OpenFGL: A Comprehensive Benchmark for Federated Graph Learning

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

Yinlin Zhu
Sun Yat-sen University
zhuylin27@mail2.sysu.edu.cn

Boyang Pang
Beijing Institute of Technology
1275854839@qq.com

Guochen Yan
Peking University
guochen_yan@outlook.com

Yeyu Yan
Beijing Jiaotong University
yanyeyu-work@foxmail.com

Zening Li
Beijing Institute of Technology
zening-li@outlook.com

Zhengyu Wu
Beijing Institute of Technology
Jeremywzy96@outlook.com

Wentao Zhang
Peking University
wentao.zhang@pku.edu.cn

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

Guoren Wang
Beijing Institute of Technology
wanggrbit@gmail.com

ABSTRACT

Federated graph learning (FGL) is a promising distributed training paradigm for graph neural networks across multiple local systems without direct data sharing. This approach inherently involves large-scale distributed graph processing, which closely aligns with the challenges and research focuses of graph-based data systems. Despite the proliferation of FGL, the diverse motivations from real-world applications, spanning various research backgrounds and settings, pose a significant challenge to fair evaluation. To fill this gap, we propose OpenFGL, a unified benchmark designed for the primary FGL scenarios: Graph-FL and Subgraph-FL. Specifically, OpenFGL includes 42 graph datasets from 18 application domains, 8 federated data simulation strategies that emphasize different graph properties, and 5 graph-based downstream tasks. Additionally, it offers 18 recently proposed SOTA FGL algorithms through a user-friendly API, enabling a thorough comparison and comprehensive evaluation of their effectiveness, robustness, and efficiency. Our empirical results demonstrate the capabilities of FGL while also highlighting its potential limitations, providing valuable insights for future research in this growing field, particularly in fostering greater interdisciplinary collaboration between FGL and data systems.

PVLDB Reference Format:
Xunkai Li, Yinlin Zhu, Boyang Pang, Guochen Yan, Zening Li, Yeyu Yan, Zhengyu Wu, Wentao Zhang, Rong-Hua Li, Guoren Wang. OpenFGL: A Comprehensive Benchmark for Federated Graph Learning. PVLDB, 18(5): 1305 - 1320, 2025. doi:10.14778/3718057.3718061

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

Proceedings of the VLDB Endowment, Vol. 18, No. 5 ISSN 2150-8097.
doi:10.14778/3718057.3718061

1 INTRODUCTION

Recently, graphs have emerged as effective tools for capturing intricate interactions among real-world entities, leading to their widespread applications. Based on this, we can translate various business applications from industrial scenarios into different graph-based downstream tasks from the machine learning perspective. To generate effective graph entity embeddings, graph neural networks (GNNs) utilize relational data stored from databases, encoding both node features and structural information for various data systems applications [9, 107, 146]. This paradigm has been widely validated, including node-level financial fraud detection [45, 94], link-level recommendation [65, 127], and graph-level bioinformatics [28, 96].

Despite their effectiveness, privacy regulations and scalability issues pose challenges to direct data sharing, complicating centralized model training [56, 134, 136]. To address this challenge, federated graph learning (FGL) has been proposed to enable collaborative training across multiple local systems [32, 90, 120, 132, 143], providing a novel distributed approach to graph-based data management [5, 78, 101, 128, 147]. Existing FGL benchmarks, such as FS-G [115] (Year: 2022) and FedGraphNN [38] (Year: 2021), offer valuable insights but still have the following limitations: (1) **Datasets**: Limited to few application domains (e.g., citation networks and recommendation). (2) **Algorithms**: Missing recent SOTA FGL methods (e.g., 8 methods in 2023, 10+ methods in 2024). (3) **Experiments**: Lack of graph-oriented federated data simulation strategies, inadequate support for various graph-based downstream tasks, and limited evaluation perspectives. While the research prospects and enthusiasm for FGL are prominent and growing [46, 74, 113, 118], the absence of a comprehensive benchmark for fair comparison impedes its development. Specifically, the diversity of downstream tasks (i.e., node, link, and graph), the unique graph properties (i.e., feature, label, and topology), and the complexity of FGL evaluation (i.e., effectiveness, robustness, and efficiency) collectively pose significant obstacles to achieving a comprehensive understanding of the current FGL landscape. Consequently, there is an emergency need to develop a standardized benchmark.

