FedGTA: Topology-aware Averaging for Federated Graph Learning

Xunkai Li
Beijing Institute of Technology
cs.xunkai.li@gmail.com

Zhengyu Wu
Beijing Institute of Technology
Jeremywzy96@outlook.com

Wentao Zhang
Mila - Québec AI Institute
HEC Montréal
wentao.zhang@mila.quebec

Yinlin Zhu
Sun Yat-sen University
ylzhuawesome@163.com

Rong-Hua Li
Beijing Institute of Technology
lironghuabit@126.com

Guoren Wang
Beijing Institute of Technology
wanggrbit@126.com

ABSTRACT

Federated Graph Learning (FGL) is a distributed machine learning paradigm that enables collaborative training on large-scale subgraphs across multiple local systems. Existing FGL studies fall into two categories: (i) FGL Optimization, which improves multi-client training in existing machine learning models; (ii) FGL Model, which enhances performance with complex local models and multi-client interactions. However, most FGL optimization strategies are designed specifically for the computer vision domain and ignore graph structure, presenting dissatisfied performance and slow convergence. Meanwhile, complex local model architectures in FGL Models studies lack scalability for handling large-scale subgraphs and have deployment limitations. To address these issues, we propose Federated Graph Topology-aware Aggregation (FedGTA), a personalized optimization strategy that optimizes through topology-aware local smoothing confidence and mixed neighbor features. During experiments, we deploy FedGTA in 12 multi-scale real-world datasets with the Louvain and Metis split. This allows us to evaluate the performance and robustness of FedGTA across a range of scenarios. Extensive experiments demonstrate that FedGTA achieves state-of-the-art performance while exhibiting high scalability and efficiency. The experiment includes ogbn-papers100M, the most representative large-scale graph database so that we can verify the applicability of our method to large-scale graph learning. To the best of our knowledge, our study is the first to bridge large-scale graph learning with FGL using this optimization strategy, contributing to the development of efficient and scalable FGL methods.

PVLDB Reference Format:
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.
doi:10.14778/3617838.3617842

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/xkLi-Allen/FedGTA.

Proceedings of the VLDB Endowment, Vol. 17, No. 1 ISSN 2150-8097.
doi:10.14778/3617838.3617842

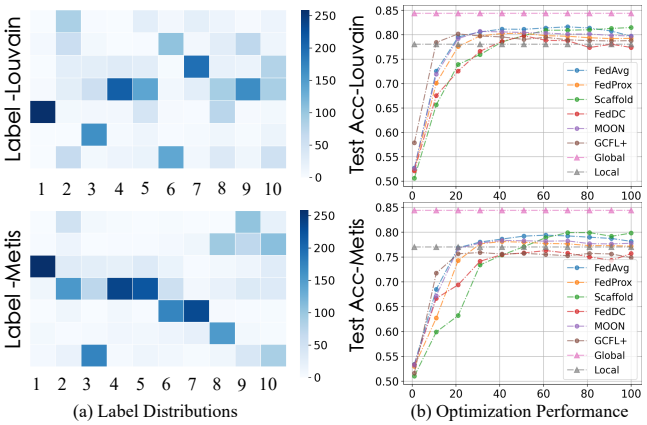
1 INTRODUCTION

Graphs are extensively utilized for modeling complex systems, primarily due to their ability to visually represent the relational information between different entities, which sets them apart from other types of data. Its growing prevalence in recommendation systems [20, 53], drug discovery [33, 43], and financial risk control [10, 40] urges the development of a correspondent graph analysis tool. Graph Neural Networks (GNNs) emerge as a promising approach to achieve state-of-the-art performance in node-level [23, 48, 58], edge-level [4, 6, 56], and graph-level [37, 44, 51] downstream tasks.

Researchers in the database community have recently focused on developing centralized data-driven pipelines of large-scale graph learning [21, 31]. However, the surge in retrieved graph data for real-world applications has generated greater interest in the decentralized settings [28, 36, 38] for large-scale graph learning [22, 27, 34]. Specifically, collecting data across different locations and sources often requires efforts from different institutions, among which information sharing may be impeded for legal or competitive reasons. For example, the disease network [39] and online transaction network [15] require the participation of multiple local clients, such as hospitals and regional institutions.

To address the above issues, one promising solution is Federated Graph Learning (FGL), a distributed training framework for GNNs. FGL approaches involve devices training their own local models using self-collected data, upon which the central server achieves optimization to obtain a global model. Despite preserving privacy, FGL can also overcome computational and storage limitations by dealing with large-scale subgraphs through scalable local models and effective optimization strategies. In a nutshell, we can summarize the current studies in FGL into the following two types: (i) FGL Optimization: applying improved federated optimization methods to the existing graph learning models; (ii) FGL Model: designing local model architectures and multi-client interactions.

A proper optimization strategy is critical in achieving multi-client collaborative training. FedAvg [35], a simple yet effective optimization strategy, which performs weighted model aggregation based on the proportional weights of the data size of participating clients against the total combined data size. Although FedAvg is topology-independent and primarily intended for CNNs or MLP, many existing FGL Model studies [7, 19, 55] still apply it and enhance their performance by well-designed model architectures.



(a) Label Distributions (b) Optimization Performance

Figure 1: Empirical analysis on Cora in 10 clients with GCN, which contains 7 different node labels. (a) The color from white to blue represents the number of nodes held by different clients in each class a gradual increase in quantity. (b) The x-axis of the line plot is the federated training round. "Global" and "Local" represent the model performance in centralized and siloed settings, respectively.

However, these models have limited scalability, which ultimately restricts their real-world applicability. To illustrate further, we present an experiment in Fig. 1, which reveals two major FGL limitations: (i) lack of investigation of federated subgraph distribution; (ii) the absence of topology-aware optimization strategies.

The first limitation underscores the label Non-independent identical distribution (Non-iid) problem in FGL as shown in Fig. 1(a), where we chose Louvain [5] and Metis [26], two widely applied federated subgraphs simulation methods based on community search. As the homogeneity assumption suggests for most real-world graphs, linked nodes are similar in both feature distributions and labels [1, 24, 49]. Such premises lead to the results concluded in Fig. 1(a). We use different colors to highlight the varying numbers of nodes distributed through the data simulation in each client. We observe that each client exhibits distinctive or similar label distributions (label Non-iid). In FGL, the label distributions held by clients are often significantly different, and this property requires extra emphasis since the aggregation between clients regardless of label distribution can lead to unsatisfying results. In fact, this is the essential reason for the sub-optimal performance of the existing FGL Model [3, 7, 11, 47, 55]. To maximize efficiency, we require a personalized optimization strategy that selects clients with similar label distribution instead of applying permutations of all possibilities. To the best of our knowledge, it has not yet been specifically addressed.

The second limitation is that most currently adopted optimization strategies overlook the topology of the graph. In Fig. 1(b), we conduct a subsequent experiment to detect the label Non-iid problem illustrated in Fig. 1(a). The results show that methods like FedProx [30], Scaffold [25], MOON [29], and FedDC [16], which are applied to solve label Non-iid problems in the computer vision domain, do not achieve competitive results compared to FedAvg and local train. GCFL+ [50] assumes that the graph topology is implicitly incorporated into the local model through uploaded gradients, but this approach fails to fully capture the topology and leads to sub-optimal performance. These results motivate us to investigate why conventional methods that perform well in the computer vision domain fail to replicate their success in FGL. Our conclusion is that these methods do not directly consider the topology properties, which are crucial elements in graph studies. In light of this finding, our main motivation is to design a topology-aware optimization strategy that specifically targets the label Non-iid problem in the federated graph collaborative training process.

In this paper, we propose Federated Graph Topology-aware Aggregation (FedGTA), a novel and scalable optimization strategy, for FGL and present state-of-the-art performance in efficiency. Specifically, each client calculates topology-aware local smoothing confidence and mixed moments of neighbor features and uploads them with correspondent model weights to the server. Then, the server is able to customize the optimization strategy for each participating client to continue federated training.

