# Federated Graph Classification over Non-IID Graphs

**Han Xie** [1] **Jing Ma** [1] **Li Xiong** [1] **Carl Yang** [1 †]

## Abstract

Federated learning has emerged as an important paradigm for training machine learning models in different domains. For graph-level tasks such as graph classification, graphs can also be regarded as a special type of data samples, which can be collected and stored in separate local systems. Similar to other domains, multiple local systems, each holding a small set of graphs, may benefit from collaboratively training a powerful graph mining model, such as the popular graph neural networks (GNNs). However, we also find that different sets of graphs, even from the same domain or same dataset, are non-IID regarding both graph structures and node features. To handle this, we propose a graph clustered federated learning (GCFL) framework that dynamically finds clusters of local systems based on the gradients of GNNs, and theoretically justify that such clusters can reduce the structure and feature heterogeneity among graphs owned by the local systems. Extensive experimental results and in-depth analysis demonstrate the effectiveness of our proposed framework.

## 1. Introduction

Federated learning (FL) as a distributed learning paradigm that trains centralized models on decentralized data has attracted much attention recently (McMahan et al., 2017b; Zhao et al., 2018; Li et al., 2020; Karimireddy et al., 2019; Kairouz et al., 2019). FL allows local systems to benefit from each other while keeping their own data private. Especially, for local systems with scarce training data or lack of diverse distributions, FL provides them with the potentiality to leverage the power of data from others, in order to facilitate the performance on their own local tasks. One important problem FL concerns is data distribution heterogeneity, since the decentralized data, collected by different institutes using different methods and aiming at different tasks, are highly likely to follow non-identical distribution. Prior works approach this problem from different aspects, including optimization process (Li et al., 2020; Karimireddy et al., 2019), personalized FL (Huang et al., 2021; Dinh et al., 2021; Fallah et al., 2020a), clustered FL (Ghosh et al., 2020; Jeong et al., 2018; Briggs et al., 2020), etc.

As more advanced techniques are developed for learning with graph data, using graphs to model and solve real-world problems becomes more popular. One important scenario of graph learning is graph classification, where models such as graph kernels (Yanardag & Vishwanathan, 2015; Shervashidze & Borgwardt, 2009; Vishwanathan et al., 2010; Yang et al., 2018) and graph neural networks (Kipf & Welling, 2017; Xu et al., 2019; Ying et al., 2018; Yang et al., 2020a;b; 2019) are used to predict graph-level labels based on the features and structures of graphs. One real scenario of graph classification is molecular property prediction, which is an important task in cheminformatics and AI medicine. In the area of bioinformatics, graph classification can be used to learn the representation of proteins and classify them into enzymes or non-enzymes. For collaboration networks, graph classification can learn from its sub-networks about the information of research areas, topics, genre, etc.

Since the key idea of FL is the sharing of underlying common information, as (Leskovec et al., 2005) discusses that real-world graphs preserve many common properties, we become curious about the question, *whether real-world graphs from heterogeneous sources (e.g., different datasets or even divergent domains) can provide useful common information among each other?* To understand this question, we first conduct preliminary data analysis to explore real-world graph properties, and try to find clues about common patterns shared among graphs across datasets. As shown in Table 1, we analyze four typical datasets from different domains, i.e., PTC_MR (molecular structures), ENZYMES (protein structures) and IMDB-BINARY (social communities), and MSRC_21 (superpixel networks). We find them to indeed share certain properties that are statistically significant compared to random graphs with the same numbers of nodes and links (generated with the Erdős–Rényi model (Erdős & Rényi, 1959; Gilbert, 1959)). Such observations confirm the claim about common patterns underlying real-

[1]Department of Computer Science, Emory University. Correspondence to: Carl Yang <j.carlyang@emory.edu>.

This work was presented at the International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML 2021 (FL-ICML'21). This workshop does not have official proceedings and this paper is non-archival. 

*Table 1.* Data analysis on important graph properties shared among real-world graphs across different domains. For example, large Kurtosis values (Pearson, 1905) indicate long-tail distribution of node degrees, which is observed in ENZYMES, IMDB-BINARY and MSRC_21; similar average shortest path lengths are observed in PTC_MR, ENZYMES and MSRC_21, although their actual graph sizes are rather different; large CC are observed in ENZYMES, IMDB-BINARY, and MSRC_21 and large LC are observed in almost all graphs.

| Property | kurtosis of degree distribution | | | avg. shortest path length | | | largest component size (LC, %) | | | clustering coefficient (CC) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | real | random | p-value | real | random | p-value | real | random | p-value | real | random | p-value |
| PTC_MR (chemical) | 2.1535 | 2.4424 | 0.9999 | 3.36 | 2.42 | $\sim 0$ | 100 | 82.68 | $\sim 0$ | 0.0095 | 0.1201 | $\sim 0$ |
| ENZYMES (biological) | 3.0106 | 2.8243 | 0.0027 | 4.44 | 2.56 | $\sim 0$ | 98.24 | 97.69 | 0.2054 | 0.4516 | 0.1425 | $\sim 0$ |
| IMDB-BINARY (social) | 8.9262 | 2.2791 | $\sim 0$ | 1.48 | 1.54 | $\sim 0$ | 100 | 99.93 | 0.0023 | 0.9471 | 0.5187 | $\sim 0$ |
| MSRC_21 (visual) | 3.6959 | 2.9714 | $\sim 0$ | 4.09 | 2.81 | $\sim 0$ | 100 | 99.43 | $\sim 0$ | 0.5147 | 0.0655 | $\sim 0$ |

world graphs, which can largely influence the graph mining models and motivates us to consider the FL of graph classification across datasets and even domains.

Although common patterns exist among graph datasets, we can still observe certain heterogeneity. In fact, the detailed graph structure distributions and node feature distributions can both diverge due to various reasons. To demonstrate this, we design and evaluate a structure heterogeneity measure and a feature heterogeneity measure in different scenarios (c.f. Section 4.1). We refer to the graphs possibly with significant heterogeneity in our cross-dataset FL setting as non-IID graphs, which concerns both structure non-IID and feature non-IID, where naïve FL algorithms like `FedAvg` (McMahan et al., 2017b) can fail and even backfire (c.f. Section 5.2). Moreover, as the heterogeneity varies from case to case, a dynamic FL algorithm is needed to keep track of such heterogeneity of non-IID graphs while conducting collaborative model training.

