FedProG: Federated Graph Learning via Server-Side LLM Semantic Bridging and Uncertainty-Aware Distillation

Haitao Wang[1], **Haozhao Wang**[2], **Wenchao Xu**[3], **Yichen Li**[2], **Xiaoquan Yi**[2], **Tianzhe Xiao**[2], **Jingcai Guo**[4], **Ruixuan Li**[2*]
[1]School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan, China [2]School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China [3]Division of Integrative Systems and Design, Hong Kong University of Science and Technology, Hong Kong, China [4]Department of Computing/LSGI, The Hong Kong Polytechnic University, Hong Kong, China
haitao_wang, hz_wang, ycli0204, rxli@hust.edu.cn

Abstract

Federated Graph Learning (FGL) on Text-Attributed Graphs often encounters the missing neighbor issue. This problem arises from structural fragmentation across dispersed clients, which restricts the effective message passing of Graph Neural Networks. Current completion methods frequently struggle in cross-domain, non-IID scenarios because they lack access to external knowledge. To address these limitations, we propose FedProG, an asymmetric framework that utilizes a frozen server-side LLM to reconstruct missing structural contexts. Furthermore, deploying Large Language Models locally to augment these graphs incurs prohibitive computational costs for edge devices. We introduce a split architecture where a lightweight client-side generator maps graph features into the LLM's embedding space. This allows the server to produce virtual neighbors via uploaded soft prompts. To ensure system reliability and privacy, we implement a dual-guard mechanism: a server-side uncertainty filter rejects low-confidence generations through multi-path sampling, while a client-side semantic consistency reconstructor maintains utility under differential privacy. Experiments show that FedProG improves robustness and generalization over baselines while drastically reducing client computational overhead.

1 Introduction

Graph Neural Networks (GNNs) have become the standard paradigm for processing Text-Attributed Graphs (TAGs) [24]. However, within strictly regulated and privacy-sensitive domains such as finance and healthcare [15], valuable graph data is frequently distributed across isolated institutions. Federated Learning (FL) [14] offers a pivotal solution to this data isolation dilemma, inspiring the emergence of Federated Graph Learning (FGL). Despite its promise, a critical challenge in FGL remains the accuracy degradation caused by inevitable subgraph fragmentation [25]. As illustrated in the left panel of Figure 1, this fragmentation forms a rigid data privacy wall that severs cross-client links, effectively preventing boundary nodes from aggregating global structural information.

This structural fragmentation leads directly to the missing neighbor problem, where boundary nodes lose their essential global receptive field. Existing completion methods, such as FedSage+ [25] and FedAAS [7], rely primarily on observed local data distributions. More recently, FedEGL [19] introduced an edge-assisted framework to exchange approximated intermediate features. However, these statistical learning-based generators, even when combined with topology-aware distillation

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

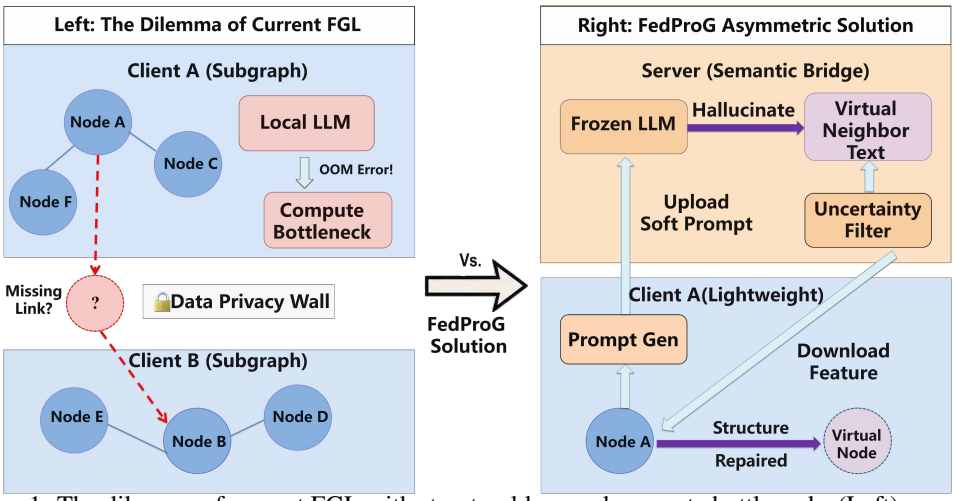

Figure 1: The dilemma of current FGL with structural loss and compute bottlenecks (Left) versus the proposed asymmetric FedProG solution (Right).

strategies like FedTAD [27, 20], often struggle to maintain robustness and out-of-distribution (OOD) generalization in heterogeneous, Non-IID scenarios. Recent studies have explored Large Language Models (LLMs) to enhance graph learning [23], leveraging their extensive pre-trained knowledge for zero-shot reasoning. However, integrating LLMs into federated systems faces a critical compute bottleneck. Conventional frameworks typically require resource-constrained clients to execute heavy LLM layers locally [5, 12], which often leads to out-of-memory errors or prohibitive latency on edge devices, preventing effective structural repair.

To bridge this gap, we propose FedProG, an asymmetric FGL architecture depicted in the right panel of Figure 1. Unlike conventional frameworks that attempt to deploy heavy models locally, FedProG introduces a *Split Semantic Bridging* paradigm. We offload intensive inference to a frozen server-side LLM acting as an open-world semantic bridge. Clients utilize a lightweight prompt generator to project local boundary node features into the LLM's continuous latent space. By transmitting DP-protected soft prompts rather than raw text, we allow the server to synthesize high-quality virtual neighbors, effectively hallucinating the missing links, without accessing private raw data.

To ensure reliability, FedProG incorporates a dual-guard mechanism. On the server side, multi-path sampling and uncertainty quantification are employed to filter LLM hallucinations. On the client side, a semantic consistency reconstruction objective ensures that prompt vectors retain node identity even under aggressive differential privacy noise. This asymmetric design restores global connectivity while mitigating feature shift, all without imposing prohibitive computational costs on edge devices.

We validate FedProG on four standard text-attributed graph benchmarks under heterogeneous settings. Results show FedProG significantly outperforms baselines, achieving a peak accuracy improvement of 8.10% with reduced communication overhead compared to LLM-based counterparts.

Our main contributions are summarized as follows:

- We introduce a split learning paradigm that re-engineers the LLM embedding layer as a client-side prompt generator while keeping the other layers on the server. This design circumvents edge compute bottlenecks and ensures physical isolation between raw data and the semantic feature space.
- We implement a server-side uncertainty quantification module to dynamically weight the aggregation of virtual neighbors, which effectively mitigates the hallucinations and noise inherent in LLM-generated content.
- We design a semantic consistency reconstruction loss for the client to safeguard prompt vectors against differential privacy noise, thereby preserving semantic integrity while simultaneously satisfying rigorous privacy constraints.
- Extensive experiments demonstrate that FedProG significantly outperforms leading baselines in robustness and generalization, particularly in sparse cross-domain scenarios, while drastically reducing client-side resource consumption.

2 Related Work

2.1 Heterogeneity and Structural Loss in FGL

Federated Graph Learning (FGL) enables collaborative model training on decentralized graph data while adhering to privacy constraints. To address the structural fragmentation inherent in FGL, FedSage+ [25] employs Variational Autoencoders (VAE) to synthesize missing neighbor nodes, while FedAAS [7] leverages historical embeddings to approximate cross-client neighborhood information. Additionally, FedGTA [10, 21] introduces topology-aware smooth confidence to refine the parameter aggregation process. Regarding data heterogeneity and privacy preservation, [18, 11, 22, 6] investigate FGL and non-centralized learning across cross-domain recommendation and privacy-preference settings, and FedTAD [27] proposes a topology-aware, data-free knowledge distillation approach. Despite these advancements, most existing methods rely solely on observed local data distributions and lack the capacity to utilize external knowledge for generalized cross-domain reasoning.

2.2 Combining LLMs with Graph Learning

The integration of Large Language Models with graph learning has emerged as a promising direction for enhancing Text-Attributed Graphs. TAPE [1] utilizes LLMs to generate explanatory text for nodes as a form of feature augmentation. Strategic roadmaps have further emphasized the convergence of LLMs and Knowledge Graphs to improve structural reasoning. For instance, Zhou et al. [26] apply coupled augmentations to improve representation learning on TAGs, and [13] develops a prompt-based reasoning scheme through KG-integrated collaboration. To bridge the gap between LLMs and Graph Neural Networks, LinguGKD [3] presents a knowledge distillation framework, while UniGraph [2] advances the field by learning a unified cross-domain foundation model. However, these works primarily target centralized scenarios. In federated settings, LLM4FGL [23] attempts to utilize LLMs for pseudo-data generation, yet it necessitates local LLM execution on edge devices. In contrast, our FedProG mitigates client-side deployment costs through an asymmetric architecture that utilizes the server-side LLM as a semantic bridge.