In summary, the main contributions of this paper are: (1) **Problem Connection.** We introduce a novel perspective for integrating large-scale graph learning with FGL. (2) **New Method.** We propose FedGTA, a novel topology-aware optimization strategy for FGL. It has been formulated into a unified framework that can be applied to any graph learning model. (3) **SOTA Performance.** We conduct experiments on 12 real-world benchmark datasets including the archetypal large-scale graph dataset, ogbn-papers100M, with prevalent GNNs. We demonstrate that FedGTA significantly outperforms the state-of-the-art baselines on both performance and efficiency.

2 PRELIMINARIES AND RELATED WORKS

In this section, we first describe the notations and problem formulation in this paper. Then we briefly discuss the difference between conventional and scalable GNNs and the FGL Optimization/Model studies. Meanwhile, summarized in Table 1, we present an analysis on the complexity bounds of existing FGL studies.

2.1 Problem Formulation

Consider a graph $G = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = n$ nodes and $|\mathcal{E}| = m$ edges, the adjacency matrix (including self-loops) is $\hat{\mathbf{A}} \in \mathbb{R}^{n \times n}$, the feature matrix is $\mathbf{X} = \{x_1, \ldots, x_n\}$ in which $x_v \in \mathbb{R}^f$ represents the feature vector of node v, and f represents the dimension of the node attributes. Besides, $\mathbf{Y} = \{y_1, \ldots, y_n\}$ is the label matrix, where $y_v \in \mathbb{R}^{|\mathcal{Y}|}$ is a one-hot vector and $|\mathcal{Y}|$ represents the number of the classes. The semi-supervised node classification task is based on the topology of labeled set $\mathcal{V}_L$ and unlabeled set $\mathcal{V}_U$, and the nodes in $\mathcal{V}_U$ are predicted with the model supervised by $\mathcal{V}_L$.

2.2 Conventional and Scalable GNNs

Graph Neural Networks (GNNs) adopt spectral graph theory and deep learning to enable graph learning. Specifically, propagation operators are defined based on topology, while trainable weights are applied to learn node attributes. In this study, we present two standards of GNNs that differ in model architecture design and their corresponding implications in real-world application scenarios.

Conventional GNNs. GCN [46] and GAT [42] are two widely used methods that employ coupled message-passing schemes to propagate information across the nodes. However, when dealing with real-world applications involving large-scale graphs, scalability becomes a major concern due to their limited capacity.

Scalable GNNs. There are two major approaches to achieving GNN scalability. One is developing sampling-based GNNs, such as

Table 1: Algorithm analysis for existing FGL Optimization/Model studies. n, m, c, and f are the number of nodes, edges, classes, and feature dimensions, respectively. s is the number of selected augmented nodes and g is the number of generated neighbors. b and T are the batch size and dynamic training round, respectively. k and K correspond to the number of times we aggregate features and moments order, respectively. Besides, N is the number of participating clients in each training round. For model-agnostic optimization strategies, we choose SGC as the local model (k-step feature propagation), and for FGL methods, we adopt the model architecture (L-layer) used in their original paper.

Method	Type	Client Mem.	Server Mem.	Inference Mem.	Client Time.	Server Time.	Inference Time
FedAvg	Optim.	$O((b+k)f+f^2)$	$O(N+Nf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2)$	$O(N)$	$O(kmf+nf^2)$
FedProx	Optim.	$O((b+k)f+2f^2)$	$O(N+Nf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+f^2)$	$O(N)$	$O(kmf+nf^2)$
Scaffold	Optim.	$O((b+k)f+2f^2)$	$O(N+2Nf^2+f^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+f^2)$	$O(N+Nf^2+f^2)$	$O(kmf+nf^2)$
MOON	Optim.	$O((3b+k)f+f^2)$	$O(N+Nf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+2nf)$	$O(N)$	$O(kmf+nf^2)$
FedDC	Optim.	$O((b+k)f+4f^2)$	$O(N+2Nf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+4f^2)$	$O(N)$	$O(kmf+nf^2)$
GCFL+	Optim.	$O((b+k)f+f^2)$	$O(N+Nf^2+TNf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2)$	$O(N+N^2(\log(N)+T^2f^2))$	$O(kmf+nf^2)$
FedGL	Model	$O(Lnf+Lf^2+n^2)$	$O(N+NLf^2)$	$O(Lnf+Lf^2+n^2)$	$O(Lmf+Lnf^2+n^2f)$	$O(N)$	$O(Lmf+Lnf^2+n^2f)$
FedSage+	Model	$O(L(n+sg)f+3Lf^2)$	$O(N+3NLf^2)$	$O(L(n+sg)f+3Lf^2)$	$O(L(m+sg)f+L(n+sg)f^2)$	$O(N)$	$O(L(m+sg)f+L(n+sg)f^2)$
FedGTA (ours)	Optim.	$O((b+k)f+f^2+kKc)$	$O(N+Nf^2+NkKc)$	$O((b+k)f+f^2)$	$O(km(f+knc)+n(f^2+c))$	$O(N+NkKc)$	$O(kmf+nf^2)$

GraphSAGE [18] randomly samples neighbors for computation in each mini-batch, Fast-GCN [8] samples a fixed number of nodes at each layer, and Cluster-GCN [12] is implemented based on graph-level clustering. However, recent studies [13, 57, 58] emphasize a decoupled mechanism due to its simple operative mechanism and superior performance. For example, SGC [48] reduces GNNs into a linear model operating on k-layers propagated features $\mathbf{X}^{(k)}$:

$$\mathbf{X}^{(k)} = \tilde{\mathbf{A}}^k\mathbf{X}^{(0)},\ \tilde{\mathbf{A}} = \hat{\mathbf{D}}^{r-1}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-r},\ \mathbf{Y} = \text{softmax}\left(\mathbf{\Theta}\mathbf{X}^{(k)}\right), \quad (1)$$

where $\mathbf{X}^{(0)} = \mathbf{X}$, $\hat{\mathbf{D}}$ is the degree matrix of $\hat{\mathbf{A}}$, $r \in [0,1]$ denotes the propagation kernel coefficient, and $\mathbf{W}$ represents weight matrix. By default $r = 0.5$, we can get the symmetric normalization adjacency matrix $\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}$ [17]. As the propagated features $\mathbf{X}^{(k)}$ can be precomputed, SGC is easy to scale to large graphs. Inspired by it, SIGN [14] proposes to concatenate the learnable propagated features: $\left[\mathbf{X}^{(0)}\mathbf{W}_0, \ldots, \mathbf{X}^{(k)}\mathbf{W}_k\right]$. S^{2}GC [59] proposes to average the spectral features: $\mathbf{X}^{(k)} = \sum_{l=0}^{k}\tilde{\mathbf{A}}^l\mathbf{X}^{(0)}$. GBP [9] further utilizes the β weighted averaging: $\mathbf{X}^{(k)} = \sum_{l=0}^{k} w_l\tilde{\mathbf{A}}^l\mathbf{X}^{(0)}, w_l = \beta(1-\beta)^l$. GAMLP [58] achieves information aggregation based on the attention mechanisms $\mathbf{X}^{(k)} = \tilde{\mathbf{A}}^k\mathbf{X}^{(0)} \| \sum_{l=0}^{k-1} w_l\mathbf{X}^{(l)}$, where attention weight w_l has multiple calculation versions.