Due to the observations that the graphs in one client can be similar to those in some clients but not the others, we get motivated by (Briggs et al., 2020) and find it intuitive to consider a clustered FL framework, which assigns local clients to multiple clusters with less data heterogeneity. To this end, we propose a novel graph-level clustered FL framework (termed GCFL) through integrating the powerful GIN model (Xu et al., 2019) into clustered FL, where the server can dynamically cluster the clients based on the gradients of GIN without additional prior knowledge, while collaboratively training multiple GINs as necessary for homogeneous clusters of clients. We theoretically analyze that the model parameters of GIN indeed reflect the structures and features of graphs, and thus using the gradients of GIN for clustering in principle can yield clusters with reduced heterogeneity of both structures and features.

We conducted extensive experiments with various settings to demonstrate the effectiveness of our framework. The experimental results show surprisingly positive results brought by our novel setting of cross-dataset/cross-domain FL for graph classification, where our GCFL framework can effectively and consistently outperform the baselines.

## 2. Related Works

**Federated Learning** Federated learning (FL) has gained increasing attention as a training paradigm under the setting where data are distributed at remote devices and models are collaboratively trained under the coordination of a central server. `FedAvg` was first proposed by (McMahan et al., 2017a) which illustrates the general setting of an FL framework. Since the original `FedAvg` relies on the optimization by SGD, data non-IID distribution will not guarantee the stochastic gradients to be an unbiased estimation of the full gradient, thus hurting the convergence of FL. In fact, multiple experiments (Zhao et al., 2018; Li et al., 2020; Karimireddy et al., 2019) have shown that the convergence will be slow and unstable, and the accuracy will degrade with `FedAvg` when data at each client are statistically heterogeneous (non-IID). (Zhao et al., 2018; Jeong et al., 2018; Huang et al., 2020) proposed different data sharing strategies to tackle the data heterogeneity problem by sharing the local device data or server-side proxy data, which still requires certain public common data, whereas other studies explored the convergence guarantee under the data non-IID setting by assuming bounded gradients (Wang et al., 2019; Yu et al., 2019) or additional noise (Khaled et al., 2020). There are also works seeking to reduce the variance of the clients (Liang et al., 2019; Karimireddy et al., 2019; Li et al., 2020). Furthermore, multiple works have been proposed to explore the connection between model-agnostic meta-learning (MAML) and personalized FL (Fallah et al., 2020b; Chen et al., 2018). They aim to learn a generalizable global model and then fine-tune it on local clients, which may still fail when data on local clients are from divergent domains with high heterogeneity. Some personalized FL works (Dinh et al., 2021; Li et al., 2021) studied the bi-level problem of optimization which decouples the local and global optimization, while each client maintaining its own model can be costly and inefficient.

**Federated Learning on Graphs** Although FL has been intensively studied with Euclidean data such as images, there exist few studies about FL for graph data. (Lalitha et al., 2019) first introduced FL on graph data, by regarding each client as a node in a graph. (Caldarola et al., 2021) studied the cross-domain heterogeneity problem in

FL by leveraging GCN to model the interaction between domains. (Chen et al., 2020) proposed a generalized federated knowledge graph embedding framework that can be applied for multiple knowledge graph embedding algorithms. Moreover, there are several works exploring the Graph Neural Networks (GNNs) under the FL setting: (Jiang et al., 2020; Zhou et al., 2020; Wu et al., 2021) focused on the privacy issue of federated GNNs; (Wang et al., 2020) incorporated model-agnostic meta-learning (MAML) into graph FL; (Zhang et al., 2021) studied the missing neighbor generation problem in the subgraph FL setting; (Wang et al., 2021) proposed a computationally efficient way of GCN architecture search with FL. Most existing works consider node classification and link prediction on graphs, which cannot be trivially applied to our graph classification setting.

## 3. Preliminaries

### 3.1. Graph Neural Networks (GNNs)

In general, given the structure and feature information of a graph $G = (V, E, X)$, where $V$, $E$, $X$ denote nodes, edges and node features, GNNs target to learn the representations of graphs, such as a node embedding $h_v \in \mathbb{R}^{d_v}$, or a graph embedding $h_G \in \mathbb{R}^{d_G}$. A GNN typically includes message propagation and neighborhood aggregation, in which each node iteratively gathers the information propagated by its neighbors, and aggregates them with its own information to update its representation. Generally, a $L$-layer GNN can be formulated as

$$h_v^{(l+1)} = \sigma(h_v^{(l)}, agg(\{h_u^{(l)}; u \in \mathcal{N}_v\})), \forall l \in [L], \quad (1)$$

where $h_v^{(l)}$ is the representation of node $v$ at the $l^{th}$ layer, and $h_v^{(0)} = x_v$ is the node feature. $\mathcal{N}_v$ is neighbors of node $v$, $agg(\cdot)$ is a aggregation function that can vary for different GNN variants, and $\sigma$ represents a activation function. For a graph-level representation $h_G$, it can be pooled from the representations of all nodes, as

$$h_G = readout(\{h_v; v \in V\}), \quad (2)$$

where $readout(\cdot)$ can be implemented as mean pooling, sum pooling, etc, which essentially aggregates the embeddings of all nodes on the graph into a single embedding vector to achieve tasks like graph classification and regression.

