness score that anchors structural sharpness to the client-averaged target. We define R_{hf} based on the Graph Laplacian $L = D - \widetilde{A}$:

$$R_{\mathrm{hf}}(\widetilde{G}) = \lambda_x \left(\frac{1}{|\mathcal{F}|} \sum_{f \in \mathcal{F}} \frac{\tilde{x}_f^\top L \tilde{x}_f}{\tilde{x}_f^\top \tilde{x}_f} \right) + \lambda_D \bar{d}, \tag{12}$$

where $\bar{d}$ is the average degree. Each client evaluates the same score on $G_{k,t}$ to obtain the local target $R_{\mathrm{hf}}^{*(k)}$. Edge deletion cannot raise R_{hf} (both the Rayleigh term and $\bar{d}$ are nonincreasing), so we

delete edges when $R_{\mathrm{hf}} > R^*_{\mathrm{hf}}$ and add edges when $R_{\mathrm{hf}} < R^*_{\mathrm{hf}}$, preferring homophilic deletions and heterophilic additions, until $|R_{\mathrm{hf}} - R^*_{\mathrm{hf}}| \le \varepsilon$.

Classification on Synthesized Data ($\mathcal{L}_{\mathrm{ce}}^{\mathrm{syn}}$). The student f_θ (also written f_t) is supervised to classify synthesized samples using only the semantic subspace features $h_S^{(s)}$.

$$\mathcal{L}_{\mathrm{ce}}^{\mathrm{syn}} = \mathbb{E}_{(\tilde{x},\tilde{y})}\Big[\mathrm{CE}\big(\mathrm{Clf}(h_S^{(s)}), \tilde{y}\big)\Big]. \tag{13}$$

Semantic-Channel Distillation ($\mathcal{L}_{\mathrm{contrast}}$). Let f_{θ^-} be the frozen teacher (the snapshot after the previous task; also written f_{t-1}) and f_θ the student (f_t). Teacher prototypes are class means of teacher semantic features on $\widetilde{G}^{1:t-1}$:

$$\begin{aligned} P_c^{(t-1)} &= \mathrm{mean}\{h_{i,S}^{(t-1)} : \tilde{y}_i = c\}, \\ \mathcal{L}_{\mathrm{contrast}} &= -\sum_i \log \frac{\exp\big(h_{i,S}^{(s)} (P_{\tilde{y}_i}^{(t-1)})^\top / \tau\big)}{\sum_c \exp\big(h_{i,S}^{(s)} (P_c^{(t-1)})^\top / \tau\big)}. \end{aligned} \tag{14}$$

Structural-Channel Distillation ($\mathcal{L}_{\mathrm{sim}}$). We distill relational reasoning patterns only on the structural channel H_R. We first identify hard nodes with high Jensen–Shannon divergence between Student and Teacher predictions:

$$\Omega = \{v \mid D_{\mathrm{JS}}(f_\theta(v) \| f_{\theta^-}(v)) > \epsilon\}. \tag{15}$$

For nodes in Ω, we construct pairwise Gram similarities $S_{ij}^{(m)} = (h_{i,R}^{(m)})^\top h_{j,R}^{(m)} / \tau'$ for $m \in \{t-1, t\}$ and minimize the row-wise distribution gap:

$$\mathcal{L}_{\mathrm{sim}} = \frac{1}{|\Omega|} \sum_{i \in \Omega} \mathrm{KL}\Big(\mathrm{softmax}(S_{i,:}^{(t-1)}) \,\Big\|\, \mathrm{softmax}(S_{i,:}^{(t)})\Big). \tag{16}$$

If $|\Omega| < 2$, we set $\mathcal{L}_{\mathrm{sim}} = 0$.

Total Objective. The final objective combines the above losses:

$$\mathcal{L}_{\mathrm{total}} = \lambda_{\mathrm{ce}} \mathcal{L}_{\mathrm{ce}}^{\mathrm{syn}} + \lambda_{\mathrm{contrast}} \mathcal{L}_{\mathrm{contrast}} + \lambda_{\mathrm{sim}} \mathcal{L}_{\mathrm{sim}}. \tag{17}$$

This KD step is the only encoder update on historical knowledge. After synthesizing $\widetilde{G}_t$, if $t > 1$ the server minimizes $\mathcal{L}_{\mathrm{total}}$ on $\widetilde{G}^{1:t-1}$ for E_d epochs (default $E_d = 60$; Table S7), then sets $f_{\theta^-} \leftarrow f_\theta$ and broadcasts θ as the next-task initialization. Federated rounds on $G_{k,t}$ still use only $\mathcal{L}_{\mathrm{ce}}$.

4 Experiment

4.1 Experiment Setup

Datasets. We evaluate FedDiG on seven benchmark datasets with different scales and structural regimes. The suite includes citation networks (Cora, CiteSeer McCallum et al. (2000); Giles et al. (1998)) and co-purchase/co-authorship graphs (Computers, Physics Shchur et al. (2018)). It also includes heterophilic graphs (Squirrel, Roman-empire Pei et al. (2020); Platonov et al. (2023)) and Years (ArXiv-year; Lim et al. (2021)), derived from ogbn-arxiv Hu et al. (2020).

Baselines. We compare FedDiG with sixteen representative baselines in four families (Table 1): federated graph learning (FedAvg, FedSage+, FedPub, FedGTA); graph continual learning adapted to the federated setting (Fed-DSLR, Fed-TWP, Fed-SSM, Fed-PromptCGL, Fed-TPP, Fed-TACO); federated continual learning (GLFC, TARGET, Lander); and federated continual graph learning (POWER, MOTION, FedFST). All methods use the same client participation mode, communication rounds, and local epochs.

Implementation details. Following prior work Huang et al. (2023); Zhang et al. (2021), we use Louvain partitioning Blondel et al. (2008) to simulate non-IID client distributions and a $60\%/20\%/20\%$ train/val/test split. All clients participate in every round and the server aggregates after each round; the generator stage also uses full participation. Task blocks are contiguous original class IDs

Table 1: Performance comparison of our method and baselines (%). AA↑: Average Accuracy, AF↓: Average Forgetting. Rows report mean±std over 5 seeds. Best results are in **bold** and second-best are underlined.

