FedGrAINS: Personalized SubGraph Federated Learning with AdaptIve Neighbor Sampling

Emir Ceyani*   Han Xie†   Baturalp Buyukates‡   Carl Yang†   Salman Avestimehr*

Abstract

Graphs are crucial for modeling relational and biological data. As datasets grow larger in real-world scenarios, the risk of exposing sensitive information increases, making privacy-preserving training methods like federated learning (FL) essential to ensure data security and compliance with privacy regulations. Recently proposed personalized subgraph FL methods have become the de-facto standard for training personalized Graph Neural Networks (GNNs) in a federated manner while dealing with the missing links across clients' subgraphs due to privacy restrictions. However, personalized subgraph FL faces significant challenges due to the heterogeneity in client subgraphs, such as degree distributions among the nodes, which complicate federated training of graph models. To address these challenges, we propose *FedGrAINS*, a novel data-adaptive and sampling-based regularization method for subgraph FL. FedGrAINS leverages generative flow networks (GFlowNets) to evaluate node importance concerning clients' tasks, dynamically adjusting the message-passing step in clients' GNNs. This adaptation reflects task-optimized sampling aligned with a trajectory balance objective. Experimental results demonstrate that the inclusion of *FedGrAINS* as a regularizer consistently improves the FL performance compared to baselines that do not leverage such regularization.

Keywords: federated learning, graph neural networks, reinforcement learning, node classification, generative models, generative flow networks

1 Introduction

Graph Neural Networks (GNNs) operate over a single graph, with nodes and edges stored in a central server. For example, in social networks, each user, along with their connections form a large graph of all users and their links [2]. However, parties may not send their private graph datasets to a central server due to privacy concerns, raising the need to train GNN models over multiple distributed graph datasets. For instance, hospitals may maintain their patient interaction networks to track physical contacts or co-diagnoses of diseases. However, they cannot share these graphs with other entities due to privacy restrictions [52]. Here, the question is how to collaboratively train GNNs across clients' distributed subgraphs without sharing the actual graph data. Federated Learning (FL) of GNNs aims to solve this problem so that each client individually trains a local GNN on their local data, while a central server aggregates locally updated GNN weights from multiple clients into a global model using federated optimization [19].

A key challenge in naïvely applying FL methods to GNNs is the potential loss of crucial information due to missing links between subgraphs distributed across parties. Recent approaches to subgraph FL address this issue by extending local subgraphs with information from other subgraphs [47, 52]. Specifically, they extend the local subgraph by either precisely adding the relevant nodes from other subgraphs from different clients [47] or by predicting the nodes using node information from the other subgraphs via graph mending [52]. However, sharing node information can compromise data privacy and lead to high communication costs. While [4] mitigates the issues of missing links and subgraph heterogeneity through personalized aggregation and local sparse masks, all existing approaches are mainly based on image data modality, failing to fully exploit the graph-structured data to address heterogeneity, thus resulting in sub-optimal solutions to personalize GNN models.

On the other hand, dropout-based data pruning in FL [21, 44] and stochastic regularization in GNNs [17, 30] have shown to be successful in regularizing deeper models without any significant overhead by randomly dropping the network weights and/or data components. Existing dropout-based data pruning approaches in FL, however, are not suitable for graph ML models due to GNNs' invariance requirements, which forces us to utilize dropout methods for graph-structured data [11, 40]. Further, applying dropout techniques naïvely to deeper GNNs without accounting for the data

---
*University of Southern California ({ceyani,avestime}@usc.edu)

†Emory University ({han.xie,j.carlyang}@emory.edu)

‡University of Birmingham (b.buyukates@bham.ac.uk)

structure can degrade node and structural information quality due to Laplacian smoothing, leading to over-smoothened representations that are uninformative in higher layers [31]. Moreover, using the same dropout parameters across different clients is also not an effective strategy due to heterogeneity over data, system, and the model, raising the need for coordination based on the users' requirements.

To effectively personalize GNN models by respecting graph structures, we propose a novel data-adaptive, sampling-based regularization method for subgraph FL, named *FedGrAINS* by estimating and selecting essential nodes at each layer of GNNs every round for every client in a federated manner. To estimate the adaptive neighborhood distribution of each node in the datasets, we leverage Generative Flow Networks (GFlowNets), the state-of-the-art generative models tailored for structured discrete objects like graphs [7]. Here, for each node in the clients' graph, we leverage GFlowNets to estimate the importance of the neighbors of a node so that we can prune the graph for each node adaptively in parallel to optimize the clients' personalized task performances. We refer to this framework as Personalized Sub*Gr*aph Federated Learning with *AdaptIve Neighbor Sampling* (*FedGrAINS*).

We extensively validate the performance of FedGrAINS across six popular graph datasets, evaluating overlapping and disjoint subgraph FL scenarios with varying numbers of clients. Our comprehensive experimental results show that FedGrAINS significantly outperforms relevant baselines. Our contributions are summarized as follows:

- We introduce a new type of heterogeneity, *node-degree heterogeneity*, showing that it cannot be handled with previous personalized FL approaches.

- We propose *FedGrAINS*, a novel framework for personalized subgraph FL. *FedGrAINS* performs layer-wise node-importance estimation using GFlowNets and dynamically updates the GNN message-passing steps to combat neighborhood heterogeneity caused by missing links and subgraph heterogeneity between multiple clients.

- We validate our personalized FL framework *FedGrAINS* across six real-world datasets under disjoint and overlapping node scenarios, demonstrating its effectiveness against relevant baselines.

2 Preliminaries

In this section, we describe GNNs and personalized subgraph FL in more detail. Then, we present the core component of our proposed algorithm, GFlowNets.

GNNs: Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with a set of N nodes $\mathcal{V} = \{v_1, \dots v_N\}$ and edges $\mathcal{E}$. $\boldsymbol{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\} \in \mathbb{R}^{N \times d}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is the node feature vector for node v_i, indicating the features of each node $i \in \{1, \dots, N\}$. According to the Message-Passing Neural Network (MPNN) paradigm [12], the flow of hidden representations in GNNs, based on the nodes' neighborhoods and features, is described as follows:

$$(2.1) \quad \boldsymbol{H}_v^{l+1} = \mathrm{UPD}^l \left(\boldsymbol{H}_v^l, \mathrm{AGG}^l \left(\left\{ \boldsymbol{H}_u^l : \forall u \in \mathcal{N}(v) \right\} \right) \right),$$

where $\boldsymbol{H}_v^l$ denotes the features of node v at the l-th layer, and $\mathcal{N}(v)$ denotes the set of neighbor nodes of node v: $\mathcal{N}(v) = \{u \in \mathcal{V} \mid (u, v) \in \mathcal{E}\}$. AGG aggregates the features of the neighbors of node v. We note that AGG can be mean or even a black-box function. UPD updates node v's representation given its previous representation and the aggregated representations from its neighbors. $\boldsymbol{H}^1$ is initialized as $\boldsymbol{X}$.

As the number of layers increases, the embedding computation for a node v incorporates information from neighbors several hops away. Increasing the number of layers expands the depth of neighbors for each node included in the training set, resulting in the exponential increase in the neighborhood size for each node, thus causing high computation time on the user side. Reducing high computation at the edge is possible by training GNNs in a subgraph federated learning setting where each user has a part of an entire graph and utilizes FL to access higher-order node information. The following subsection describes the federated training process for GNNs.