### 3.2. The `FedAvg` algorithm

`FedAvg` (McMahan et al., 2017b) is the first basic FL algorithm and is commonly used as the starting point for more advance FL framework design. The key idea of `FedAvg` is to aggregate the updated model parameters transmitted from local clients and then re-distribute the averaged parameters to clients. Specifically, given $m$ clients in total, at each communication round $t$, the server first samples a partition of clients $\{\mathbb{S}_i\}^{(t)}$. For each client $\mathbb{S}_i$ in $\{\mathbb{S}_i\}^{(t)}$, it locally trains the model downloaded from the server with its own data distribution $\mathcal{D}_i$ for $E_{local}$ epochs. The client $\mathbb{S}_i$ then transmits its updated parameters $w_i^{(t)}$ to the server, and the server will aggregate these updates by

$$w^{(t+1)} = \sum_{i=1}^{m} \frac{|D_i|}{|D|} w_i^{(t)}, \quad (3)$$

where $|D_i|$ is the size of data samples in $\mathbb{S}_i$ and $|D|$ is the total size of samples over all clients. Next, the server broadcasts the new parameters $w^{(t+1)}$ to remote clients, and at the $(t+1)$ round clients use $w^{(t+1)}$ to start their local training for another $E_{local}$ epochs.

## 4. The `GCFL` Framework

### 4.1. Non-IID Structures and Features across Clients

From Table 1 we notice that real-world graphs tend to share certain general properties across different graphs, datasets and even domains, which motivates the graph-level FL framework. However, there still exist differences when the detailed graph structures and node features are being considered. In Table 2, we present the average pair-wise structure and feature heterogeneity among graphs in a single dataset, a single domain, and multiple domains. Specifically, for structure heterogeneity, we use the Anonymous Walk Embeddings (AWEs) (Ivanov & Burnaev, 2018) to generate representations for graphs, and compute the Jensen-Shannon (JS) distance between the AWEs of each pair of graphs; for feature heterogeneity, we calculate the empirical distribution of feature similarity between all pairs of linked nodes in each graph, and compute the JS divergence between the feature similarity distributions of each pair of graphs.

As we can observe in Table 2, both graph structures and features demonstrate different levels of heterogeneity within a single dataset, a single domain, and across domains. We refer to graphs with such structure and feature heterogeneity as non-IID graphs. Intuitively, directly applying naïve FL algorithms like `FedAvg` on clients with non-IID graphs can be ineffective and even backfiring. To be specific, structure heterogeneity makes it difficult for a model to capture the universally important graph structure patterns across different clients, whereas feature heterogeneity makes it hard for a model to learn the universally appropriate message propagation functions across different clients. How can we leverage the shared graph properties among clients while addressing the non-IID structures and features across clients?

### 4.2. Problem Formulation

Motivated by our real graph data analysis in Tables 1 and 2, we propose a novel framework of **G**raph **C**lustered **F**ederated **L**earning (GCFL). The main idea of GCFL is

*Table 2.* Summary of the average structure/feature heterogeneity for some datasets. In general, the structure heterogeneity increases from the settings of one dataset to across-dataset, and to across-domain. However, the feature heterogeneity is more case-by-case, and the high variances indicate that graphs could have large feature divergence even within the same dataset. Additionally, it is not necessarily true that one dataset itself should be more homogeneous (e.g., IMDB-BINARY).

| dataset | IMDB-BINARY (social) | COX2 (molecules) | COX2 (molecules)<br>PTC_MR (molecules) | COX2 (molecules)<br>ENZYMES (proteins) | COX2 (molecules)<br>IMDB-BINARY (social) |
|---|---|---|---|---|---|
| avg. struc. hetero. | 0.4406 ($\pm$0.0397) | 0.3246 ($\pm$0.0145) | 0.3689 ($\pm$0.0540) | 0.5082 ($\pm$0.0399) | 0.6079 ($\pm$0.0331) |
| avg. feat. hetero. | 0.1785 ($\pm$0.1226) | 0.0427 ($\pm$0.0314) | 0.1837 ($\pm$0.1065) | 0.1912 ($\pm$0.1000) | 0.1642 ($\pm$0.1006) |

to jointly find clusters of clients with graphs of similar structures and features, and train the graph mining models with `FedAvg` among clients in the same clusters.

Specifically, we are inspired by the Clustered Federated Learning (CFL) framework on Euclidean data (Sattler et al., 2020) and consider a CFL setting with one central server and a set of $n$ local clients $\{\mathbb{S}_1, \mathbb{S}_2, \ldots, \mathbb{S}_n\}$. Different from the traditional FL setting, the server can dynamically cluster the clients into a set of clusters $\{\mathbb{C}_1, \mathbb{C}_2, \ldots\}$ and maintain $m$ cluster-wise models. In our GCFL setting, each local client $\mathbb{S}_i$ owns a set of graphs $\mathcal{G}_i = \{G_1, G_2, \ldots\}$, where each $G_j = (V_j, E_j, X_j, y_j) \in \mathcal{G}_i$ is a graph data sample with a set of nodes $V_j$, a set of edges $E_j$, node features $X_j$, and a graph class label $y_j$. The task on each local client $\mathbb{S}_i$ is graph classification that predicts the class label $\hat{y}_j = h_k^*(G_j)$ for each graph $G_j \in \mathcal{G}_i$, where $h_k^*$ is the collaboratively learned optimal graph mining model for cluster $\mathbb{C}_k$ to which $\mathbb{S}_i$ belongs. Our goal is to minimize the loss function $F(\Theta_k) := \mathrm{E}_{\mathbb{S}_i \in \mathbb{C}_k}[f(\theta_{k,i}; \mathcal{G}_i)]$, for all clusters $\{\mathbb{C}_k\}$. The function $f(\theta_{k,i}; \mathcal{G}_i)$ is a local loss function for client $\mathbb{S}_i$ which belongs to cluster $\mathbb{C}_k$. In the meantime, we also aim to maintain a dynamic cluster assignment $\Gamma(\mathbb{S}_i) \rightarrow \{\mathbb{C}_k\}$ based on the FL process.

### 4.3. Technical Design

GNNs are demonstrated to be powerful for learning graph representations and have been wildly used in graph mining. More importantly, the model parameters and their gradients of GNNs can reflect the graph structure and feature information (more details in Section 4.4). Thus, we use GNNs as the graph mining model in our GCFL framework.