Dataset	Cora		CiteSeer		Computers		Physics		Squirrel		Years		Roman-empire		Avg	
Method	AA↑	AF↓	AA↑	AF↓	AA↑	AF↓	AA↑	AF↓	AA↑	AF↓	AA↑	AF↓	AA↑	AF↓	AA↑	AF↓
FedAvg [AISTATS17]	18.2 ±1.2	96.4 ±1.6	18.4 ±0.9	83.5 ±2.1	27.9 ±1.5	98.4 ±0.9	47.7 ±2.3	98.9 ±0.8	27.9 ±1.1	56.5 ±3.4	28.8 ±0.9	64.4 ±2.7	18.8 ±1.4	69.9 ±2.8	26.8 ±0.9	81.1 ±1.4
FedSage+ [NeurIPS21]	17.5 ±1.0	96.0 ±1.8	29.7 ±2.2	84.7 ±1.5	15.9 ±1.7	90.3 ±2.1	27.9 ±1.3	98.9 ±0.6	28.4 ±1.9	60.6 ±3.2	28.1 ±1.1	64.4 ±2.3	19.1 ±1.5	73.2 ±3.0	23.8 ±1.1	81.2 ±1.6
FedPub [ICML23]	17.1 ±1.1	78.0 ±2.6	17.6 ±0.7	64.9 ±3.2	20.5 ±1.9	87.3 ±2.0	27.1 ±0.9	98.4 ±0.5	23.7 ±2.0	59.2 ±2.9	27.4 ±1.3	62.4 ±3.1	17.3 ±1.4	44.2 ±2.7	21.5 ±1.0	70.6 ±1.8
FedGTA [PVLDB23]	24.5 ±1.6	95.8 ±1.2	26.6 ±1.8	82.8 ±2.5	27.9 ±2.1	97.9 ±1.2	49.4 ±2.7	96.2 ±1.8	29.7 ±1.5	56.8 ±3.2	28.5 ±1.2	64.1 ±2.5	14.1 ±1.8	70.4 ±2.4	28.7 ±1.2	80.6 ±1.7
Fed-DSLR [WWW24]	65.8 ±3.4	42.4 ±4.2	43.1 ±2.8	41.0 ±3.6	57.0 ±4.2	49.5 ±3.9	25.4 ±1.5	33.3 ±1.1	30.1 ±2.2	21.5 ±4.3	38.0 ±2.5	24.6 ±3.7	32.6 ±2.9	43.0 ±3.4	41.7 ±2.4	36.5 ±2.8
Fed-TWP [AAAI21]	38.7 ±2.2	**18.8** ±0.9	29.4 ±1.9	20.3 ±0.8	44.2 ±3.0	10.4 ±0.9	30.2 ±2.5	51.7 ±3.7	30.0 ±1.7	**6.2** ±1.2	34.1 ±2.1	14.9 ±2.1	13.1 ±1.5	12.2 ±2.3	31.4 ±1.6	19.2 ±1.4
Fed-SSM [ICDM22]	68.1 ±3.3	21.6 ±2.1	70.5 ±3.5	19.6 ±2.1	51.2 ±3.9	36.8 ±3.2	29.4 ±2.2	44.5 ±3.1	30.5 ±2.1	24.1 ±3.5	34.8 ±2.4	23.6 ±3.2	27.2 ±2.5	39.4 ±3.7	44.5 ±2.2	29.9 ±2.9
Fed-PromptCGL [SCIS26]	31.3 ±2.5	94.0 ±1.9	42.3 ±2.7	80.0 ±2.9	23.6 ±1.7	92.4 ±1.5	23.1 ±1.4	97.1 ±0.9	25.4 ±2.2	51.9 ±3.2	26.8 ±1.8	71.3 ±2.7	16.9 ±1.6	76.2 ±2.4	27.1 ±1.7	80.4 ±2.0
Fed-TPP [NeurIPS24]	62.1 ±3.1	38.2 ±3.9	61.6 ±3.4	34.8 ±3.2	48.3 ±3.7	40.5 ±3.5	32.6 ±2.5	47.2 ±3.3	26.7 ±2.1	9.1 ±1.2	33.5 ±2.2	26.8 ±2.9	22.1 ±2.4	36.9 ±3.1	41.0 ±2.1	33.4 ±2.5
Fed-TACO [NeurIPS24]	71.0 ±3.5	30.1 ±3.2	58.8 ±3.2	48.9 ±3.7	52.8 ±3.9	43.6 ±3.5	36.5 ±2.9	51.8 ±3.6	30.6 ±2.2	32.1 ±3.1	35.6 ±2.5	29.2 ±2.9	29.8 ±2.7	42.5 ±3.4	45.0 ±2.3	39.7 ±2.7
GLFC [CVPR22]	20.2 ±1.7	63.7 ±3.3	10.3 ±1.4	65.7 ±2.9	13.8 ±0.9	89.8 ±1.7	24.6 ±0.7	31.8 ±0.8	29.4 ±2.3	13.4 ±2.6	37.4 ±1.9	21.7 ±3.2	12.8 ±1.0	37.6 ±2.5	21.2 ±1.5	46.2 ±2.1
TARGET [ICCV23]	19.5 ±1.4	66.5 ±2.8	19.4 ±1.2	68.3 ±3.4	13.1 ±0.7	93.2 ±1.6	26.7 ±1.5	98.8 ±0.7	28.8 ±1.8	55.7 ±3.6	27.5 ±1.0	64.3 ±2.5	12.2 ±0.8	44.3 ±2.2	21.0 ±0.8	70.2 ±1.9
Lander [CVPR24]	32.9 ±2.5	96.2 ±1.4	18.9 ±1.6	82.7 ±2.6	28.1 ±2.3	97.9 ±0.9	38.3 ±2.8	99.2 ±0.6	26.6 ±1.9	56.8 ±3.1	28.4 ±1.5	64.4 ±2.8	14.8 ±1.9	67.5 ±2.7	26.9 ±1.2	80.7 ±1.5
POWER [KDD25]	56.3 ±2.3	57.6 ±2.4	54.8 ±2.0	47.3 ±2.8	31.4 ±1.9	49.8 ±2.5	67.3 ±3.5	39.7 ±3.9	26.8 ±1.9	27.4 ±3.4	28.7 ±1.4	35.9 ±2.6	26.3 ±2.4	19.6 ±2.8	41.7 ±1.8	39.6 ±2.2
MOTION [NeurIPS25]	61.2 ±3.1	35.6 ±3.6	57.3 ±2.4	38.9 ±2.8	49.6 ±3.8	44.2 ±2.2	63.7 ±2.9	33.4 ±4.1	29.4 ±1.8	25.7 ±2.4	34.1 ±2.3	21.6 ±3.3	27.6 ±2.1	21.8 ±1.9	46.1 ±2.2	31.6 ±2.7
FedFST [KDD26]	69.7 ±3.1	23.3 ±2.2	66.5 ±3.3	**12.6** ±1.7	51.8 ±3.5	32.6 ±3.1	61.2 ±3.2	38.4 ±3.7	29.6 ±2.5	18.7 ±2.9	33.4 ±2.4	19.2 ±2.6	**33.5** ±2.5	13.9 ±2.2	49.4 ±2.3	22.7 ±2.4
FedDiG	**75.6** ±4.3	20.3 ±2.1	**75.6** ±3.9	14.7 ±2.0	**58.1** ±3.5	**9.1** ±2.7	**70.1** ±3.9	**29.3** ±4.2	**32.1** ±2.4	11.1 ±1.9	**38.2** ±2.2	**12.4** ±2.5	**33.5** ±2.8	**6.8** ±1.6	**54.7** ±3.2	**14.8** ±2.0

$[(t-1)C_{\text{task}},\, tC_{\text{task}})$; leftover tail IDs $[C_{\text{eff}}, C)$ are discarded (Table S5). Each task is trained and evaluated on the subgraph of its class block, and inter-task edges are excluded for every method. Unless otherwise stated, we use $K = 3$ clients, 15 communication rounds, 5 local epochs, a two-layer GCN, $r = 50\%$, 80 synthetic nodes per class, and five seeds reported as mean±std (remaining defaults in Table S7).

4.2 Comparison with Baselines

Table 1 reports results on seven datasets, where AA and AF denote the average accuracy and average forgetting after the final task, respectively. Overall, FedDiG achieves the highest mean AA and the lowest mean AF across the seven benchmarks under the unified protocol.

A consistent trend appears across baseline families. Conventional federated graph learning methods (FedAvg McMahan et al. (2017), FedSage+ Zhang et al. (2021), FedPub Baek et al. (2023), FedGTA Li et al. (2023)) are built for stationary distributions and forget heavily when tasks evolve. Fed-DSLR Choi et al. (2024) is competitive in AA by replaying real nodes, whereas FedDiG rehearses on stored synthetic subgraphs. Fed-TWP Liu et al. (2021) attains low AF via parameter regularization, but its rigid constraints limit plasticity. Image-oriented federated continual methods (GLFC Dong et al. (2022), TARGET Zhang et al. (2023), Lander Tran et al. (2024)) lack graph-aware mechanisms. Dedicated FCGL baselines POWER Zhu et al. (2025), MOTION Wan et al. (2025), and FedFST Guo et al. (2026) improve over generic alternatives, but they do not explicitly disentangle semantic and structural information. FedDiG achieves the highest mean AA and the lowest mean AF across the seven benchmarks under the unified protocol.

4.3 Ablation Study

Each ablation disables one term ($\mathcal{L}_{\text{align}}$, $\mathcal{L}_{\text{topo}}$, $\mathcal{L}_{\text{contrast}}$, or $\mathcal{L}_{\text{sim}}$) by setting its coefficient to zero, while the federated schedule remains unchanged. When $\mathcal{L}_{\text{topo}}$ is removed, link-predictor training is also omitted. All ablation results are reported as mean±std over 5 seeds in Table 2; mask-ratio and split controls are in Table 3.

Table 2: Ablation study results (%) on Cora, CiteSeer and Years datasets (mean±std over 5 seeds), where the best and second-best results are highlighted in **bold** and underline.

