Rethinking Federated Graph Learning: A Data Condensation Perspective

Hao Zhang[1,2], **Xunkai Li**[3], **Yinlin Zhu**[4] and **Lianglin Hu**[1]
[1] Computer Network Information Center, Chinese Academy of Sciences, Beijing
[2] University of Chinese Academy of Sciences, Beijing, China
[3] Beijing Institute of Technology, Beijing, China
[4] Sun Yat-sen University, Guangzhou, China
hzhang@cnic.cn, cs.xunkai.li@gmail.com, zhuylin27@mail2.sysu.edu.cn, hull@cnic.cn

Abstract

Federated graph learning is a widely recognized technique that promotes collaborative training of graph neural networks (GNNs) by multi-client graphs. However, existing approaches heavily rely on the communication of model parameters or gradients for federated optimization and fail to adequately address the data heterogeneity introduced by intricate and diverse graph distributions. Although some methods attempt to share additional messages among the server and clients to improve federated convergence during communication, they introduce significant privacy risks and increase communication overhead. To address these issues, we introduce the concept of a condensed graph as a novel optimization carrier to address FGL data heterogeneity and propose a new FGL paradigm called FedGM. Specifically, we utilize a generalized condensation graph consensus to aggregate comprehensive knowledge from distributed graphs, while minimizing communication costs and privacy risks through a single transmission of the condensed data. Extensive experiments on six public datasets consistently demonstrate the superiority of FedGM over state-of-the-art baselines, highlighting its potential for a novel FGL paradigm.

1 Introduction

Graph Neural Networks (GNNs) have emerged as a robust machine learning paradigm to learn expressive representations of graph-structured data through message passing, exhibiting remarkable performance across various AI applications, such as molecular interactions[Huang *et al.*, 2020]. However, most existing GNNs adopt a centralized training strategy where graph data need to be collected together before training. In practical industrial scenarios, large-scale graphs are collected and stored on edge devices. Meanwhile, regulations such as GDPR [Voigt and Von dem Bussche, 2017] highlight the importance of data privacy and impose restrictions on the transmission of local data, which often contains sensitive information. This has led to the exploration of leveraging collective intelligence through distributed data silos to enable collaboration in graph learning [Li *et al.*, 2022a].

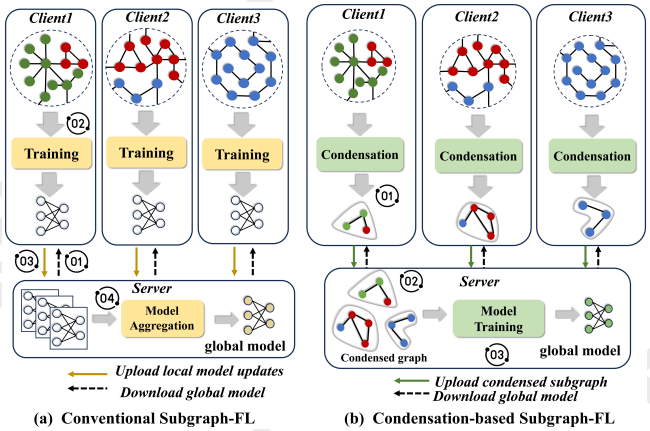


Figure 1: (a) The conventional subgraph-FL framework in the subgraph heterogeneity scenario where node colors represent different labels. (b) The condensation-based subgraph-FL framework, which trains a robust global model by integrating condensed knowledge.

To this end, Federated Graph Learning (FGL) has been proposed, extending Federated Learning (FL) to graph-structured data. Its core idea is to harness collective intelligence for the collaborative training of powerful GNNs, thereby advancing AI-driven insights in federated systems. Given the diversity of graph-based downstream tasks, this paper focuses on subgraph-FL, the instance of FGL on a semi-supervised node classification paradigm. To enhance understanding, we present case study within healthcare systems.

Case Study. In different regions, residents visit various hospitals (e.g., independent clients), all of which are centrally managed by government organizations (e.g., the trusted server). Each hospital maintains a subgraph in its database, containing demographics, living conditions, and patient interactions. These subgraphs form a global patient network. Notably, due to privacy regulations, geographic isolation, and competitive concerns, centralized data storage is not feasible. Fortunately, subgraph-FL enables federated training through multi-client collaboration without direct data sharing. For tasks such as predicting the spread of infections during a pandemic [Bertozzi *et al.*, 2020], developing a federated collaborative paradigm based on distributed scenarios is essential.

Specifically, as illustrated in Fig.1(a), the iterative training

process of conventional subgraph-FL consists of four steps: (1) all clients download the latest global model from the server; (2) each client trains the model on its privately stored subgraph; (3) after local training, the clients upload the model parameters or gradients to the central server; (4) the server aggregates the model parameters or gradients to update the global model, which is then broadcast back to clients.

Based on this, most subgraph-FL approaches [Fu *et al.*, 2022] rely on model parameters or gradients as the optimization carriers for federated training. These carriers are primarily derived from Computer Vision (CV)-based FL collaborative paradigm [Lim *et al.*, 2020], which struggles to capture the client-specific local convergence tendencies due to the complex topology in FGL, thus failing to adequately address the unique challenges of subgraph heterogeneity [Baek *et al.*, 2023]. Notably, current FGL methods [Li *et al.*, 2024b; Li *et al.*, 2024a] typically face a trade-off between optimization and privacy, as they seek to enhance model gradient-based federated convergence by sharing more messages. Further analysis and empirical studies will be provided in Sec. 3. Despite the significant contributions of existing methods, the inherent trade-off dilemma in CV-based federated optimization carriers limits the upper bound of FGL convergence, motivating us to propose a new collaborative paradigm.

Inspired by Graph Condensation (GC) [Zhao *et al.*, 2020; Jin *et al.*, 2021], a comprehensive and explicit approach based on condensed graphs holds promise in addressing the aforementioned limitations. Specifically, condensed graphs can effectively capture the complex relationships between nodes and topology, providing a more suitable means of information transmission. This implies that we can use the condensed graph as the optimization carrier for FGL, replacing traditional model parameters or gradients. Furthermore, a recent study [Dong *et al.*, 2022] suggests that data condensation via gradient matching can safeguard privacy. This makes condensed graph-based approaches highly promising as a robust framework for FGL data heterogeneity. The core idea of our method is to integrate generalized condensation subgraph consensus to acquire comprehensive and reliable knowledge.