Personalized Subgraph FL: We assume that there exists a global graph $\mathcal{G}_{global} = (\mathcal{V}_{global}, \mathcal{E}_{global})$, where $\mathcal{V}_{global}$ and $\mathcal{E}_{global}$ are the set of nodes and edges of the global graph, respectively. $\boldsymbol{X}_{global}$ is the associated node feature set of the global graph. In the subgraph FL system, we have one central server S and M clients with distributed subgraph datasets: $\mathcal{D}_i = \{\mathcal{G}_i, \boldsymbol{Y}_i\} = \{\{\mathcal{V}_i, \mathcal{E}_i, \boldsymbol{X}_i\}, \boldsymbol{Y}_i\}$ where $\boldsymbol{X}_i$ and $\boldsymbol{Y}_i$ are node feature and label sets of the i-th client, respectively, for $i \in [M]$, where $\mathcal{V}_{global} = \bigcup_{i=1}^{M} \mathcal{V}_i$. For an edge $e_{v,u} \in \mathcal{E}_{global}$, where $v \in \mathcal{V}_i, u \in \mathcal{V}_j$, we have $e_{v,u} \notin \mathcal{E}_i \cup \mathcal{E}_j$. That is, $e_{v,u}$ might exist in reality but is missing from the whole system, as this link is between the local data of two distinct clients i and j. The system exploits an FL framework to collaboratively learn M personalized node classifiers f_i, parameterized with ϕ_i, on isolated subgraphs in clients without raw graph data sharing. Finding personalized models is equivalent to minimizing

the aggregated empirical risk defined as:

$$(2.2) \qquad \min_{\phi_1, \dots \phi_M} \frac{1}{M} \sum_{i=1}^{M} \mathcal{R}_i \left(f_i(\phi_i) \right),$$

where the empirical risk for the i-th client is defined as $\mathcal{R}_i \left(f_i(\phi_i) \right) = \mathbb{E}_{(\mathcal{G}_i, \boldsymbol{Y}_i) \sim \mathcal{D}_i} \left[\mathcal{L}_i \left(f_i \left(\phi_i; \mathcal{G}_i \right), \boldsymbol{Y}_i \right) \right]$, and $\mathcal{L}_i \left(f_i \left(\phi_i; \mathcal{G}_i \right), \boldsymbol{Y}_i \right)$ is the i-th client's average loss over the training data.

GFlowNets: GFlowNets are the state-of-the-art generative models for generating structured objects, such as molecules and graphs, from a finite set of states $\mathcal{S}$§ in proportion to a reward function $R(s_T) : \mathcal{S}_T \to \mathbb{R}_+$, that is assigned to terminal states s_T [7]. At a particular time t, based on the current state s_t, we perform an action a_t to propagate into the next state s_{t+1} with the help of the forward transition function P_F. The consecutive state-action pairs starting from an initial state $s_0 \in \mathcal{S}_0 \subset \mathcal{S}$ and ending at a terminal state $s_T \in \mathcal{S}_T \subset \mathcal{S}$ form a trajectory $\tau = (s_0, a_0, \dots, s_T)$. The key idea in GFlowNets is to learn a flow network function $F(s)$ that ensures that the total flow into each state equals the flow out. Hence, the probability of reaching any terminal state is proportional to its reward $R(s_n)$, achieved by aligning the forward and backward policies P_F and P_B, parameterized with neural networks, ensuring that the final distribution over terminal states reflects the desired reward distribution. Therefore, the GFlowNet learning problem is equivalent to learning forward and backward policies $\pi = \{P_F(s_t|s_{t-1}), P_B(s_{t-1}|s_t)\}$. Trajectory balance (TB) [34] is the most common learning objective designed to improve credit assignment and learning policies in GFlowNets. Given τ, the TB is defined as:

$$(2.3) \qquad \mathcal{L}_{TB}(\tau) = \left(\log \frac{Z(s_0) \prod_{t=1}^{n} P_F(s_t|s_{t-1})}{R(s_n) \prod_{t=1}^{n} P_B(s_{t-1}|s_t)} \right)^2,$$

where $Z(s_o) : \mathcal{S}_0 \to \mathbb{R}_+$ computes the total flow of the network from starting state s_0, which is also known as the normalizing constant.

In the next section, we outline our GFlowNet modeling approach to characterize the neighborhood importance distribution for each node and explain how we learn and balance global and local neighborhood distributions in federated settings.

3 Personalized Subgraph Federated Learning with Adaptive Neighbor Sampling

In this section, we introduce *FedGrAINS*, our personalized subgraph FL method. We detail the design of GFlowNet to estimate neighborhood importance distributions for each node (Section 3.1), the scalable sam-

pling procedure (Section 3.1.1), and the personalized subgraph FL framework *FedGrAINS* (Section 3.2).

3.1 Adaptive Neighborhood Distribution Estimation with GFlowNets: Given graph data, the adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ allows us to determine the neighbors of every node $v \in \mathcal{V}$. At a given layer ℓ, the AGG operator given in (2.1) aggregates the features of v's neighbors based on $\boldsymbol{A}$ and requires higher-order propagation to compute neighbors' messages [15]. As a result, the neighborhood size grows exponentially with the number of layers, resulting in over-smoothing. It is possible to regularize models based on randomly dropping nodes, edges, and even messages [11,40]. However, these methods operate independently of the task, data, and classifier, leading to suboptimal performance [31]. Guided by the intuition that the aggregation step in message-passing GNNs is crucial for good performance, we aim to adaptively choose and sample the neighbors at every layer to minimize the classification loss using GFlowNets. Since every MPNN layer requires an adjacency matrix to compute hidden representations, we sample a sequence of adjacency matrices over all layers. Thus, for GFlowNet to sample important neighbors, we define the state $s \in \mathcal{S}$ as a sequence of adjacency matrices $s = (\boldsymbol{A}_0, \dots, \boldsymbol{A}_\ell)$ sampled up to the current step. The action a_ℓ to construct the adjacency matrix for the next layer $\boldsymbol{A}_{\ell+1}$ is to choose k nodes without replacement among the neighbors of nodes in $\boldsymbol{A}_\ell$. Then, in our design, obtaining the adjacency matrix of the next layer is equivalent to transitioning to a new state.

Using the modeling described above, in an L-layer GNN, we construct a sequence of L adjacency matrices to reach a terminating state. We aim to perform task-aware sampling with the optimal GFlowNet sampling policy to minimize the expected classification loss :

$$
\begin{aligned}
R(s_L) &= R(\boldsymbol{A}_0, \dots, \boldsymbol{A}_L) \\
&= \exp(-\alpha \cdot \mathcal{L}\left(F\left(\phi; \mathcal{G}\right), \boldsymbol{Y}\right)) \\
(3.4) \quad &= \exp(-\alpha \cdot \mathcal{L}\left(F\left(\phi; \mathcal{V}, (\boldsymbol{A}_0, \dots, \boldsymbol{A}_L)\right), \boldsymbol{Y}\right)),
\end{aligned}
$$

where α is a *scaling parameter*. $\boldsymbol{A}_0$ is the adjacency matrix of the graph at the starting state, and $\boldsymbol{A}_1, \dots, \boldsymbol{A}_L$ are the adjacency matrices sampled at every layer.