	Cora		CiteSeer		Years	
Ablation	AA↑	AF↓	AA↑	AF↓	AA↑	AF↓
w/o $\mathcal{L}_{\text{align}}$	$64.6_{\pm 2.9}$	$23.8_{\pm 1.6}$	$57.4_{\pm 3.1}$	$19.3_{\pm 2.4}$	$32.8_{\pm 1.9}$	$15.1_{\pm 1.9}$
w/o $\mathcal{L}_{\text{topo}}$	$58.4_{\pm 3.5}$	$27.4_{\pm 2.2}$	$56.8_{\pm 3.0}$	$35.4_{\pm 3.7}$	$28.5_{\pm 2.1}$	$17.6_{\pm 2.5}$
w/o $\mathcal{L}_{\text{contrast}}$	$\underline{71.4_{\pm 3.2}}$	$\underline{21.9_{\pm 1.9}}$	$65.5_{\pm 2.7}$	$24.2_{\pm 2.6}$	$\underline{34.2_{\pm 1.9}}$	$\underline{14.2_{\pm 2.2}}$
w/o $\mathcal{L}_{\text{sim}}$	$66.1_{\pm 2.9}$	$25.0_{\pm 2.1}$	$\underline{71.6_{\pm 3.5}}$	$\underline{16.9_{\pm 1.8}}$	$31.9_{\pm 2.4}$	$15.3_{\pm 2.3}$
FedDiG	$\mathbf{75.6}_{\pm 4.3}$	$\mathbf{20.3}_{\pm 2.1}$	$\mathbf{75.6}_{\pm 3.9}$	$\mathbf{14.7}_{\pm 2.0}$	$\mathbf{38.2}_{\pm 2.2}$	$\mathbf{12.4}_{\pm 2.5}$

Table 3: Sensitivity to the mask ratio r (left) and the channel split (right) (%).

	Cora		CiteSeer		Squirrel	
Mask r	AA↑	AF↓	AA↑	AF↓	AA↑	AF↓
25%	68.3	34.7	70.5	24.6	28.6	17.5
50%	**75.6**	**20.3**	**75.6**	**14.7**	**32.1**	**11.1**
75%	64.1	39.8	64.2	30.1	28.0	18.3

	Cora		CiteSeer		Squirrel	
Setting	AA↑	AF↓	AA↑	AF↓	AA↑	AF↓
w/o split	66.8	35.2	68.5	27.4	26.4	20.3
I^{sem} only	72.3	25.7	71.8	20.6	29.4	15.8
Random mask	54.3	51.7	59.2	40.6	22.8	28.9
FedDiG	**75.6**	**20.3**	**75.6**	**14.7**	**32.1**	**11.1**

Table 2 separates two complementary sources of retention. Removing the topology branch causes the largest AA loss on all three datasets (Cora: 75.6 → 58.4, CiteSeer: 75.6 → 56.8, Years: 38.2 → 28.5) and more than doubles CiteSeer AF. Because this row also omits LP training, Table S4 isolates the two switches: removing only $\mathcal{L}_{\text{topo}}$ gives Cora/CiteSeer AA 65.2/65.5, while removing only the LP gives 67.2/63.4. Both exceed the complete-branch ablation, showing that topology supervision and transferred link prediction are complementary. Semantic objectives are also necessary: without $\mathcal{L}_{\text{align}}$, AA falls to 64.6/57.4/32.8; $\mathcal{L}_{\text{contrast}}$ provides additional, dataset-dependent reinforcement.

Mask-ratio sensitivity. Table 3 (left) exhibits a consistent balance: allocating 25% of dimensions to H_S weakens semantic capacity, whereas 75% leaves too little capacity for H_R and produces a larger Cora penalty (64.1 AA, 39.8 AF). CiteSeer and heterophilic Squirrel show the same rise-then-fall pattern, indicating that both channels require sufficient capacity.

Necessity of the dual-gradient mask. Table 3 (right) distinguishes specialization from an architectural effect. On Cora, I^{sem} alone improves w/o split from 66.8/35.2 to 72.3/25.7, confirming that semantic gradients locate class-discriminative coordinates. Its remaining gap to FedDiG (75.6/20.3), together with the poor random-mask result (54.3/51.7), supports the additional contribution of the structural gradient in identifying the complementary relational channel. The same ordering holds on CiteSeer and Squirrel.

Loss-weight sensitivity. Figure 3 sweeps λ_{align} and $\lambda_{\text{contrast}}$ over $\{0.1, 1, 5, 10, 20\}$ and $\lambda_{\text{topo}}/\lambda_{\text{sim}}$ over $\{0.001, 0.01, 0.05, 0.1, 1\}$ on Cora.

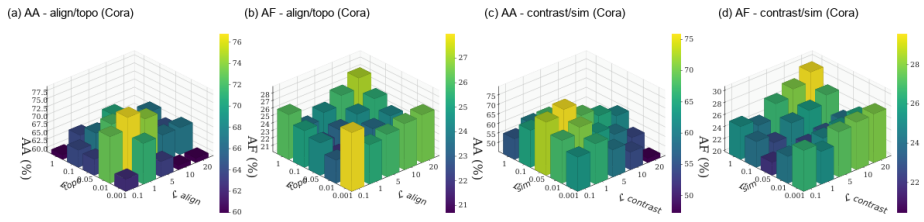

Figure 3: Loss-weight sensitivity on Cora. Defaults in Table S7 lie in the jointly competitive region.

Figure 3 shows a broad high-AA region at moderate λ_{align} and small-to-moderate λ_{topo}. Stronger topology regularization can reduce AF but eventually lowers AA, exposing the stability–plasticity trade-off. The flatter $\lambda_{\text{contrast}}$–$\lambda_{\text{sim}}$ surface around $(0.1, 0.05)$ places the shared defaults in a robust region rather than at an isolated, dataset-specific peak.

Channel probes. Figure 4 tests whether the mask separates class and structure.

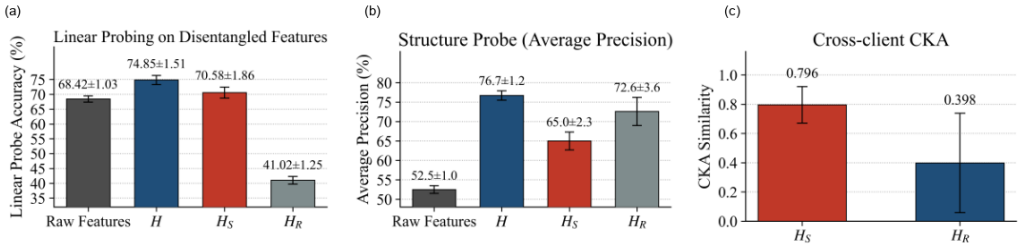

Figure 4: Channel probes: (a) class linear probe on Cora; (b) structure probe on Squirrel (AP); (c) cross-client centered kernel alignment (CKA) on Cora.

Figure 4 confirms specialization rather than perfect isolation. H_S retains most of the class information in H (70.58% versus 74.85%), whereas H_R drops to 41.02%. The ordering reverses for the structure probe: H_R preserves 72.6% AP, close to the full representation (76.7%), while H_S reaches 65.0%. Moreover, cross-client CKA is twice as high for H_S as for H_R (0.796 vs. 0.398), suggesting greater cross-client consistency in H_S than in H_R, consistent with topology-specific variation across clients. These results directly support applying alignment/contrastive losses to H_S and topology/similarity losses to H_R.

Robustness and retention. Figure 5 tests the split under perturbation and task drift.

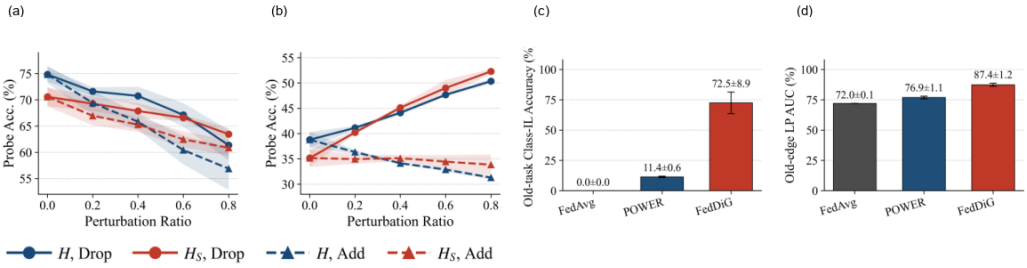

Figure 5: Robustness and retention: edge-drop/add tests on (a) Cora and (b) Roman-empire; (c–d) old-class accuracy and old-edge AUC on Cora.