In this paper, we propose a new FGL paradigm based on condensed subgraphs, as illustrated in Fig.1(b): (1) Each client performs local subgraph condensation using gradient matching and then uploads the condensed subgraph to the server; (2) The central server optimizes the condensed knowledge from a global perspective, resulting in a global-level condensed graph; (3) The server then trains a robust global model on the condensed graph and returns it to the clients.

Based on our proposed FGL paradigm, we introduce FedGM, a specialized dual-stage framework, as follows: **Stage 1:** Each client independently performs local subgraph condensation through gradient matching between the real subgraph and the condensed subgraph, without any communication, and then uploads the condensed knowledge to the server. Notably, multiple clients execute data condensation locally and in parallel. The central server then integrates these condensed subgraphs into a global-level condensed graph. It implies that Stage 1 is completed. And clients and the server only need to perform a single communication round to upload the locally condensed subgraphs, making this a one-shot FGL process. **Stage 2:** To achieve performance comparable to directly training on the implicit global real graph, FedGM employs federated gradient matching to optimize the condensed features. This approach leverages global class-wise knowledge to reinforce and consolidate the condensation consensus through multiple rounds of federated optimization. Subsequently, the central server trains a global robust GNN using the global condensed graph and distributes it to all clients.

Our contributions are as follows: (1) **New Framework**. To the best of our knowledge, we are the first to introduce condensed graphs as a novel optimization carrier to address the challenge of subgraph heterogeneity. (2) **New Paradigm**. We propose FedGM, a dual-stage paradigm that integrates generalized condensed subgraph consensus to obtain comprehensive knowledge while minimizing communication costs and reducing the risk of privacy breaches through a single transmission of condensed data between clients and the server. (3) **SOTA Performance**. Extensive experiments on six datasets demonstrate the consistent superiority of FedGM over state-of-the-art baselines, with improvements of up to 4.3%.

2 Notations and Problem Formalization

2.1 Notations

Graph Neural Networks. Consider a graph $G = \{\mathbf{A}, \mathbf{X}, \mathbf{Y}\}$ consisting of N nodes, where $\mathbf{X} \in \mathbb{R}^{N\times d}$ is the d-dimensional node feature matrix and $\mathbf{Y} \in \{1, ..., C\}^N$ denotes the node labels over C classes. $\mathbf{A} \in \mathbb{R}^{N\times N}$ is the adjacency matrix, with entry $\mathbf{A}_{i,j} > 0$ denoting an observed edge from node i to j, and $\mathbf{A}_{i,j} = 0$ otherwise. Building upon this, most GNNs can be subsumed into the deep message-passing framework [Wu *et al.*, 2020a].

Graph Condensation. Graph condensation is proposed to learn a synthetic graph with $N' \ll N$ nodes from the real graph G, denoted by $\mathcal{S}_k = \{\mathbf{A'}, \mathbf{X'}, \mathbf{Y'}\}$ with $\mathbf{A'} \in \mathbb{R}^{N'\times N'}$, $\mathbf{X'} \in \mathbb{R}^{N'\times d}$, $\mathbf{Y'} \in \{1, ..., C\}^{N'}$, such that a GNN $f(\cdot)$ solely trained on $\mathcal{S}$ can achieve comparable performance to the one trained on the original graph. In other words, graph condensation can be considered as a process of minimizing the loss defined on the models trained on the real graph G and the synthetic graph $\mathcal{S}$:

$$\mathcal{S} = \arg\min_{\mathcal{S}} \mathcal{L}(\text{GNN}_{\theta_\mathcal{S}}(G), \text{GNN}_{\theta_G}(G)), \tag{1}$$

where $\text{GNN}_{\theta_\mathcal{S}}$ and GNN_{θ_G} denote the GNN models trained on $\mathcal{S}$ and G, respectively; $\mathcal{L}$ represents the loss function used to measure the difference of these two models.

Subgraph Federated Learning. In subgraph-FL, the k-th client has a subgraph $G_k = \{\mathbf{A}_k, \mathbf{X}_k, \mathbf{Y}_k\}$ of an implicit global graph $G_{glo} = \{\mathbf{A}_{glo}, \mathbf{X}_{glo}, \mathbf{Y}_{glo}\}$ (i.e., $\mathbf{A}_k \subseteq \mathbf{A}_{glo}, \mathbf{X}_k \subseteq \mathbf{X}_{glo}, \mathbf{Y}_k \subseteq \mathbf{Y}_{glo}$). Each subgraph consists of N_k nodes, where $\mathbf{X}_k \in \mathbb{R}^{N_k\times d}$ is the d-dimensional node feature matrix and $\mathbf{Y_k} \in \{1, ..., C\}^{N_k}$ denotes the node labels over C classes. Typically, the training process for the t-th communication round in subgraph-FL with the FedAvg aggregation can be described as follows: (i) *Initialization:* This step occurs only at the first communication round ($t = 1$). The server sets the local GNN parameters of k clients to the

global GNN parameters $\bar{\theta}$, using $\theta_k \leftarrow \bar{\theta} \forall k$. (ii) *Local Updates:* Each local GNN performs training on the local data G_k to minimize the task loss $\mathcal{L}(G_k; \theta_k)$, and then updating the parameters: $\theta_k \leftarrow \theta_k - \eta \nabla \mathcal{L}$. (iii) *Global Aggregation:* After local training, the server aggregates local knowledge with respect to the number of training instances, i.e., $\bar{\theta} \sum_{k=1}^{K} \leftarrow \frac{N_k}{N} \theta_k$ with $N = \sum_k N_k$, and distributes the updated global parameters $\bar{\theta}$ to clients selected at the next round.