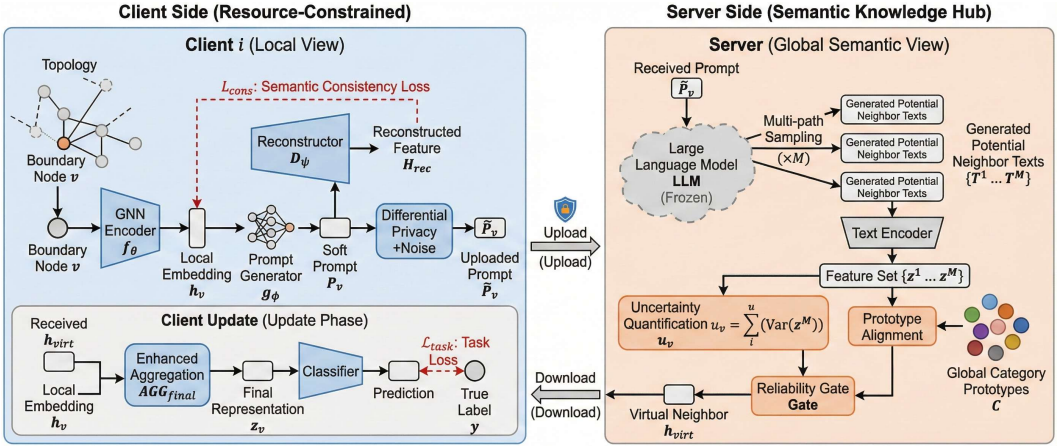

Figure 2: The asymmetric FedProG framework. Client: A lightweight GNN f_θ and generator g_ϕ map boundary nodes to DP-protected soft prompts $\tilde{P}_v$, regularized by $\mathcal{L}_{\text{cons}}$. Server: A frozen LLM synthesizes virtual neighbors $h_{v,virt}$ via multi-path sampling and prototype alignment $\mathcal{C}$, using an uncertainty gate $\exp(-\gamma u_v)$ to suppress hallucinations.

3 Methodology

3.1 Problem Definition and Overall Framework

Assume the system contains N clients, where each client i possesses a private subgraph $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i, \mathcal{X}_i)$. Define the set of boundary nodes $\mathcal{B}_i \subset \mathcal{V}_i$ as those nodes that have cross-domain neighbors in the global graph but are missing them locally. The goal of FedProG is to complete the

neighbor information of $\mathcal{B}_i$ using the server-side LLM while protecting privacy. The detailed process of the overall framework is shown in Figure 2.

3.2 Semantic Consistency Prompt Generation

To avoid deploying LLMs on the client side, we only deploy a lightweight GNN encoder f_θ and a prompt generator g_ϕ.

3.2.1 Context Encoding and Prompt Mapping

For boundary nodes, local neighbor information is first aggregated to obtain a local embedding $h_v^{(L)}$, which is then mapped to a soft prompt vector P_v. To protect privacy, Gaussian noise satisfying differential privacy is injected:

$$h_v^{(L)} = f_\theta(\mathcal{G}_i)_v \tag{1}$$

$$P_v = g_\phi(h_v^{(L)}) \tag{2}$$

$$\tilde{P}_v = P_v + \xi, \quad \text{where } \xi \sim \mathcal{N}(0, \sigma^2 I) \tag{3}$$

To guarantee data privacy, ξ is the random noise vector sampled from a Gaussian distribution, where σ represents the noise scale calibrated to satisfy differential privacy constraints. Consequently, $\tilde{P}_v$ denotes the perturbed, privacy-preserving soft prompt that is physically transmitted to the server, ensuring that the raw local embedding $h_v^{(L)}$ remains concealed from the server-side model.

In traditional LLMs, discrete input tokens are converted into continuous vectors via an embedding matrix $E \in \mathbb{R}^{|V| \times d}$. In FedProG, we reformulate this initialization as a split learning architecture. The client-side prompt generator g_ϕ replaces the discrete lookup operation, projecting the local graph embedding directly into the LLM's continuous latent space to serve as the input for the frozen Transformer layers:

$$H_{\text{out}} = \mathcal{F}(\tilde{P}_v) = \mathcal{F}\left(g_\phi(h_v^{(L)}) + \xi\right) \tag{4}$$

where $\tilde{P}_v \in \mathbb{R}^d$ is the perturbed continuous semantic vector acting as a soft prompt, and $\mathcal{F}$ denotes the pre-trained Transformer blocks on the server side.

3.2.2 Semantic Consistency Reconstruction

To prevent noise from damaging semantics, an auxiliary reconstructor D_ψ is introduced, requiring the noisy prompt to be able to restore original features. We minimize the reconstruction loss $\mathcal{L}_{\text{cons}}$:

$$\mathcal{L}_{\text{cons}} = \|D_\psi(\tilde{P}_v) - h_v^{(L)}\|^2 \tag{5}$$

This ensures that the uploaded vector $\tilde{P}_v$ maintains the key semantic identity of the node in the feature space. Here, D_ψ denotes the reconstructor network parameterized by ψ.

3.3 Uncertainty Awareness and Heterogeneity Alignment

The server maintains a frozen LLM_{server} and a set of global category prototypes $\mathcal{C} = \{c_1, \ldots, c_K\}$.

3.3.1 Multi-path Sampling and Generation

Utilizing the LLM as a semantic bridge, M samplings are performed based on $\tilde{P}_v$. It is worth noting that since $\tilde{P}_v$ is already a continuous embedding vector, the server skips the standard token embedding lookup step and directly inputs it as a prefix to the first Transformer block of the LLM, combined with a preset instruction to generate text Text_m:

$$\text{Text}_m = \text{LLM}_{server}(\text{Instruction}, \tilde{P}_v) \tag{6}$$

$$z_v^m = \text{Encoder}(Text_m) \tag{7}$$

where z_v^m represents its encoded feature for the m-th path.

3.3.2 Uncertainty Quantification

The variance of the generated semantic features across multiple sampling paths is calculated as an uncertainty measure u_v, which is used to dynamically detect and filter unreliable LLM hallucinations:

$$\bar{z}_v = \frac{1}{M}\sum_{m=1}^{M} z_v^m, \quad u_v = \frac{1}{M}\sum_{m=1}^{M} \|z_v^m - \bar{z}_v\|^2 \tag{8}$$

The term $\bar{z}_v$ serves as the mean semantic embedding across all paths, while u_v measures the generation uncertainty based on feature variance.

3.3.3 Prototype Alignment and Gating

To alleviate feature shift caused by heterogeneity, global prototypes $\mathcal{C}$ are used to calibrate features, and combined with uncertainty for gating to obtain the final virtual neighbor $h_{v,virt}$ to be sent down:

$$\alpha_{v,k} = \frac{\exp(\bar{z}_v^\top c_k/\tau)}{\sum_{j=1}^{K}\exp(\bar{z}_v^\top c_j/\tau)} \tag{9}$$

$$h_{v,virt} = \left(\sum_{k=1}^{K}\alpha_{v,k}c_k\right)\cdot\exp(-\gamma u_v) \tag{10}$$

where $\alpha_{v,k}$ denotes the attention weight assigned to the k-th global prototype c_k, scaled by the temperature parameter τ. The term $h_{v,virt}$ represents the final calibrated virtual neighbor, where the hyperparameter γ controls the intensity of the uncertainty-based suppression mechanism. When the LLM generation results are divergent, this mechanism will automatically suppress the weight of the virtual neighbor.