Figure 5 connects this specialization to continual retention. As the perturbation ratio increases, H_S has a smaller accuracy decline than the full H under both edge deletion and addition; the advantage is clearest for added edges, whose noisy messages directly contaminate an entangled representation. This robustness transfers to old tasks: FedDiG retains 72.5% old-class accuracy versus 11.4% for POWER and 0.0% for FedAvg, while old-edge AUC rises from 72.0%/76.9% to 87.4%. The simultaneous gains on class and edge retention support the central claim that protecting one channel does not require sacrificing the other.

5 Conclusion and Discussion

Summary. FedDiG separates semantic and structural channels through post-hoc dual-gradient masking and performs dual-channel distillation on task-wise synthetic subgraphs. Across seven benchmarks, it achieves the highest mean AA and the lowest mean AF under the unified protocol. Future work will explore dynamic channel separation.

Reproducibility Statement

We provide the training protocol, metrics, dataset construction rules, hyperparameter analyses, and method pseudocode in the appendix. All reported main results use the same federated protocol across methods, including Louvain client partitioning, sequential task blocks, $K = 3$ clients, 15 communication rounds, 5 local epochs, a two-layer GCN backbone unless otherwise stated, and five predefined random seeds. Additional experiments report client-scale sweeps, partial client participation, and fine-grained ablations. Mask-ratio sensitivity is reported in Table 3. Default loss weights, replay budget, and generator architecture are listed in Table S7.

AI Use Statement

In this work, we used generative AI tools (large language models) to draft and revise manuscript text, improve readability, format LaTeX tables and references, search and organize related work, and assist with code editing. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

References

Jinheon Baek, Wonyong Jeong, Jiongdao Jin, Jaehong Yoon, and Sung Ju Hwang. Personalized subgraph federated learning. In *ICML*, pp. 1396–1415. PMLR, 2023.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, 2008.

Seungyoon Choi, Wonjoong Kim, Sungwon Kim, Yeonjun In, Sein Kim, and Chanyoung Park. DSLR: Diversity enhancement and structure learning for rehearsal-based graph continual learning. In *WWW*, pp. 733–744, 2024.

Jiahua Dong, Lixu Wang, Zhen Fang, Gan Sun, Shichao Xu, Xiao Wang, and Qi Zhu. Federated class-incremental learning. In *CVPR*, pp. 10164–10173, 2022.

Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S Yu. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *CIKM*, pp. 315–324, 2020.

Xingbo Fu, Binchi Zhang, Yushun Dong, Chen Chen, and Jundong Li. Federated graph machine learning: A survey of concepts, techniques, and applications. *ACM SIGKDD Explorations*, 24(2): 32–47, 2022.

C Lee Giles, Kurt D Bollacker, and Steve Lawrence. CiteSeer: An automatic citation indexing system. In *ACM DL*, pp. 89–98, 1998.

Hanyao Guo, Zihan Tan, Wenke Huang, Bin Yang, and Mang Ye. FedFST: Mitigating spectral catastrophic forgetting in federated graph continual learning. In *KDD*, pp. 1404–1414, 2026. doi: 10.1145/3770855.3817730.

Xiaoxue Han, Zhuo Feng, and Yue Ning. A topology-aware graph coarsening framework for continual graph learning. In *NeurIPS*, volume 37, 2024.

Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang, Han Xie, Lichao Sun, Lifang He, Liangwei Yang, Philip S Yu, Yu Rong, et al. FedGraphNN: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145*, 2021.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. LightGCN: Simplifying and powering graph convolution network for recommendation. In *SIGIR*, pp. 639–648, 2020.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS*, volume 33, pp. 22118–22133, 2020.

Wenke Huang, Guancheng Wan, Mang Ye, and Bo Du. Federated graph semantic and structural learning. In *IJCAI*, pp. 3830–3838, 2023.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13):3521–3526, 2017.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *MLSys*, volume 2, pp. 429–450, 2020.

Xunkai Li, Zhengyu Wu, Wentao Zhang, Yinlin Zhu, Rong-Hua Li, and Guoren Wang. FedGTA: Topology-aware averaging for federated graph learning. *PVLDB*, 17(1):41–50, 2023.

Derek Lim, Xiuyu Li, Felix Hohne, and Ser-Nam Lim. New benchmarks for learning on non-homophilous graphs. In *Workshop on Graph Learning Benchmarks, WWW*, 2021.

Huihui Liu, Yiding Yang, and Xinchao Wang. Overcoming catastrophic forgetting in graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8653–8661, 2021. doi: 10.1609/aaai.v35i10.17049.

Rui Liu, Pengwei Xing, Zichao Deng, Anran Li, Cuntai Guan, and Han Yu. Federated graph neural networks: Overview, techniques, and challenges. *IEEE TNNLS*, 36(3):4279–4295, 2025.

Ziqi Liu, Chaochao Chen, Xinxing Yang, Jun Zhou, Xiaolong Li, and Le Song. Heterogeneous graph neural networks for malicious account detection. In *CIKM*, pp. 2077–2085, 2018.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *NeurIPS*, volume 30, 2017.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 2017.

Chaoxi Niu, Guansong Pang, Ling Chen, and Bing Liu. Replay-and-forget-free graph class-incremental learning: A task profiling and prompting approach. In *NeurIPS*, volume 37, 2024.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-GCN: Geometric graph convolutional networks. In *ICLR*, 2020.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *ICLR*, 2023.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

Jonathan M Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M Donghia, Craig R MacNairy, Shawn French, Lindsey A Carfrae, Zohar Bloom-Ackermann, et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702, 2020.

Minh-Tuan Tran, Trung Le, Xuan-May Le, Mehrtash Harandi, and Dinh Phung. Text-enhanced data-free approach for federated class-incremental learning. In *CVPR*, pp. 23870–23880, 2024.

Frank Wan, Fengyuan Ran, Ruikang Zhang, Wenke Huang, Xuankun Rong, Guibin Zhang, Yuxin Wu, Bo Du, and Mang Ye. MOTION: Multi-sculpt evolutionary coarsening for federated continual graph learning. In *NeurIPS*, pp. 78309–78333, 2025.

Guancheng Wan, Wenke Huang, and Mang Ye. Federated graph learning under domain shift with generalizable prototypes. In *AAAI*, 2024.

Qi Wang, Rui Mao, Ye Yuan, and Tianfei Zhou. Prompt-driven continual graph learning. *Science China Information Sciences*, 2026.

Han Xie, Jing Ma, Li Xiong, and Carl Yang. Federated graph classification over non-IID graphs. In *NeurIPS*, volume 34, pp. 18839–18852, 2021.

Jie Zhang, Chen Chen, Weiming Zhuang, and Lingjuan Lyu. TARGET: Federated class-continual learning via exemplar-free distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4782–4793, 2023.

Ke Zhang, Carl Yang, Xiaoxiao Li, Lichao Sun, and Siu Ming Yiu. Subgraph federated learning with missing neighbor generation. In *NeurIPS*, volume 34, pp. 6671–6682, 2021.

Xikun Zhang, Dongjin Song, and Dacheng Tao. Sparsified subgraph memory for continual graph representation learning. In *ICDM*, pp. 1335–1340. IEEE, 2022.

Fan Zhou and Chengtai Cao. Overcoming catastrophic forgetting in graph neural networks with experience replay. In *AAAI*, volume 35, pp. 4714–4722, 2021.

Yinlin Zhu, Miao Hu, and Di Wu. Federated continual graph learning. In *KDD*, pp. 4203–4213, 2025.

A Communication Efficiency

Table S1 reports communication and wall-clock cost on Cora (3 clients, 2-layer GCN).

Table S1: Communication and computation overhead on Cora. Training time is total wall-clock.

Method	Comm./round	Training Time (s)	AA (%)
FedAvg	2.11 MB	15.5	18.2
Fed-TWP	2.11 MB	21.1	38.7
POWER	2.49 MB	202.0	56.3
FedDiG	3.12 MB	268.7	75.6

Communication. The extra 1.01 MB over FedAvg comes from uploading ϕ, ψ, and R^*_{hf} in the generator stage.

Computation. FedDiG is $1.33\times$ POWER on Cora. Table S2 attributes almost all of the extra time to generator training; G_{ϕ_t} is trained once per task, not every round.

Table S2: Runtime breakdown on Cora.