Specifically, our GCFL framework dynamically clusters clients by leveraging their transmitted gradients $\{\Delta\theta_i\}_{i=1}^n$, in order to maximize the collaboration among more homogeneous clients and eliminate the harm from heterogeneous clients. According to (Sattler et al., 2020), if the data distribution of clients are highly heterogeneous, FL cannot jointly optimize all local loss functions, which means that the norm of gradients are greater than zero. Here, we introduce a hyper-parameter $\varepsilon_1$ as a criterion

$$\delta_{mean} = \| \sum_{i \in [n]} \Delta\theta_i \| < \varepsilon_1. \tag{4}$$

to check for stopping the general FL stage in GCFL. In the meantime, if some gradients have a large norm, which means that they fail to approach to their stationary points, a clustering step is needed to eliminate the negative influence among heterogeneous clients. We then introduce the second criterion with a hyper-parameter $\varepsilon_2$ to split the clusters when

$$\delta_{max} = \max(\|\Delta\theta_i\|) > \varepsilon_2 > 0. \tag{5}$$

The GCFL framework follows a top-down bi-partitioning mechanism. At each communication round $t$, the server receives $m$ sets of gradients $\{\{\Delta\theta_{i_1}\}, \{\Delta\theta_{i_2}\}, \ldots, \{\Delta\theta_{i_m}\}\}$ from clients in clusters $\{\mathbb{C}_1, \mathbb{C}_2, \ldots, \mathbb{C}_m\}$. For a cluster $\mathbb{C}_k$, if $\delta_{mean}^k$ and $\delta_{max}^k$ satisfy the Eqs. 4 and 5, the server will calculate a cluster-wise cosine similarity matrix $\alpha_k$, and use it to perform an agglomerative clustering, which divides the cluster $\mathbb{C}_k \rightarrow \{\mathbb{C}_{k1}, \mathbb{C}_{k2}\}$. The clustering mechanism based on Eqs. 4 and 5 can automatically and dynamically determine the number of clusters along the FL, while the two hyper-parameters $\varepsilon_1$ and $\varepsilon_2$ can be easily set through some simple experiments following (Sattler et al., 2020).

For a client $\mathbb{S}_i$ in cluster $\mathbb{C}_k$, it tries to find $\hat{\theta}_{k,i}$ that is close to the real solution $\theta_{k,i}^* = \arg\min_{\theta_i \in \Theta_k} f(\theta_{k,i}; \mathcal{G}_i)$. At a communication round $t$, the client $\mathbb{S}_k$ transmits its gradient to the server

$$\Delta\theta_{k,i}^t = \hat{\theta}_{k,i}^t - \theta_{k,i}^{t-1}. \tag{6}$$

Since the server maintains the cluster assignments, it can aggregate the gradients cluster-wise by

$$\theta_k^{t+1} = \theta_k^t + \sum_{i \in [n_k]} \Delta\theta_{k,i}^t. \tag{7}$$

### 4.4. Theoretical Analysis

We theoretically analyze that the gradient-based FL algorithm on GNNs can in principle reduce the structure and feature heterogeneity in clusters, by proving that the gradients of GNNs can reflect the structures and features of their training graphs.

**Definition 4.1** *Let a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ which maps from the metric space $(\mathcal{X}, d)$ to $(\mathcal{Y}, d')$, the function $f$ is considered to have $\delta$ distortion if $\forall u, v \in \mathcal{X}$, $\frac{1}{\delta} d(u, v) \leq d'(f(u), f(v)) \leq d(u, v)$.*

**Theorem 4.1** *(Bourgain theorem (Bourgain, 1985)) Given an $n$-point metric space $(\mathcal{X}, d)$ and an embedding function $f$ as defined above, $\forall u, v \in \mathcal{X}$, there exist an embedding mapped from $(\mathcal{X}, d)$ to $\mathbb{R}^k$ with the distortion of the embedding being $O(\log n)$.*

**Problem 1.** *GCFL which involves the communication of the gradients between graphs with heterogeneous structures distributed among different clients, the structure and feature difference can be captured by the GNN gradients.*

For simplicity, we solve Problem 1 with the GNN of Simple Graph Convolutions (SGC) (Wu et al., 2019), through the following two propositions.

**Proposition 4.1** *Given a graph $G$ with fixed structure represented by the normalized graph Laplacian $\mathcal{L} = \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}$, feature represented with $X$, and an SGC $f(\mathcal{L}, X) = softmax(\mathcal{L}^K X\Theta)$ with weights $\Theta$ trained on graph $G$. If we have another graph $G'$ with different structure $\mathcal{L}'$, the weight difference $||\Theta' - \Theta||_2$ is bounded with the structure difference.*

**Proposition 4.2** *Given a graph $G$ with fixed structure represented by the normalized graph Laplacian $\mathcal{L} = \widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}$, feature represented with $X$, and an SGC $f(\mathcal{L}, X) = softmax(\mathcal{L}^K X\Theta)$ with weights $\Theta$ trained on graph $G$. If we have another graph $G'$ with different feature $\mathcal{X}'$, the weight difference $||\Theta' - \Theta||_2$ is bounded with the feature difference.*

We prove proposition 4.1 and 4.2 in the Appendix. We use the *Bourgain theorem* to bound the difference between embeddings generated with different graph structures/features, and prove that the feature and structure information of a graph is incorporated into the model weights (gradients). By proving that the model weights (gradients) are bounded with the structure/feature difference, we show that the gradient will change with the structure and feature. This further justifies that our proposed gradient based clustering framework GCFL can capture the structure and feature information.

# 5. Experiments

## 5.1. Experimental Settings

**Datasets** We use 13 graph classification datasets (Morris et al., 2020) from 3 domains including 7 molecule datasets (MUTAG, BZR, COX2, DHFR, PTC_MR, AIDS, NCI1), 3 protein datasets (ENZYMES, DD, PROTEINS), and 3 social network datasets (COLLAB, IMDB-BINARY, IMDB-MULTI), each with a set of graphs. Node features are available in some datasets, and graph labels are either binary or multi-class. Details of the datasets are presented in the Appendix.