2.3 FGL Optimization

FedAvg [35] is the most widely used optimization strategy for FL. It implements model aggregation by using a simple weighted average of the model parameters received from each participating client. The weights are proportional to the training data size. Its generic form with N participating clients and learning rate η is defined as

$$\begin{aligned}\widetilde{\mathbf{W}}^t &= \sum_{i=1}^{N}\frac{n_i}{n}\mathbf{W}_i^{t-1}, \forall i, \mathbf{W}_i^{t-1} = \widetilde{\mathbf{W}}^{t-1} - \eta\nabla f,\\ \mathbf{W}_i^t &= \widetilde{\mathbf{W}}^t - \eta\nabla f\left(\widetilde{\mathbf{W}}^t, (\mathbf{A}_i, \mathbf{X}_i, \mathbf{Y}_i)\right)\\ &= \widetilde{\mathbf{W}}^t + \eta\sum_{i\in\mathcal{V}_l}\sum_j \mathbf{Y}_{ij}\log\left(\text{softmax}(\hat{\mathbf{Y}})_{ij}\right),\end{aligned} \quad (2)$$

where n_i and n represent the i-th client and the global data size, $\nabla f(\cdot)$ represents the gradients. It can be obtained by any reasonable loss function that evolves with downstream tasks. $\mathbf{W}_i^t$ and $\widetilde{\mathbf{W}}^t$ represent the local model weights held by the i-th client in round t and the aggregated model weights received from the server.

Despite its effectiveness, FedAvg fails to solve the weight-shifting problem caused by Non-iid data. To address this challenge, several methods have been proposed, but they largely focus on the computer vision domain rather than on the graphs. FedProx [30] limits the deviation of the local model from the global model. Scaffold [25] introduces server and client control variables to control the model's updated direction. MOON [29] introduces model-contrastive loss in the local training. FedDC [16] utilizes learnable local drift variables to bridge the above gap. GCFL+ [50] utilizes weight clustering techniques to custom model aggregation in graph classification.

2.4 FGL Model

Graph data has been demonstrated to be superior in multiple applications. Recently, FGL [19, 32, 45] has received a lot of attention due to its unique advantage in training GNN models collaboratively without sharing collected data for safety concerns.

As mentioned in Section 1, there are two main strategies for improving FGL studies. The first strategy involves optimizing the FGL process and applying it to existing GNNs. We discussed some previous works related to this in Section 2.3. The second strategy involves improving the model architectures and using FedAvg as the default method, presented by FedSage, FedGNN, and FedGL. FedSage+ [55] implements local subgraph augmentation via the missing neighbor generator. FedGNN [47] attempts to propose a federated graph recommendation model with security guarantees. FedGL [7] proposes to use the overlapping subgraph nodes to implement global supervision among multi-clients.

To further illustrate, we provide the algorithmic complexity of each method in Table 1, where "Optim." represents FGL Optimization and "Model" denotes FGL Models. For the k-layer SGC model with batch size b, the precomputed results are bounded by a space complexity of $O((b+k)f)$. The overhead for linear regression is $O(f^2)$. For the server performing FedAvg, it needs to receive the model weights and the number of samples participating in this round. Its space complexity and time complexity are bounded as $O(N+Nf^2)$ and $O(N)$. As discovered by previous studies [9, 57, 58], the dominating term is $O(kmf)$ or $O(Lmf)$ when the graph is large since feature learning can be accelerated by parallel computation. The full large-graph propagation becomes extremely difficult and thus, FedGL and FedSage+ lead to unacceptable space-time overhead because of the $O(n^2)$ term and $O(m+n+2sg)$ term.

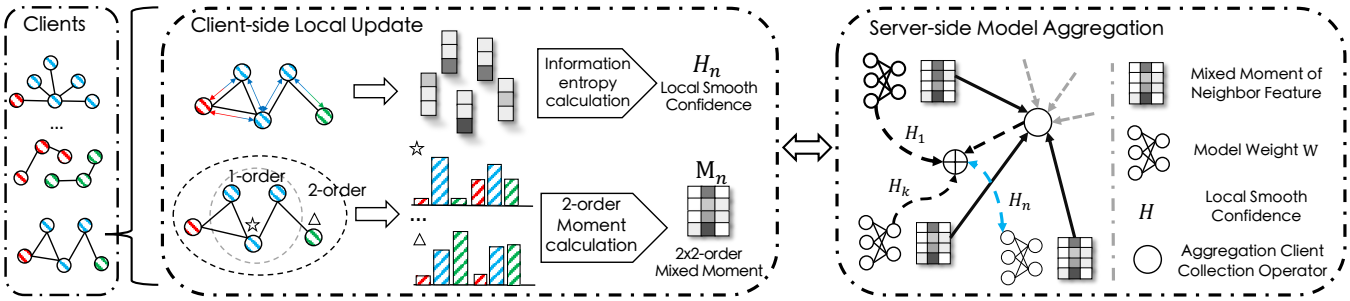


Figure 2: Overview of our proposed FedGTA framework. The different colors of the nodes represent the different labels.

3 FEDGTA FRAMEWORK

As an optimization strategy, FedGTA provides a novel perspective that integrates both large-scale graph learning and FGL into its design as shown in Fig. 2. To begin with, on the **Client-side**, participating clients encode topology and node attributes. After that, they calculate the local smoothing confidence and mixed moments of neighbor features, which are then uploaded to the server. On the **Server-side**, FedGTA performs personalized model aggregation for each client based on the mixed moments of neighbor features and uses the local smoothing confidence as the aggregation weights.

Local Smoothing Confidence. Since GNNs trained by the smoothing graph are more confident with their prediction [17, 24, 57], we grant it with higher contribution in the model aggregation process.

Mixed Moments of Neighbor Features. By using the mixed moments of neighbor features to measure the distribution of subgraphs, we can limit the model aggregation process to only those clients with similar subgraph distributions.

3.1 The Proposed FedGTA

To encode topology and node attributes, we introduce k-step Non-parameters Label Propagation (Non-param LP), which establishes relationships among the current node and its k-hop neighbors

$$\hat{\mathbf{Y}} = \text{Softmax}\left(\text{Encoder}(\mathbf{A}, \mathbf{X})\right),$$
$$\hat{\mathbf{Y}}^k(v_i) = \alpha\hat{\mathbf{Y}}^0(v_i) + (1-\alpha)\sum_{j\in\mathcal{N}_i}\frac{1}{\sqrt{\tilde{d}_i\tilde{d}_j}}\hat{\mathbf{Y}}^{k-1}(v_j), \tag{3}$$

where $\text{Encoder}(\cdot,\cdot)$ represent any embedding model and $\hat{\mathbf{Y}} \in \mathbb{R}^{n\times|\mathcal{Y}|}$ denotes the soft label matrix. We follow the approximate calculation of the personalized PageRank [17], where $\mathcal{N}_i$ represents the one-hop neighbors of i. We default set $\alpha = 1/2, k = 5$ to encode deep structural information. Then, we obtain the topology-aware soft label matrix. Based on this, each client calculates the quantitative metrics to perform model optimization on the server side. The complete algorithm can be obtained as described in Algorithm1.

Client-Local Smoothing Confidence. The key insight is to quantify smoothness by the entropy of local predictions

$$H = \sum_{i=1}^{|\mathcal{V}|}\sum_{j=1}^{|\mathcal{Y}|}\mathbf{D}_{ii}\left(e^{-1} - \left(-\hat{\mathbf{Y}}_{ij}^k \log \hat{\mathbf{Y}}_{ij}^k\right)\right). \tag{4}$$

Since the model based on the smoothing subgraph tends to produce clearer predictions with a lower entropy value, modification is needed as we want to represent higher confidence figuratively. Therefore we subtract it from the theoretical maximum e^{-1}, and then we obtain H by considering both the local neighbors and the number of samples, with the addition of the degree matrix $\mathbf{D}_{ii}$.