Forward Probability: The GFlowNet of *FedGrAINS* is a neural network that learns the forward probabilities $P_F(s_{\ell+1}|s_\ell)$, the probability of sampling an adjacency matrix $\boldsymbol{A}_{\ell+1}$ for the layer $\ell + 1$ given the previous layers' adjacency matrices $\boldsymbol{A}_0, \dots, \boldsymbol{A}_\ell$. For each node v_i, where $i \in 1, \dots, N$, the GFlowNet models the probability that node v_i in the neighborhood of the sampled nodes at layer ℓ, $\mathcal{N}(\mathcal{S}^\ell)$, will be included in $\mathcal{V}^{\ell+1}$, the set of nodes sampled for layer $\ell + 1$.
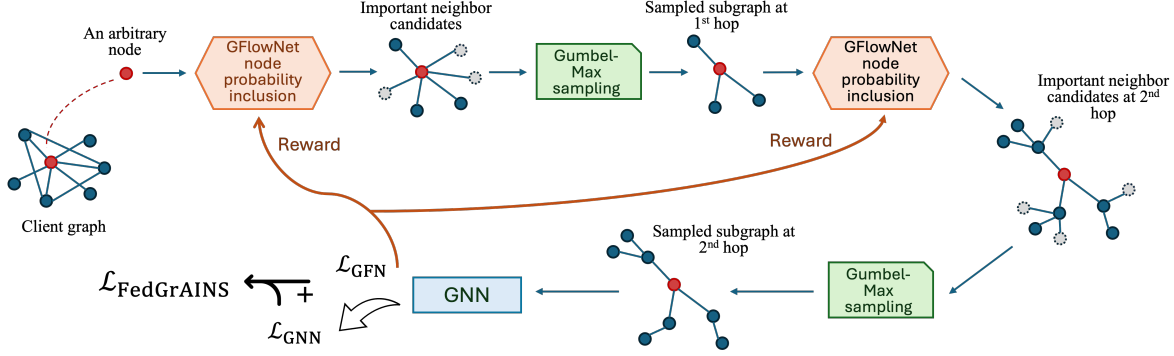
Figure 1: The illustration of the proposed joint training scheme of 2-layer GNN and GFlowNet for our personalized subgraph FL framework, *FedGrAINS* from the client side. For any node v in the client's graph, we first estimate the important neighbors with GFlowNet, then sample k nodes using the Gumbel-Max trick [23]. After choosing the important nodes for two layers, we input the sampled subgraph to the GNN. After obtaining our novel loss, we backpropagate through the classification and the trajectory balance losses, with the reward function defined as the cross-entropy loss at a specific round. With classification loss as a reward function, GFlowNet adapts itself to select and sample neighbors vital for optimal task performance.

We represent the inclusion probability of each node in the sample using a Bernoulli distribution [11, 40]. In this formulation, $P_F(s_{l+1}|s_l)$ becomes tractable, with each node represented as a Bernoulli-distributed random variable. Each node is associated with a logit indicating its likelihood of being included in the sampled set. Thus, if p_i denotes the probability that the node v_i in in the neighborhood of the sampled nodes at layer ℓ will be included in $\mathcal{V}^{\ell+1}$, then the forward probability transition function between layers ℓ and $\ell+1$ is $P_F(s_{\ell+1}|s_\ell) = \prod_{v_i \in \mathcal{V}^{\ell+1}} p_i \prod_{v_j \in \mathcal{N}(\mathcal{S}^\ell) \backslash \mathcal{V}^{\ell+1}} (1-p_j)$, where the first product computes the probabilities of the nodes that are in the newly sampled set $\mathcal{V}^{\ell+1}$, and the second one computes the probabilities of nodes that are not within the newly sampled set $\mathcal{V}^{\ell+1}$.

Backward Probability: TB requires us to define the backward transition function $P_B(s_\ell|s_{\ell+1})$ [34]. In our setup, however, it is not required as the state representation, $s = (\boldsymbol{A}_0, \ldots, \boldsymbol{A}_\ell)$, exactly saves the trajectory that was taken through the flow network to get to state s. Thus, the backward mapping is unique and deterministic. So, $P_B(s_l|s_{l+1}) = 1$.

Trajectory Balance Condition: Given τ and model weights ϕ_{GFN}, we define TB condition [34] as:

$$\mathcal{L}^{GFN}(\tau; \phi_{GFN}) = \left(\log \frac{Z \prod_{l=1}^L P_F(s_l|s_{l-1}; \phi_{GFN})}{R(s_L)} \right)^2$$

(3.5)

$$= \left(\log Z + \sum_{l=1}^L \log P_F(s_l|s_{l-1}; \phi_{GFN}) + \alpha \cdot \mathcal{L}^{GNN} \right)^2.$$

Notice that we model the normalizer $Z(s_0)$ in (2.3) as a constant in our loss formulation to skip the additional optimization during parameter estimation. After estimation, we describe the node sampling procedure of our proposed personalized subgraph FL framework.

3.1.1 Sampling and Off-Policy Training: We want to sample k important ones out of N graph nodes without replacement at every message-passing step. However, our forward transition function P_F contains N independent Bernoulli distributions, and we are unlikely to sample exactly k nodes from this distribution. Instead, we use the Gumbel-max trick [23, 29, 50], which selects a set of nodes $\mathcal{V}_k^\ell$ by perturbing the log probabilities randomly and taking the top-k among those:

$$(3.6) \qquad \mathcal{V}_k^l = \mathsf{top}(k, \log p_1 + G_1, \ldots, \log p_n + G_n),$$

where $G_i \sim \text{Gumbel}(0,1)$. The sampling process shown above differs from the sampling from the forward transition function P_F sequentially one node at a time. Thus, we train the GFlowNet in an off-policy setting using the trajectory balance loss without any change [35]. In addition, the off-policy training is less expensive in computation and more robust to high-variance problems in distribution estimation than direct gradient-estimation methods such as REINFORCE [1]. Next, we explain how the adaptive importance sampling framework assists the (personalized) subgraph FL in Section 3.2.

3.2 Federated Training of Node Classifier and Importance Sampler: In *FedGrAINS*, each client desires an optimal yet personalized performance in the

node classification task while not sharing any data with others. For this, we formulate the joint training of the node classifier and the importance sampler as depicted in Figure 1. First, for each layer of local GNN, we estimate the important nodes to build layer-wise adjacency matrices for each layer using the GFlowNet and Gumbel-max trick as defined in Section 3.1.1. Then, using the estimated layer-wise adjacency matrices, we perform node classification using the local model. More details on the forward pass and the training loop of *Fed-GrAINS* client can be found in Appendix[1]. For each client, we formulate the following personalized subgraph federated optimization:

(3.7)
$$\min_{\{\phi_{GNN}, \phi_{GFN}^1, \ldots, \phi_{GFN}^M\}} \mathcal{L}_{\text{FedGrAINS}}^i \left(\phi_{GNN}, \phi_{GFN}^i \right).$$

where $\mathcal{L}_{\text{FedGrAINS}}^i \left(\phi_{GNN}, \phi_{GFN}^i \right) = \mathcal{L}_i^{GFN} + \mathcal{L}^{GNN}$ is the loss function of the i-th client, $\mathcal{L}_i^{GFN}$ is defined as in Equation 3.5 and $\mathcal{L}^{GNN} = \mathcal{R}_i \left(f_i(\phi_{GNN}) \right)$ is the classification loss.