Phase	FedDiG	FedAvg
Local train + aggregation	10.38 s	9.41 s
Generator training	220.61 s	0 s
Spectral/graph adjustment	33.66 s	0 s
Other	4.05 s	6.09 s
Total wall time	268.7 s	15.5 s

B Hyperparameters Analysis

The main-paper heatmaps (Figure 3) sweep λ_{align} and $\lambda_{\text{contrast}}$ over $\{0.1, 1, 5, 10, 20\}$ and $\lambda_{\text{topo}}/\lambda_{\text{sim}}$ over $\{0.001, 0.01, 0.05, 0.1, 1\}$ on Cora. Instead of a single global trend, different losses affect performance in different ways.

Semantic Alignment ($\mathcal{L}_{\text{align}}$). This term mainly affects class discrimination in the semantic channel. A moderate increase improves AA because replayed samples carry clearer class cues, while an overly strong setting reduces replay diversity and weakens adaptation to new tasks.

Topology Preservation ($\mathcal{L}_{\text{topo}}$). This term directly controls whether replay preserves local connectivity and neighborhood relations. Strengthening it usually improves retention (lower AF), but excessive emphasis shifts optimization toward structure reconstruction and can reduce plasticity.

Semantic-Channel Distillation ($\mathcal{L}_{\text{contrast}}$). This term aligns student semantic features on replayed historical samples with the frozen teacher prototypes of their classes.

Structural-Channel Distillation ($\mathcal{L}_{\text{sim}}$). This term preserves relational geometry between samples during knowledge transfer. With moderate strength, cross-task transfer is more stable. When this term is too strong, optimization may over-rely on neighbor-driven relations, which can blur semantic class boundaries and reduce final AA.

The selected settings in Figure 3 come from a region where AA and AF are both competitive. The corresponding default coefficients are listed in Table S7.

C Additional Experimental Results

We further report client-scale sweeps, partial participation, and more fine-grained ablations.

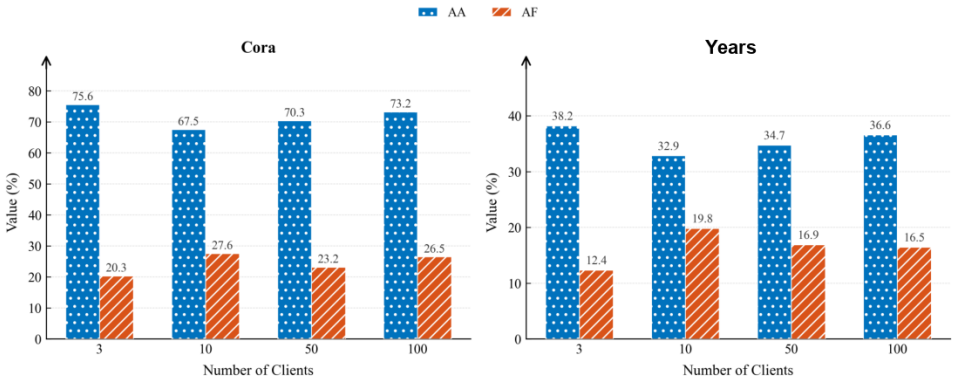

Figure 6: Client-scale ablation on Cora and Years.

Table S3: Partial-participation results on Cora under a 10-client split (%).

	FedDiG		Fed-TPP		POWER		Fed-DSLR	
Active clients	AA	AF	AA	AF	AA	AF	AA	AF
10/10	67.5	27.6	61.2	35.9	52.8	52.3	64.5	47.5
7/10	65.1	29.6	43.5	49.8	50.1	55.1	62.3	46.5
5/10	63.4	31.7	36.1	45.1	47.4	56.8	58.8	51.2
3/10	58.5	36.6	28.2	43.0	43.6	61.2	54.1	56.8

Table S3 shows that FedDiG remains the most accurate method as fewer clients participate in each round. Forgetting increases under severe participation sparsity, but the method degrades smoothly rather than collapsing.

Mask-ratio sensitivity for $r \in \{25\%, 50\%, 75\%\}$ is reported in Table 3 in the main paper. The default $r = 50\%$ remains the best setting on Cora, CiteSeer, and Squirrel.

Dual-gradient disentanglement (Table 3 in the main paper) is stronger than removing the split, using only I^{sem}, or using a random mask. The semantic-only improvement over no split shows that class gradients are informative, while its gap to the full method supports combining semantic and structural importance in R_d.

The main-paper row w/o $\mathcal{L}_{\text{topo}}$ (Table 2; Cora AA 58.4, CiteSeer 56.8) turns off the full topology branch: $\lambda_{\text{topo}} = 0$ and link-predictor training is skipped. The rows here isolate one switch: w/o $\mathcal{L}_{\text{topo}}$ only zeros the topology loss while keeping the link predictor, and w/o LP only removes the predictor. The milder Cora/CiteSeer drops (65.2 / 65.5) therefore do not contradict Table 2; they show that the topology branch is not explained by a single component. Cora is especially sensitive to hard-node selection, while CiteSeer degrades more when the link predictor is removed.

Table S4: Fine-grained component ablations (%). Each row disables one component; Table 2 removes the full topology branch.

	Cora		CiteSeer	
Variant	AA	AF	AA	AF
w/o spectral adjustment	70.4	26.4	68.0	23.5
w/o hard-node selection	64.8	32.7	71.6	20.2
w/o LP only	67.2	29.8	63.4	29.1
w/o $\mathcal{L}_{\text{topo}}$ only	65.2	25.1	65.5	21.3

D Dataset Statistics

Table S5 summarizes the raw graph statistics and task counts used in our experiments.

Table S5: Dataset statistics and task construction. Class IDs are the original 0-based labels; leftover tail IDs $[C_{\text{eff}}, C)$ are discarded.

Dataset	**#Nodes**	**#Edges**	**#Feat.**	**#Cls.**	C_{task}	T	**Dropped IDs**
Cora McCallum et al. (2000)	2,708	10,556	1,433	7	2	3	6
CiteSeer Giles et al. (1998)	3,327	9,104	3,703	6	2	3	–
Computers Shchur et al. (2018)	13,752	491,722	767	10	3	3	9
Physics Shchur et al. (2018)	34,493	495,924	8,415	5	2	2	4
Squirrel Pei et al. (2020)	5,201	217,073	2,089	5	2	2	4
Roman-empire Platonov et al. (2023)	22,662	65,854	300	18	4	4	16, 17
Years (ArXiv-year) Lim et al. (2021); Hu et al. (2020)	169,343	1,166,243	128	5	2	2	4

All statistics are measured on raw global graphs before Louvain client partitioning and task-block construction. Edge counts follow a directed edge-list convention. Task t uses original class IDs in $[(t-1)C_{\text{task}},\, tC_{\text{task}})$, so $C_{\text{eff}} = T \cdot C_{\text{task}}$.

E Notation

Table S6 lists the symbols used in the method.

Table S6: Notation and Definitions.