2.2 Problem Formalization

The proposed condensation-based Subgraph-FL framework is as follows: Firstly, each client performs local subgraph condensation, then uploads condensed knowledge to the server. Specifically, suppose that a client k is tasked with learning a local condensed subgraph with $N' < N$ nodes from the real subgraph G_k, denoted as $\mathcal{S}_k = \{\mathbf{A}'_k, \mathbf{X}'_k, \mathbf{Y}'_k\}$ with $\mathbf{A}'_k \in \mathbb{R}^{N' \times N'}$, $\mathbf{X}'_k \in \mathbb{R}^{N' \times d}$, $\mathbf{Y}'_k \in \{1, ..., C\}^{N'}$. The central server leverages the global perspective to optimize the condensed subgraphs, resulting in a global-level condensed graph $\mathcal{S}_{glo} = \{\mathbf{A}'_{glo}, \mathbf{X}'_{glo}, \mathbf{Y}'_{glo}\}$. Subsequently, the server trains a robust global model on the condensed graph and returns it to the clients. The motivation for this design is to obtain a powerful model trained on the condensed graph $\mathcal{S}_{glo}$, achieving performance comparable to one directly trained on the implicit global graph G_{glo}:

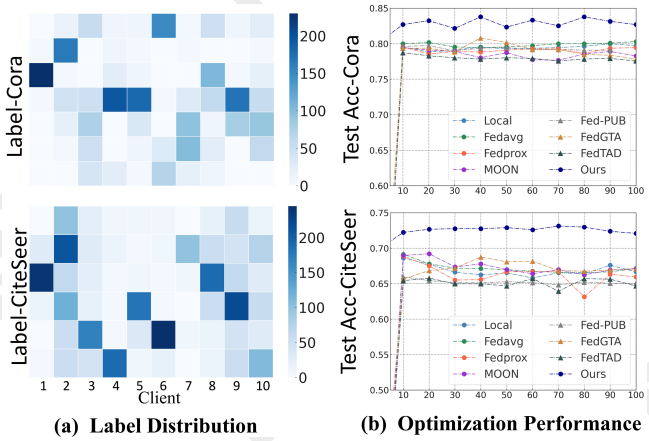
$$\begin{aligned} &\min_{\mathcal{S}} \mathcal{L}(\text{GNN}_{\theta_\mathcal{S}}(\mathbf{A}_{glo}, \mathbf{X}_{glo}), \mathbf{Y}_{glo}) \\ &\text{s.t}\, \theta_\mathcal{S} = \arg\min_{\theta} \mathcal{L}(\text{GNN}_\theta(\mathbf{A}'_{glo}, \mathbf{X}'_{glo}), \mathbf{Y}'_{glo}) \end{aligned} \tag{2}$$

where GNN_θ denotes the GNN model parameterized with θ, $\theta_\mathcal{S}$ denotes the parameters of the model trained on $\mathcal{S}_{glo}$, and $\mathcal{L}$ is the loss function used to measure the difference between model predictions and ground truth (i.e. cross-entropy loss).

3 Empirical Analysis

In this section, we empirically explore subgraph heterogeneity to investigate the FGL optimization dilemma. According to our observations, existing methods have the following limitations: (i) Using model parameters or gradients as primary information carriers overlooks the complex interplay between features and topology within local heterogeneous subgraphs, resulting in sub-optimal performance; (ii) many existing methods require the upload of additional information (e.g., subgraph embeddings, or mixed moments), raising privacy concerns. The in-depth analysis is provided as follows.

In CV-based FL, data heterogeneity refers to variations among clients in terms of features, labels, data quality, and data quantity [Qu *et al.*, 2022; Li *et al.*, 2022b]. This variability presents substantial challenges for effective federated training and optimization. In this work, we focus on the heterogeneity of features and labels, considering their strong correlation, widely highlighted by practical applications. Unlike data heterogeneity in conventional FL, subgraph heterogeneity is influenced by diverse topologies across clients [Li *et al.*, 2024a]. According to the homophily assumption [Wu *et al.*, 2020b], connected nodes tend to share similar feature



(a) Label Distribution (b) Optimization Performance

Figure 2: (a) Label distribution based on random data split, where the color gradient from white to blue indicates the increasing number of nodes held by different clients in each class. (b) Optimization performance of various methods under subgraph heterogeneity scenarios. The x-axis of the line plot represents federated training rounds, with "Local" indicating model performance in siloed settings.

distributions and labels. However, with the increasing deployment of GNNs in real-world applications, topology heterophily has emerged [Zhu *et al.*, 2021; Luan *et al.*, 2022], where the connected nodes exhibit contrasting attributes. Due to community-based subgraph data collection methods (i.e., the community can be viewed as the client), subgraphs from different communities often exhibit diverse topological structures, leading to Non-independent and identically distributed labels, as shown in Fig.2(a). Specifically, we observe strong homophily at Client 1 in Cora and CiteSeer, as the majority of nodes belong to the same label class. In contrast, at Cora-Client 3 and CiteSeer-Client 10, we observe the presence of heterophily, as label distributions approach uniformity. In summary, clients often exhibit diverse label distributions and topologies, a characteristic of distributed graphs that demands attention. Conventional federated training, which neglects subgraph heterogeneity, leads to underperformance.

To address these issues, FED-PUB [Baek *et al.*, 2023] measures subgraph similarity by transmitting subgraph embeddings for personalized aggregation. FedGTA [Li *et al.*, 2024b] shares mixed moments and local smoothing confidence for topology-aware aggregation. FedSage+[Zhang *et al.*, 2021] and FedGNN[Wu *et al.*, 2021] aim to reconstruct potentially missing edges among clients, thereby aligning local objectives at the data level. FedTAD [Zhu *et al.*, 2024] introduces topology-aware, data-free knowledge distillation. Despite the considerable efforts of subgraph heterogeneity, these methods rely on uploading additional information, leading to privacy concerns and higher communication overhead.