Algorithm 1 FedGTA-Client Update

```
for each communication round t = 1, ..., T do
    for each local model update e = 1, ..., E do
        Update local model weights W according to the Eq (2);
    end for
    Calculate the topology-aware label distribution by Eq (3);
    Local Smoothing Confidence:
    Calculate the entropy of soft label predictions;
    Execute smoothness quantification H based on the Eq (4);
    Mixed Moments of Neighbor Features:
    Calculate the K-order moments M based on the Eq (5);
    Each client uploads the relevant H, M, and model weight W;
end for
```

Algorithm 2 FedGTA-Server Aggregation

```
for each communication round t = 1, ..., T do
    for each client i = 1, ..., N do
        Calculate the set of model aggregation for the current
        client i based on M, ε, and Eq (6);
        Execute federated model optimization based on the model
        aggregation sets I_i, H_i, and Eq (7) to get W̃_i;
        Server sends global model W̃_i to each local client i;
    end for
end for
```

Client-Mixed Moments of Neighbor Features. Here we introduce the mixed moments of neighbor features, which is used to generalize local subgraph. Specifically, we compute the K-order mixed moments of k-step propagated soft labels $\mathbf{M} \in \mathbb{R}^{(k\times K)\times|\mathcal{Y}|}$. We present the formal representation of central moments as an example

$$\mathbf{M}(\hat{y}_i^k) = \mathbb{E}\left(\left(\hat{y}_i^1 - \mu_i^1\right)^1\right)||\ldots||\mathbb{E}\left(\left(\hat{y}_i^k - \mu_i^k\right)^K\right),$$
$$\mathbb{E}\left(\left(\hat{y}_i^k - \mu_i^k\right)^K\right) = \left(\alpha\hat{y}_i^0 + (1-\alpha)\sum_{j\in\mathcal{N}_i}\frac{1}{\sqrt{\tilde{d}_i\tilde{d}_j}}\hat{y}_j^{k-1} - \frac{1}{|\mathcal{Y}|}\sum\hat{y}_i^k\right)^K, \tag{5}$$

where $\cdot||\cdot$ is concatenation and μ_i^1 denotes the mean value of $\hat{y}_i^1$. Due to the differences in the subgraphs held by clients, it becomes imperative to employ appropriate quantification metrics, guiding the server toward performing model aggregation tailored to each client. Recognizing the significance of topology in graphs and its potential correlation with features, we utilize the mixed moments of neighbor features to realize topology-aware quantification of discrepancies among multi-client subgraphs. After that, participating clients upload their metrics and model weights to the server.

Server-Model Aggregation. Based on the above metrics, the server computes the similarity of the mixed moments of neighbor features and utilizes a data-driven context threshold to obtain the set of other clients for which the current client performs model aggregation. The intuition is to avoid clients with high variability interfering with each other during the federated collaborative training process. The computation process is formally defined as

$$\begin{aligned} &\mathbb{I}_i = \{j|\text{sim}(i,j) \geq \epsilon\} \cup i,\ \forall i,j \in \text{Set}(N),\ j \neq i, \\ &\text{sim}(i,j) = \frac{\sum_{p=1}^{k\times K} \mathbf{M}_i^p \cdot \mathbf{M}_j^p}{\sqrt{\sum_{p=1}^{k\times K}(\mathbf{M}_i^p)^2}\sqrt{\sum_{p=1}^{k\times K}(\mathbf{M}_j^p)^2}}, \end{aligned} \tag{6}$$

where ϵ denotes the threshold and $\mathbb{I}_i$ represents the model aggregation set for client i. $\text{Set}(N)$ represents the set of all participating clients in the current round. In fact, the cosine similarity in Eq (6) can be replaced with any reasonable metric. Based on this, the server performs personalized weighted model aggregation for each participating client based on the local smoothing confidence

$$\forall i, \widetilde{\mathbf{W}}_i^{t+1} \leftarrow \sum_{j,k\in\mathbb{I}_i} \frac{H_i}{\sum H_j}\mathbf{W}_j^t,\ \mathbf{W}_j^t \leftarrow \widetilde{\mathbf{W}}_j^t - \eta\nabla f. \tag{7}$$

Notably, with the involvement of mixed moments of neighbor features, the model aggregation process based on the comparison of local smoothing confidence can become more accurate and efficient as it will only consider the client with similar subgraph distribution. The complete algorithm can be referred to as Algorithm2.

3.2 Complexity Analysis

We provide the complexity analysis of our proposed FedGTA and other FGL Optimization/Model studies in Table 1. For the client side of FedGTA, the computation complexity for calculating the entropy for each row of soft label $\hat{\mathbf{Y}} \in \mathbb{R}^{n\times c}$ is $O(nc)$, and the computation complexity for moments is $O(k^2mnc)$. On the server side, the computation complexity for calculating the cosine similarity of N clients' moments is $O(NkKc)$. The computation complexity of our method only depends on the training-independent model-agnostic sparse matrix multiplication, while other FGL optimization strategies adopt coupled training mechanisms, whose additional loss terms lead to excessive computation cost accompanied by the local training process. More analysis and experiments of algorithmic complexity can be referred to Section 4.5.

4 EXPERIMENTS

In this section, we conduct a wide range of experiments to verify the effectiveness of FedGTA. To begin with, we introduce 12 graph benchmark datasets as the global graph and two subgraph simulation strategies widely used in the FGL. Then, we introduce the baseline backbone GNNs and FGL approaches and detailed experimental setup. After that, we aim to answer the following questions: **Q1**: Compared with other state-of-the-art FGL Optimization/Model studies, can FedGTA achieve better performance? **Q2**: What is the generalization ability of FedGTA in the field of FGL? **Q3**: Where does the performance gain of FedGTA come from? **Q4**: How does FedGTA perform in terms of efficiency and scalability?

4.1 Experimental Setup

Datasets. For a comprehensive comparison, we evaluate FedGTA and other baselines under both transductive and inductive settings. For transductive settings, we conduct experiments on 3 small-scale citation networks (Cora, Citeseer, PubMed) [52], 2 medium-scale user-item datasets (Amazon Computer, Amazon Photo), 2 medium-scale Coauthor datasets (Coauthor CS, Coauthor Physics) [41], and 3 large-scale OGB datasets (ogbn-arxiv, ogbn-products, ogbn-papers100M) [22]. For inductive settings, we conduct experiments on 2 datasets of medium and large scales: Flickr and Reddit [54]. More details about the above datasets can be found in Table 2. Based on this, we provide Louvain [5] and Metis [26] split, which are widely used in FGL [3, 19, 45, 55]. Specifically, we apply Louvain on the global graph to assign discovered communities to multi-clients. In the Metis split, we assign nodes to each client based on the given number of clients. Notably, since the ogbn-papers100M dataset contains a large number of unlabeled nodes, we only perform Louvain split on it. This is because we can control the amount of labeled data contained by each client by assigning communities.

Baselines. For FGL Optimization, we compare FedGTA with FedAvg [35], FedProx [30], Scaffold [25], MOON [29], FedDC [16], and GCFL+ [50]. For FGL Model, we conduct comparisons on recently proposed FedGL [7] and FedSage+ [55]. For local models, we utilize simple and scalable GCN [46], GraphSage [18], SGC [48], SIGN [14], S^2GC [59], GBP [9], and GAMLP [58]. Based on this, the results we present are calculated by 10 runs. Unless otherwise stated, we adopt GAMLP as the local model and employ all datasets with Louvain 10 clients split, except for ogbn-papers100M, which is divided into 500 clients. Notably, we experiment with multiple existing scalable GNN models in separate modules to validate the generalizability of our optimization strategy and avoid complex charts, making the results more reader-friendly.