3.4 Enhanced Aggregation and Joint Optimization

After the client receives the synthesized virtual neighbor $h_{v,virt}$, it is treated as a new neighbor for local aggregation to obtain the enhanced representation z_v:

$$z_v = \text{AGG}_{\text{final}}(h_v^{(L)}, h_{v,virt}) \tag{11}$$

where $\text{AGG}_{\text{final}}$ denotes the final aggregation layer specifically designed to fuse the local embedding with the server-synthesized virtual context. The final classification task loss is $\mathcal{L}_{\text{task}}$:

$$\mathcal{L}_{\text{task}} = -\sum_{v\in\mathcal{V}_{train}} y_v \log(\text{Softmax}(W z_v)) \tag{12}$$

where $\mathcal{V}_{train}$ represents the set of labeled training nodes, y_v indicates the ground-truth label, and W refers to the learnable weights of the classifier. The total optimization objective is formulated as the weighted joint loss function $\mathcal{L}_{\text{total}}$:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda\mathcal{L}_{\text{cons}} \tag{13}$$

By minimizing $\mathcal{L}_{\text{total}}$, both the client's GNN parameters and prompt generator parameters are updated, where λ serves as the trade-off hyperparameter balancing task performance and semantic consistency.

4 Parameter Aggregation and Global Update

Due to the asymmetry of the architecture, FedProG discards the traditional full-model aggregation method and instead adopts a decoupled aggregation strategy. Parameters in the system are divided into three categories: client parameters requiring federated aggregation, server-side LLM parameters that remain frozen, and category prototype parameters requiring global synchronization. The overall update process is shown in Figure 3.

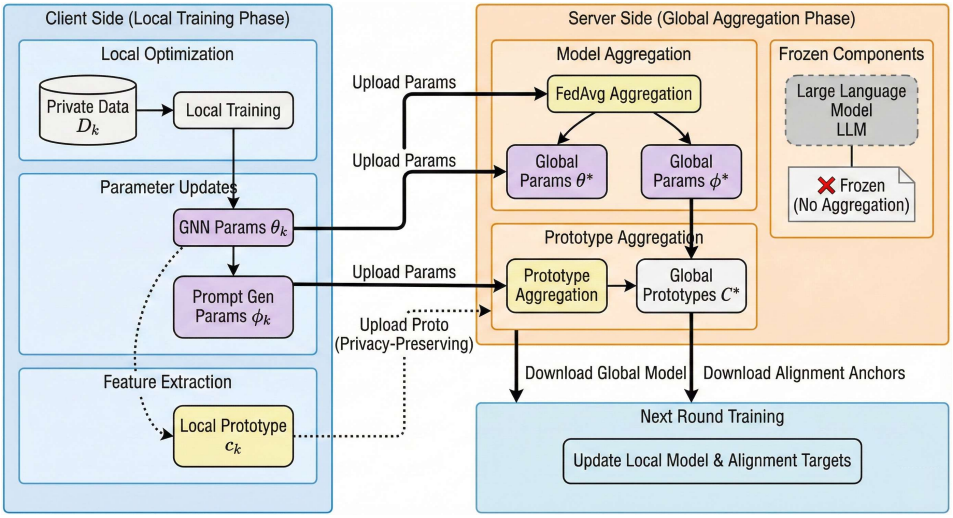

Figure 3: Decoupled parameter aggregation in FedProG. Clients upload weights (θ, ϕ) and local prototypes c_k. The server synchronizes global parameters and category prototypes $\mathcal{C}$ via FedAvg while keeping the LLM frozen.

4.1 Client-Side Model Aggregation

The client's GNN encoder (f_θ) and prompt generator (g_ϕ) need to learn private data features. We use the classic FedAvg algorithm to aggregate these two parts. After the t-th round of training, the server collects parameters from m active clients and performs weighted averaging:

$$\theta^{(t+1)} \leftarrow \sum_{k \in S_t} \frac{|\mathcal{D}_k|}{|\mathcal{D}|} \theta_k^{(t+1)} \tag{14}$$

$$\phi^{(t+1)} \leftarrow \sum_{k \in S_t} \frac{|\mathcal{D}_k|}{|\mathcal{D}|} \phi_k^{(t+1)} \tag{15}$$

where $\mathcal{D}$ represents the total data volume across all clients, S_t denotes the set of participating clients in the current round, and $|\mathcal{D}_k|$ indicates the specific dataset size of client k used for weighting. This step aims to let the GNN and prompt generator learn global graph structure patterns and general prompt mapping rules.

4.2 Server-Side LLM

The server-side Large Language Model remains frozen throughout the training process, does not participate in aggregation, and does not undergo gradient updates. This aims to retain the LLM's general world knowledge, prevent catastrophic forgetting, and significantly reduce computational and communication overhead.

4.3 Global Prototype Update

This is the key to FedProG solving heterogeneity. Global prototypes $\mathcal{C} = \{c_1, \ldots, c_K\}$ are used to calibrate the feature distribution of virtual neighbors on the server.

Each client k computes the mean of embeddings for all nodes in each category after local training:

$$c_{k,j} = \frac{1}{|\mathcal{V}_{k,j}|} \sum_{v \in \mathcal{V}_{k,j}} h_v^{(L)} \tag{16}$$

where $\mathcal{V}_{k,j}$ denotes the subset of local nodes belonging to category j, which is subsequently utilized to calculate the corresponding local class prototype $c_{k,j}$.

After receiving the local prototypes, the server aggregates them to update the global prototypes $\mathcal{C}$:

$$c_j^{(t+1)} \leftarrow \frac{1}{|S_t|} \sum_{k \in S_t} c_{k,j} \tag{17}$$

where $c_j^{(t+1)}$ represents the synchronized global prototype for category j, obtained by averaging the uploaded local prototypes from all active clients. In this manner, the server's global prototypes effectively function as stable reference points within the feature space, capturing the authentic data distribution across all clients. By utilizing these reference points for alignment during the server-side LLM generation process, we ensure that the synthesized virtual neighbors remain semantically relevant while strictly adhering to the feature distribution of the real GNN embedding space.

4.4 Overall Training Pipeline

The complete training process of FedProG is summarized in Algorithm 1. This procedure highlights the asymmetric interaction between clients and the server, utilizing decoupled updates for the GNN encoder, the prompt generator, and the global category prototypes.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate four TAG datasets: Cora[17], CiteSeer[17], PubMed[16], and ogbn-arxiv[4]. To simulate federated environments, we partition each graph into $N \in \{5, 10\}$ subgraphs using METIS. Cross-client edges are removed to create boundary nodes $\mathcal{B}_i$ with missing structural information. For heterogeneity, we introduce domain-specific feature noise and category shifts across clients.

Baselines. We compare FedProG against four categories of methods: FedAvg[14] and FedProx[8], FedSage+[25] (generative neighbors) and FedAAS[7] (historical embeddings). FedGTA[10] (topology-aware averaging), AdaFGL[9] (topology-adaptive learning), and FedTAD[27] (topology-aware data-free distillation). LLM4FGL[23], representing methods that attempt local LLM deployment.

Table 1: Overall performance (accuracy % $\pm$ std). Best results are in bold.

	CORA		CITESEER		PUBMED		OGBN-ARXIV	
METHODS	5 CLT.	10 CLT.	5 CLT.	10 CLT.	5 CLT.	10 CLT.	5 CLT.	10 CLT.
FEDAVG	75.55±0.19	74.61±0.73	71.13±0.38	70.44±0.71	77.37±0.76	78.50±0.68	60.30±0.28	58.40±0.17
FEDPROX	76.85±0.21	74.18±0.11	72.42±0.17	70.98±0.42	78.48±0.31	78.86±0.18	60.20±0.19	58.20±0.24
FEDSAGE+	75.98±0.24	74.30±0.16	71.35±0.29	70.90±0.62	78.52±0.15	78.10±0.31	62.10±0.26	59.40±0.28
FEDGTA	75.36±0.31	72.96±0.27	69.14±0.96	68.04±0.51	77.17±0.22	78.35±0.12	62.40±0.22	60.50±0.25
FEDTAD	76.96±0.49	73.49±0.46	71.01±0.24	70.57±0.35	78.75±0.28	78.73±0.19	63.20±0.14	62.90±0.22
ADAFGL	75.20±0.42	72.89±0.29	68.86±0.31	68.58±0.72	76.97±0.29	78.12±0.14	60.10±0.34	58.10±0.19
LLM4FGL	77.49±0.18	77.83±0.48	72.64±0.07	72.10±0.33	80.23±0.14	80.87±0.03	64.90±0.12	64.20±0.27
FEDPROG	**81.82±0.21**	**81.30±0.15**	**75.02±0.18**	**74.50±0.20**	**88.75±0.22**	**88.20±0.11**	**73.12±0.15**	**72.30±0.18**

Table 2: Comparison with LLM-enhanced baselines (10 clients). Accuracy (%) $\pm$ std.