Considering the capability of condensed graphs to capture complex node-to-topology relationships while preserving privacy, we propose the condensation-based subgraph-FL framework. To validate the effectiveness of condensed knowledge as an optimization carrier, we conduct experiments using GCN in a federated learning setting with 10 clients across two common datasets, Cora [Kipf and Welling,

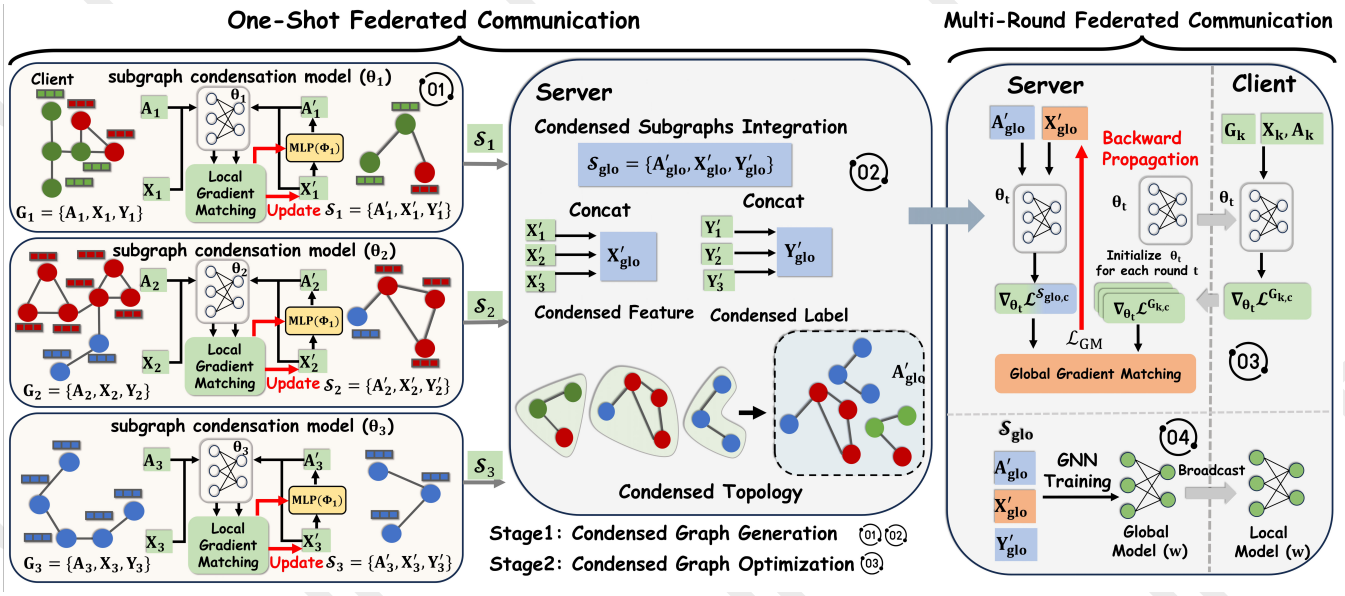


Figure 3: Overview of our proposed FedGM paradigm. We first perform local subgraph condensation and the central server integrates the condensed subgraphs. Subsequently, the server receives class-wise gradients from real subgraphs in federated communication to enhance the quality of condensed knowledge. Ultimately, the global model is trained on the condensed graph and then distributed to the clients.

2016a] and CiteSeer [Kipf and Welling, 2016a], as shown in Fig.2(b). The results demonstrate improved model performance and faster convergence of the condensation-based subgraph-FL method. Superior performance indicates a better capability to tackle subgraph heterogeneity, while faster convergence translates into reduced communication costs.

4 Method

The overview of our proposed FedGM is depicted in Fig.3. In *Stage 1*, each client involves a local process of standard graph condensation by one-step gradient matching and uploads condensed subgraphs to the central server. The server subsequently integrates these into a global-level graph. In *Stage 2*, we introduce federated optimization and perform multiple rounds of communication to leverage the class-wise knowledge, enhancing the quality of the condensed features.

4.1 Stage 1: Condensed Graph Generation

In the first stage, our task is to integrate the condensation consensus from clients to generate a global condensed graph. Since the real graph is distributed among multiple participants, direct access to the real graph to obtain the condensed graph, as in Eq.(2), is prohibited. Therefore, we perform subgraph condensation on each client to achieve local optimization and then integrate these subgraphs on the server via a single round of federated communication. Considering privacy protection requirements and condensation quality, we adopt the gradient alignment advocated by [Jin *et al.*, 2021; Jin *et al.*, 2022] as the local learning task. Unlike GraphGAN [Wang *et al.*, 2018] and GraphVAE [Simonovsky and Komodakis, 2018], which synthesize high-fidelity graphs by capturing data distribution, its goal is to generate informative graphs for training GNNs rather than "real-looking" graphs.

Client-Side Subgraph Condensation. The FedGM aims to provide a flexible graph learning paradigm by enabling each client to perform local subgraph condensation under local conditions without requiring real-time synchronization. The client generate the condensed subgraph through one-step gradient matching[Jin *et al.*, 2021; Jin *et al.*, 2022], where a GNN is updated using real subgraph and condensed subgraph, respectively, and their resultant gradients are encouraged to be consistent, as show on the left side of Fig.4. The local optimization objective can be formulated as:

$$\min_{\mathcal{S}_k} E_{\theta_k \sim P_{\theta_k}}[D(\nabla_{\theta_k}\mathcal{L}_1, \nabla_{\theta_k}\mathcal{L}_2)], \tag{3}$$

$$\mathcal{L}_1 = \mathcal{L}(GNN_{\theta_k}(\mathbf{A}'_k, \mathbf{X}'_k), \mathbf{Y}'_k), \tag{4}$$

$$\mathcal{L}_2 = \mathcal{L}(GNN_{\theta_k}(\mathbf{A}_k, \mathbf{X}_k), \mathbf{Y}_k), \tag{5}$$

where $D(\cdot,\cdot)$ represents a distance function, and the subgraph condensation model parameters θ_k for client k are initialized from the distribution of random initialization P_{θ_k}. In each condensation round, each client initializes its subgraph condensation model to calculate the gradients for the real subgraph and the condensed subgraph. By taking different parameter initializations drawn from the distribution P_{θ_k}, the learned $\mathcal{S}_k$ can avoid over fitting a specific initialization.