Hyperparameters. The hyperparameters in the local model are set according to the original paper if available. Otherwise, we perform automatic hyperparameter optimization via the Optuna toolkit [2]. Specifically, we explore the optimal values for feature propagation steps (k) and model layers (L) within the ranges of 2 to 20 and 2 to 6. Regarding the percentage of selected augmented nodes and the number of generated neighbors, we conduct a grid search from {0.01, 0.05, 0.1, 0.5} and {2, 5, 10} respectively. The hidden dimension for the small dataset is set to 64 with the number of local epochs set to 3. For medium or large-scale datasets, we set 256, and 5, respectively. We default perform 100 rounds, and the coefficient of the gradient regularization terms is determined through a grid search with values {0.001, 0.01, 0.1}. The optimal window size of gradient dynamic clustering ranges from 2 to 10. For our proposed FedGTA, the order of moments (K) and the similarity threshold (ϵ) are explored within the ranges of 2 to 20 and 0 to 1.

Table 2: The statistical information of the experimental datasets.

Dataset	#Nodes	#Features	#Edges	#Classes	#Train/Val/Test	#Task	Description
Cora	2,708	1,433	5,429	7	20%/40%/40%	Transductive	citation network
CiteSeer	3,327	3,703	4,732	6	20%/40%/40%	Transductive	citation network
PubMed	19,717	500	44,338	3	20%/40%/40%	Transductive	citation network
Amazon Photo	7,487	745	119,043	8	20%/40%/40%	Transductive	co-purchase graph
Amazon Computer	13,381	767	245,778	10	20%/40%/40%	Transductive	co-purchase graph
Coauthor CS	18,333	6,805	81,894	15	20%/40%/40%	Transductive	co-authorship graph
Coauthor Physics	34,493	8,415	247,962	5	20%/40%/40%	Transductive	co-authorship graph
ogbn-arxiv	169,343	128	2,315,598	40	60%/20%/20%	Transductive	citation network
ogbn-products	2,449,029	100	61,859,140	47	10%/5%/85%	Transductive	co-purchase graph
ogbn-papers100M	111,059,956	128	1,615,685,872	172	1200k/200k/146k	Transductive	citation network
Flickr	89,250	500	899,756	7	44k/22k/22k	Inductive	image network
Reddit	232,965	602	11,606,919	41	155k/23k/54k	Inductive	social network

Table 3: Transductive performance on FGL Optimization/Model studies. "OOM" stands for out-of-memory error. "Global" represents the training and inference using the complete global graph under centralized conditions. The best result is bold. The second result is underlined.

Model	Optimization	Cora	CiteSeer	PubMed	Amazon Photo	Amazon Computer	Coauthor CS	Coauthor Physics	ogbn arxiv	ogbn products	ogbn papers100M
GCN	Global	84.6±0.3	72.1±0.2	90.3±0.1	92.8±0.3	84.3±0.4	92.5±0.2	93.1±0.6	73.8±0.3	76.3±0.2	OOM
	FedAvg	80.7±0.3	68.4±0.3	85.9±0.1	89.6±0.5	80.3±0.4	87.4±0.3	88.5±0.5	66.7±0.4	71.7±0.2	58.4±0.3
	FedProx	80.5±0.2	68.7±0.3	85.8±0.1	88.8±0.7	80.5±0.6	87.5±0.6	88.6±0.7	67.1±0.5	72.6±0.3	58.7±0.4
	Scaffold	81.3±0.4	68.3±0.3	85.7±0.2	89.5±0.8	80.4±0.5	86.1±0.5	89.3±0.7	66.9±0.7	72.4±0.3	58.8±0.5
	MOON	80.7±0.4	69.0±0.3	85.8±0.1	89.3±0.8	80.2±0.5	87.7±0.5	89.2±0.7	67.3±0.5	72.3±0.2	58.2±0.5
	FedDC	81.0±0.2	68.4±0.2	86.2±0.2	89.7±0.6	80.8±0.5	87.8±0.5	89.1±0.7	67.5±0.5	72.0±0.2	58.8±0.6
	GCFL+	80.5±0.1	68.1±0.2	85.0±0.1	89.9±0.4	79.4±0.3	87.4±0.2	88.6±0.3	66.8±0.2	71.7±0.2	58.0±0.2
	FedGTA	**82.1±0.3**	**70.6±0.3**	**88.0±0.1**	**91.4±0.7**	**82.7±0.5**	**90.0±0.2**	**91.2±0.5**	**70.3±0.4**	**74.8±0.3**	**60.6±0.3**
GAMLP	Global	85.7±0.5	75.9±0.4	91.1±0.1	93.1±0.5	86.0±0.6	93.7±0.4	93.6±1.0	80.5±0.6	84.2±0.3	68.8±0.1
	FedAvg	82.2±0.3	70.7±0.4	86.9±0.1	90.4±0.4	80.6±0.3	89.3±0.4	89.2±0.5	71.4±0.7	79.0±0.3	62.1±0.2
	FedProx	82.0±0.8	70.6±0.6	86.8±0.1	90.3±0.6	80.7±0.5	88.7±0.5	89.3±1.0	72.3±1.1	78.8±0.5	63.2±0.3
	Scaffold	82.6±0.7	71.1±0.5	86.5±0.2	89.8±0.8	80.8±0.8	88.6±0.6	89.4±0.9	71.6±0.8	79.2±0.5	63.1±0.4
	MOON	81.9±0.4	70.9±0.4	87.2±0.2	90.5±0.7	80.5±0.7	89.2±0.5	90.0±0.8	72.5±0.9	79.3±0.5	62.7±0.3
	FedDC	83.0±0.5	70.8±0.5	87.0±0.1	89.6±0.7	80.8±0.6	88.8±0.6	89.8±1.0	71.9±0.8	78.9±0.5	63.0±0.3
	GCFL+	82.7±0.7	71.6±0.3	86.5±0.1	90.5±0.4	80.4±0.3	89.5±0.3	89.2±0.4	71.5±0.3	78.5±0.2	63.0±0.2
	FedGTA	**83.8±0.6**	**74.3±0.6**	**88.4±0.1**	**91.5±0.5**	**83.9±0.4**	**91.2±0.4**	**91.8±0.5**	**74.3±0.7**	**81.6±0.4**	**66.5±0.3**
FedGL	FedAvg	81.1±0.6	70.6±0.6	86.5±0.4	89.7±1.0	81.7±0.8	88.4±0.8	88.8±1.2	71.4±1.5	OOM	OOM
FedSage+	FedAvg	82.7±0.9	72.0±1.0	87.1±0.5	90.7±1.3	82.4±1.5	89.2±1.4	90.0±1.6	71.1±1.8	OOM	OOM

Experiment Environment. The experiments are conducted on the machine with Intel(R) Xeon(R) Gold 6230R CPU @ 2.10GHz, and NVIDIA GeForce RTX 3090 with 24GB memory and CUDA 11.8. The operating system is Ubuntu 18.04.6 with 216GB memory.

4.2 Performance Comparison

To answer **Q1**, we report the transductive performance in Table 3, where FedGTA outperforms other baselines. Specifically, compared to the second result, FedGTA achieves an average improvement of 2.33% and 2.54% when using GCN and GAMLP. While FedGL and FedSage+ outperform methods using GCN as the local model in some cases, they cannot achieve more competitive results and even fail to handle large-scale scenarios due to their limited scalability.

The experiment results in Table 4 show that FedGTA consistently outperforms all the baselines under the inductive setting. Compared to the most competitive MOON and FedDC, FedGTA has a lead of more than 3.5% and 2.2%, respectively. Notably, we only compare FedGTA with other FGL optimization studies (**bold** or underline).

4.3 Generalization

To answer **Q2**, we demonstrate that FedGTA can be applied to a large variety of GNN variants: GCN, GAMLP, SIGN, S^2GC, SGC, GraphSAGE, and GBP are shown in Table 3, Table 4, and Table 6. Building upon the above backbone GNNs, to test the effectiveness of our proposed FedGTA, we evaluate it on both coupled and sampling-based GNN models, which differ in the orderings of feature propagation and transformation. Through the above experiments, we observe that FedGTA consistently outperforms the other FGL optimization baselines in both GNN categories.