In this paper, we propose OpenFGL, which integrates 2 commonly used FGL scenarios, 42 datasets in 18 application domains, 8 graph-specific distributed data simulation strategies, 18 recently proposed SOTA algorithms, and 5 graph-based downstream tasks. These components are implemented with a unified API to facilitate fair comparisons and future development in a user-friendly manner. Based on this foundation, we provide a comprehensive evaluation of existing FGL algorithms, drawing the following valuable insights. For **Effectiveness** we advocate for quantifying the statistics in distributed graphs to define the graph-based federated heterogeneity formally. For **Robustness**, to facilitate the practical deployment of existing FGL algorithms, we emphasize the significant potential of personalized, multi-client collaboration, and privacy-preserving techniques in addressing challenges such as data noise, data sparsity, low client participation, and generalization in complex applications. For **Efficiency**, considering the industry-scale datasets, we encourage FGL developers to prioritize algorithmic scalability and propose innovative federated collaborative paradigms that bring substantial benefits in improving efficiency. To further illustrate the advantages of our proposed OpenFGL compared to existing FGL benchmarks, we provide a clear description in Table 1.

Our contributions. (1) *Comprehensive Benchmark*. We propose OpenFGL, which integrates 2 scenarios with 42 publicly datasets. Based on this, we propose 8 practical distributed settings from the perspective of data heterogeneity. Meanwhile, we integrate 18 FGL algorithms and advocate 3 orthogonal evaluation perspectives to establish comprehensive baselines (See Table 1). We believe that FGL can inspire new research directions within the data systems community, particularly in scalable, privacy-preserving, and distributed data processing. (2) *Valuable Insights*. Leveraging the user-friendly API integrated into OpenFGL, we conducted extensive empirical studies and derived 10 valuable conclusions (See Sec. 4.1-Sec. 4.4). Building upon these findings, we provide 6 key insights from the perspectives of effectiveness, robustness, and efficiency, outlining promising research directions for the future FGL community (See Sec. 5). (3) *Open-sourced Library and Detailed Repository*. We develop an easy-to-use and open-source library to support ongoing FGL studies, allowing users to evaluate their algorithms or datasets with ease. Additionally, we conduct a comprehensive review of existing FGL studies and release a beginner-friendly repository to facilitate the growth of the FGL community. The code and related tutorial are available at https://github.com/xkLi-Allen/OpenFGL.

2 PROBLEM STATEMENT

In this section, we briefly review the FGL training pipeline in the following 2 most representative scenarios. To begin with, from a data perspective, each client regards graphs (Graph-FL) and nodes in a subgraph (Subgraph-FL) as data samples. Subsequently, FGL aims to achieve collaborative training based on these clients and a trusted server. Formally, we consider a FGL system consisting of K clients, where the k-th client manages a private dataset denoted as $\mathcal{D}^{(k)} = \{\mathcal{G}_i^{(k)}\}_{i=1}^{N_T}$. Here, N_T is the task-specific description, where N_T denotes the number of graph samples under Graph-FL, while $N_T = 1$ exists under Subgraph-FL. To provide a detailed description, we take FedAvg [85] as an example. Its training process within the T communication round is outlined in four key steps:

1. Receive Message. Each client initializes its local model with the unified parameters from the server at the t-th round $\mathbf{W}_k^T \leftarrow \widehat{\mathbf{W}}^T$;
2. Local Update. Each client performs local training using its private data, i.e., $\min_{\mathbf{W}_k^T} \mathcal{L}_{task}(\mathcal{D}^{(k)})$ to obtain $\mathbf{W}_k^{T+1}$, where $\mathcal{L}_{task}$ denotes the task-specific optimization objectives.
3. Upload Message. Each client uploads their local updated models $\mathbf{W}_k^{T+1}$ and the number of data samples D_i (i.e., graphs, nodes, or edges, depending on the downstream task) to the server.
4. Global Aggregation. The server aggregates the updated models to obtain $\widehat{\mathbf{W}}^{T+1}$ for the next communication, i.e., $\widehat{\mathbf{W}}^{T+1} \leftarrow \frac{1}{D}\sum_{k=1}^{K} D_k \mathbf{W}_k^{T+1}$, where D is the total number of data samples.