We design two settings that follow different data partitioning schemes. One setting is to randomly distribute graphs from a single dataset to a number of clients, with each client holding a distinct set of 30-50 graphs, among which 10% are held out for testing. In the other setting, we use multiple datasets either from a single domain or multiple domains. Each client holds a distinct set of 50 graphs from one dataset, among which 10% are held out for testing. In the first setting, we use NCI1, PROTEINS, and IMDB-BINARY from 3 domains and distribute them to 80, 30, 20 clients, respectively. In the second setting, we create 3 data groups including MOLECULES which consists of 7 datasets from the molecule domain distributed into 7 clients, BIOCHEM where we add 3 datasets from the protein domain into MOLECULES and distribute them into 10 clients, MIX where we add 3 datasets from the social domain into BIOCHEM and distribute them into 13 clients.

**Baselines** We use `self-train`[1] as the first baseline to test whether FL can bring improvements to each client through collaborative training. In `self-train`, each client first downloads the same randomly initialized model from the server and then trains locally without communications. Then we implement two widely used FL baselines `FedAvg` (McMahan et al., 2017a) and `FedProx` (Li et al., 2020), the latter of which can deal with data and system heterogeneity in non-graph FL. For the graph classification model, we use the same GIN design (Xu et al., 2019), which represents the state-of-the-art GNN for graph-level tasks. We fix the GIN architecture and hyper-parameters through all baselines.

**Parameter settings** We use the two-layer GINs with hidden size of 64. We use a batch size of 32, and an Adam (Kingma & Ba, 2017) optimizer with learning rate 0.001 and weight decay $5e^{-4}$. The $\mu$ for `FedProx` is set to 0.01. For all FL methods, the local epoch $E$ is set to 3. The two important hyper-parameters $\varepsilon_1$ and $\varepsilon_2$ as clustering criteria vary in different groups of data, which are set through offline training for about 50 rounds following (Sattler et al., 2020). We run all experiments for five random repetitions on a server with 8 24GB NVIDIA TITAN RTX GPUs.

## 5.2. Experimental Results

**Federated graph classification within single datasets** Conceptually, clients in this setting are more homogeneous. As can be seen from the results in Table 3 (upper part), our framework can obviously improve the performance of graph classification over local clients. For NCI1 distributed on 80 clients, GCFL achieves 5.57% performance gains over `self-train`, and it helps 17 more clients than `FedAvg` and 12 more clients than `FedProx`. For PROTEINS on the total 30 clients, the average performance gain over `self-train` is more significant, i.e., 12.88%. The GCFL framework is able to improve all 30 clients. For IMDB-BINARY on 20 clients,

[1] We do not have a global model as baseline as datasets are from various domains and their tasks are divergent.

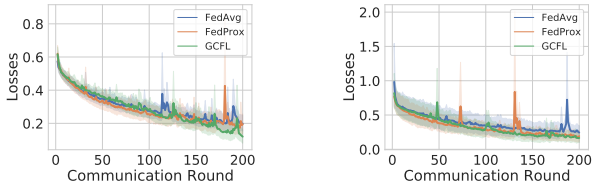
*Table 3.* Average accuracy and minimum gain over `self-train` on all clients, and the ratio of clients which get improved.

| Single dataset multiple client setting | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset (# clients) | NCI1 (80) | | | PROTEINS (30) | | | IMDB-BINARY (20) | | |
| Accuracy | average | min gain | ratio | average | min gain | ratio | average | min gain | ratio |
| `self-train` | 0.5874(±0.018) | — | — | 0.5979(±0.028) | — | — | 0.6178(±0.024) | — | — |
| `FedAvg` | 0.6013(±0.021) | -0.0423 | 56/80 | 0.6477(±0.027) | -0.0125 | 28/30 | 0.6122(±0.031) | -0.0300 | 5/20 |
| `FedProx` | 0.6019(±0.020) | -0.0443 | 60/80 | 0.6430(±0.023) | -0.0339 | 28/30 | 0.6117(±0.028) | -0.0380 | 10/20 |
| GCFL | **0.6201**(±0.015) | -0.0326 | **73**/80 | **0.6749**(±0.023) | 0.0268 | **30**/30 | **0.6345**(±0.020) | -0.0100 | **17**/20 |
| Multiple datasets multiple client setting | | | | | | | | | |
| Dataset (# domains) | MOLECULES (1) | | | BIOCHEM (2) | | | MIX (3) | | |
| Accuracy | average | min gain | ratio | average | min gain | ratio | average | min gain | ratio |
| `self-train` | 0.6992(±0.027) | — | — | 0.6405(±0.022) | — | — | 0.6136(±0.022) | — | — |
| `FedAvg` | 0.7133(±0.029) | -0.0211 | 4/7 | 0.6539(±0.030) | -0.0237 | 6/10 | 0.6307(±0.027) | -0.0322 | 9/13 |
| `FedProx` | 0.7082(±0.025) | -0.0468 | 3/7 | 0.6507(±0.032) | -0.0433 | 7/10 | 0.6237(±0.026) | -0.0383 | 8/13 |
| GCFL | **0.7356**(±0.029) | -0.0010 | **6**/7 | **0.6785**(±0.018) | -0.0017 | **9**/10 | **0.6526**(±0.026) | -0.0068 | **12**/13 |

both `FedAvg` and `FedProx` fail to improve the clients on average, and `FedAvg` can help only 5 clients. This is consistent with the results shown in Table 2 that IMDB-BINARY itself has relatively high structure and feature heterogeneity, which makes `FedAvg` ineffective. Our GCFL framework can still improve the performance on IMDB-BINARY on average, and help 17 out of 20 clients. These experimental results demonstrate that our frameworks are effective on the single-dataset multi-client FL setting.