As we claimed, FedGTA is an optimization strategy suitable for FGL, and a natural idea is to combine it with the existing FGL Model studies. In Table 5, we present the experimental results of combining FedGTA and other competitive strategies with the FGL Model studies. As shown in the Tabel5, the test accuracy of FedGTA could improve FedGL and FedSage+ by an average of more than 2.5% on three datasets. Therefore, we conclude that FedGTA can generalize to different types of GNNs and existing FGL Model studies well.

Table 4: Inductive performance under 10 clients Metis split.

Model	Optimization	Flickr	Reddit
SIGN	FedAvg	48.10±0.31	91.31±0.10
	FedProx	48.58±0.27	91.15±0.12
	Scaffold	48.98±0.32	90.58±0.08
	MOON	49.34±0.27	91.37±0.11
	FedDC	48.76±0.20	91.46±0.05
	GCFL+	48.58±0.16	90.54±0.07
	FedGTA	**50.89±0.18**	**93.79±0.06**
S^2GC	FedAvg	48.75±0.30	92.31±0.09
	FedProx	48.81±0.18	92.67±0.07
	Scaffold	48.65±0.29	92.39±0.13
	MOON	49.36±0.26	92.65±0.07
	FedDC	48.91±0.26	93.30±0.12
	GCFL+	49.24±0.14	93.06±0.03
	FedGTA	**51.32±0.19**	**95.07±0.08**

Table 5: Performance gain in FGL Model under 10 clients Metis split.

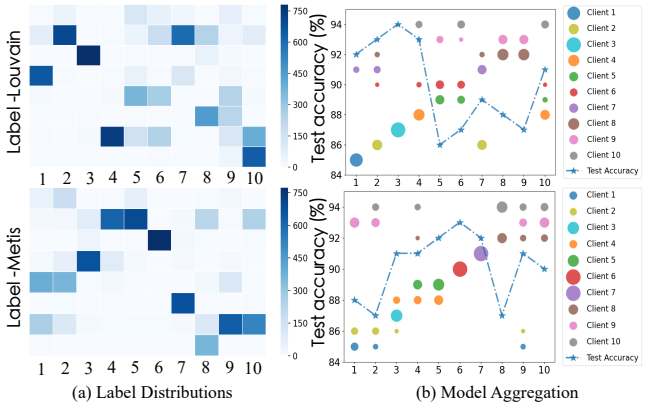
Model	Optimization	ogbn-arxiv	Flickr	Reddit
FedGL	FedAvg	70.5±1.2	47.9±0.3	89.1±0.3
	MOON	70.9±1.4	48.1±0.4	88.7±0.3
	FedDC	70.3±1.9	47.8±0.5	89.3±0.5
	FedGTA	**72.3±0.9**	**50.5±0.3**	**92.0±0.2**
FedSage+	FedAvg	69.8±1.9	48.1±0.5	90.2±0.3
	MOON	70.4±1.7	48.5±0.5	90.2±0.3
	FedDC	69.8±2.1	48.2±0.7	90.4±0.5
	FedGTA	**72.5±1.4**	**51.4±0.4**	**92.9±0.3**

Table 6: Ablation study on three scalable GNN models.

Model	Component	ogbn-products		Reddit	
		Louvain	Metis	Louvain	Metis
SGC	w/o Mom.	72.9±0.3	71.8±0.3	91.4±0.1	91.9±0.1
	w/o Conf.	73.6±0.2	73.1±0.1	92.5±0.1	92.6±0.1
	FedGTA	**74.2±0.1**	**73.6±0.2**	**93.0±0.1**	**93.1±0.1**
GBP	w/o Mom.	77.1±0.5	77.7±0.5	92.4±0.1	92.4±0.1
	w/o Conf.	77.5±0.3	78.1±0.4	93.3±0.1	93.0±0.1
	FedGTA	**78.2±0.3**	**78.7±0.4**	**93.7±0.1**	**93.4±0.1**
SAGE	w/o Mom.	73.8±0.5	73.1±0.3	88.4±0.1	88.0±0.1
	w/o Conf.	75.5±0.2	75.6±0.3	90.2±0.1	89.9±0.1
	FedGTA	**76.8±0.1**	**76.2±0.3**	**90.7±0.1**	**90.2±0.1**

4.4 Method Interpretability

To answer **Q3**, we focus on the implementation of FedGTA on the client and server side. Specifically, we present the ablation experiments shown in Table 6 to investigate the contribution of local smoothing confidence and mixed moments of neighbor features computed on the client side. Subsequently, their efficacy hinges on the server's ability to perform efficient model aggregation, thereby achieving optimized federated training. The visualization of server-side model aggregation based on our method is presented in Fig. 3.



(a) Label Distributions (b) Model Aggregation

Figure 3: The visualization of model aggregation in Amazon Photo with 10 clients split, which contains 8 different node labels. The circle size corresponds to aggregation weight, from large to small.

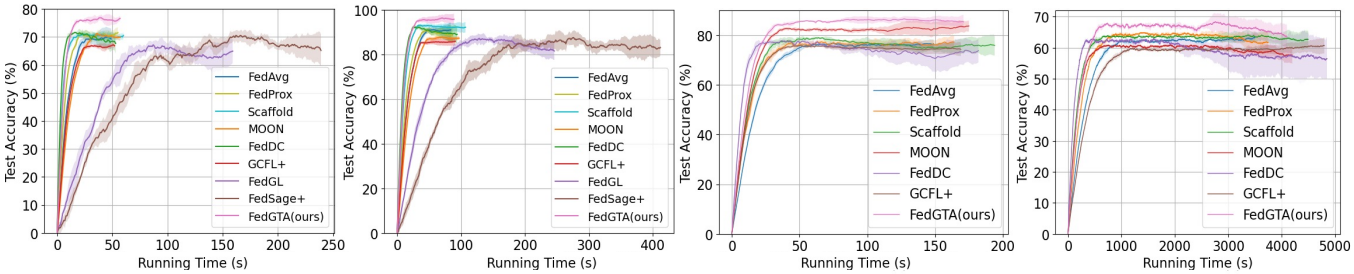
In the ablation study, we use "Mom." and "Conf." to represent the mixed moments of neighbor features and local smoothing confidence. We also use "SAGE" to refer to the GraphSage model. According to the experimental results shown in Table 6, we observe that both Mom. and Conf. contribute to the improvement of the optimization process significantly. Furthermore, the combination of them reduces the variance during the federated training process.

In the visualization part, we analyze the model aggregation process by referring to the label distributions of each client as shown in Fig. 3. Based on our design, we aim to achieve personalized aggregation for each client only combines with those with similar label distribution while the smoothness of each subgraph determines the aggregation weight. Clearly, for the local subgraph, the steeper the label distribution, the more likely it has a smooth topology. This is because connected nodes are more likely to have similar feature distributions or the same label. In Fig. 3(b), we select the best aggregation round for presentation. According to the circle categories in multi-clients, we observe that FedGTA successfully customizes the model aggregation targets for each client. To further clarify, the circle sizes of clients reflect both their local smoothing confidence and corresponding aggregation weights. This demonstrates the efficiency of our method, as subgraphs with more smoothing play a dominant role in the model aggregation process.

4.5 Efficiency and Scalability Analysis

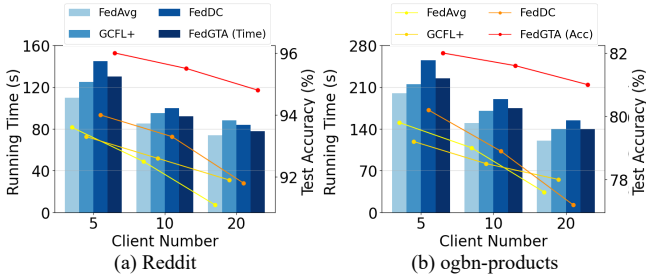
To answer **Q4**, we report the running time in Fig. 4 and Fig. 5, which includes both local training and model aggregation on the server. Notably, FedGL and FedSage+ consume an extended amount of running time due to their complex local model architectures and additional cross-client interactions. In contrast, FGL optimization strategies based on scalable GNNs are more efficient. Among these optimization studies, FedGTA exhibits the most stable and superior performance as evidenced by the curve and shaded trends in Fig. 4.