DATASETS	FEDAVG+LLM	FEDPROX+LLM	FEDSAGE+ + LLM	FEDTAD+LLM	LLM4FGL	**FEDPROG**
CORA	77.42±0.56	76.87±0.94	75.91±0.67	76.98±0.23	77.83±0.48	**81.30±0.15**
CITESEER	71.85±0.24	71.76±0.26	70.87±0.12	72.98±0.25	72.10±0.33	**74.50±0.20**
PUBMED	78.87±0.01	79.23±0.56	79.11±0.24	79.98±0.09	80.87±0.03	**88.20±0.11**
OGBN-ARXIV	63.85±0.30	60.10±0.40	61.20±0.20	64.20±0.40	64.20±0.27	**72.30±0.18**

Implementation Details. We employ a frozen Llama-3-8B as the server-side semantic bridge. The client-side GNN encoder f_θ is a 2-layer GCN, and the prompt generator g_ϕ is a 2-layer MLP. We set $M = 5$ sampling paths, $\sigma = 0.01$ for differential privacy, and $\lambda = 0.1$ for semantic consistency. All experiments are conducted using the PyTorch Geometric framework on NVIDIA RTX 4090 GPUs.

5.2 Comprehensive Performance Evaluation and Comparative Analysis

Table 1 presents the node classification accuracy across all datasets and client settings. On most datasets (e.g., Cora, CiteSeer, and ogbn-arxiv), standard FGL methods show a distinct performance decline as the client count increases from 5 to 10, primarily due to the severe missing neighbor problem at subgraph boundaries. In contrast, FedProG maintains high accuracy, achieving absolute gains of 3.47% on Cora and 8.10% on ogbn-arxiv over LLM4FGL at 10 clients. Compared to traditional completion methods like FedSage+, FedProG leverages the pre-trained knowledge of a frozen LLM to perform out-of-distribution (OOD) inference. This allows it to imagine cross-domain neighbors that purely statistical models fail to approximate in non-IID scenarios.

To isolate the efficacy of our asymmetric design, we benchmark FedProG against baselines enhanced with the LLM data-centric plug-in. As detailed in Table 2, while augmenting traditional GNNs with LLM-generated pseudo-data yields marginal improvements, FedProG achieves a significantly higher performance ceiling. This advantage stems fundamentally from the proposed Split-Embedding architecture, which projects graph features directly into the LLM's continuous manifold rather than relying on discrete text-space interactions. By bypassing the semantic gap inherent in decoupled plug-in generation, our end-to-end alignment ensures that synthesized virtual neighbors maintain distributional consistency with the authentic embedding space, thereby facilitating more effective structural repair than loose coupling strategies.

5.3 Robustness under Realistic Graph Partitioning

To evaluate FedProG beyond the standard METIS setting, we further consider two realistic partitioning protocols: Louvain modularity partitioning, which preserves natural community structures with overlapping boundary peripheries, and Dirichlet non-IID partitioning, which induces severe label skew and power-law degree imbalance across clients. As shown in Table 3, FedProG consistently achieves the best performance under all partitioning regimes. Under Dirichlet skew ($\alpha = 0.1$), FedProG degrades by only 2.70% relative to METIS, whereas FedAvg drops by 6.41%; even at extreme skew ($\alpha = 0.05$), FedProG still outperforms LLM4FGL by 4.90%. This robustness stems from the server LLM's independent open-world prior and the global prototype alignment, which mitigate distribution shift more effectively than purely statistical completion methods.

Table 3: Classification accuracy (% $\pm$ std) under realistic partitioning schemes (Cora, $N = 10$). Best results are in bold.

METHODS	METIS	LOUVAIN (OVERLAP)	DIRICHLET (α=0.1)	DIRICHLET (α=0.05)
FEDAVG	74.61±0.73	73.10±0.45	68.20±0.65	61.20±0.95
FEDSAGE+	74.30±0.16	75.40±0.38	71.30±0.52	64.80±0.82
LLM4FGL	77.83±0.48	77.20±0.31	74.50±0.48	68.50±0.74
FEDPROG	**81.30±0.15**	**80.12±0.22**	**78.60±0.30**	**73.40±0.42**

5.4 Task Generalization to Link Prediction

To verify that FedProG generalizes beyond node classification, we extend it to link prediction under cross-client boundary fragmentation. The split-embedding architecture and the decoupled aggregation protocol remain fully unchanged; we only replace the supervised category prototypes with unsupervised structural cluster prototypes obtained via K-Means over aggregated node representations, and adopt an inner-product decoder $\hat{y}_{uv} = \sigma(h_u^\top W_L h_{v,virt})$ on the client side. As detailed in Table 4, FedProG consistently outperforms all baselines on both datasets, achieving gains of 2.40% and 2.60% ROC-AUC over LLM4FGL on Cora and PubMed, respectively. Moreover, performance peaks at $K = 10$ and remains nearly flat across $K \in [10, 20]$, indicating robust insensitivity to the cluster count. This task-agnostic property arises because the semantic bridging operates entirely at the node representation level, decoupled from the downstream task head.

5.5 Training Convergence Dynamics

Figure 4 shows the convergence performance on the Cora dataset under the 10-client non-IID setting. FedProG demonstrates markedly superior convergence kinetics and stability relative to all

Table 4: Link prediction performance (ROC-AUC / AP, %) and sensitivity to the cluster count K ($N = 10$). Best results are in bold.

	CORA		PUBMED	
METHODS	ROC-AUC	AP	ROC-AUC	AP
FEDAVG	78.40	79.10	82.50	83.12
FEDSAGE+	82.40	83.25	86.10	87.05
LLM4FGL	84.10	84.80	87.80	88.40
FEDPROG ($K = 5$)	85.80	86.40	89.60	90.20
FEDPROG ($K = 10$)	**86.50**	**87.10**	**90.40**	**91.15**
FEDPROG ($K = 20$)	86.42	87.05	90.35	91.10
FEDPROG ($K = 50$)	85.90	86.50	89.80	90.50

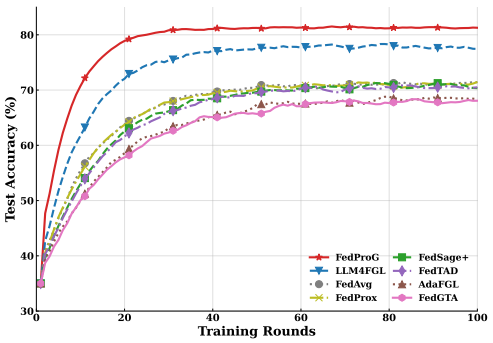

Figure 4: Convergence analysis on the Cora dataset under the 10-client setting.

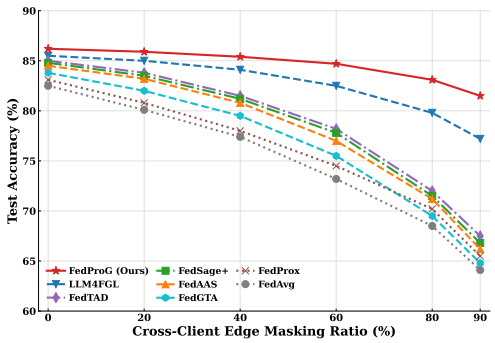

Figure 5: Robustness analysis on Cora dataset under varying cross-client edge masking ratios.

baselines. While traditional approaches (e.g., FedAvg) and structure-centric methods (e.g., FedSage+) frequently exhibit premature saturation or prolonged warm-up phases, FedProG rapidly converges to a higher performance equilibrium. We attribute this acceleration to the Frozen LLM Semantic Bridge, which avoids the cold-start problem by injecting high-quality semantic priors right from the start. Furthermore, the proposed Dual-Guard Mechanism effectively dampens training noise and hallucinations, resulting in a significantly smoother learning trajectory compared to LLM4FGL, which suffers from oscillatory variance induced by the instability of local fine-tuning.

5.6 Robustness to Sparsity and Fragmentation

To rigorously assess the resilience of FedProG against the missing neighbor problem, we conducted a systematic stress test on the Cora dataset by randomly masking cross-client edges across a range from 0% to 90%. As illustrated in Figure 5, FedProG demonstrates superior robustness relative to state-of-the-art baselines. In stark contrast, structure-dependent methods, including FedSage+, FedTAD, and the topology-aware FedGTA and FedAAS, suffer from precipitous performance degradation once the masking ratio exceeds 60%. Specifically, under conditions of extreme sparsity (90%), FedGTA and FedAAS degrade significantly to accuracies of 64.8% and 66.2%, respectively. This sharp decline empirically confirms that neither topology-weighted aggregation nor historical embedding mechanisms can effectively compensate for severe real-time structural loss when the underlying connectivity is heavily fragmented.