To facilitate efficient learning of $\mathcal{S}_k$, a common practice is to reduce the trainable pieces in the condensed subgraph $\mathcal{S}_k = \{\mathbf{A}'_k, \mathbf{X}'_k, \mathbf{Y}'_k\}$ to only the node features $\mathbf{X}'_k$. Concretely, the labels $\mathbf{Y}'_k$ can be predefined to match the distribution of different classes in the real subgraph, while each client condenses the graph structure by leveraging a function to parameterize the adjacency matrix $\mathbf{A}'_k$ to prevent overlooking the implicit correlations between condensed node features and condensed structure [Jin *et al.*, 2021]:

$$\mathbf{A}'_{ij} = \sigma([\mathrm{MLP}_\Phi([\mathrm{x}'_i; \mathrm{x}'_j]) + \mathrm{MLP}_\Phi([\mathrm{x}'_j; \mathrm{x}'_i])]/2), \tag{6}$$

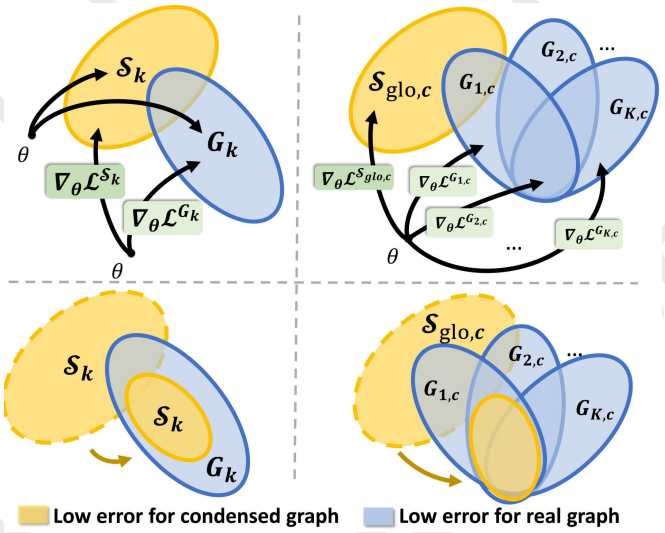


Figure 4: This is the representation of the local and global gradient matching in the model parameter space. The gradient matching iteratively optimizes the condensed data by minimizing the distance between gradients generated by the real and condensed data on the model, ultimately aligning the low-loss region of the condensed data within the low-loss region of the real data. The blue intersecting region in the right panel represents shared intra-class knowledge.

where MLP_Φ is a multi-layer perceptron (MLP) parameterized with Φ, $[\cdot;\cdot]$ indicates concatenation and σ is the sigmoid function. The client condenses the subgraph by optimizing alternately $\mathbf{X}'_k$ and Φ_k. After condensation, the client transfers $\mathcal{S}_k$ to the server via one-shot federated communication.

Server-Side Condensed Subgraphs Integration. To construct a global condensed graph, the server concatenates the features and labels from each client's condensed subgraph:

$$\mathbf{X}'_{\text{glo}} = \begin{bmatrix} \mathbf{X}'_1 \\ \mathbf{X}'_2 \\ \vdots \\ \mathbf{X}'_K \end{bmatrix}, \quad \mathbf{Y}'_{\text{glo}} = \begin{bmatrix} \mathbf{Y}'_1 \\ \mathbf{Y}'_2 \\ \vdots \\ \mathbf{Y}'_K \end{bmatrix}, \tag{7}$$

where $\mathbf{X}'_k$ and $\mathbf{Y}'_k$ represent the condened features and labels from client k, respectively. Unlike real-world graph structures, the condensed topology lacks tangible meaning, which is only relevant to the passage of condensed knowledge within the GNNs. To avoid disrupting the knowledge representing each client's subgraph within condensed data, we retain the topology of each condensed subgraph. Specifically, the global condensed adjacency matrix $\mathbf{A}'_{\text{glo}}$ is represented as:

$$\mathbf{A}'_{\text{glo}}[i,j] = \begin{cases} \mathbf{A}'_k[i,j], \text{ if } i,j \in \mathcal{V}_k; \\ 0, \text{otherwise}, \end{cases} \tag{8}$$

where $\mathcal{V}_k$ denotes the set of nodes from client k. Consequently, we obtain an initial global condensed graph, consisting of multiple connected components. However, there remains a significant gap between the quality of the condensed graph and our desired target due to the limitations of the narrow local scope. Therefore, it is crucial to effectively optimize the condensed graph by leveraging a global perspective.

4.2 Stage 2: Condensed Graph Optimization

There is a common phenomenon of class imbalance at the client level in scenarios with subgraph heterogeneity, which obviously leads to poorer feature quality for condensed nodes, especially for the minority classes. We observe that majority classes on one client often correspond to minority classes on other clients in scenarios with subgraph heterogeneity, as shown in Fig. 1(b). Therefore, we aim to collect the class-wise gradients generated by each subgraph and perform federated gradient matching to optimize the condensed features. Our intuition is that the shared intra-class knowledge between clients provides a basis based on the global perspective, which iteratively reduces the gap between the condensed graph and the real graph via class-wise gradient matching, as illustrated on the right side of Fig. 4.

In the second stage, FedGM introduces condensed graph optimization, which is performed over multi-round federated communication. In the t-th iteration, the server samples the gradient generation model parameters θ_t from random initialization distribution P_{θ_t} and sends it to each client.

Clinet-Side Gradient Generation. On each client, the real subgraphs generate class-wise gradients through the gradient generation model. For class c, and the generated gradient is:

$$\bigtriangledown_{\theta_t}\mathcal{L}^{G_{k,c}} = \bigtriangledown_{\theta_t}\mathcal{L}(GNN_{\theta_t}(\mathbf{A}_{k,c}, \mathbf{X}_{k,c}), \mathbf{Y}_c), \tag{9}$$

where $\mathbf{A}_{k,c}$ denotes the adjacency matrix composed of the c-class nodes and their neighbors and $\mathbf{X}_{k,c}$ denote the features corresponding to the c-class nodes. To ease the presentation, we adopt the following nations for every client k:

$$\bigtriangledown_{\theta_t}\mathcal{L}^{G_k} = [\bigtriangledown_{\theta_t}\mathcal{L}^{G_{k,1}}, \bigtriangledown_{\theta_t}\mathcal{L}^{G_{k,2}}, ..., \bigtriangledown_{\theta_t}\mathcal{L}^{G_{k,C}}]. \tag{10}$$

Upon having the class-wise gradients obtained, clients share the generated gradients $\bigtriangledown_{\theta_t}\mathcal{L}^{G_k}$ with the server.