**Federated graph classification across multiple datasets** According to our data analysis in Tables 1 and 2, clients in such a setting are more heterogeneous. As can be seen in Table 3 (lower part), our framework GCFL can significantly improve the performance of clients with distinct datasets. We conduct experiments with multiple datasets in two settings: single domain (using the data group MOLECULES), and across domains (using the data groups BIOCHEM and MIX). The results show $5.21\% - 6.36\%$ improvements of our frameworks compared to `self-train`. A noticeable result is that our GCFL framework improves almost all clients' performance for all 3 data groups. These results indicate that graphs across datasets or even across domains are able to help each other through proper FL, which is a surprising and interesting start point for further study.

**Convergence analysis** We visualize the testing loss with respect to the communication round to show the convergence of GCFL compared with the standard federated learning baselines. Figure 1 shows the training curves on two settings, which illustrates that GCFL achieves similar convergence rate as `FedProx`, which is the state-of-the-art FL framework dealing with non-IID Euclidean data. We also notice that both GCFL and `FedProx` can converge to a lower loss compared with `FedAvg`, which corroborates our consideration of the non-IID problem in our setting.




(a) oneDS:PROTEINS (b) multiDS:MIX

*Figure 1.* Average with standard deviation of the training curves of all clients.

**More results in Appendix** In Table 3, we averaged the accuracy across all clients for presentation simplicity. To understand the detailed performance by clients, we present different Violin plots in the Appendix. Besides, we also show more results regarding various settings (overlapping clients and real vs. synthetic node features) in the Appendix.

## 6. Conclusion

In this work, we propose a novel setting of cross-dataset and cross-domain federated graph classification. The technique (GCFL) we develop allow multiple data owners holding structure and feature non-IID graphs to collaboratively train powerful graph classification neural networks without the need of direct data sharing. As the first trial, we focus on the effectiveness of FL in this setting and have not carefully studied other issues such as data privacy, although it is intuitive to preserve the privacy of clients by introducing an encryption mechanism (e.g. applying orthonormal transformations), and to prevent from adversarial scenarios by clustering out the malicious clients. Due to its evident motivations and proofs on the effective FL in a new setting, we believe this work can serve as a stepping stone for many interesting future studies.

## References


Bourgain, J. On lipschitz embedding of finite metric spaces in hilbert space. *Israel Journal of Mathematics*, 52 (1): 46–52, 1985.

Briggs, C., Fan, Z., and Andras, P. Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In *IJCNN*, 2020.

Caldarola, D., Mancini, M., Galasso, F., Ciccone, M., Rodolà, E., and Caputo, B. Cluster-driven graph federated learning over multiple domains. In *ICMLW*, 2021.

Chen, F., Luo, M., Dong, Z., Li, Z., and He, X. Federated meta-learning with fast convergence and efficient communication. *arXiv preprint arXiv:1802.07876*, 2018.

Chen, M., Zhang, W., Yuan, Z., Jia, Y., and Chen, H. Fede: Embedding knowledge graphs in federated setting. *arXiv preprint arXiv:2010.12882*, 2020.

Dinh, C. T., Tran, N. H., and Nguyen, T. D. Personalized federated learning with moreau envelopes. In *NeurIPS*, 2021.

Erdős, P. and Rényi, A. On random graphs. 1. *Publicationes Mathematicae.*, 6:290–297, 1959.

Fallah, A., Mokhtari, A., and Ozdaglar, A. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In *NeurIPS*, 2020a.

Fallah, A., Mokhtari, A., and Ozdaglar, A. Personalized federated learning: A meta-learning approach. In *NeurIPS*, 2020b.

Ghosh, A., Chung, J., Yin, D., and Ramchandran, K. An efficient framework for clustered federated learning. In *NeurIPS*, 2020.

Gilbert, E. Random graphs. *Annals of Mathematical Statistics.*, 30 (4):1141–1144, 1959.

Huang, L., Yin, Y., Fu, Z., Zhang, S., Deng, H., and Liu, D. Loadaboost: Loss-based adaboost federated machine learning on medical data. *PLoS ONE*, 15(4):e0230706, 2020.

Huang, Y., Chu, L., Zhou, Z., Wang, L., Liu, J., Pei, J., and Zhang, Y. Personalized cross-silo federated learning on non-iid data. In *AAAI*, 2021.

Ivanov, S. and Burnaev, E. Anonymous walk embeddings. In *ICML*, 2018.

Jeong, E., Oh, S., Kim, H., Park, J., Bennis, M., and Kim, S.-L. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. In *NIPSW*, 2018.

Jiang, M., Jung, T., Karl, R., and Zhao, T. Federated dynamic gnn with secure aggregation. *arXiv preprint arXiv:2009.07351*, 2020.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14 (1), 2019.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *ICML*, 2019.

Khaled, A., Mishchenko, K., and Richtárik, P. Tighter theory for local sgd on identical and heterogeneous data. In *AISTATS*, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2017.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Lalitha, A., Kilinc, O. C., Javidi, T., and Koushanfar, F. Peer-to-peer federated learning on graphs. *arXiv preprint arXiv:1901.11173*, 2019.

Leskovec, J., Chakrabarti, D., Kleinberg, J., and Faloutsos, C. Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. In *ECML-PKDD*, 2005.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, 2020.

Li, T., Hu, S., Beirami, A., and Smith, V. Ditto: Fair and robust federated learning through personalization. In *ICML*, 2021.

Liang, X., Shen, S., Liu, J., Pan, Z., Chen, E., and Cheng, Y. Variance reduced local sgd with lower communication complexity. *arXiv preprint arXiv:1912.12844*, 2019.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, 2017a.

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017b.

Morris, C., Kriege, N. M., Bause, F., Kersting, K., Mutzel, P., and Neumann, M. Tudataset: A collection of benchmark datasets for learning with graphs. In *ICMLW*, 2020.

Pearson, K. Das fehlergesetz und seine verallgemeinerungen durch fechner und pearson. a rejoinder [the error law and its generalizations by fechner and pearson. a rejoinder]. *Biometrika*, 4 (1-2):169–212, 1905.

Sattler, F., Müller, K.-R., and Samek, W. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *TNNLS*, pp. 1–13, 2020.

Shervashidze, N. and Borgwardt, K. M. Fast subtree kernels on graphs. In *NIPS*, 2009.