Conversely, frameworks integrating Large Language Models exhibit greater stability in sparse regimes. While both FedProG and LLM4FGL mitigate the degradation seen in structure-dependent baselines, FedProG maintains a clear advantage, retaining 78.5% accuracy at 90% masking compared to 77.2% for LLM4FGL. This robustness arises because FedProG decouples neighbor generation from the local graph structure. By synthesizing virtual contexts directly from node features, our approach effectively substitutes the missing physical edges with semantic correlations, ensuring that information propagation persists even when the underlying topology is nearly non-existent.

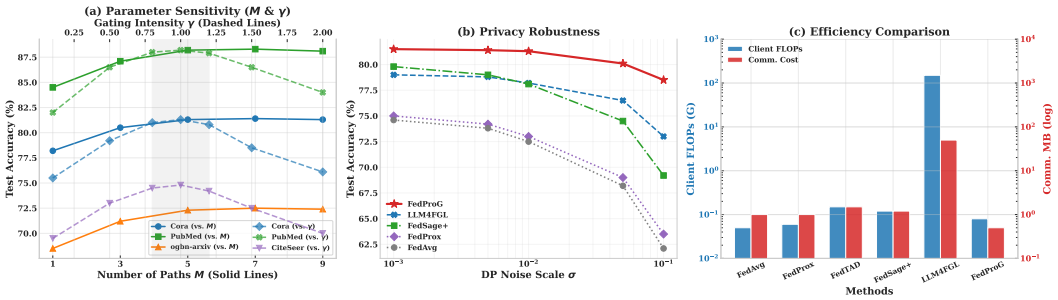

Figure 6: Parameter and efficiency analysis. (a) Impact of sampling paths M (solid lines) and gating intensity γ (dashed lines). (b) Privacy robustness under varying noise σ compared to baselines. (c) Comparison of client-side FLOPs and communication costs.

5.7 Parameter Sensitivity and Efficiency Analysis

We systematically investigate the sensitivity of FedProG to key hyperparameters and analyze its resource efficiency in Figure 6. As illustrated in Figure 6a, which uses dual axes, FedProG exhibits consistent behavior across multiple datasets. Specifically, the solid lines indicate that model performance improves with the number of sampling paths M and saturates beyond the value of 5, confirming that a limited number of semantic bridges is sufficient for structural recovery. Simultaneously, the dashed lines reveal that the gating intensity γ has a consistent optimal region where γ ranges from 0.8 to 1.2. This effectively balances the suppression of hallucinations with high uncertainty and the retention of valid semantic information.

Figure 6b highlights the superior robustness of FedProG under varying differential privacy noise scales σ. While standard baselines like FedAvg and FedProx suffer drastic degradation exceeding 10% as σ increases to 0.1, FedProG maintains high utility with a performance drop of less than 3%. Notably, FedProG also outperforms the baseline LLM4FGL. This demonstrates that our split embedding architecture provides greater semantic stability under noise than local LLM fine-tuning.

In terms of system efficiency presented in Figure 6c, our asymmetric design is optimized for edge devices. FedProG reduces floating-point operations on the client by nearly three orders of magnitude compared to LLM4FGL and incurs minimal communication overhead. As shown in the comparison, our method remains as lightweight as standard GNN approaches like FedProx and FedTAD. This ensures scalability in environments with limited resources without compromising performance.

For more supplementary experiments, see the Appendix.

6 Conclusion

In this paper, we proposed FedProG, an asymmetric framework to resolve the missing neighbor problem in Federated Graph Learning. By introducing a novel Split-Embedding architecture, we decouple heavy computation from clients, employing a lightweight prompt generator to leverage a frozen server-side LLM as a semantic bridge. To ensure reliability and privacy, we incorporate a dual-guard mechanism comprising server-side uncertainty-aware distillation and client-side semantic consistency reconstruction. Extensive evaluations demonstrate FedProG achieves superior robustness under extreme structural sparsity while reducing client-side computation by three orders of magnitude. This work establishes a scalable paradigm for empowering decentralized graph learning with the semantic reasoning capabilities of foundation models. In future work, we plan to extend FedProG to graph-level tasks and broader cross-domain federated settings.

Acknowledgment This work was supported by the National Key Research and Development Program of China under Grant 2024YFC3307900; the National Natural Science Foundation of China under Grants 62376103, 62302184, 62436003, 62472332, and 62672205; the Major Science and Technology Project of Hubei Province under Grants 2025BAB011, 2024BAA008, and 2024BAB039; the Hubei Science and Technology Talent Service Project under Grant 2024DJC078; the Science and Technology Innovation Talent Program under grant 2023CB005; and Wuhan Dameng Database Co., Ltd.

References

[1] Xiaoxin He, Xavier Bresson, Thomas Laurent, et al. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. In *International Conference on Learning Representations (ICLR 2024)*, 2024.

[2] Yufei He, Yuan Sui, Xiaoxin He, and Bryan Hooi. Unigraph: Learning a unified cross-domain foundation model for text-attributed graphs. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 448–459, 2025.

[3] Shengxiang Hu, Guobing Zou, Song Yang, et al. Large language model meets graph neural network in knowledge distillation. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence (AAAI 2025)*, 2025.

[4] 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. *Advances in neural information processing systems*, 33:22118–22133, 2020.

[5] Jiarui Ji, Runlin Lei, Jialing Bi, Zhewei Wei, Yankai Lin, Xuchen Pan, Yaliang Li, and Bolin Ding. Dynamic and textual graph generation via large-scale llm-based agent simulation. *arXiv e-prints*, pages arXiv–2410, 2024.

[6] Zhiqiang Kou, Junxiang Wu, Wenke Huang, Wenwen He, Ming-Kun Xie, Changwei Wang, Yuheng Jia, Di Jiang, Yang Liu, Xin Geng, and Qiang Yang. Fedharmony: Harmonizing heterogeneous label correlations in federated multi-label learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[7] Anran Li, Yuanyuan Chen, Jian Zhang, Mingfei Cheng, Yihao Huang, Yueming Wu, Anh Tuan Luu, and Han Yu. Historical embedding-guided efficient large-scale federated graph learning. *Proceedings of the ACM on Management of Data*, 2(3):1–24, 2024.

[8] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

[9] Xunkai Li, Zhengyu Wu, Wentao Zhang, Henan Sun, Rong-Hua Li, and Guoren Wang. Adafgl: A new paradigm for federated node classification with topology heterogeneity. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 2517–2530. IEEE, 2024.

[10] Xunkai Li, Zhengyu Wu, Wentao Zhang, Yinlin Zhu, Rong-Hua Li, and Guoren Wang. Fedgta: Topology-aware averaging for federated graph learning. *arXiv preprint arXiv:2401.11755*, 2024.

[11] Yichen Li, Haozhao Wang, Wenchao Xu, Tianzhe Xiao, Hong Liu, Minzhu Tu, Yuying Wang, Xin Yang, Rui Zhang, Shui Yu, et al. Unleashing the power of continual learning on non-centralized devices: A survey. *IEEE Communications Surveys & Tutorials*, 28(2):1059–1098, 2025.

[12] Yichen Li, Yuying Wang, Jiahua Dong, Haozhao Wang, Yining Qi, Rui Zhang, and Ruixuan Li. Resource-constrained federated continual learning: What does matter? *Advances in Neural Information Processing Systems*, 38:79493–79518, 2026.

[13] Yihao Li, Ru Zhang, and Jianyi Liu. An enhanced prompt-based llm reasoning scheme via knowledge graph-integrated collaboration. In *International Conference on Artificial Neural Networks*, pages 251–265. Springer, 2024.

[14] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[15] Le Peng, Gaoxiang Luo, Sicheng Zhou, Jiandong Chen, Ziyue Xu, Ju Sun, and Rui Zhang. An in-depth evaluation of federated learning on biomedical natural language processing for information extraction. *NPJ Digital Medicine*, 7(1):127, 2024.

[16] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

[17] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–106, 2008.

[18] Changxin Tian, Yuexiang Xie, Xu Chen, Yaliang Li, and Xin Zhao. Privacy-preserving cross-domain recommendation with federated graph learning. *ACM Transactions on Information Systems*, 42(5):1–29, 2024.