3 BENCHMARK DESIGN

3.1 Data-level FGL Scenarios

In this section, we distinguish FGL scenarios based on the types of downstream tasks and the storage forms of local data at each client, categorizing them as Graph-FL, Subgraph-FL, and Node-FL. This serves as the guideline for proposing the concept of data-level FGL scenarios, emphasizing the data-centric description of real-world FGL applications. Notably, this section focuses on the pre-experiment preparation from a data perspective, whereas Sec. 3.3 emphasizes a more in-depth empirical analysis through a comprehensive evaluation across 3 orthogonal perspectives. In OpenFGL, we focus on the two prevalent FGL scenarios: (1) Graph-FL. The growing integration with graph-based techniques and AI4Science applications, such as drug discovery, has motivated this scenario, in which clients consider graphs as the data samples and pursue collaborative training between clients to acquire powerful models while preserving data privacy. (2) Subgraph-FL. Realistic applications in this scenario include node-level fraud detection for financial security and link-level user-item interactions for recommendation, with data stored in a distributed manner. Clients treat their data as subgraphs of a larger and more comprehensive global graph and focus on utilizing nodes and edges as data samples for training. Due to regulatory constraints, clients seek a collaborative training scheme to develop well-trained models without direct data sharing.

Notably, Node-FL is also a significant paradigm of FGL, which is widely used in graph-based spatial-temporal analysis, such as sensor networks [102] and traffic flow prediction [135], where nodes are only aware of their local context within the broader network. Although Node-FL has been widely mentioned, we have not integrated it into OpenFGL. This is because most Node-FL studies are tailored to specific scenarios and involve experimental setups that are highly diverse and closely aligned with particular application contexts. These characteristics make Node-FL less suitable for being included in a unified benchmark evaluation, where consistency across scenarios is essential for comprehensive comparisons.

Datasets. To comprehensively evaluate existing FGL algorithms, we have compiled a substantial collection of public datasets from various domains. Specifically, Regarding Graph-FL scenario, we conduct experiments on the compounds networks (MUTAG, BZR, COX2, DHFR, PTC-MR, AIDS, NCI1, hERG, ogbg-molhiv, ogbg-molpcba) [18, 29, 40, 42, 97, 104, 109], protein networks (ENZYMES, DD, PROTEINS, ogbg-ppa) [12, 20, 40, 42], collaboration network (COLLAB) [53], movie network (IMDB-B/M) [126], super-pixel

Table 1: FGL benchmark comparison, where D.D. denotes dataset domain. Effe. and Effi. represent effectiveness and efficiency.

FGL Benchmarks	D.D.	FGL Algorithms	Federated Data Simulation	Tasks	Evaluations	Conclusions
FS-G [115]	7	3 (Year: 2021)	Label, Topology	3	Effe. and Effi.	5 (Effe.+Hyperparameter)
FedGraphNN [38]	6	0 (FL+GNNs)	LDA-based Feature, Label	3	Effe. and Effi.	5 (Effe.+Effi.+Security)
OpenFGL (Ours)	18	18 (Year: 2021-2024)	Cross-domain and Graph-based Feature, Label, Topology	5	Table 2	10 (Effe. + Robustness + Effi.) and 6 Promising Directions

Table 2: An overview of OpenFGL.

Data	***Graph-FL Scenario***	***Subgraph-FL Scenario***
Datasets	Protein, Collaboration, Movie Network...	Citation, Purchase, Wiki, Syntax Network...
Simulation	Feature, Label, Topology, Cross-domain	Feature, Label, Community, Cross-domain
Tasks	Graph Regression, Graph Classification	Node Classification, Link Prediction, Node Clustering
Method	***Algorithms***	
GNN	GCN, GAT, GraphSAGE, SGC, GCNII, GIN, TopKPooling, SAGPooling, EdgePooling, PANPooling	
FL	FedAvg, FedProx, Scaffold, FedDC, MOON, FedProto, FedNH, FedTGP	
FGL	GCFL+, FedStar, FedSage+, Fed-PUB, FedGTA, FGSSL, FedGL, AdaFGL, FGGP, FedDEP, FedTAD	
Experiment	***Evaluations***	
Data Analysis	Feature KL Divergence, Label Distribution, Topology Statics	
Effectiveness	MSE, RMSE, Accuracy, Precision, Recall, F1, AUC-ROC, AP, Clustering-accuracy, NMI, ARI	
Robustness	Noise, Sparsity, Client Active Fraction, Federated Scenario Generalization, DP-based Privacy Preserve	
Efficiency	Convergence, Scalability, Communication, FLOPS, Time&Space Complexity	