Vishwanathan, S. V. N., Schraudolph, N. N., Kondor, R., and Borgwardt, K. M. Graph kernels. *JMLR*, 11:1201–1242, 2010.

Wang, B., Li, A., Li, H., and Chen, Y. Graphfl: A federated learning framework for semi-supervised node classification on graphs. *arXiv preprint arXiv:2012.04187*, 2020.

Wang, C., Chen, B., Li, G., and Wang, H. Fl-agcns: Federated learning framework for automatic graph convolutional network search. In *ICML*, 2021.

Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., and Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 37(6): 1205–1221, 2019.

Wu, C., Wu, F., Cao, Y., Huang, Y., and Xie, X. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv preprint arXiv:2102.04925*, 2021.

Wu, F., Zhang, T., Souza Jr, A. H. d., Fifty, C., Yu, T., and Weinberger, K. Q. Simplifying graph convolutional networks. In *ICML*, 2019.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *ICLR*, 2019.

Yanardag, P. and Vishwanathan, S. Deep graph kernels. In *KDD*, 2015.

Yang, C., Liu, M., Zheng, V. W., and Han, J. Node, motif and subgraph: Leveraging network functional blocks through structural convolution. In *ASONAM*, 2018.

Yang, C., Zhuang, P., Shi, W., Luu, A., and Li, P. Conditional structure generation through graph variational generative adversarial nets. In *NIPS*, 2019.

Yang, C., Xiao, Y., Zhang, Y., Sun, Y., and Han, J. Heterogeneous network representation learning: A unified framework with survey and benchmark. In *TKDE*, 2020a.

Yang, C., Zhang, J., and Han, J. Co-embedding network nodes and hierarchical labels with taxonomy based generative adversarial nets. In *ICDM*, 2020b.

Ying, R., You, J., Morris, C., Ren, X., Hamilton, W. L., and Leskovec, J. Hierarchical graph representation learning with differentiable pooling. In *NeurIPS*, 2018.

You, J., Ying, R., and Leskovec, J. Position-aware graph neural networks. In *ICML*, pp. 7134–7143, 2019.

Yu, H., Yang, S., and Zhu, S. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *AAAI*, 2019.

Zhang, K., Yang, C., Li, X., Sun, L., and Yiu, S. M. Subgraph federated learning with missing neighbor generation. In *ICML-FL*, 2021.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

Zhou, J., Chen, C., Zheng, L., Wu, H., Wu, J., Zheng, X., Wu, B., Liu, Z., and Wang, L. Vertically federated graph neural network for privacy-preserving node classification. *arXiv preprint arXiv:2005.11903*, 2020.

# A. Missing Proofs in Section 4.4

## A.1. Proof of Proposition 4.1

Assume the structure difference between graph $G$ and $G'$ is bounded with

$$||\mathcal{L}' - \mathcal{L}||_2^2 = ||E_{\mathcal{L}}||_2^2 \leq \epsilon_{\mathcal{L}}, \tag{8}$$

The difference between the original weights $\Theta$ and the weights trained on the new graph structure $\Theta'$ is represented as

$$\begin{aligned} ||\Theta' - \Theta||_2^2 &= ||(\mathcal{L}'X)^{-1}Y' - (\mathcal{L}X)^{-1}Y||_2^2 \\ &= ||X^{-1}(\mathcal{L}'^{-1}Y' - \mathcal{L}^{-1}Y)||_2^2. \end{aligned} \tag{9}$$

Given that $||\mathcal{L} \cdot \mathcal{L}'||_2^2 = ||\mathcal{L} \cdot (\mathcal{L} + E_{\mathcal{L}})||_2^2 \geq ||\mathcal{L}E_{\mathcal{L}}||_2^2$. Let $||\mathcal{L}E_{\mathcal{L}}||_2^2 = \delta_{\mathcal{L}}$ , then we can get $||\mathcal{L}'^{-1} - \mathcal{L}^{-1}||_2^2 = ||E_{\mathcal{L}^{-1}}|| \leq \frac{\epsilon_{\mathcal{L}}}{\delta_{\mathcal{L}}}$ .

Given the Bourgain theorem (You et al., 2019), the difference between the embedding $Y$ and $Y'$ is bounded with

$$||\mathcal{Y}' - \mathcal{Y}||_2^2 = ||E_Y||_2^2 \leq \epsilon_Y, \tag{10}$$

The weight difference can then be bounded with

$$\begin{aligned} ||\Theta' - \Theta||_2^2 &\leq ||X^{-1}||_2^2||(\mathcal{L}^{-1} + E_{\mathcal{L}^{-1}})(Y + E_Y) - \mathcal{L}^{-1}Y||_2^2 \\ &= ||X^{-1}||_2^2||\mathcal{L}^{-1}E_Y + E_{\mathcal{L}^{-1}}Y + E_{\mathcal{L}^{-1}}E_Y||_2^2 \\ &\leq ||X^{-1}||_2^2\Big[\epsilon_Y||\mathcal{L}^{-1}||_2^2 + \frac{\epsilon_{\mathcal{L}}}{\delta_{\mathcal{L}}}||Y||_2^2 + \frac{\epsilon_{\mathcal{L}}\epsilon_Y}{\delta_{\mathcal{L}}}\Big]. \end{aligned} \tag{11}$$

With the trained SGC, the feature and graph structure is fixed as $X$ and $\mathcal{L}$. Thus the weight difference is bounded with $X$ and $\mathcal{L}$.