According to Table 1, the time complexity of FedGTA is mainly related to the model-agnostic sparse matrix multiplication of soft labels ($c \ll f$): $O(k^2mnc)$ and $O(NkKc)$, which is not dependent on the local training process. In contrast, in the client-side model training, FedProx and Scaffold introduce a gradient regularization



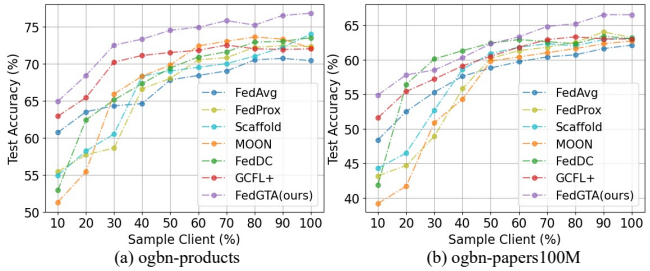
(a) ogbn-arxiv (b) Reddit (c) ogbn-products (d) ogbn-papers100M

Figure 4: Convergence curves of our proposed FedGTA and baseline models on 4 large-scale benchmark graph datasets. Curves represent the local model training phase and model aggregation phase on the server side. The shaded area is the result range of multiple runs.



(a) Reddit (b) ogbn-products

Figure 5: Training efficiency with different numbers of clients.



(a) ogbn-products (b) ogbn-papers100M

Figure 6: Performance with different clients participating.

term $O(f^2)$, MOON and FedDC rely on the outputs from different federated rounds to construct additional model-contrastive loss $O(2nf)$ and model-drift loss $O(4f^2)$. Due to this reason, as the local data scale increases and the models become complex, the training cost becomes larger, leading to an increase in the time complexity and unstable performance of the aforementioned optimization methods. This is validated in Fig. 4 and Fig. 5 of our study. On the server side, Scaffold introduces additional time complexity of $O(Nf^2 + f)$ for updating global control variables. The time complexity term of GCFL+ $O((N^2(\log(N) + T^2f^2))$ becomes sensitive to the window size T used for dynamic gradient clustering and the number of participating clients N. The above inference is also confirmed with Fig. 5. Although FGL optimization strategies, including our methods, share similar inference efficiency, we include the additional experiment discussing the differences in inference efficiency between various GNN models on the ogbn-arxiv dataset with 10-clients Louvain split(time reported in second). For FGL Model studies, FedGL(1.10±0.12) and FedSage(1.73±0.18) proved to be the least efficient and unapplicable for solving scalability. While decoupled GNN models such as SGC(0.12±0.03), SIGN(0.19±0.07), and GAMLP(0.25±0.08) demonstrate less time cost.

In practical scenarios of FGL, there are often a large number of clients, which makes it necessary to select a subset of clients to participate in each round to reduce communication and time-space costs. This amounts to performing Louvain 50 clients split for ogbn-products and 500 clients split for ogbn-papers100M. In Fig. 6, we present the experimental results. According to the experimental results, we conclude that the accuracy level of model and embeddings comparison-based approaches, such as MOON and FedDC, significantly dropped due to the high heterogeneity of subgraphs when the participation ratio is small. In contrast, personalized strategies, such as FedGTA and GCFL+, exhibit robustness. However, compared with FedGTA, the implicit utilization of structural topology in GCFL+ causes its inability to produce competitive performance.

5 CONCLUSION

This paper presents the first integration of large-scale graph learning with FGL, motivated by the need for analyzing real-world applications. Large-scale graph learning can be computationally intensive and space-consuming, which can be effectively solved with FGL due to its decentralized structure. In this paper, we first discuss the flaws of existing FGL approaches. Specifically, FGL Model studies lack scalability due to complex models, and most FGL optimization strategies adopted by FGL fail to recognize the topology. To address the above issues, we propose FedGTA, which is the first topology-aware optimization strategy for FGL. Experimental results demonstrate that FedGTA significantly outperforms competitive baselines in terms of model performance and generalizability.

FedGTA incorporates Non-param LP, which allows for the explicit consideration of both model prediction and topology in each client. This strategy is straightforward and user-friendly. However, a promising avenue for improvement is to leverage additional information provided by local models during training, such as k-layer propagated features. Moreover, we employ personalized model aggregation based on mixed moments of neighbor features' similarity, which has shown effectiveness in a data-driven context. Nevertheless, there is potential for exploring an adaptive aggregation mechanism that considers the impact of topology on the FGL.

ACKNOWLEDGMENTS

This work was partially supported by (i) the National Key Research and Development Program of China 2021YFB3301301, (ii) NSFC Grants U2241211, 62072034, and (iii) CCF-Huawei Populus Grove Fund. Rong-Hua Li is the corresponding author of this paper.

REFERENCES

[1] Sergi Abadal, Akshay Jain, Robert Guirado, Jorge López-Alonso, and Eduard Alarcón. 2021. Computing graph neural networks: A survey from algorithms to accelerators. *ACM Computing Surveys, CSUR* 54, 9 (2021), 1–38.

[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD.*

[3] Jinheon Baek, Wonyong Jeong, Jiongdao Jin, Jaehong Yoon, and Sung Ju Hwang. 2023. Personalized Subgraph Federated Learning. (2023).

[4] Maciej Besta, Raphael Grob, Cesare Miglioli, Nicola Bernold, Grzegorz Kwasniewski, Gabriel Gjini, Raghavendra Kanakagiri, Saleh Ashkboos, Lukas Gianinazzi, Nikoli Dryden, et al. 2022. Motif prediction with graph neural networks. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD.*

[5] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[6] Lei Cai, Jundong Li, Jie Wang, and Shuiwang Ji. 2021. Line graph neural networks for link prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).

[7] Chuan Chen, Weibo Hu, Ziyue Xu, and Zibin Zheng. 2021. FedGL: federated graph learning framework with global self-supervision. *arXiv preprint arXiv:2105.03170* (2021).

[8] Jie Chen, Tengfei Ma, and Cao Xiao. 2018. FastGCN: fast learning with graph convolutional networks via importance sampling. In *International conference on learning representations, ICLR.*

[9] Ming Chen, Zhewei Wei, Bolin Ding, Yaliang Li, Ye Yuan, Xiaoyong Du, and Ji-Rong Wen. 2020. Scalable graph neural networks via bidirectional propagation. *Advances in neural information processing systems, NeurIPS* (2020).

[10] Dawei Cheng, Yi Tu, Zhen-Wei Ma, Zhibin Niu, and Liqing Zhang. 2019. Risk Assessment for Networked-guarantee Loans Using High-order Graph Attention Representation.. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI.*

[11] Tsz-Him Cheung, Weihang Dai, and Shuhan Li. 2021. FedSGC: Federated Simple Graph Convolution for Node Classification. In *International Workshop on Federated and Transfer Learning for Data Sparsity and Confidentiality in Conjuncation with IJCAI.*

[12] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. 2019. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, KDD.*

[13] Wenzheng Feng, Yuxiao Dong, Tinglin Huang, Ziqi Yin, Xu Cheng, Evgeny Kharlamov, and Jie Tang. 2022. Grand+: Scalable graph random neural networks. In *Proceedings of the ACM Web Conference, WWW.*

[14] Fabrizio Frasca, Emanuele Rossi, Davide Eynard, Ben Chamberlain, Michael Bronstein, and Federico Monti. 2020. Sign: Scalable inception graph neural networks. *arXiv preprint arXiv:2004.11198* (2020).