networks (MNISTSuperPixels) [87], point cloud (ShapeNet) [133], and syntax trees (ogbg-code2) [42]. As for Subgraph-FL scenario, we perform experiments on the citation networks (Cora, Citeseer, PubMed, FedDBLP, ogbn-arxiv) [42, 115, 131], co-purchase networks (Amazon-Computers, Amazon-Photo, ogbn-products) [42, 100], and co-author networks (CS, Physics) [100], wiki-page networks (Chameleon, Squirrel) [91, 93], actor network (Actor) [91], game synthetic network (Minesweeper) [93], crowd-sourcing network (Tolokers) [93], syntax network (Roman-empire) [93], rating network (Amazon-rating) [93], social network (Questions) [93], and point cloud networks (PCPNet, S3DIS) [6, 34].

Remarkably, besides these datasets being collected across various application domains, they exhibit diverse graph characteristics, encompassing rich or poor node attributes at the feature level, homophily or heterophily, and sparsity or density at the topology level. These graph properties facilitate the evaluation of the adaptability and robustness of existing FGL algorithms across various and intricate experimental settings, highlighting their strengths and revealing potential limitations from a data-centric perspective. More details can be found in Tables. 3, 4 and [1] (A.1).

Simulation Strategies. In response to policy constraints on acquiring distributed graphs, we draw inspiration from federated learning in computer vision to simulate generalized federated scenarios by partitioning existing datasets into distributed subsets. This strategy is similar to recent FGL benchmarks [38, 115]. In our proposed OpenFGL, we integrate 8 federated data simulation strategies driven by practical applications, in which we ensure the graph data distributed to each client exhibits similar patterns in feature, label, or topology while maintaining a controllable heterogeneity across clients. Specifically, for both Graph-FL and Subgraph-FL, we implement 3 simulation strategies widely used in graph-independent FL. In the context of Graph-FL, we introduce a topology-oriented simulation strategy called Topology Shift, which distributes graphs based on degree distribution. The inspiration for this approach stems from key insights offered by recent FGL studies [105, 122]: in Graph-FL, the structure Non-iid resulting from topology shift is the primary challenge in collaborative optimization, where node features and labels appear less significant by comparison. As for Subgraph-FL, existing strategies predominantly utilize community detection algorithms such as Louvain [11] and Metis [49] to identify clusters with dense intra-community connections. Then, these clusters' nodes are subsequently allocated to local clients to construct corresponding induced subgraphs for partitioning. These strategies all operate under a common assumption: that the private data collected by each local agent in the real world contains dense internal connections but is loosely connected across clients [41, 47, 58, 62]. Despite their effectiveness, limitations persist in their application. This is because Subgraph-FL primarily focuses on node-level and link-level tasks, where node profiles (i.e., node features and labels) are crucial. However, these strategies do not consider label distribution for federated data simulation. Hence, we introduce Metis-based Label Imbalance Split and Louvain-based Label Imbalance Split. These methods, refining the aforementioned strategies, carefully consider label distribution during cluster allocation to better simulate realistic and generalized distributed scenarios. We outline these strategies in Table 5, with descriptions provided below:

Table 3: The statistical information of Graph-FL datasets.

Graph-FL	Graphs	Nodes	Edges	Features	Classes	Train/Val/Test	Description
MUTAG	188	17.93	19.79	7	2	80%/10%/10%	Compounds Network
BZR	405	35.75	38.36	56	2	80%/10%/10%	Compounds Network
COX2	467	41.22	43.45	38	2	80%/10%/10%	Compounds Network
DHFR	467	42.43	44.54	56	2	80%/10%/10%	Compounds Network
PTC-MR	344	14.29	14.69	18	2	80%/10%/10%	Compounds Network
AIDS	2,000	15.69	16.20	42	2	80%/10%/10%	Compounds Network
NCI1	4,110	29.87	32.30	37	2	80%/10%/10%	Compounds Network
hERG	10,572	29.39	94.09	8	-	80%/10%/10%	Compounds Network
ogbg-molhiv	41,127	25.50	27.50	9	2	80%/10%/10%	Compounds Network
ogbg-molpcba	437,929	26.00	28.10	9	2	80%/10%/10%	Compounds Network
ENZYMES	600	32.63	62.14	21	6	80%/10%/10%	Protein Network
DD	1,178	284.32	715.66	89	2	80%/10%/10%	Protein Network
PROTEINS	1,113	39.06	72.82	4	2	80%/10%/10%	Protein Network
ogbg-ppa	158,100	243.40	2,266.10	4	37	49%/29%/22%	Protein Network
COLLAB	5,000	74.49	2,457.78	degree	3	80%/10%/10%	Collaboration Network
IMDB-BINARY	1,000	19.77	96.53	degree	2	80%/10%/10%	Movie Network
IMDB-MULTI	1,500	13.00	65.94	degree	3	80%/10%/10%	Movie Network
ShapeNet	16,881	2616.20	KNN	3	50	40%/40%/40%	Point Cloud Network
MNISTSuperPixels	70,000	75.00	1393.03	1	10	43%/43%/14%	Super-pixel Network
ogbg-code2	452,741	125.20	124.20	4	275	90%/5%/5%	Abstract Syntax Trees