Notation	Definition
K, T	The number of clients and sequential tasks.
k, t	Client index and task index.
$C_{\text{task}}, C_{\text{eff}}$	Classes per task, and classes kept after dropping leftover tail IDs $[C_{\text{eff}}, C)$.
$G_{k,t}$	Local graph data of client k at task t.
$V_{k,t}, E_{k,t}, Y_{k,t}$	Node set, edge set, and label set of $G_{k,t}$.
f_θ	Global GNN parameterized by θ.
$\mathcal{D}_{t'}$	Data distribution of task t'.
$\mathcal{L}_{\text{ce}}$	Local cross-entropy on client task data.
$\mathcal{L}_{\text{ce}}^{\text{gen}}, \mathcal{L}_{\text{ce}}^{\text{syn}}$	Cross-entropy on generated samples (client) and on synthesized replay (server).
$H \in \mathbb{R}^{N \times D}$	Node representation matrix (N nodes, D dimensions).
$I_d^{\text{sem}}, I_d^{\text{str}}$	Post-hoc CE / structural-probe sensitivity of dimension d.
$g_\omega, \mathcal{L}_{\text{str}}$	Attribution probe on frozen H and its BCE; distinct from LP_ψ.
$\mathcal{E}_{\text{str}}^+, \mathcal{E}_{\text{str}}^-$	Observed edges and sampled non-edges for $\mathcal{L}_{\text{str}}$.
R_d	Relative importance $\log(I_d^{\text{sem}} + \epsilon_0) - \log(I_d^{\text{str}} + \epsilon_0)$.
$M \in \{0,1\}^D$	Binary mask from the top-r coordinates of R_d.
r	Share of hidden dimensions kept as H_S.
H_S, H_R	Semantic and structural channels after masking.
G_{ϕ_t}	Per-task conditional generator for task t.
$\mathcal{B}_t, \widetilde{G}^{1:t-1}$	Stored synthetic subgraphs and their disjoint union.
z, e_c	Gaussian noise and class embedding for generator.
$\tilde{h}, \tilde{x}$	Generated hidden vector and its normalized feature.
LP_ψ	Generator-side link predictor; topology prior for synthesis.
$p_{ij}, P_{ij}, \widetilde{A}_{ij}$	Directed, symmetrized edge score, and synthetic adjacency.
τ_{lp}	Link predictor threshold.
$\sigma, \text{softmax}$	Sigmoid (edge scores q_{ij}, p_{ij}) and softmax (KL arguments).
$\mathcal{L}_{\text{align}}$	Semantic prototype-alignment loss on classes $\mathcal{C}_{k,t}$ observed by client k.
$p_{c,S}^{\text{real}}, p_{c,S}^{\text{syn}}$	Real/synthetic class mean of H_S after encoding $G_{k,t}$ / $\widetilde{G}_{k,t}$ with frozen f_θ.
$\mathcal{E}^+, \mathcal{E}^-$	High-/low-p_{ij} generated pairs for $\mathcal{L}_{\text{topo}}$.
$\mathcal{L}_{\text{topo}}$	Topology-preservation BCE (positives and negatives).
$\mathcal{L}_{\text{gen}}$	Total generator objective.
$\lambda_{\text{cls}}, \lambda_{\text{align}}, \lambda_{\text{topo}}$	Generator-side loss weights.
$\phi_t^{(g)}, \psi_t^{(g)}$	Aggregated task-t generator and link-predictor parameters.
n_k, N_{total}	Client size $n_k = \lvert V_{k,t} \rvert$ and $N_{\text{total}} = \sum_k n_k$.
R_{hf}^*	Aggregated high-frequency roughness target.
$\widetilde{G}_t = (\widetilde{X}_t, \widetilde{A}_t, \widetilde{Y}_t)$	Synthetic subgraph materialized for task t.
$L, \bar{d}, \mathcal{F}$	Graph Laplacian, average degree, and feature-index set.
$R_{\text{hf}}(\widetilde{G})$	High-frequency roughness score.
λ_x, λ_D	Feature and degree trade-off coefficients.
f_t, f_{t-1}	Student f_θ and teacher f_{θ^-} (previous-task snapshot).
$\mathcal{L}_{\text{contrast}}$	Semantic-channel contrastive distillation loss.
$P_c^{(t-1)}, \tau$	Teacher prototype on $\widetilde{G}^{1:t-1}$ and temperature.
$\Omega, D_{\text{JS}}, \epsilon$	Hard-node set, JS divergence, and selection threshold; $\mathcal{L}_{\text{sim}} = 0$ if $\lvert\Omega\rvert < 2$.
$S_{ij}^{(m)}, \tau'$	Pairwise Gram similarity on H_R and its temperature.
$\mathcal{L}_{\text{sim}}$	Structural-channel similarity distillation loss.
$\mathcal{L}_{\text{total}}$	Final server-side distillation objective.
$\lambda_{\text{ce}}, \lambda_{\text{contrast}}, \lambda_{\text{sim}}$	Server-side distillation loss weights.

F METHOD PSEUDOCODE

Algorithm 1 summarizes the training pipeline of FedDiG.

Algorithm 1 FedDiG for Federated Continual Graph Learning

Require: Client task streams $\{G_{k,t}\}$, communication rounds R, local epochs E, generator rounds R_g, generator epochs E_g, distillation epochs E_d
Ensure: Trained global model f_θ

Partition the full graph into client task streams and initialize f_θ and $f_{\theta^-} \leftarrow f_\theta$
for $t = 1$ to T **do**
  for $r = 1$ to R **do**
    for all clients k in parallel **do**
      Train f_{θ_k} from f_θ on $G_{k,t}$ by $\mathcal{L}_{\text{ce}}$ and upload (θ_k, n_k)
    end for
    Aggregate local models $\theta \leftarrow \sum_k \frac{n_k}{\sum_j n_j}\theta_k$ and broadcast θ
  end for
  for all clients k in parallel **do**
    Train a fresh link predictor $\text{LP}_{\psi_{k,t}}$ on $G_{k,t}$
    Extract H_k with frozen aggregated f_θ on $G_{k,t}$
    Train probe $g_{\omega_{k,t}}$ on frozen H_k by $\mathcal{L}_{\text{str}}$ (update only ω)
    Compute I^{sem} by Eq. 2 and I^{str}, R_d by Eqs. 3–4
    Select the top-r coordinates of R_d as M_k and split $H_{k,S} = H_k \odot M_k$, $H_{k,R} = H_k \odot (1 - M_k)$
    Compute $R_{\text{hf}}^{*(k)}$ on $G_{k,t}$ by Eq. 12
  end for
  for $q = 1$ to R_g **do**
    for all clients k in parallel **do**
      for $e_g = 1$ to E_g **do**
        Generate $\widetilde{G}_{k,t}$ with $(G_{\phi_{k,t}}, \text{LP}_{\psi_{k,t}})$, encode it by frozen f_θ, apply M_k, and update $G_{\phi_{k,t}}$ by
        $\mathcal{L}_{\text{gen}} = \lambda_{\text{cls}}\mathcal{L}_{\text{ce}}^{\text{gen}} + \lambda_{\text{align}}\mathcal{L}_{\text{align}}(p_{c,S}^{\text{real}}, p_{c,S}^{\text{syn}}) + \lambda_{\text{topo}}\mathcal{L}_{\text{topo}}$
      end for
      Upload $(\phi_{k,t}, \psi_{k,t}, R_{\text{hf}}^{*(k)})$
    end for
    Aggregate $(\phi_{k,t}, \psi_{k,t}, R_{\text{hf}}^{*(k)})$ by Eq. 11 (do not average M_k) and broadcast the current-task generator and LP
  end for
  Synthesize $\widetilde{G}_t$, calibrate it to R_{hf}^* by Eq. 12, and append it to $\mathcal{B}_t$
  if $t > 1$ **then**
    Form $\widetilde{G}^{1:t-1} = \bigsqcup_{t'<t} \widetilde{G}_{t'}$
    Train a probe g_ω on frozen teacher hidden states of $\widetilde{G}^{1:t-1}$ by $\mathcal{L}_{\text{str}}$
    Recompute M_t from R_d using teacher CE and $\mathcal{L}_{\text{str}}$ gradients, and split H_S/H_R
    for $e = 1$ to E_d **do**
      Update θ by $\mathcal{L}_{\text{total}} = \lambda_{\text{ce}}\mathcal{L}_{\text{ce}}^{\text{syn}} + \lambda_{\text{contrast}}\mathcal{L}_{\text{contrast}}(H_S) + \lambda_{\text{sim}}\mathcal{L}_{\text{sim}}(H_R)$
    end for
  end if
  Set $f_{\theta^-} \leftarrow f_\theta$ and evaluate all seen tasks
end for
return f_θ

G Implementation Details and Evaluation Protocol

Following prior work Huang et al. (2023); Zhang et al. (2021), we use Louvain partitioning Blondel et al. (2008) to simulate non-IID client distributions and a $60\%/20\%/20\%$ train/val/test split. In federated training, all clients participate in every communication round, and the server aggregates local updates after each round; the generator stage also uses full client participation by default. For task construction, original class IDs are grouped into contiguous non-shuffled blocks $[(t-1)C_{\text{task}}, tC_{\text{task}})$ for $t = 1, \ldots, T$. Only complete blocks are kept: $C_{\text{eff}} = \lfloor C/C_{\text{task}} \rfloor \cdot C_{\text{task}}$, $T = C_{\text{eff}}/C_{\text{task}}$, and leftover tail IDs $[C_{\text{eff}}, C)$ are discarded (Table S5). Each task is trained and evaluated on the subgraph of its class block; inter-task edges are excluded for all methods.

Across all experiments, we use 15 communication rounds and 5 local training epochs per round; unless otherwise stated, we use $K = 3$ clients, a two-layer GCN backbone with hidden size 64, semantic mask ratio $r = 50\%$, and a replay budget of 80 synthetic nodes per class. Unless otherwise stated, all results are averaged over five predefined random seeds and reported as mean±std.