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

[16] Liang Gao, Huazhu Fu, Li Li, Yingwen Chen, Ming Xu, and Cheng-Zhong Xu. 2022. Feddc: Federated learning with non-iid data via local drift decoupling and correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR.*

[17] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Combining Neural Networks with Personalized PageRank for Classification on Graphs. In *International Conference on Learning Representations, ICLR.*

[18] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems, NeurIPS* (2017).

[19] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang, Han Xie, Lichao Sun, Lifang He, Liangwei Yang, Philip S Yu, Yu Rong, et al. 2021. FedGraphNN: A Federated Learning Benchmark System for Graph Neural Networks. In *ICLR 2021 Workshop on Distributed and Private Machine Learning (DPML).*

[20] Li He, Xianzhi Wang, Dingxian Wang, Haoyuan Zou, Hongzhi Yin, and Guandong Xu. 2023. Simplifying Graph-based Collaborative Filtering for Recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, WSDM.*

[21] Mossad Helali, Essam Mansour, Ibrahim Abdelaziz, Julian Dolby, and Kavitha Srinivas. 2022. A Scalable AutoML Approach Based on Graph Neural Networks. *Proc. VLDB Endow.* (2022).

[22] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems, NeurIPS* (2020).

[23] Youpeng Hu, Xunkai Li, Yujie Wang, Yixuan Wu, Yining Zhao, Chenggang Yan, Jian Yin, and Yue Gao. 2021. Adaptive hypergraph auto-encoder for relational data clustering. *IEEE Transactions on Knowledge and Data Engineering* (2021).

[24] Qian Huang, Horace He, Abhay Singh, Ser-Nam Lim, and Austin R Benson. 2021. Combining label propagation and simple models out-performs graph neural networks. *International Conference on Learning Representations, ICLR* (2021).

[25] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning, ICML.*

[26] George Karypis and Vipin Kumar. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing* 20, 1 (1998), 359–392.

[27] Arpandeep Khatua, Vikram Sharma Mailthody, Bhagyashree Taleka, Tengfei Ma, Xiang Song, and Wen-mei Hwu. 2023. IGB: Addressing The Gaps In Labeling, Features, Heterogeneity, and Size of Public Graph Datasets for Deep Learning Research. *arXiv preprint arXiv:2302.13522* (2023).

[28] Runze Lei, Pinghui Wang, Junzhou Zhao, Lin Lan, Jing Tao, Chao Deng, Junlan Feng, Xidian Wang, and Xiaohong Guan. 2023. Federated Learning over Coupled Graphs. *IEEE Transactions on Parallel and Distributed Systems* 34, 4 (2023), 1159–1172.

[29] Qinbin Li, Bingsheng He, and Dawn Song. 2021. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR.*

[30] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems, MLSys* (2020).

[31] Ningyi Liao, Dingheng Mo, Siqiang Luo, Xiang Li, and Pengcheng Yin. 2022. SCARA: Scalable Graph Neural Networks with Feature-Oriented Optimization. *Proceedings of the VLDB Endowment* (2022).

[32] Rui Liu and Han Yu. 2022. Federated Graph Neural Networks: Overview, Techniques and Challenges. *arXiv preprint arXiv:2202.07256* (2022).

[33] Xuan Liu, Congzhi Song, Feng Huang, Haitao Fu, Wenjie Xiao, and Wen Zhang. 2022. GraphCDR: a graph neural network method with contrastive learning for cancer drug response prediction. *Briefings in Bioinformatics* 23, 1 (2022), bbab457.

[34] Zhiwei Liu, Liangwei Yang, Ziwei Fan, Hao Peng, and Philip S Yu. 2022. Federated social recommendation with graph neural network. *ACM Transactions on Intelligent Systems and Technology* 13, 4 (2022), 1–24.

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

[36] Sergi Nadal, Alberto Abelló, Oscar Romero, Stijn Vansummeren, and Panos Vassiliadis. 2021. Graph-driven federated data management. *IEEE Transactions on Knowledge and Data Engineering, TKDE* 35, 1 (2021), 509–520.

[37] Dai Quoc Nguyen, Tu Dinh Nguyen, and Dinh Phung. 2022. Universal graph transformer self-attention networks. In *Companion Proceedings of the Web Conference, WWW.*

[38] Qiying Pan, Yifei Zhu, and Lingyang Chu. 2023. Lumos: Heterogeneity-aware Federated Graph Learning over Decentralized Devices. *arXiv preprint arXiv:2303.00492* (2023).

[39] Liang Peng, Nan Wang, Nicha Dvornek, Xiaofeng Zhu, and Xiaoxiao Li. 2022. Fedni: Federated graph learning with network inpainting for population-based disease prediction. *IEEE Transactions on Medical Imaging* (2022).

[40] Nitendra Rajput and Karamjit Singh. 2022. Temporal Graph Learning for Financial World: Algorithms, Scalability, Explainability & Fairness. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD.*

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

[42] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. In *International Conference on Learning Representations, ICLR.*

[43] Hong Wang, Chong Dai, Yuqi Wen, Xiaoqi Wang, Wenjuan Liu, Song He, Xiaochen Bo, and Shaoliang Peng. 2023. GADRP: graph convolutional networks and autoencoders for cancer drug response prediction. *Briefings in Bioinformatics* 24, 1 (2023), bbac501.

[44] Xiyuan Wang and Muhan Zhang. 2022. How Powerful are Spectral Graph Neural Networks. In *Proceedings of the 39th International Conference on Machine Learning, ICML.*

[45] Zhen Wang, Weirui Kuang, Yuexiang Xie, Liuyi Yao, Yaliang Li, Bolin Ding, and Jingren Zhou. 2022. FederatedScope-GNN: Towards a Unified, Comprehensive and Efficient Package for Federated Graph Learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD.*

[46] Max Welling and Thomas N Kipf. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations, ICLR.*

[47] Chuhan Wu, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. 2021. Fedgnn: Federated graph neural network for privacy-preserving recommendation. In *International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML*.

[48] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International conference on machine learning, ICML*.

[49] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph neural networks in recommender systems: a survey. *ACM Computing Surveys, CSUR* 55, 5 (2022), 1–37.

[50] Han Xie, Jing Ma, Li Xiong, and Carl Yang. 2021. Federated graph classification over non-iid graphs. *Advances in Neural Information Processing Systems, NeurIPS* (2021).

[51] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? *International Conference on Learning Representations, ICLR* (2019).

[52] Zhilin Yang, William Cohen, and Ruslan Salakhudinov. 2016. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning, ICML*. PMLR.

[53] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR*.

[54] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. 2020. Graphsaint: Graph sampling based inductive learning method. In *International conference on learning representations, ICLR*.

[55] Ke Zhang, Carl Yang, Xiaoxiao Li, Lichao Sun, and Siu Ming Yiu. 2021. Subgraph federated learning with missing neighbor generation. *Advances in Neural Information Processing Systems, NeurIPS* (2021).

[56] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. *Advances in neural information processing systems, NeurIPS* (2018).

[57] Wentao Zhang, Yu Shen, Zheyu Lin, Yang Li, Xiaosen Li, Wen Ouyang, Yangyu Tao, Zhi Yang, and Bin Cui. 2022. Pasca: A graph neural architecture search system under the scalable paradigm. In *Proceedings of the ACM Web Conference, WWW*.

[58] Wentao Zhang, Ziqi Yin, Zeang Sheng, Yang Li, Wen Ouyang, Xiaosen Li, Yangyu Tao, Zhi Yang, and Bin Cui. 2022. Graph Attention Multi-Layer Perceptron. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD*.

[59] Hao Zhu and Piotr Koniusz. 2021. Simple spectral graph convolution. In *International conference on learning representations, ICLR*.