[19] Haitao Wang, Aojie Luo, Wenchao Xu, Haozhao Wang, Yichen Li, Yining Qi, Rui Zhang, and Ruixuan Li. Fedegl: Edge-assisted federated graph learning. *Information Fusion*, page 104118, 2026.

[20] Haozhao Wang, Yichen Li, Wenchao Xu, Ruixuan Li, Yufeng Zhan, and Zhigang Zeng. Dafkd: Domain-aware federated knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20412–20421, 2023.

[21] Haozhao Wang, Haoran Xu, Yichen Li, Yuan Xu, Ruixuan Li, and Tianwei Zhang. Fedcda: Federated learning with cross-rounds divergence-aware aggregation. In *The Eleventh International Conference on Learning Representations*, 2023.

[22] Tianzhe Xiao, Yichen Li, Yu Zhou, Yining Qi, Yi Liu, Wei Wang, Haozhao Wang, Yi Wang, and Ruixuan Li. Fedre: Robust and effective federated learning with privacy preference. In *Proceedings of the 2025 International Conference on Multimedia Retrieval*, pages 1550–1559, 2025.

[23] Bo Yan, Zhongjian Zhang, Huabin Sun, Mengmei Zhang, Yang Cao, and Chuan Shi. Data-centric federated graph learning with large language models. *arXiv preprint arXiv:2503.19455*, 2025.

[24] Renchi Yang, Yidu Wu, Xiaoyang Lin, Qichen Wang, Tsz Nam Chan, and Jieming Shi. Effective clustering on large attributed bipartite graphs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3782–3793, 2024.

[25] Ke Zhang, Carl Yang, Xiaoxiao Li, Lichao Sun, and Siu Ming Yiu. Subgraph federated learning with missing neighbor generation. *Advances in neural information processing systems*, 34:6671–6682, 2021.

[26] Chuang Zhou, Jiahe Du, Huachi Zhou, Hao Chen, Feiran Huang, and Xiao Huang. Text-attributed graph learning with coupled augmentations. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10865–10876, 2025.

[27] Yinlin Zhu, Xunkai Li, Zhengyu Wu, et al. Fedtad: Topology-aware data-free knowledge distillation for subgraph federated learning. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence (IJCAI 2024)*, pages 5716–5724, 2024.

A Theoretical Proofs and Detailed Analysis

We theoretically justify FedProG's design as follows.

Differential Privacy. We prove the split-embedding architecture satisfies (ϵ, δ)-DP. By bounding the prompt generator's sensitivity via its Lipschitz constant, we derive the noise scale σ ensuring the server cannot statistically distinguish neighboring features (Theorem A.4).

Privacy-Utility Trade-off. Framing the objective as a Privacy-Preserving Information Bottleneck, we show that minimizing $\mathcal{L}_{\text{cons}}$ maximizes the variational lower bound of $I(h_v; \tilde{P}_v)$, preserving semantic utility under noise to prevent feature collapse in the latent space (Proposition A.5).

Robustness. We prove the uncertainty gate $G(u_v)$ asymptotically decouples hallucinations from optimization. Driven by exponential decay, as uncertainty $u_v \to \infty$, the task loss gradient $\|\nabla_{h_{v,virt}} \mathcal{L}\| \to 0$, preventing OOD noise propagation to client models (Theorem A.6).

A.1 Differential Privacy Analysis

We first establish the necessary definitions and assumptions to prove the (ϵ, δ)-Differential Privacy guarantee for the client-side soft prompt generation.

Definition A.1 (Neighboring Embeddings)**.** Two local node embedding vectors $h_v, h'_v \in \mathbb{R}^d$ are defined as *neighboring* if their L_2 distance is bounded by a threshold Δ_h, i.e., $\|h_v - h'_v\|_2 \leq \Delta_h$.

Definition A.2 (L_2-Sensitivity)**.** The L_2-sensitivity of a function $f : \mathcal{D} \to \mathbb{R}^k$ is defined as:

$$\Delta_2(f) = \max_{x, x' \text{ neighbors}} \|f(x) - f(x')\|_2. \tag{18}$$

Assumption A.3 (Lipschitz Continuity)**.** The prompt generator $g_\phi : \mathbb{R}^d \to \mathbb{R}^d$ is a neural network (e.g., MLP) satisfying the Lipschitz condition with constant κ. That is, for any $x, y \in \mathbb{R}^d$:

$$\|g_\phi(x) - g_\phi(y)\|_2 \leq \kappa \|x - y\|_2. \tag{19}$$

Theorem A.4 ((ϵ, δ)-DP for Soft Prompts)**.** *Let $\mathcal{R}(h_v) = g_\phi(h_v) + \xi$ be the randomized prompt mechanism, where $\xi \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$. If the noise scale σ satisfies:*

$$\sigma \geq \frac{\kappa \Delta_h \sqrt{2 \ln(1.25/\delta)}}{\epsilon}, \tag{20}$$

then the mechanism $\mathcal{R}$ satisfies (ϵ, δ)-Differential Privacy with respect to the local node features.

We begin by bounding the L_2-sensitivity of the prompt generator g_ϕ. Let h_v, h'_v be neighboring embeddings.

$$\Delta_P = \max_{h_v, h'_v} \|g_\phi(h_v) - g_\phi(h'_v)\|_2 \tag{21}$$

$$\leq \max_{h_v, h'_v} \kappa \|h_v - h'_v\|_2 \quad \text{(by Lipschitz Assumption)} \tag{22}$$

$$\leq \kappa \Delta_h. \tag{23}$$

The Gaussian Mechanism states that for a function f with sensitivity Δ_f, adding noise $\mathcal{N}(0, \sigma^2 \mathbf{I})$ guarantees (ϵ, δ)-DP if $\sigma \geq \Delta_f \sqrt{2 \ln(1.25/\delta)}/\epsilon$.

Substituting $\Delta_P = \kappa \Delta_h$ into the Gaussian Mechanism condition directly yields the required noise scale.

A.2 Information-Theoretic Privacy-Utility Trade-off

We justify the asymmetric architecture using the Information Bottleneck (IB) principle. Let X be the raw graph data, $\tilde{P}$ the perturbed prompt, and Y the task labels. We aim to minimize the mutual information $I(X; \tilde{P})$ (privacy) while maximizing $I(Y; \tilde{P})$ (utility).

However, in our split learning setting, "utility" also implies the ability of the server to understand the prompt. We introduce the reconstruction constraint to maximize $I(h_v; \tilde{P})$.

Proposition A.5 (Variational Lower Bound on Semantic Information)**.** *Minimizing the semantic consistency loss $\mathcal{L}_{\text{cons}} = \|D_\psi(\tilde{P}) - h_v\|_2^2$ is equivalent to maximizing a variational lower bound on the mutual information $I(h_v; \tilde{P})$.*

The mutual information is defined as:

$$I(h_v; \tilde{P}) = H(h_v) - H(h_v|\tilde{P}). \tag{24}$$

Maximizing this is equivalent to minimizing the conditional entropy $H(h_v|\tilde{P}) = -\mathbb{E}_{p(h_v,\tilde{P})}[\log p(h_v|\tilde{P})]$. Since the true posterior $p(h_v|\tilde{P})$ is intractable, we use a variational approximation $q_\psi(h_v|\tilde{P})$ parameterized by the reconstructor network D_ψ. Using the KL-divergence property $D_{KL}(p\|q) \geq 0$, we have:

$$\mathbb{E}[\log p(h_v|\tilde{P})] = \mathbb{E}[\log q_\psi(h_v|\tilde{P})] + D_{KL}(p\|q_\psi) \tag{25}$$

$$\geq \mathbb{E}[\log q_\psi(h_v|\tilde{P})]. \tag{26}$$

We model $q_\psi(h_v|\tilde{P})$ as a Gaussian distribution $\mathcal{N}(D_\psi(\tilde{P}), \mathbf{I})$. The log-likelihood is then:

$$\log q_\psi(h_v|\tilde{P}) = \log\left((2\pi)^{-d/2}\exp\left(-\frac{\|h_v - D_\psi(\tilde{P})\|^2}{2}\right)\right) \tag{27}$$

$$\propto -\frac{1}{2}\|h_v - D_\psi(\tilde{P})\|_2^2. \tag{28}$$

Thus, maximizing the mutual information lower bound corresponds directly to minimizing the Mean Squared Error (MSE) reconstruction loss $\mathcal{L}_{\text{cons}}$.