Additional implementation settings. Mask-ratio sensitivity is in Table 3; client-scale sweeps are in Figure 6 (Appendix C); loss-weight sensitivity is in Figure 3. The remaining defaults are as follows. Class prototypes are the mean of semantic-channel hidden features H_S. Client-side $p_{c,S}^{\mathrm{syn}}$ is the class mean of $H_S^{\mathrm{syn}} = f_\theta^{\mathrm{hid}}(\widetilde{G}_{k,t}) \odot M_k$ with θ frozen; $p_{c,S}^{\mathrm{real}}$ uses the same encoder on $G_{k,t}$. Each client computes M from the dual-gradient ranking R_d of the frozen current-task aggregated model f_θ on $G_{k,t}$: I^{sem} from $\mathcal{L}_{\mathrm{ce}}$ and I^{str} from a label-free probe g_ω trained on frozen H. The same frozen f_θ is used by $\mathcal{L}_{\mathrm{ce}}^{\mathrm{gen}}$. The server does not average client masks, but retrains a probe on the stored synthetic adjacency and recomputes R_d from teacher f_{θ^-} on $\widetilde{G}^{1:t-1}$ before distillation. Neither attribution updates the encoder. Generator parameters are still size-weighted averaged because G_ϕ emits input-space features and M_k is only a frozen readout. Replay and distillation use 80 synthetic nodes per class in each stored subgraph. The generator is an MLP that maps a 128-d noise vector concatenated with a class embedding through hidden widths 64-128-256 to the input feature dimension; the link predictor is a two-layer MLP with hidden size 64 and threshold $\tau_{\mathrm{lp}} = 0.4$, trained on $G_{k,t}$ with BCE over real edges and sampled non-edges. $\mathcal{L}_{\mathrm{topo}}$ applies the same BCE to generated nodes, taking the top-m and bottom-m frozen link-predictor scores as $\mathcal{E}^+$ and $\mathcal{E}^-$. After synthesizing $\widetilde{G}_t$, the server stores that subgraph and does not keep a persistent generator across tasks. For $t > 1$, distillation updates the global encoder f_θ for E_d epochs on the disjoint union of previously stored synthetic subgraphs by $\mathcal{L}_{\mathrm{total}}$, then sets $f_{\theta^-} \leftarrow f_\theta$.

Table S7: Default hyperparameters used unless otherwise stated. Mask ratio r and loss-weight sweeps are in the main paper (Table 3, Figure 3); client-scale sweeps are in Figure 6.

Setting	Default
Clients K	3
Hidden size / layers	64 / 2-layer GCN
Replay budget	80 synthetic nodes per class
$\lambda_{\mathrm{cls}}, \lambda_{\mathrm{align}}, \lambda_{\mathrm{topo}}$	10, 1, 0.01
$\lambda_{\mathrm{ce}}, \lambda_{\mathrm{contrast}}, \lambda_{\mathrm{sim}}$	10, 0.1, 0.05
λ_x, λ_D	0.5, 0.5
Generator rounds / epochs R_g, E_g	3, 200
Distillation epochs E_d	60
Learning rates (GNN / gen / KD)	0.01, 0.005, 0.002
Noise dim / τ_{lp}	128 / 0.4
Structural probe g_ω	2-layer MLP on $[h_i \| h_j]$; H frozen

After finishing task t, we evaluate the current global model on the test split of every seen task $i \le t$. Importantly, prediction is performed over the cumulative label space of all seen classes, $Y_{\le t} = \bigcup_{j=1}^{t} Y_j$, rather than only the classes of task i. Let $a_{t,i}$ denote the test accuracy on task i after completing training up to task t. We report the final average accuracy (AA) and average forgetting (AF) as

$$\mathrm{AA} = \frac{1}{T} \sum_{i=1}^{T} a_{T,i}, \tag{18}$$

and

$$\mathrm{AF} = \frac{1}{T-1} \sum_{i=1}^{T-1} (a_{i,i} - a_{T,i}). \tag{19}$$

Here, $a_{i,i}$ is the accuracy of task i immediately after task i is learned, while $a_{T,i}$ is its final accuracy after the last task. Therefore, AA measures the mean task accuracy at the end of training, and AF measures the mean forgetting on previously learned tasks.

H Dataset Details

We evaluate FedDiG on seven real-world graph datasets spanning citation networks, co-purchase/co-authorship networks, heterophilic benchmarks, and a temporal citation graph. The statistics for these datasets are provided in Table S5. For all datasets, we use Louvain partitioning to simulate non-IID client distributions and a $60\%/20\%/20\%$ train/val/test split. The details are as follows:

- **Cora** McCallum et al. (2000). Cora is a citation network of 2,708 scientific publications classified into 7 research topics. Each node represents a paper and is described by a 1,433-dimensional binary bag-of-words feature vector indicating the presence or absence of specific words. Edges represent citation links between papers. Cora is one of the most widely used benchmarks for evaluating graph neural networks on node classification tasks, due to its moderate size, clear community structure, and strong homophily.
- **CiteSeer** Giles et al. (1998). CiteSeer is a citation network of 3,327 scientific documents classified into 6 categories. Each node is represented by a 3,703-dimensional binary word-presence feature vector. Similar to Cora, edges denote citation relationships between documents. CiteSeer is sparser than Cora and has a higher feature dimensionality relative to graph size, which makes it a complementary benchmark for testing model performance under different sparsity regimes.
- **Computers** Shchur et al. (2018). Amazon-Computers is a co-purchase graph extracted from Amazon product data. Nodes represent 13,752 computer-related products, and edges indicate that two products are frequently co-purchased. Node features are 767-dimensional bag-of-words vectors derived from product reviews, and nodes are classified into 10 product categories. Compared to citation networks, this dataset is denser and larger, posing additional challenges for scalability and message-passing efficiency.
- **Physics** Shchur et al. (2018). Coauthor-Physics is a co-authorship network derived from the Microsoft Academic Graph. It contains 34,493 author nodes connected by edges representing co-authorship on physics papers. Node features are 8,415-dimensional keyword vectors extracted from each author's publications, and nodes are classified into 5 research fields. The high feature dimensionality and large graph size make Physics a demanding benchmark for evaluating model scalability and the ability to handle rich node attributes.
- **Squirrel** Pei et al. (2020). Squirrel is a heterophilic page–page network from English Wikipedia, introduced in the Geom-GCN benchmark. Its 5,201 nodes represent Wikipedia articles and edges represent mutual hyperlinks. Node features encode the presence of particular nouns in the articles, and nodes are classified into 5 classes based on their average monthly web traffic. Unlike citation or co-purchase graphs, Squirrel exhibits low homophily, meaning connected nodes often belong to different classes. This property tests whether models can handle non-homophilic message-passing patterns effectively.
- **Roman-empire** Platonov et al. (2023). Roman-empire is a heterophilic word-dependency graph proposed by Platonov et al. as part of a more rigorous heterophily benchmark suite. It contains 22,662 nodes (words in a text about the Roman Empire), 300-dimensional fastText word-embedding features, and 18 syntactic role classes. Edges are derived from syntactic dependency relations, producing extremely low edge homophily (≈ 0.05). This dataset was specifically designed to address issues in prior heterophily benchmarks such as small size, class imbalance, and duplicate nodes, making it a reliable testbed for evaluating GNNs under strong heterophily.
- **Years (ArXiv-year)** Lim et al. (2021); Hu et al. (2020). Years is the ArXiv-year benchmark of Lim et al., derived from the ogbn-arxiv citation network in the Open Graph Benchmark. It consists of 169,343 arXiv computer-science paper nodes connected by 1,166,243 directed citation edges. Each node has a 128-dimensional feature vector obtained from averaging word embeddings of the paper's title and abstract. In the ArXiv-year construction, publication year is discretized into 5 temporal bins as class labels. This dataset is the largest in our benchmark suite and introduces a temporal non-stationarity dimension, testing the model's ability to handle evolving semantics over time.

I Baseline Details

We compare FedDiG against sixteen representative baselines spanning four methodological families. Below we describe each baseline in detail.

Federated graph learning methods.