4 Experiments

4.1 Experimental Setup

Datasets As proposed by [4, 52], we generate distributed subgraphs by dividing the graph datasets into a certain number of clients so that each FL client has a subgraph that is a part of the original graph. Specifically, we use six datasets: Cora, CiteSeer, Pubmed, and ogbn-arxiv for citation graphs [22, 42]; Computer and Photo for product graphs [36, 43]. We partition the graphs into subgraphs using the METIS graph partitioning algorithm [25], which allows specifying the number of subsets without requiring further merging, unlike Louvain partitioning [8]. METIS first coarsens the original graph using maximal matching methods [26], then computes a minimum edge-cut partition on the coarsened graph followed by the projection of the partitioned coarsened graph back onto the original graph.

In our experiments, we consider two subgraph settings: *Disjoint subgraphs* setting, in which each client has a unique set of nodes and no overlapping nodes between subgraphs. In this setup, we use the METIS outputs as they provide non-overlapping partitions. We consider 5, 10, or 20 clients for the disjoint setting. In the *overlapping subgraphs* setting, clients may have common nodes between their datasets. Since METIS produces non-overlapping partitions, we introduce overlap by randomly sampling subgraphs multiple times from the partitioned graph. First, we divide the original

---

graph into 2, 6, or 10 disjoint subgraphs using METIS. Then, for each METIS partition, we randomly sample half of the nodes and their associated edges five times to create subgraphs. This process results in 10, 30, or 50 clients for the overlapping case, generating shared nodes across subgraphs from the same METIS partition. It is important to note that the number of clients in our proposed settings is not uniform due to the need for overlapping nodes. For each subgraph, except for the ogbn-arxiv dataset, train, validation, and test node random sampling ratios are 20%, 40%, and 40%, respectively. Thus, for the ogbn-arxiv dataset, train, validation, and test node random sampling ratios are 5%, 47.5%, and 47.5%, respectively, as the ogbn-arxiv dataset has a significantly higher number of nodes, as in Table 4.

Baselines 1) Local: A baseline that locally trains models without weight sharing. **2) FedAvg** [37]: The most popular and naive FL baseline. **3) FedPer** [3]: A personalized FL baseline without sharing personalized layers adopted to subgraph data. **4) FedSage+** [52]: A subgraph FL baseline that proposes the subgraph FL problem with missing links and introduces the first solution by training local models with a global fake neighborhood generator. **5) GCFL** [49]: A graph FL baseline which formulates graph level FL as clustered FL [41], adopted for subgraph FL. **6) FED-PUB** [4]: State-of-the-art personalized subgraph FL framework, which includes similarity matching and weight masking. **7) *FedGrAINS***: Our personalized subgraph FL framework with adaptive neighborhood sampling using GFlowNets. Further details for the baselines can be found in Appendix[1].

Implementation Details For our experiments, we employ two layers of Graph Convolutional Networks (GCNs) [28] with ReLU activation as the base GNN for the flow networks P_F and P_B, as well as for the global and local models. The important hyperparameters are a hidden layer dimensionality of 128 and a learning rate of 0.01. We set the sparsity and proximal terms for the FED-PUB framework to 0.001 [4].

FedGrAINS requires three additional hyperparameters: the sampling budget k, the learning rate for the GFlowNet, β_{GFN}, and the reward scaling parameter α. We set the reward scaling to 10^5, while β_{GFN} is set to 0.001 for all experiments (more detailed analysis for GFlowNet hyperparameters can be found in Section 4.2). We tune these parameters based on classification performance on the validation set. We utilize Adam optimizer [27] for all models as local optimizer and perform FL (with full participation) over 100 communication rounds with one local training epoch for the Cora, Cite-

Seer, and Pubmed datasets and 200 rounds with three local training epochs for the Computer, Photo, and ogbn-arxiv datasets, considering the dataset size. We measure the node classification accuracy on subgraphs on the client side and then average over all clients.

4.2 Experimental Results

Main Results Tables 1 and 2 show node classification results on the disjoint and overlapping subgraph scenarios, respectively. The primary motivation behind the proposed *FedGrAINS* is its applicability to existing subgraph FL algorithms. For a fairer comparison with Fed-PUB [4], we combine our data-adaptive regularization framework with their personalized aggregation and sparse masking. Other baselines do not include personalized aggregation. In addition, we did not test how the inclusion of our proposed framework affects the performance of FedSage+ as our proposed framework tries to filter unimportant nodes. In contrast, FedSage+ tries to increase the number of neighbors with fake neighbors, which provides a more complicated setup for us to observe the effect of learnable node filtering only. Based on experimental outcomes, we make the following observations: First, regardless of the FL algorithm, classification performance decreases when the number of clients increases due to the increased number of partitions creating more missing links between the clients (See Appendix B.1.1[1] for the missing links counts for each dataset). The increasing number of missing links also leads to severe node-degree heterogeneity(see Appendix B.1.2[1] within clients. This results in more challenging local training and collaboratively learning generalizable models with other clients.

Second, the disjoint setting is more challenging than the overlapping one for the same number of clients because the subgraphs in the non-overlapping setting are entirely disjoint and more heterogeneous than the overlapped case. Moreover, the non-overlapping setting has fewer nodes to learn from due to the overlap generation scheme explained in the experimental design. As shown in Table 1, *FedGrAINS* consistently outperforms all existing baselines in this challenging scenario, demonstrating its efficacy.

Finally, the inclusion of *FedGrAINS* as a personalization method consistently enhances the performance of all baselines. Specifically, the GFlowNet-based importance sampler improves the accuracy of all algorithms (except FedSage+) by at least 1%. Additionally, the advantages of *FedGrAINS* are most evident when comparing the results of FedAvg with our personalization method to those of FedPer [3], GCFL+ [49], and FedSage+ [52]. Moreover, the performance of both

FedSage+ and FedAvg with *FedGrAINS* are very close for Cora, CiteSeer, and PubMed datasets. However, our personalized FL framework demonstrates significantly better performance for the Amazon-Computer, Amazon-Photo, and ogbn-arxiv datasets. We attribute this success to the size of the graph dataset, which resulted in a higher average degree than citation networks.

In comparison with FED-PUB [4], FED-PUB remains superior due to its use of personalized aggregation. To further assess the impact of *FedGrAINS*, we evaluate its performance when integrated with base algorithms that also employ personalized aggregation, such as FED-PUB, i.e., FED-PUB + FedGrAINS. Our results show that incorporating the importance sampler consistently and significantly enhances FED-PUB performance. Therefore, we can conclude that *FedGrAINS* can be seamlessly integrated into most subgraph FL algorithms without incurring additional communication overhead or compromising privacy.