*Table 4.* The statistics of datasets.

| dataset | statistics | | | | | dataset | statistics | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | #graphs | avg. #nodes | avg. #edges | #classes | node features | | #graphs | avg. #nodes | avg. #edges | #classes | node features |
| MUTAG | 188 | 17.93 | 19.79 | 2 | original | ENZYMES | 600 | 32.63 | 62.14 | 6 | original |
| BZR | 405 | 35.75 | 38.36 | 2 | original | DD | 1178 | 284.32 | 715.66 | 2 | original |
| COX2 | 467 | 41.22 | 43.45 | 2 | original | PROTEINS | 1113 | 39.06 | 72.82 | 2 | original |
| DHFR | 467 | 42.43 | 44.54 | 2 | original | COLLAB | 5000 | 74.49 | 2457.78 | 3 | degree |
| PTC_MR | 344 | 14.29 | 14.69 | 2 | original | IMDB-BINARY | 1000 | 19.77 | 96.53 | 2 | degree |
| AIDS | 2000 | 15.69 | 16.20 | 2 | original | IMDB-MULTI | 1500 | 13.00 | 65.94 | 3 | degree |
| NCI1 | 4110 | 29.87 | 32.30 | 2 | original | | | | | | |

### A.2. Proof of Proposition 4.2

Assume the feature difference between graph $G$ and $G'$ is bounded with

$$||X' - X||_2^2 = ||E_X||_2^2 \leq \epsilon_X, \tag{12}$$

The difference between the original weights $\Theta$ and the weights trained on the new graph structure $\Theta'$ is represented as

$$||\Theta' - \Theta||_2^2 = ||(\mathcal{L}X')^{-1}Y' - (\mathcal{L}X)^{-1}Y||_2^2 \tag{13}$$

Given that $||X \cdot X'||_2^2 = ||X \cdot (X + E_X)||_2^2 \geq ||XE_X||_2^2$ . Let $||XE_X||_2^2 = \delta_X$ , then we can get $||X'^{-1} - X^{-1}||_2^2 = ||E_{X^{-1}}|| \leq \frac{\epsilon_X}{\delta_X}$.

Given the Bourgain theorem (You et al., 2019), the difference between the embedding $Y$ and $Y'$ is bounded with

$$||\mathcal{Y}' - \mathcal{Y}||_2^2 = ||E_Y||_2^2 \leq \epsilon_Y, \tag{14}$$

The weight difference can then be bounded with

$$\begin{aligned}
||\Theta' - \Theta||_2^2 &= ||(X'^{-1}\mathcal{L}^{-1}Y' - X^{-1}\mathcal{L}^{-1}Y||_2^2 \\
&= ||(X'^{-1}\mathcal{L}^{-1}(Y + E_Y) - X^{-1}\mathcal{L}^{-1}Y||_2^2 \\
&= ||(X'^{-1}\mathcal{L}^{-1}Y + X'^{-1}\mathcal{L}^{-1}E_Y - X^{-1}\mathcal{L}^{-1}Y||_2^2 \\
&= ||(X'^{-1}\mathcal{L}^{-1}Y - X^{-1}\mathcal{L}^{-1}Y + X'^{-1}\mathcal{L}^{-1}E_Y||_2^2 \\
&= ||(X'^{-1} - X^{-1})\mathcal{L}^{-1}Y + X'^{-1}\mathcal{L}^{-1}E_Y||_2^2 \\
&= ||E_{X^{-1}}\mathcal{L}^{-1}Y + (X + E_X)^{-1}\mathcal{L}^{-1}E_Y||_2^2 \\
&= ||E_{X^{-1}}\mathcal{L}^{-1}Y + (\mathcal{L}X + \mathcal{L}E_X)^{-1}E_Y||_2^2 \\
&\leq \frac{\epsilon_X}{\delta_X}||\mathcal{L}^{-1}Y||_2^2 + \frac{\epsilon_X^2\epsilon_Y}{\delta_X}||(\mathcal{L}X)^{-1}||_2^2 \\
&\quad + \epsilon_X\epsilon_Y||(\mathcal{L}X)^{-1}||_2^4
\end{aligned} \tag{15}$$

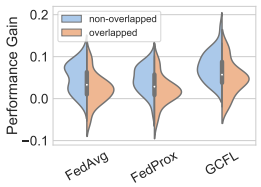
With the trained SGC, the feature and graph structure is fixed as $X$ and $\mathcal{L}$. Thus the weight difference is bounded.

## B. More Detailed Experiment Results

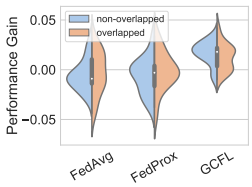
**Violin plots instead of tables** Figures 2 and 3 show the detailed experiment results regarding more various client settings.In Figure 2 and 3, each violin represents a distribution of all clients' performance gain using one algorithm. In Figures 2 and 3, the blue left sides of violins are corresponding to the results in the main tables 3.

**Overlapping versus non-overlapping** For distributing one dataset to multiple clients, we compare the two settings of allowing overlapping (same graphs appearing multiple clients) and not. As can be seen in Figure 2, our framework can also improve on overlapped clients. Although in Figure 2a, overlapped clients show less performance gains.

**Original node features versus other features** Apart from the original node features, we also use one-hot node degree features and extra continuous node attributes, in order to study the influence of node features. Figure (3a, 3b) and (3c, 3d) show the comparisons between original features and one-hot degree features, and between original features and extra node attributes, respectively. Overall, our framework can consistently improve when using one-hot degree features and continuous node attributes.

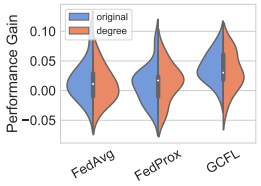



(a) oneDS: PROTEINS

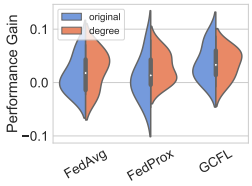



(b) oneDS: IMDB-BINARY

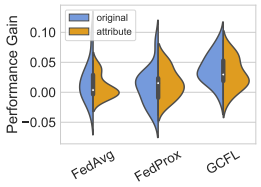
*Figure 2.* Distributions of performance gains of all clients with overlapped versus non-overlapped data partitioning.



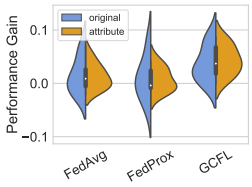

(a) multiDSs: MIX




(b) multiDSs: MOLECULES




(c) multiDSs: MIX




(d) multiDSs: MOLECULES

*Figure 3.* Distributions of performance gains of all clients using different node features.