Sever-Side Gradient Matching. For class c, the server calculates the gradient generated by the implicit global graph based on the number of c-class condensed nodes from clients:

$$\bigtriangledown_{\theta_t}\mathcal{L}^{G_c} = \sum_{k=1}^{K} \frac{N'_{k,c}}{N'_c} \bigtriangledown_{\theta_t} \mathcal{L}^{G_{k,c}}. \tag{11}$$

The gradients generated by the condensed graph is as follows:

$$\bigtriangledown_{\theta_t}\mathcal{L}^{\mathcal{S}_{glo,c}} = \bigtriangledown_{\theta_t}\mathcal{L}(GNN_{\theta_t}(\mathbf{A}'_{glo}, \mathbf{X}'_{glo}), \mathbf{Y}'_c), \tag{12}$$

Then we empower the server to match the gradients of the implicit global graph and the condensed graph for each category using the distance function. And we simplify our objective as

$$\min_{\mathbf{X}'_{\text{glo}}} E_{\theta_t \sim P_{\theta_t}}[D(\bigtriangledown_{\theta_t}\mathcal{L}^G, \bigtriangledown_{\theta_t}\mathcal{L}^{\mathcal{S}_{\text{glo}}})], \tag{13}$$

$\bigtriangledown_{\theta_t}\mathcal{L}^G$ and $\bigtriangledown_{\theta_t}\mathcal{L}^{\mathcal{S}_{\text{glo}}}$ in the above equation are defined as

$$\begin{aligned} \bigtriangledown_{\theta_t}\mathcal{L}^G &= [\bigtriangledown_{\theta_t}\mathcal{L}^{G_1}, \bigtriangledown_{\theta_t}\mathcal{L}^{G_2}, ..., \bigtriangledown_{\theta_t}\mathcal{L}^{G_C}], \\ \bigtriangledown_{\theta_t}\mathcal{L}^{\mathcal{S}_{\text{glo}}} &= [\bigtriangledown_{\theta_t}\mathcal{L}^{\mathcal{S}_{glo,1}}, \bigtriangledown_{\theta_t}\mathcal{L}^{\mathcal{S}_{glo,2}}, ..., \bigtriangledown_{\theta_t}\mathcal{L}^{\mathcal{S}_{glo,C}}]. \end{aligned} \tag{14}$$

In the multi-round communication process, the central server optimizes the features using the class-specific gradients, ultimately obtaining the desired condensed graph. The second-stage method of FedGM is presented in Algorithm 1.

Algorithm 1 FedGM-Condensed Graph Optimization

Input: Rounds, T; Local real subgraphs, $\{G_k\}_{k=1}^{K}$; Initial condensed graph, $\mathcal{S}_{glo}$
Output: Optimized condensed graph, $\mathcal{S}'_{glo}$

/* Client Execution */
for each communication round $t = 1, ..., T$ **do**
  Update the gradient generation model θ_t; **for** each class $c = 1, ..., C$ **do**
    Sample on the real subgraph according to the class c $(A_{k,c}, X_{k,c}, Y_{k,c}) \sim G_k$;
    Calculate loss and get the gradient via Eq.9
  Upload the number of samples of each category and the corresponding gradient to the central server
/* Server Execution */
for each communication round $t = 1, ..., T$ **do**
  initialize $\theta_t \sim P_{\theta_t}$;
  for each client $k = 1, ..., K$ **do**
    Send the gradient generation model θ_t to client k;
    Receive the number of class samples and the corresponding gradient
  for each class $c = 1, ..., C$ **do**
    Calculate the condensed graph gradient via Eq.12;
    Calculate the real graph gradient via Eq.11;
  Update the condensed features $\mathbf{X}'_{glo}$ via Eq.13;

5 Experiments

In this section, we conduct experiments to verify the effectiveness of FedGM. We introduce 6 benchmark graph datasets across 5 domains and the simulation strategy for the subgraph-FL scenario. And we present 8 evaluated state-of-the-art baselines. Specifically, we aim to answer the following questions: **Q1**: Compared with other state-of-the-art federated optimization strategies, can FedGM achieve better performance? **Q2**: Where does the performance gain of FedGM come from? **Q3**: Is FedGM sensitive to the hyperparameters? **Q4**: What is the time complexity of FedGM?

5.1 Datasets and Simulation Method

We evaluate FedGM on six public benchmark graph datasets across five domains, including two citation networks (Cora, Citeseer) [Kipf and Welling, 2016a], one co-authorship network (CS) [Shchur *et al.*, 2018], one co-purchase network (Amazon Photo), one task interaction network (Tolokers) [Platonov *et al.*, 2023], and one social network (Actor) [Tang *et al.*, 2009]. More details can be found in Table 1. To simulate the distributed subgraphs in subgraph-FL, we employ the Louvain algorithm [Blondel *et al.*, 2008] to achieve graph partitioning across 10 clients, which is based on modularity optimization and widely used in the subgraph-FL fields.

Dataset	#Nodes	#Features	#Edges	#Classes
Cora	2,708	1,433	5,429	7
CiteSeer	3,327	3,703	4,732	6
Photo	7,487	745	119,043	8
Actor	7,600	931	33,544	5
Tolokers	11,758	10	519,000	2
CS	18,333	6,805	81,894	15

Table 1: Statistics of the six public benchmark graph datasets.

5.2 Baselines and Experimental Settings

Baselines. We compare the proposed FedGM with four conventional FL optimization strategies (FedAvg [McMahan *et al.*, 2017], FedProx [Li *et al.*, 2020], SCAFFOLD [Karimireddy *et al.*, 2020], MOON [Li *et al.*, 2021]), two personalized subgraph-FL optimization strategies (Fed-PUB[Baek *et al.*, 2023], FedGTA[Li *et al.*, 2024b]), one subgraph-FL optimization strategy (FedTAD [Zhu *et al.*, 2024]), and one subgraph-FL framework (FedSage+ [Zhang *et al.*, 2021]).