Hyperparameter Analysis on the GFlowNet To understand the operation regime of our personalization method, we vary important hyperparameters for the GFlowNet-based sampler: the learning rate of the GFlowNet, β_{GFN}, and the reward scaling factor α. We use the log uniform distribution to sample the aforementioned hyperparameters with the values from the following ranges respectively, $[1e^{-6}, 1e^{-2}]$ and $[1e^2, 1e^6]$. The setting is on the Cora dataset with the overlapping node scenario, where we set the number of local epochs to 1 and the number of clients to 10. The results are shown in Table 3. According to Table 3, the relationship between the performance and the reward scaling factor α is highly non-linear. In addition, the learning rate β_{GFN} is not as effective as the scaling factor α.

5 Related Works

GNNs and Random dropping GNNs: GNNs are the de-facto tools to learn the representations of nodes, edges, and entire graphs [5, 15, 24, 48, 53]. Most existing GNNs fall under the message-passing neural network framework [12], which iteratively represents a node representation by aggregating features from its neighboring nodes and itself [14, 16, 28]. However, the current paradigm for GNNs is restrictive regarding the development of deeper architectures due to the issues of over-smoothing and overfitting [28, 31]. To remedy this, dropout and its variants are utilized by randomly removing the node features, edges, or messages during training [11, 40, 44]. However, these regularizers are independent of the graph topology, limiting their efficacy under data and structural heterogeneity. Moreover, state-of-the-art GNNs on node classification and link

Table 1: **Results for the disjoint node scenario.** The reported results are mean and standard deviation over three different runs. The best performances in terms of accuracy are emphasized in bold.

Methods	Cora			CiteSeer			PubMed		
	5 Clients	**10 Clients**	**20 Clients**	**5 Clients**	**10 Clients**	**20 Clients**	**5 Clients**	**10 Clients**	**20 Clients**
Local	80.10 ± 0.76	77.43 ± 0.49	72.75 ± 0.89	70.10 ± 0.25	68.77 ± 0.35	64.51 ± 0.28	85.30 ± 0.24	84.88 ± 0.32	82.66 ± 0.65
FedAvg	79.63 ± 4.37	72.06 ± 2.18	69.50 ± 3.58	70.24 ± 0.47	68.32 ± 2.59	65.12 ± 2.15	84.87 ± 0.41	78.92 ± 0.39	78.21 ± 0.25
FedAvg + *FedGrAINS*	82.92 ± 1.11	80.16 ± 1.62	81.17 ± 1.22	73.09 ± 1.55	74.63 ± 1.35	70.19 ± 2.55	85.83 ± 0.35	85.72 ± 0.46	84.39 ± 0.46
FedPer	81.33 ± 1.24	78.76 ± 0.25	78.24 ± 0.36	70.36 ± 0.34	70.31 ± 0.36	66.95 ± 0.46	85.88 ± 0.25	85.62 ± 0.23	84.90 ± 0.37
FedPer + *FedGrAINS*	82.15 ± 0.67	80.16 ± 0.34	79.33 ± 0.43	72.15 ± 0.41	71.23 ± 0.49	69.15 ± 0.63	87.91 ± 0.32	87.13 ± 0.36	86.05 ± 0.43
GCFL+	80.36 ± 0.57	78.37 ± 0.89	77.19 ± 1.30	70.52 ± 0.64	69.71 ± 0.79	66.80 ± 0.95	85.77 ± 0.38	84.94 ± 0.35	84.10 ± 0.43
GCFL+ + *FedGrAINS*	82.44 ± 1.15	79.46 ± 1.61	81.42 ± 1.40	72.79 ± 1.14	74.64 ± 1.48	70.56 ± 2.18	85.68 ± 0.47	85.37 ± 0.67	84.29 ± 0.51
FedSage+	80.09 ± 1.28	74.07 ± 1.46	72.68 ± 0.95	70.94 ± 0.21	69.03 ± 0.59	65.20 ± 0.73	86.03 ± 0.28	82.89 ± 0.37	79.71 ± 0.35
FED-PUB	83.72 ± 0.18	81.45 ± 0.12	81.10 ± 0.64	72.40 ± 0.26	71.83 ± 0.61	66.89 ± 0.14	86.81 ± 0.12	86.09 ± 0.17	84.66 ± 0.54
FED-PUB + *FedGrAINS*	**84.23 ± 0.28**	**82.68 ± 0.22**	**81.27 ± 0.61**	**73.57 ± 0.31**	**72.48 ± 0.42**	**68.61 ± 0.19**	**88.11 ± 0.39**	**87.91 ± 0.57**	**87.23 ± 0.59**

Methods	Amazon-Computer			Amazon-Photo			ogbn-arxiv		
	5 Clients	**10 Clients**	**20 Clients**	**5 Clients**	**10 Clients**	**20 Clients**	**5 Clients**	**10 Clients**	**20 Clients**
Local	89.18 ± 0.15	88.25 ± 0.21	84.34 ± 0.28	91.85 ± 0.12	89.56 ± 0.09	85.83 ± 0.17	66.87 ± 0.09	66.03 ± 0.14	65.43 ± 0.21
FedAvg	88.03 ± 1.68	81.82 ± 1.71	78.19 ± 0.86	89.26 ± 1.80	85.31 ± 1.67	82.59 ± 1.18	66.24 ± 0.45	64.09 ± 0.83	62.47 ± 1.19
FedAvg + *FedGrAINS*	87.59 ± 0.66	85.21 ± 0.42	83.18 ± 0.11	**92.95 ± 0.29**	**91.48 ± 0.60**	90.60 ± 0.79	68.13 ± 0.58	66.65 ± 0.45	66.21 ± 0.72
FedPer	88.94 ± 0.25	88.26 ± 0.17	87.85 ± 0.29	91.30 ± 0.33	89.97 ± 0.27	88.30 ± 0.18	67.02 ± 0.19	66.02 ± 0.27	65.25 ± 0.31
FedPer + *FedGrAINS*	89.45 ± 0.34	88.75 ± 0.27	88.33 ± 0.44	92.07 ± 0.38	90.75 ± 0.37	89.07 ± 0.22	67.68 ± 0.27	66.45 ± 0.39	65.93 ± 0.42
GCFL+	89.07 ± 0.45	88.74 ± 0.49	87.81 ± 0.36	90.78 ± 0.69	90.22 ± 0.85	89.23 ± 1.07	66.97 ± 0.11	66.38 ± 0.14	65.30 ± 0.34
GCFL+ + *FedGrAINS*	89.63 ± 0.72	89.13 ± 0.81	88.62 ± 0.54	91.14 ± 0.47	90.73 ± 0.62	89.93 ± 0.67	68.14 ± 0.44	67.82 ± 0.45	67.15 ± 0.56
FedSage+	89.78 ± 0.71	84.39 ± 1.06	79.75 ± 0.90	90.89 ± 0.44	86.82 ± 0.78	83.10 ± 0.70	66.91 ± 0.12	65.30 ± 0.13	62.63 ± 0.24
FED-PUB	90.25 ± 0.07	89.73 ± 0.16	88.20 ± 0.18	93.20 ± 0.15	92.46 ± 0.19	90.59 ± 0.35	67.62 ± 0.11	66.35 ± 0.16	63.90 ± 0.27
FED-PUB + *FedGrAINS*	**91.24 ± 0.23**	**90.41 ± 0.25**	**88.95 ± 0.15**	**94.17 ± 0.33**	**93.23 ± 0.60**	**91.42 ± 0.18**	**68.86 ± 0.43**	**68.44 ± 0.56**	**66.43 ± 0.77**

prediction tasks are tailored for a single graph, limiting their applicability in real-world systems with locally distributed graphs that are not shared across participants, which gives rise to the use of FL for GNNs [9, 19].