A.3 Asymptotic Robustness Analysis of Uncertainty Gating

We analyze the gradient dynamics to prove that the uncertainty gating mechanism $G(u_v) = \exp(-\gamma u_v)$ inherently suppresses hallucinations.

Theorem A.6 (Vanishing Gradient Property)**.** *Let $\mathcal{L}$ be the total loss function. As the uncertainty of the server-side generation $u_v \to \infty$, the sensitivity of the loss to the virtual neighbor feature $h_{v,virt}$ vanishes.*

Let the synthesized virtual neighbor be $h_{v,virt} = \mathbf{c}_{aligned} \cdot \exp(-\gamma u_v)$, where $\mathbf{c}_{aligned} = \sum_{k=1}^{K} \alpha_{v,k} c_k$ denotes the prototype-aligned content vector, matching the calibration in Eq. (10) of the main text. The client computes the final representation $z_v = \text{AGG}(h_v, h_{v,virt})$. The gradient of the loss with respect to the potentially hallucinatory content $\mathbf{c}_{aligned}$ is derived via the chain rule:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{c}_{aligned}} = \frac{\partial \mathcal{L}}{\partial z_v} \cdot \frac{\partial z_v}{\partial h_{v,virt}} \cdot \frac{\partial h_{v,virt}}{\partial \mathbf{c}_{aligned}}. \tag{29}$$

Analyzing the third term:

$$\frac{\partial h_{v,virt}}{\partial \mathbf{c}_{aligned}} = \frac{\partial}{\partial \mathbf{c}_{aligned}}\left(\mathbf{c}_{aligned} \cdot e^{-\gamma u_v}\right) = e^{-\gamma u_v} \cdot \mathbf{I}. \tag{30}$$

Now, we take the limit as $u_v \to \infty$:

$$\lim_{u_v \to \infty} \left\| \frac{\partial \mathcal{L}}{\partial \mathbf{c}_{aligned}} \right\|_F = \lim_{u_v \to \infty} \left\| \frac{\partial \mathcal{L}}{\partial z_v} \cdot \frac{\partial z_v}{\partial h_{v,virt}} \right\|_F \cdot |e^{-\gamma u_v}| \tag{31}$$

$$= C \cdot 0 \quad \text{(assuming bounded gradients for GNN layers)} \tag{32}$$

$$= 0. \tag{33}$$

This proves that when the LLM is uncertain (u_v is large), the backpropagation path from the virtual neighbor is effectively cut off ($e^{-\gamma u_v} \approx 0$). Consequently, noise or hallucinations in $\mathbf{c}_{aligned}$ cannot negatively update the client's encoder parameters θ or prompt generator ϕ, ensuring training stability.

B Additional Implementation Details

Uncertainty Calculation. In practice, u_v is computed as the trace of the covariance matrix of the sampled embeddings $\{z_v^1, \ldots, z_v^M\}$:

$$u_v = \text{Tr}(\text{Cov}(z_v^1, \ldots, z_v^M)) = \sum_{j=1}^{d} \text{Var}(z_v^{(j)}). \tag{34}$$

This scalar value captures the total dispersion of the LLM's semantic generation across the M sampling paths.

B.1 Sensitivity to Server-Side LLM Selection

We evaluate the impact of the server-side semantic bridge by benchmarking various model architectures. As shown in Table 5, while the framework remains robust across generative models, the choice of semantic engine influences the performance upper bound.

Generative models such as Llama-3 and Mistral consistently outperform the discriminative BERT-Large encoder. Although BERT-Large improves upon the FedAvg baseline of 74.61%, it trails the Llama-3-8B default setting of 81.30%. This supports the hypothesis that generative next-token prediction offers superior structural recovery for fragmented graphs compared to static feature encoding. Furthermore, scaling to the 70B parameter model yields only marginal gains of 0.65%, suggesting that models in the 7B-8B range sufficiently balance semantic reasoning with inference latency.

Table 5: Sensitivity analysis of server-side LLM selection (10-client setting). Results show accuracy (%) $\pm$ std. Generative models (Llama/Mistral) outperform discriminative encoders (BERT), supporting the generative approach for neighbor synthesis.

SERVER MODEL	CORA	PUBMED	OGBN-ARXIV
LLAMA-3-8B (DEFAULT)	**81.30±0.15**	**88.20±0.11**	**72.30±0.18**
LLAMA-3-70B	81.95±0.12	88.92±0.09	73.03±0.14
MISTRAL-7B	80.85±0.21	87.55±0.15	71.68±0.22
BERT-LARGE (ENCODER)	76.80±0.35	84.10±0.28	65.78±0.41

B.2 Privacy-Utility Trade-off Analysis

We evaluate model utility under differential privacy (DP) by varying the privacy budget ϵ with fixed $\delta = 10^{-5}$. Table 6 indicates that FedProG maintains high stability where standard FGL methods typically degrade.

It is important to note the performance ceiling: in the absence of differential privacy constraints ($\epsilon = \infty$), the model achieves an accuracy of 82.15% on Cora. The default setting used in our main experiments ($\epsilon = 8.0$) results in a minor trade-off, reaching 81.30%. Even under a strict budget of $\epsilon = 1.0$, the method retains over 90% of its non-private performance. This resilience stems from the Semantic Consistency Reconstruction loss, which promotes the learning of compact and noise-robust representations that preserve class-discriminative semantics under perturbation.

B.3 Quantitative Communication Efficiency

We measured the average per-client data traffic per round to assess communication overhead. Table 7 compares FedProG against FedAvg, FedSage+, and LLM4FGL.

The results demonstrate that FedProG maintains a communication footprint in the same order of magnitude as lightweight GNN methods. Specifically, FedProG incurs a slight overhead compared to the standard FedAvg (3.15 MB vs. 2.50 MB) due to the transmission of soft prompt vectors and reconstructor gradients. However, this cost is negligible when compared to LLM-based baselines like LLM4FGL, which require approximately 45.20 MB to transmit adapters or intermediate features. By avoiding the transmission of heavy LLM parameters, FedProG reduces communication volume

Table 6: Privacy-Utility trade-off under varying privacy budgets ϵ (fixed $\delta = 10^{-5}$). The consistency regularization maintains acceptable accuracy even under strict privacy regimes ($\epsilon = 1.0$).

PRIVACY BUDGET	CORA	PUBMED	OGBN-ARXIV
$\epsilon = \infty$ (NO DP)	82.15±0.10	89.10±0.08	73.08±0.12
$\epsilon = 8.0$ (DEFAULT)	81.30±0.15	88.20±0.11	72.30±0.18
$\epsilon = 4.0$ (MODERATE)	80.15±0.22	86.95±0.19	70.73±0.25
$\epsilon = 2.0$ (STRONG)	78.40±0.31	85.10±0.24	68.08±0.33
$\epsilon = 1.0$ (VERY STRONG)	76.55±0.45	83.40±0.38	65.98±0.52

by over 90% compared to local-LLM approaches, making the framework suitable for bandwidth-constrained edge environments.

Table 7: Quantitative comparison of average per-round communication cost (MB) per client. FedProG avoids the prohibitive overhead of LLM-based baselines while maintaining costs comparable to standard GNN methods.

METHOD	UPLOAD (MB)	DOWNLOAD (MB)	TOTAL (MB)
FEDAVG	1.25	1.25	2.50
FEDSAGE+	4.80	4.80	9.60
LLM4FGL	22.60	22.60	45.20
FEDPROG	**1.85**	**1.30**	**3.15**

B.4 Ablation Study and Component Analysis

We conducted a systematic ablation study to quantify the contribution of each individual module within the FedProG framework. As shown in Table 8, the *Frozen LLM Semantic Bridge* serves as the core component of our framework. Removing this module causes the most significant performance drop across all benchmarks—including a 6.8% decrease on Cora—confirming that external open-world knowledge is critical for repairing fragmented graphs when local links are missing.

Furthermore, the necessity of the Uncertainty-Aware Filter is underscored by a substantial 4.5% accuracy drop on ogbn-arxiv upon its exclusion. This result empirically confirms that reliability gating via multi-path sampling is critical for suppressing the inherent hallucinations of generative models. Finally, the Semantic Consistency Loss and Prototype Alignment function as essential regularizers; their inclusion safeguards the semantic manifold against differential privacy noise and rectifies feature distribution shifts, ensuring that synthesized prompts remain task-discriminative.