Hyperparameters. For conventional framework, we employ a 2-layer GCN [Kipf and Welling, 2016b] with 256 hidden units as the backbone for both the clients and the central server. The local training epoch is set to 3. Notably, model-specific baselines such as FedSage+ [Zhang *et al.*, 2021] adhere to the custom architectures specified in their original papers. In the FedGM framework, the local subgraph condensation model, gradient generation model, and the model employed for evaluation are all implemented as 2-layer GCNs with 256 hidden units, and the condensed graph structure generation model is implemented as a 3-layer MLP with 128 hidden units. In the first stage, the number of local condensation epochs is 1000. Based on this, we perform the hyperparameter search for FedGM using the Optuna framework [Akiba *et al.*, 2019] on the ratio r of condensed nodes to real nodes within the ranges of 0 to 1. For all methods, the learning rate for the GNN is set to 1e-2, the weight decay is set to 5e-4, and the dropout rate is set to 0.0. The federated training is conducted over 100 rounds. For each experiment, we report the mean and variance results of 3 standardized training runs.

5.3 Results and Analysis

Result 1: the answer to Q1. The comparison results are presented in Table 2. According to observations, the proposed FedGM overall outperforms the baseline. Specifically, compared with FedAvg, FedGM brings at most 4.3% performance improvement; Compared with Fed-PUB, FedGM can achieve a performance improvement of at most 4.1%. Moreover, FedGM consistently outperforms existing SOTA methods in varying numbers of clients. Notably, as the number of clients increases, the performance advantage of FedGM becomes more pronounced. Specifically, with 20 clients, FedGM achieves a 13.4% performance improvement over FedAvg, as shown in Fig.5. The convergence curves of FedGM and baselines are shown in Fig.2(b). It is observed that FedGM has a good effect at the beginning of federated communication, which shows that FedGM is suitable for subgraph-FL scenarios with limited communication overhead. In addition, FedGM demonstrates robust performance stability across various client settings, with accuracy fluctuations remaining within a margin of 2% as shown in Fig.5.

Result 2: the answer to Q2. Stage 2 builds upon the foundation established in Stage 1. To answer **Q2**, we conducted an ablation study to investigate the effectiveness of both Stage

Methods	Cora	CiteSeer	Photo	Actor	Tolokers	CS	All Avg.
FedAvg [McMahan *et al.*, 2017]	79.66 ±0.15	73.34 ±0.12	90.24 ±0.23	30.84 ±0.15	77.99 ±0.03	87.61 ±0.08	73.28
FedProx [Li *et al.*, 2020]	80.08 ±0.22	73.11 ±0.71	90.07 ±0.33	30.72 ±0.19	78.01 ±0.02	88.42 ±0.24	73.40
SCAFFOLD [Karimireddy *et al.*, 2020]	79.31 ±0.48	73.48 ±0.00	84.99 ±0.92	28.90 ±0.08	78.21 ±0.25	86.60 ±0.52	71.92
MOON [Li *et al.*, 2021]	79.48 ±0.30	73.41 ±0.62	89.43 ±1.19	30.86 ±0.13	77.98 ±0.06	87.66 ±0.05	73.14
Fed-PUB [Baek *et al.*, 2023]	80.44 ±0.25	70.92 ±0.09	90.63 ±0.34	28.32 ±0.47	78.20 ±0.22	88.65 ±0.33	72.86
FedSage+ [Zhang *et al.*, 2021]	76.19 ±1.03	71.78 ±0.79	90.50 ±0.41	29.88 ±0.67	78.44 ±0.65	87.76 ±0.34	72.46
FedGTA [Li *et al.*, 2024b]	78.35 ±0.47	65.51 ±0.15	**91.17** ±0.16	28.53 ±0.15	78.16 ±0.13	88.34 ±0.08	71.65
FedTAD [Zhu *et al.*, 2024]	79.30 ±0.17	73.41 ±0.39	80.17 ±6.30	30.93 ±0.27	78.02 ±0.41	87.83 ±0.18	71.61
FedGM (Ours)	**83.23** **±0.13**	**73.95** **±0.65**	90.85 **±0.22**	**31.33** **±0.31**	**78.49** **±0.39**	**89.51** **±0.08**	**74.56**
FedGM (w/o Stage 2)	82.54 ±0.28	72.62 ±0.36	84.91 ±1.12	30.42 ±0.65	78.07 ±0.23	88.97 ±0.30	72.92

Table 2: **Performance comparison of FedGM and baselines**, where the best and second results are highlighted in **bold** and underline.

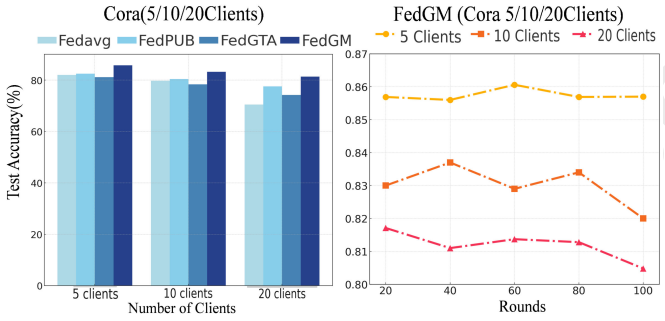
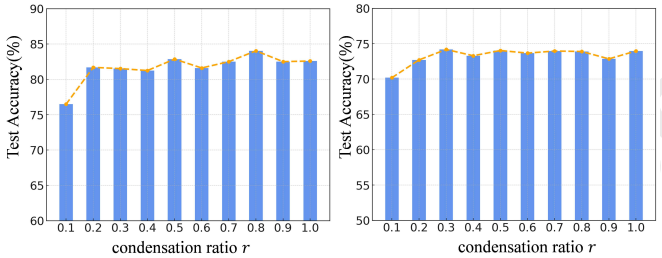


Figure 5: Performance of FedGM with different numbers of clients.



(a) Cora (b) CiteSeer

Figure 6: Sensitive analysis for condensation ratio r

1 and Stage 2, as shown in Table 2. After Stage 1, our method achieves an average accuracy of 72.92%, surpassing other state-of-the-art methods on two datasets, demonstrating the feasibility of the condensation-based FGL paradigm. In addition, our method consistently achieves superior performance compared to its single-stage variant across all datasets (e.g., increasing from 84.91% to 90.85% and from 72.92% to 74.56%), highlighting the pivotal role of the second stage in enhancing the model's representational capacity and robustness. This further validates that leveraging global intra-class knowledge contributes to improving the quality of the condensed graphs, reinforcing the efficacy of FedGM.