Federated Learning (FL): Federated Learning allows multiple users to learn a collaborative model under the guidance of a centralized server without sharing raw data [32, 37]. One of the primary bottlenecks in this learning paradigm is data heterogeneity across different entities. To address data heterogeneity, proximal term regularization has been proposed to prevent clients' local models from diverging excessively from their local training data [33]. For clients with extreme levels of heterogeneity, learning a single global model may not be suitable. Personalizing client models relaxes the constraint of developing one global model, allowing for the optimization of average empirical risk across clients through various methods, such as distillation [18], meta-learning [10], and knowledge separation [3]. However, graph-structured data presents additional challenges due to the connections between instances, leading to missing edges and community structures within private subgraphs, unlike more common modalities like images and text [19, 52].

FL over Graphs: Recent studies primarily categorize the use of FL frameworks for training GNNs into subgraph-level and graph-level federated learning methods [19, 46]. Graph-level methods operate under the condition that different clients have their own set of graph datasets that are not part of a bigger graph (e.g., molecular graphs), and recent works [20, 45, 49] focus on the label heterogeneity among clients' non-IID graphs. On the other hand, the subgraph-level FL poses a unique graph-structural challenge: the existence of missing yet probable links between subgraphs, as each subgraph is a part of the larger global graph. To deal with the aforementioned problem, existing methods [47, 51, 52] augment the nodes by requesting the node information in the other subgraphs and then connecting the existing nodes with the augmented ones. However, requesting node information from other clients increases communication overhead and the risk of privacy leakage. Recently, [4] introduced a novel problem of heterogeneity among subgraph communities [13, 39] across clients' subgraphs and proposed a personalized subgraph FL framework that employs personalized aggregation and sparse masking. Unlike other methods, this work focuses on adjusting node information without requiring additional data. We employ a local generative model to estimate the importance of nodes to minimize classification loss. Furthermore, our approach does not incur any extra communication overhead and serves as an off-the-shelf personalization method on top of any existing personalized (sub-)graph FL algorithms.

6 Conclusion & Future Work

This work addresses the challenges of node-degree heterogeneity and local overfitting in subgraph FL

Table 2: **Results for the overlapping node scenario.** The reported results are mean and standard deviation over three different runs. The best performances in terms of accuracy are emphasized in bold.

	Cora			CiteSeer			Pubmed		
Methods	10 Clients	30 Clients	50 Clients	10 Clients	30 Clients	50 Clients	10 Clients	30 Clients	50 Clients
Local	78.14 ± 0.15	73.60 ± 0.18	69.87 ± 0.40	68.94 ± 0.29	66.13 ± 0.49	63.70 ± 0.92	84.90 ± 0.05	83.27 ± 0.24	80.88 ± 0.19
FedAvg	78.55 ± 0.49	69.56 ± 0.79	65.19 ± 3.88	68.73 ± 0.46	65.02 ± 0.59	63.85 ± 1.31	84.66 ± 0.11	80.62 ± 0.46	80.18 ± 0.50
FedAvg + *FedGrAINS*	79.52 ± 1.25	76.99 ± 1.69	79.45 ± 2.28	70.50 ± 0.93	**69.80** ± 1.78	70.33 ± 2.47	84.59 ± 0.35	84.32 ± 0.51	83.85 ± 0.69
FedPer	78.84 ± 0.32	73.46 ± 0.43	72.54 ± 0.52	70.42 ± 0.26	65.09 ± 0.48	64.04 ± 0.46	85.76 ± 0.14	83.45 ± 0.15	81.90 ± 0.23
FedPer + *FedGrAINS*	79.73 ± 0.47	76.32 ± 0.63	75.83 ± 0.55	71.12 ± 0.37	67.23 ± 0.68	65.85 ± 0.51	86.43 ± 0.27	85.15 ± 0.38	83.20 ± 0.47
GCFL+	78.60 ± 0.25	73.41 ± 0.36	73.13 ± 0.87	69.80 ± 0.34	65.17 ± 0.32	64.71 ± 0.67	85.08 ± 0.21	83.77 ± 0.17	80.95 ± 0.22
GCFL+ + *FedGrAINS*	79.63 ± 1.01	77.01 ± 1.81	**79.81** ± 2.82	70.51 ± 1.80	69.29 ± 1.77	**70.45** ± 2.66	84.51 ± 0.34	84.25 ± 0.62	83.81 ± 0.62
FedSage+	79.01 ± 0.30	72.20 ± 0.76	66.52 ± 1.37	70.09 ± 0.26	66.71 ± 0.18	64.89 ± 0.25	86.07 ± 0.06	83.26 ± 0.08	80.48 ± 0.20
FED-PUB	79.65 ± 0.17	75.42 ± 0.48	73.13 ± 0.29	70.43 ± 0.27	67.41 ± 0.36	65.13 ± 0.40	85.60 ± 0.10	85.19 ± 0.15	84.26 ± 0.19
FED-PUB + *FedGrAINS*	**81.48** ± 0.23	**77.85** ± 0.41	75.71 ± 0.33	**71.95** ± 0.37	68.45 ± 0.40	67.25 ± 0.59	**86.74** ± 0.45	**86.08** ± 0.57	**85.72** ± 0.43