- **FedAvg** McMahan et al. (2017) [AISTATS 2017]. FedAvg is the foundational federated learning algorithm proposed by McMahan et al. Local clients independently train models on their own data and then send model updates to a central server, which computes a weighted average of client parameters to produce a global model. This global model is sent back to clients for further local training. FedAvg reduces communication overhead by transmitting model parameters rather than raw data and is widely used as a baseline in federated settings. However, it assumes stationary data distributions and does not incorporate any mechanism to mitigate catastrophic forgetting when data evolves over time.
- **FedSage+** Zhang et al. (2021) [NeurIPS 2021]. FedSage+ extends the standard FedAvg framework to address the missing neighbor problem in subgraph federated learning. When a global graph is partitioned across clients, cross-client edges are severed, causing each client to observe an incomplete neighborhood. FedSage+ trains a missing neighbor generator alongside the GraphSage backbone so that each client can impute plausible features for missing neighbors during local training. This alleviates the distributional bias from incomplete subgraphs but does not account for temporal task drift or catastrophic forgetting.
- **FedPub** Baek et al. (2023) [ICML 2023]. FedPub (Personalized Subgraph Federated Learning) addresses the heterogeneity among client subgraphs by shifting from a single global model to jointly improving interrelated local GNNs. It introduces two techniques: (i) functional embeddings that encode each local GNN's behavior on random input graphs, enabling the server to compute model-level similarities for weighted aggregation without accessing client data; and (ii) personalized sparse masks that allow each client to selectively update only the parameters relevant to its local subgraph structure. FedPub is designed for stationary subgraph distributions and does not explicitly handle continual task evolution.
- **FedGTA** Li et al. (2023) [PVLDB 2023]. FedGTA introduces topology-aware averaging for federated graph learning. Existing federated optimization strategies ignore graph structure during aggregation, which can lead to poor convergence and sub-optimal performance. FedGTA addresses this by incorporating local smoothing confidence and mixed neighbor features into the aggregation process, so that the server accounts for the underlying graph topology when combining client models. FedGTA achieves strong scalability across multi-scale graphs but is designed for static graph distributions without continual learning capabilities.

Graph continual learning methods (federated adaptation).

- **Fed-DSLR** Choi et al. (2024) [WWW 2024]. DSLR (Diversity enhancement and Structure Learning for Rehearsal-based graph continual learning) addresses two key limitations of existing rehearsal-based graph continual learning methods: (i) concentration bias, where replayed nodes cluster around class centers and overfit to central representations; and (ii) irrelevant neighbors, where stored nodes may connect to uninformative neighbors that degrade replay quality. DSLR introduces coverage-based diversity to balance class representativeness with within-class variety, and graph structure learning to ensure replayed nodes connect to truly informative neighbors. In our federated adaptation (Fed-DSLR), the DSLR replay strategy is applied locally at each client within the standard federated protocol.
- **Fed-TWP** Liu et al. (2021) [AAAI 2021]. TWP (Topology-aware Weight Preserving) is a plug-and-play module for mitigating catastrophic forgetting in GNNs during continual learning. Unlike CNN-oriented continual learning methods that only penalize changes to task-important parameters, TWP explicitly considers graph topology by stabilizing parameters critical to neighborhood aggregation. This preserves the model's ability to capture local structural patterns learned in previous tasks. In our federated adaptation (Fed-TWP), the TWP regularization is applied during local training, and model updates are aggregated following the standard federated protocol.
- **Fed-SSM** Zhang et al. (2022) [ICDM 2022]. SSM (Sparsified Subgraph Memory) is a graph continual learning replay method that stores sparsified computation subgraphs to retain explicit topological information from previous tasks. In our federated adaptation (Fed-SSM), each client maintains its local sparsified subgraph memory and participates in the same federated aggregation protocol as other baselines.
- **Fed-PromptCGL** Wang et al. (2026) [SCIS 2026]. PromptCGL is a prompt-driven continual graph learning method that learns task-specific prompts while keeping the backbone GNN fixed,

using hierarchical prompting to model both feature- and topology-level variations. In our federated adaptation (Fed-PromptCGL), prompt learning is performed locally on client task streams and aggregated under the standard federated protocol.

- **Fed-TPP** Niu et al. (2024) [NeurIPS 2024]. TPP (Task Profiling and Prompting) is a replay-free and forget-free graph class-incremental learning method that predicts task identity through graph task profiling and learns task-specific graph prompts. In our federated adaptation (Fed-TPP), each client applies the task-profiling and prompting mechanism locally, while the server follows the same federated model aggregation schedule.
- **Fed-TACO** Han et al. (2024) [NeurIPS 2024]. TACO (Topology-aware Graph Coarsening) stores previous-task information as a reduced graph and updates this memory through topology-preserving graph coarsening. In our federated adaptation (Fed-TACO), topology-aware coarsened memories are maintained locally and the resulting client models are aggregated under the common federated training protocol.

Federated continual learning methods.

- **GLFC** Dong et al. (2022) [CVPR 2022]. GLFC (Global-Local Forgetting Compensation) is the first work to formally address federated class-incremental learning. It tackles catastrophic forgetting from both local and global perspectives: locally, it applies a class-aware gradient compensation loss and a class-semantic relation distillation loss to preserve old-class knowledge; globally, it introduces a proxy server mechanism that selects the best historical global model to assist local relation distillation and to handle non-IID class imbalance across clients. Although effective for image-based federated continual learning, GLFC does not incorporate graph-specific mechanisms such as topology preservation.
- **TARGET** Zhang et al. (2023) [ICCV 2023]. TARGET proposes exemplar-free distillation for federated class-continual learning. It alleviates catastrophic forgetting without storing any raw data from previous tasks by leveraging two strategies: (i) model-level knowledge transfer that uses the previously trained global model as a teacher to guide the current model; and (ii) data-level synthetic generation via a generator that produces synthetic samples simulating the global data distribution on each client. TARGET is designed for image classification and does not account for graph topology or message-passing dependencies.
- **Lander** Tran et al. (2024) [CVPR 2024]. Lander (Label Text Centered Data-Free Knowledge Transfer) introduces text-enhanced data-free replay for federated class-incremental learning. It leverages label text embeddings from pretrained language models as anchor points in the latent space: during training, sample embeddings are constrained to remain near these anchors via a bounding loss, enriching the surrounding latent space; during knowledge transfer, these anchors guide the synthesis of high-quality pseudo-samples for rehearsal. Lander achieves strong results on image benchmarks but lacks graph-aware components for preserving structural knowledge.

Federated continual graph learning.

- **POWER** Zhu et al. (2025) [KDD 2025]. POWER is an early dedicated framework for Federated Continual Graph Learning (FCGL). It addresses two core challenges: Local Graph Forgetting, where client GNNs lose prior knowledge when adapting to new tasks, and Global Expertise Conflict, where inconsistent client expertise causes sub-optimal global aggregation. To mitigate local forgetting, POWER employs experience-node replay with maximum coverage, selecting and storing representative nodes that maximize local-global structural coverage. To resolve global conflicts, it introduces pseudo prototype reconstruction and trajectory-aware knowledge transfer at the server. However, POWER retains real node data for replay and does not explicitly disentangle semantic from structural contributions, leaving structural forgetting under-addressed under heterogeneous topologies.
- **MOTION** Wan et al. (2025) [NeurIPS 2025]. MOTION is a dedicated FCGL framework that targets topology preservation at task transitions. On the client, Graph Topology-preserving Multi-Sculpt Coarsening (G-TMSC) uses similarity-guided multi-expert fusion to coarsen historical graphs while retaining critical subgraph structure. On the server, the Graph-Aware Evolving Parameter Adaptive Engine (G-EPAE) builds a topology-sensitive compatibility matrix and reweights client increments to reduce aggregation conflicts. MOTION therefore preserves topol-

ogy at the graph-memory and aggregation levels, but does not disentangle semantic and structural channels in the representation.

- **FedFST** Guo et al. (2026) [KDD 2026]. FedFST studies spectral catastrophic forgetting in federated graph continual learning. It decomposes forgetting into high-frequency inconsistency forgetting, where newly emerging node inconsistencies disrupt message passing, and low-frequency consistency forgetting, where over-adaptation to new tasks dilutes global semantic coherence. Historical High-Frequency Knowledge Restoration (HHKR) reconstructs high-frequency inconsistency knowledge, while Historical Low-Frequency Semantic Transfer (HLST) stabilizes low-frequency consistency by spectral distillation. FedFST is spectral-aware but does not explicitly partition representation dimensions into semantic and structural channels.