Result 3: the answer to Q3. To answer Q3, we assess the performance of FedGM under diverse condensation ratios. The sensitivity analysis on the Cora and CiteSeer datasets is presented in Fig.6. Overall, most values cluster near the maximum, reflecting consistently high accuracy under the majority of conditions. FedGM is insensitive to the condensation ratio, and there is no significant dependence.

Result 4: the answer to Q4. To answer **Q4**, we provide the complexity analysis of FedGM. In Stage 1, condensation graph generation costs $\mathcal{O}(rM)$, where r denotes the condensation ratio, and M denotes the size of the labeled dataset. A single transmission of condensed data implies a lower risk of privacy leakage. In Stage 2, condensation graph optimization costs $\mathcal{O}(KTN_{\Theta_{GNN}})$, where K denotes the number of participating clients, T denotes the number of the federal communication rounds, and Θ_{GNN} denotes the size of GNN gradients or parameters associated with gradient matching. Notably, FedAvg represents the lowest communication cost among federated learning processes, and its time complexity is also $\mathcal{O}(KTN_{\Theta_{GNN}})$. Unlike FedAvg, where the shared model parameters represent a trained GNN model, FedGM leverages the shared parameters primarily for generating gradients rather than direct model deployment. This distinction implies a reduced privacy risk during the federated process.

6 Conclusion

In this paper, we conduct an in-depth analysis of the trade-off dilemma caused by the poor performance of conventional federated carriers in handling subgraph heterogeneity. Considering the capability of condensed graphs to capture complex node-to-topology relationships while preserving privacy, we are the first to propose a new condensation-based FGL paradigm. Specifically, we propose FedGM, a dual-stage paradigm that integrates generalized condensation consensus to capture comprehensive knowledge. Experimental results demonstrate FedGM outperforms state-of-the-art baselines.

References

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

[Baek *et al.*, 2023] Jinheon Baek, Wonyong Jeong, Jiongdao Jin, Jaehong Yoon, and Sung Ju Hwang. Personalized subgraph federated learning. In *International conference on machine learning*, pages 1396–1415. PMLR, 2023.

[Bertozzi *et al.*, 2020] Andrea L Bertozzi, Elisa Franco, George Mohler, Martin B Short, and Daniel Sledge. The challenges of modeling and forecasting the spread of covid-19. *Proceedings of the National Academy of Sciences*, 117(29):16732–16738, 2020.

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

[Dong *et al.*, 2022] Tian Dong, Bo Zhao, and Lingjuan Lyu. Privacy for free: How does dataset condensation help privacy? In *International Conference on Machine Learning*, pages 5378–5396. PMLR, 2022.

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

[Huang *et al.*, 2020] Kexin Huang, Cao Xiao, Lucas M Glass, Marinka Zitnik, and Jimeng Sun. Skipgnn: predicting molecular interactions with skip-graph networks. *Scientific reports*, 10(1):21092, 2020.

[Jin *et al.*, 2021] Wei Jin, Lingxiao Zhao, Shichang Zhang, Yozen Liu, Jiliang Tang, and Neil Shah. Graph condensation for graph neural networks. *arXiv preprint arXiv:2110.07580*, 2021.

[Jin *et al.*, 2022] Wei Jin, Xianfeng Tang, Haoming Jiang, Zheng Li, Danqing Zhang, Jiliang Tang, and Bing Yin. Condensing graphs via one-step gradient matching. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 720–730, 2022.

[Karimireddy *et al.*, 2020] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pages 5132–5143. PMLR, 2020.

[Kipf and Welling, 2016a] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Kipf and Welling, 2016b] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Li *et al.*, 2020] 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.

[Li *et al.*, 2021] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10713–10722, 2021.

[Li *et al.*, 2022a] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. In *2022 IEEE 38th international conference on data engineering (ICDE)*, pages 965–978. IEEE, 2022.

[Li *et al.*, 2022b] Zonghang Li, Yihong He, Hongfang Yu, Jiawen Kang, Xiaoping Li, Zenglin Xu, and Dusit Niyato. Data heterogeneity-robust federated learning via group client selection in industrial iot. *IEEE Internet of Things Journal*, 9(18):17844–17857, 2022.

[Li *et al.*, 2024a] 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. *arXiv preprint arXiv:2401.11750*, 2024.

[Li *et al.*, 2024b] 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.

[Lim *et al.*, 2020] Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE communications surveys & tutorials*, 22(3):2031–2063, 2020.

[Luan *et al.*, 2022] Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. Revisiting heterophily for graph neural networks. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 1362–1375. Curran Associates, Inc., 2022.

[McMahan *et al.*, 2017] 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.

[Platonov *et al.*, 2023] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at evaluation of gnns under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*, 2023.

[Qu *et al.*, 2022] Liangqiong Qu, Yuyin Zhou, Paul Pu Liang, Yingda Xia, Feifei Wang, Ehsan Adeli, Li Fei-Fei, and Daniel Rubin. Rethinking architecture design for tackling data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10061–10071, 2022.

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

[Simonovsky and Komodakis, 2018] Martin Simonovsky and Nikos Komodakis. Graphvae: Towards generation of small graphs using variational autoencoders. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*, pages 412–422. Springer, 2018.

[Tang *et al.*, 2009] Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, page 807–816, New York, NY, USA, 2009. Association for Computing Machinery.

[Voigt and Von dem Bussche, 2017] Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017.

[Wang *et al.*, 2018] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. Graphgan: Graph representation learning with generative adversarial nets. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[Wu *et al.*, 2020a] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[Wu *et al.*, 2020b] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[Wu *et al.*, 2021] Chuhan Wu, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv preprint arXiv:2102.04925*, 2021.

[Zhang *et al.*, 2021] 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.

[Zhao *et al.*, 2020] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020.

[Zhu *et al.*, 2021] Jiong Zhu, Ryan A. Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K. Ahmed, and Danai Koutra. Graph neural networks with heterophily. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11168–11176, May 2021.

[Zhu *et al.*, 2024] Yinlin Zhu, Xunkai Li, Zhengyu Wu, Di Wu, Miao Hu, and Rong-Hua Li. Fedtad: Topology-aware data-free knowledge distillation for subgraph federated learning. *arXiv preprint arXiv:2404.14061*, 2024.