	Amazon-Computer			Amazon-Photo			ogbn-arxiv		
Methods	10 Clients	30 Clients	50 Clients	10 Clients	30 Clients	50 Clients	10 Clients	30 Clients	50 Clients
Local	88.50 ± 0.20	86.66 ± 0.00	87.04 ± 0.02	92.17 ± 0.12	90.16 ± 0.12	90.42 ± 0.15	62.52 ± 0.07	61.32 ± 0.04	60.04 ± 0.04
FedAvg	88.99 ± 0.19	83.37 ± 0.47	76.34 ± 0.12	92.91 ± 0.07	89.30 ± 0.22	74.19 ± 0.57	63.56 ± 0.02	59.72 ± 0.06	60.94 ± 0.24
FedAvg + *FedGrAINS*	89.25 ± 0.32	86.26 ± 0.62	85.43 ± 0.62	93.05 ± 0.37	91.05 ± 0.52	87.81 ± 0.88	65.01 ± 0.23	61.36 ± 0.57	61.04 ± 0.35
FedPer	89.30 ± 0.04	87.99 ± 0.23	88.22 ± 0.27	92.88 ± 0.24	91.23 ± 0.16	90.92 ± 0.38	63.97 ± 0.08	62.29 ± 0.04	61.24 ± 0.11
FedPer + *FedGrAINS*	89.87 ± 0.41	89.23 ± 0.34	88.41 ± 0.33	93.27 ± 0.31	91.71 ± 0.23	91.02 ± 0.41	64.04 ± 0.11	63.03 ± 0.21	61.93 ± 0.27
GCFL+	89.01 ± 0.22	87.24 ± 0.09	87.02 ± 0.22	92.45 ± 0.10	90.58 ± 0.11	90.54 ± 0.08	63.24 ± 0.02	61.66 ± 0.10	60.32 ± 0.01
GCFL+ + *FedGrAINS*	89.82 ± 0.66	89.13 ± 0.45	88.41 ± 0.45	92.75 ± 0.34	91.07 ± 0.28	90.69 ± 0.46	64.22 ± 0.13	63.35 ± 0.27	62.29 ± 0.37
FedSage+	89.24 ± 0.15	81.33 ± 1.20	76.72 ± 0.39	92.76 ± 0.05	88.69 ± 0.99	72.41 ± 1.36	63.24 ± 0.02	59.90 ± 0.12	60.95 ± 0.09
FED-PUB	89.98 ± 0.08	89.15 ± 0.06	88.76 ± 0.14	93.22 ± 0.07	92.01 ± 0.07	**91.71** ± 0.11	64.18 ± 0.04	63.34 ± 0.12	62.55 ± 0.12
FED-PUB + *FedGrAINS*	**90.41** ± 0.23	**90.05** ± 0.28	**89.73** ± 0.13	**93.36** ± 0.21	**92.85** ± 0.60	91.45 ± 0.29	**67.38** ± 0.25	**65.13** ± 0.33	**64.17** ± 0.17

β_{GFN}	α	Validation Accuracy [%]
1e-3	1e4	65.33 ± 3.11
5e-2	1e4	68.27 ± 1.67
1e-3	1e5	**82.92** ± 1.11
1e-2	1e5	79.42 ± 0.37
1e-3	1e6	55.27 ± 1.59
1e-2	1e6	61.26 ± 0.97

Table 3: Sensitivity analysis on GFlowNet hyperparameters β_{GFN} and α. We report the hyperparameters' influence on the model performance on validation set for the Cora dataset with 10 clients in the disjoint setting.

by proposing *FedGrAINS*, an adaptive neighborhood sampling-based regularization method. For each client, *FedGrAINS* employs a GFlowNet model to assess the node-importance to maximize the node classification performance. On the server side, these node importance distributions are aggregated in a personalized manner at every round, enabling adaptive generalization. By leveraging a generative and data-driven sampling approach, *FedGrAINS* ensures consistent performance, effectively handling missing links and subgraph heterogeneity in an intuitive yet elegant manner. Our experimental results demonstrate that incorporating *FedGrAINS* as a regularization method significantly enhances the perfor-

mance and generalization of subgraph FL techniques.

We plan to extend our personalization framework to other federated graph learning settings [9, 20, 51]. As long as we have access to a tractable reward [6], we can leverage the proposed *FedGrAINS*. We assumed that $Z(s_0)$ is constant to reduce the workload at the edge. We defer the investigation of how the estimation of $Z(s_0)$ might enhance personalization to future work [38]. Finally, we would like to obtain theoretical insights into our framework using the tools presented in [17], evaluating how effectively GFlowNet alleviates the oversmoothing and over-fitting tendencies of client GNNs while allowing uncertainty quantification at the edge at no additional cost.

Acknowledgements

References

[1] K. AHMED, Z. ZENG, M. NIEPERT, AND G. V. DEN BROECK, *SIMPLE: A gradient estimator for k-subset sampling*, in The Eleventh International Conference on Learning Representations, 2023.

[2] S. ALAM, T. ZHANG, T. FENG, H. SHEN, Z. CAO, D. ZHAO, J. KO, K. SOMASUNDARAM, S. S.

Narayanan, S. Avestimehr, et al., *Fedaiot: A federated learning benchmark for artificial intelligence of things*, arXiv preprint arXiv:2310.00109, (2023).

[3] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, *Federated learning with personalization layers*, 2019.

[4] J. Baek, W. Jeong, J. Jin, J. Yoon, and S. J. Hwang, *Personalized subgraph federated learning*, in Proceedings of the 40th International Conference on Machine Learning, 2023, pp. 1396–1415.

[5] J. Baek, M. Kang, and S. J. Hwang, *Accurate learning of graph representations with graph multiset pooling*, in 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021, 2021.

[6] E. Bengio, M. Jain, M. Korablyov, D. Precup, and Y. Bengio, *Flow network based generative models for non-iterative diverse candidate generation*, Advances in Neural Information Processing Systems, 34 (2021), pp. 27381–27394.

[7] Y. Bengio, S. Lahlou, T. Deleu, E. J. Hu, M. Tiwari, and E. Bengio, *Gflownet foundations*, arXiv preprint arXiv:2111.09266, (2021).

[8] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, *Fast unfolding of communities in large networks*, Journal of statistical mechanics: theory and experiment, 2008 (2008), p. P10008.

[9] F. Chen, P. Li, T. Miyazaki, and C. Wu, *Fedgraph: Federated graph learning with intelligent sampling*, IEEE Transactions on Parallel & Distributed Systems, 33 (2022), pp. 1775–1786.

[10] A. Fallah, A. Mokhtari, and A. Ozdaglar, *Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach*, Advances in neural information processing systems, 33 (2020), pp. 3557–3568.

[11] T. Fang, Z. Xiao, C. Wang, J. Xu, X. Yang, and Y. Yang, *Dropmessage: Unifying random dropping for graph neural networks*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, 2023.

[12] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, *Neural message passing for quantum chemistry*, in Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, vol. 70 of Proceedings of Machine Learning Research, PMLR, 2017, pp. 1263–1272.

[13] M. Girvan and M. E. J. Newman, *Community structure in social and biological networks*, Proceedings of the National Academy of Sciences, 99 (2002), pp. 7821–7826.

[14] W. Hamilton, Z. Ying, and J. Leskovec, *Inductive representation learning on large graphs*, in Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., vol. 30, Curran Associates, Inc., 2017.

[15] W. L. Hamilton, *Graph representation learning*, Synthesis Lectures on Artificial Intelligence and Machine Learning, 14 (2020), pp. 1–159.

[16] D. K. Hammond, P. Vandergheynst, and R. Gribonval, *Wavelets on graphs via spectral graph theory*, Applied and Computational Harmonic Analysis, 30 (2011), pp. 129–150.

[17] A. Hasanzadeh, E. Hajiramezanali, S. Boluki, M. Zhou, N. Duffield, K. Narayanan, and X. Qian, *Bayesian graph neural networks with adaptive connection sampling*, in International conference on machine learning, PMLR, 2020, pp. 4094–4104.

[18] C. He, M. Annavaram, and S. Avestimehr, *Group knowledge transfer: Federated learning of large cnns at the edge*, Advances in Neural Information Processing Systems, 33 (2020), pp. 14068–14080.

[19] C. He, K. Balasubramanian, E. Ceyani, C. Yang, H. Xie, L. Sun, L. He, L. Yang, P. S. Yu, Y. Rong, et al., *Fedgraphnn: A federated learning system and benchmark for graph neural networks*, arXiv preprint arXiv:2104.07145, (2021).