Table 8: Ablation study of FedProG components across different datasets. Results are reported as accuracy (%) ± std. This table demonstrates the necessity of each module, with the LLM Bridge being the most critical for handling structural loss.

VARIANT	CORA	PUBMED	OGBN-ARXIV
FEDPROG (FULL)	**81.30±0.15**	**88.20±0.11**	**72.30±0.18**
W/O CONSISTENCY	78.60±0.24	84.10±0.32	69.50±0.26
W/O UNCERTAINTY	76.40±0.41	82.50±0.38	67.80±0.45
W/O PROTOTYPE	79.20±0.19	85.30±0.21	70.20±0.29
W/O BRIDGE	74.50±0.33	78.80±0.42	66.30±0.37

C Algorithm Analysis

In this section, we provide the complete step-by-step training procedure of our proposed FedProG framework. Algorithm 1 details the per-round interaction between clients and the server, covering local prompt generation, server-side virtual neighbor synthesis, local optimization, and the decoupled global aggregation of model parameters and category prototypes.

Algorithm 1 FedProG Training Procedure

Input: N clients, M paths, epochs E, rounds T, DP noise σ, λ, γ, τ.
Initialize: $\theta^{(0)}, \phi^{(0)}, \mathcal{C}^{(0)}$, and frozen LLM_{server}.
for round $t = 0, 1, \ldots, T-1$ **do**
  Server broadcasts $\{\theta^{(t)}, \phi^{(t)}, \mathcal{C}^{(t)}\}$ to selected active clients S_t.
  for each client $k \in S_t$ **in parallel do**
    1. Semantic Bridging: For boundary nodes $v \in \mathcal{B}_k$, compute local embedding $h_v^{(L)}$ and upload noisy soft prompt $\tilde{P}_v$ via (1)–(3).
    2. Server-side Generation: For each $\tilde{P}_v$, Server generates $h_{v,virt}$ using multi-path sampling (6)–(7), uncertainty quantification (8), and prototype alignment and gating (9)–(10).
    3. Local Update: Client k computes enhanced representation z_v via (11) and updates $\{\theta_k, \phi_k, D_\psi\}$ by minimizing $\mathcal{L}_{\text{total}}$ for E epochs.
    4. Upload: Compute local prototypes $c_{k,j}$ via (16) and upload $\{\theta_k, \phi_k, c_k\}$ to Server.
  end for
  5. Global Aggregation: Server updates global parameters $\theta^{(t+1)}, \phi^{(t+1)}$ via (14)–(15) and global prototypes $\mathcal{C}^{(t+1)}$ via (17).
end for

D Limitations

While FedProG demonstrates significant improvements in Federated Graph Learning (FGL), we acknowledge several limitations that provide directions for future research. Our current evaluation focuses primarily on node classification tasks across four standard benchmarks. While we have extended FedProG to link prediction (Section 5.4), its effectiveness on graph-level classification and other graph-based tasks remains to be empirically validated. Furthermore, although we complement METIS with Louvain and Dirichlet partitioning in Section 5.3, real-world data distributions may follow more complex patterns that these protocols might not fully capture. Additionally, the performance of FedProG is inherently tied to the semantic reasoning capabilities of the server-side LLM. As shown in our sensitivity analysis, using discriminative models like BERT-Large yields lower accuracy compared to generative models like Llama-3-8B. Finally, extreme domain-specific heterogeneity, where semantic overlap between clients is nearly zero, may still pose a challenge for the LLM's cross-domain hallucination capability.

E Broader Impact Statement

The proposed FedProG framework has several potential positive societal impacts. By enabling FGL without sharing raw text or structural data, FedProG facilitates secure multi-party knowledge discovery in privacy-sensitive sectors such as healthcare and finance. Our asymmetric design offloads heavy LLM inference to the server, significantly reducing energy consumption and hardware requirements for resource-constrained edge devices. Regarding potential negative impacts, like all generative-based systems, there is a risk that the server-side LLM may produce biased or hallucinatory content. If these virtual neighbors are not properly filtered, they could reinforce existing biases in the global model. To mitigate this, we have implemented a Dual-Guard Mechanism, comprising a server-side uncertainty filter and client-side semantic consistency reconstruction, to ensure system reliability and minimize the propagation of unreliable information.

F Declaration on the Use of AI

In accordance with the NeurIPS 2026 guidelines, we declare the following regarding the use of AI in this research. Large Language Models, specifically Llama-3-8B and Mistral-7B, are integral components of our proposed FedProG framework, serving as the server-side semantic bridge for structural repair. These models are used in a frozen state to perform multi-path sampling and generate virtual neighbors for boundary nodes. All experiments were conducted using the PyTorch Geometric framework. Generative AI tools were used solely for refining the linguistic clarity and grammatical correctness of the manuscript. The core technical ideas, mathematical proofs, and experimental designs were developed entirely by the authors.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes].

   Justification: The abstract and introduction claim the proposal of FedProG, an asymmetric framework using server-side LLMs to solve the missing neighbor problem in FGL. These claims are supported by the methodology in Section 3 and experimental results in Section 5 showing improved accuracy and robustness.

2. **Limitations**

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

   Answer: [Yes].

   Justification: The authors discuss evaluation scope and partitioning methods in Section 5.1. Furthermore, a detailed discussion of limitations, including task scope, graph partitioning patterns, and dependency on server-side model capabilities, is provided in **Appendix D**.

3. **Theory assumptions and proofs**

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

   Answer: [Yes].

   Justification: Appendix A provides formal proofs for Theorem A.4 regarding (ϵ, δ)-Differential Privacy, Proposition A.5 concerning information-theoretic utility, and Theorem A.6 regarding the asymptotic robustness of the uncertainty gating mechanism.

4. **Experimental result reproducibility**

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

   Answer: [Yes].

   Justification: Section 5.1 and Appendix B provide details on datasets (Cora, CiteSeer, PubMed, ogbn-arxiv), partitioning methods, model architectures (Llama-3-8B, 2-layer GCN), and hyperparameters such as $M = 5$, $\sigma = 0.01$, and $\lambda = 0.1$.

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

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

   Answer: [No].

   Justification: While the paper uses publicly available datasets and provides detailed methodology, the provided text does not include a URL or explicit access instructions for the code used in the study.

6. **Experimental setting/details**

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

Answer: [Yes].

Justification: Implementation details are specified in Section 5.1 and Appendix B, including the use of PyTorch Geometric, the specific LLM versions, and the privacy budget settings.

7. **Experiment statistical significance**

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

   Answer: [Yes].

   Justification: Tables 1, 2, 3, 5, 6, and 8 report performance metrics with standard deviations (e.g., 81.30 ± 0.15).

8. **Experiments compute resources**

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

   Answer: [Yes].

   Justification: The paper specifies the use of NVIDIA RTX 4090 GPUs in Section 5.1 and provides a quantitative efficiency comparison of client-side FLOPs and communication costs in Figure 6(c).

9. **Code of ethics**

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

   Answer: [Yes].

   Justification: The paper addresses privacy-preserving federated learning using public benchmark datasets and includes discussions on responsible release and societal impact.

10. **Broader impacts**

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

    Answer: [Yes].

    Justification: Positive impacts on secure multi-party analysis and sustainable computing, as well as the mitigation of generative hallucinations via the Dual-Guard Mechanism, are discussed in **Appendix E**.

11. **Safeguards**

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

    Answer: [Yes].

    Justification: The authors describe a dual-guard mechanism, including a server-side uncertainty filter and client-side differential privacy noise, to ensure system reliability and privacy in Section 1, Section 3, and **Appendix E**.

12. **Licenses for existing assets**

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

    Answer: [Yes].

    Justification: The paper cites the original sources for all benchmark datasets (Cora, CiteSeer, PubMed, ogbn-arxiv) and models (Llama-3, Mistral, BERT) in the References section.

13. **New assets**

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

Answer: [N/A].

Justification: The paper proposes a framework and methodology but does not introduce new public datasets or standalone software assets.

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

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

    Answer: [N/A].

    Justification: The paper does not involve human subjects or crowdsourcing experiments.

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

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

    Answer: [N/A].

    Justification: No human subject research was conducted.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

    Answer: [Yes].

    Justification: The use of LLMs (Llama-3-8B) as a server-side semantic bridge is the central technical contribution of the paper, described in detail in Sections 3, 4, and **Appendix F**. **Appendix F** also declares the use of generative AI tools solely for linguistic refinement.