[20] C. He, E. Ceyani, K. Balasubramanian, M. Annavaram, and S. Avestimehr, *Spreadgnn: Serverless multi-task federated learning for graph neural networks*, AAAI, (2022).

[21] S. Horvath, S. Laskaridis, M. Almeida, I. Leontiadis, S. Venieris, and N. Lane, *Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout*, Advances in Neural Information Processing Systems, 34 (2021), pp. 12876–12889.

[22] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, *Open graph benchmark: Datasets for machine learning on graphs*, Advances in neural information processing systems, 33 (2020), pp. 22118–22133.

[23] I. A. Huijben, W. Kool, M. B. Paulus, and R. J. Van Sloun, *A review of the gumbel-max trick and its extensions for discrete stochasticity in machine learning*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 45 (2022), pp. 1353–1371.

[24] J. Jo, J. Baek, S. Lee, D. Kim, M. Kang, and S. J. Hwang, *Edge representation learning with hypergraphs*, in Advances in Neural Information Processing Systems, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds., vol. 34, Curran Associates, Inc., 2021, pp. 7534–7546.

[25] G. Karypis, *Metis: Unstructured graph partitioning and sparse matrix ordering system*, Technical report, (1997).

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

[27] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.

[28] T. N. Kipf and M. Welling, *Semi-supervised clas-*

sification with graph convolutional networks, in 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings, 2017.

[29] W. KOOL, H. VAN HOOF, AND M. WELLING, Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement, in International Conference on Machine Learning, PMLR, 2019, pp. 3499–3508.

[30] J. LI, S. TIAN, R. WU, L. ZHU, W. ZHAO, C. MENG, L. CHEN, Z. ZHENG, AND H. YIN, Less can be more: Unsupervised graph pruning for large-scale dynamic graphs, arXiv preprint arXiv:2305.10673, (2023).

[31] Q. LI, Z. HAN, AND X.-M. WU, Deeper insights into graph convolutional networks for semi-supervised learning, in Proceedings of the AAAI conference on artificial intelligence, vol. 32, 2018.

[32] Q. LI, Z. WEN, Z. WU, S. HU, N. WANG, Y. LI, X. LIU, AND B. HE, A survey on federated learning systems: vision, hype and reality for data privacy and protection, IEEE Transactions on Knowledge and Data Engineering, (2021).

[33] T. LI, A. K. SAHU, M. ZAHEER, M. SANJABI, A. TALWALKAR, AND V. SMITH, Federated optimization in heterogeneous networks, in Proceedings of Machine Learning and Systems 2020, MLSys 2020, Austin, TX, USA, March 2-4, 2020, mlsys.org, 2020.

[34] N. MALKIN, M. JAIN, E. BENGIO, C. SUN, AND Y. BENGIO, Trajectory balance: Improved credit assignment in gflownets, Advances in Neural Information Processing Systems, 35 (2022), pp. 5955–5967.

[35] N. MALKIN, S. LAHLOU, T. DELEU, X. JI, E. HU, K. EVERETT, D. ZHANG, AND Y. BENGIO, Gflownets and variational inference, arXiv preprint arXiv:2210.00580, (2022).

[36] J. MCAULEY, C. TARGETT, Q. SHI, AND A. VAN DEN HENGEL, Image-based recommendations on styles and substitutes, in Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval, 2015, pp. 43–52.

[37] H. B. MCMAHAN, E. MOORE, D. RAMAGE, S. HAMPSON, AND B. A. Y ARCAS, Communication-efficient learning of deep networks from decentralized data, in AISTATS, 2017.

[38] L. PAN, D. ZHANG, A. COURVILLE, L. HUANG, AND Y. BENGIO, Generative augmented flow networks, in The Eleventh International Conference on Learning Representations, 2023.

[39] F. RADICCHI, C. CASTELLANO, F. CECCONI, V. LORETO, AND D. PARISI, Defining and identifying communities in networks, Proceedings of the national academy of sciences, 101 (2004), pp. 2658–2663.

[40] Y. RONG, W. HUANG, T. XU, AND J. HUANG, Dropedge: Towards deep graph convolutional networks on node classification, in International Conference on Learning Representations, 2020.

[41] F. SATTLER, K.-R. MÜLLER, AND W. SAMEK, Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints, IEEE transactions on neural networks and learning systems, 32 (2020), pp. 3710–3722.

[42] P. SEN, G. NAMATA, M. BILGIC, L. GETOOR, B. GALLIGHER, AND T. ELIASSI-RAD, Collective classification in network data, AI magazine, 29 (2008), pp. 93–93.

[43] O. SHCHUR, M. MUMME, A. BOJCHEVSKI, AND S. GÜNNEMANN, Pitfalls of graph neural network evaluation, arXiv preprint arXiv:1811.05868, (2018).

[44] N. SRIVASTAVA, G. HINTON, A. KRIZHEVSKY, I. SUTSKEVER, AND R. SALAKHUTDINOV, Dropout: A simple way to prevent neural networks from overfitting, Journal of Machine Learning Research, 15 (2014), pp. 1929–1958.

[45] Y. TAN, Y. LIU, G. LONG, J. JIANG, Q. LU, AND C. ZHANG, Federated learning on non-iid graphs via structural knowledge sharing, arXiv preprint arXiv:2211.13009, (2022).

[46] Z. WANG, W. KUANG, Y. XIE, L. YAO, Y. LI, B. DING, AND J. ZHOU, Federatedscope-gnn: Towards a unified, comprehensive and efficient package for federated graph learning, in KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022, A. Zhang and H. Rangwala, eds., ACM, 2022, pp. 4110–4120.

[47] C. WU, F. WU, Y. CAO, Y. HUANG, AND X. XIE, Fedgnn: Federated graph neural network for privacy-preserving recommendation, KDD, (2021).

[48] Z. WU, S. PAN, F. CHEN, G. LONG, C. ZHANG, AND P. S. YU, A comprehensive survey on graph neural networks, IEEE Trans. Neural Networks Learn. Syst., 32 (2021), pp. 4–24.

[49] H. XIE, J. MA, L. XIONG, AND C. YANG, Federated graph classification over non-iid graphs, in Advances in Neural Information Processing Systems, vol. 34, Curran Associates, Inc., 2021, pp. 18839–18852.

[50] S. M. XIE AND S. ERMON, Reparameterizable subset sampling via continuous relaxations, arXiv preprint arXiv:1901.10517, (2019).

[51] Y. YAO AND C. JOE-WONG, Fedgcn: Convergence and communication tradeoffs in federated training of graph convolutional networks, arXiv preprint arXiv:2201.12433, (2022).

[52] K. ZHANG, C. YANG, X. LI, L. SUN, AND S. M. YIU, Subgraph federated learning with missing neighbor generation, in Advances in Neural Information Processing Systems, vol. 34, Curran Associates, Inc., 2021, pp. 6671–6682.

[53] J. ZHOU, G. CUI, S. HU, Z. ZHANG, C. YANG, Z. LIU, L. WANG, C. LI, AND M. SUN, Graph neural networks: A review of methods and applications, AI Open, 1 (2020), pp. 57–81.