Table 4: The statistical information of Subgraph-FL datasets.

Subgraph-FL	Nodes	Edges	Features	Classes	Train/Val/Test	Description
Cora	2,708	5,429	1,433	7	20%/40%/40%	Citation Network
CiteSeer	3,327	4,732	3,703	6	20%/40%/40%	Citation Network
PubMed	19,717	44,338	500	3	20%/40%/40%	Citation Network
FedDBLP	52,202	271,054	1,639	4	50%/20%/30%	Citation Network
ogb-arxiv	169,343	231,559	128	40	60%/20%/20%	Citation Network
Amazon-Photo	7,487	119,043	745	8	20%/40%/40%	Co-purchase Network
Amazon-Computers	13,381	245,778	767	10	20%/40%/40%	Co-purchase Network
ogb-products	2,449,029	61,859,140	100	47	10%/5%/85%	Co-purchase Network
Co-author CS	18,333	81,894	6,805	15	20%/40%/40%	Co-author Network
Co-author Physics	34,493	247,962	8,415	5	20%/40%/40%	Co-author Network
Chameleon	2,277	36,101	2,325	5	48%/32%/20%	Wiki-page Network
Chameleon Filter	890	13,584	2,325	5	48%/32%/20%	Wiki-page Network
Squirrel	5,201	216,933	2,089	5	48%/32%/20%	Wiki-page Network
Squirrel Filter	2,223	65,718	2,089	5	48%/32%/20%	Wiki-page Network
Actor	7,600	29,926	931	5	50%/25%/25%	Actor Network
Minesweeper	10,000	39,402	7	2	50%/25%/25%	Game Synthetic Network
Tolokers	11,758	519,000	10	2	50%/25%/25%	Crowd-sourcing Network
Roman-empire	22,662	32,927	300	18	50%/25%/25%	Article Syntax Network
Amazon-ratings	24,492	93,050	300	5	50%/25%/25%	Rating Network
Questions	48,921	153,540	301	2	50%/25%/25%	Social Network
PCPNet	100,000	KNN	5	-	26%/10%/64%	Point Cloud Network
S3DIS	4,096	KNN	6	3	45%/45%/10%	Point Cloud Network

Feature Distribution Skew is a graph-independent strategy to simulate feature distribution shifts [46, 113]. In OpenFGL, we utilize this approach to create more challenging and realistic scenarios for evaluating FGL algorithms [60]. Specifically, we implement various feature operations: (1) Adding Gaussian or Laplacian noise to introduce variability; (2) Applying scaling operations to simulate different magnitudes of features; (3) Employing mathematical transformations to further diversify the feature distributions.

Label Distribution Skew is a graph-independent strategy [118, 132]. In our implementation, we use the α-based Dirichlet distribution to create imbalanced label distributions across clients [10]. This approach ensures varied and imbalanced label distributions, simulating real-world data scenarios. The α in Dirichlet distribution controls the concentration of probabilities across label classes, with larger values leading to more uniform distributions and smaller values resulting in sparser and more concentrated distributions.

Table 5: A summary of our proposed graph-specific data simulation strategies.

Federated Simulation	Scenarios	Feature	Label	Topology	Implemented By
Feature Distribution Skew	Both	✓	✗	✗	FS-G, OpenFGL
Label Distribution Skew	Both	✗	✓	✗	FS-G, FedGraphNN, OpenFGL
Cross Domain Data Skew	Both	✓	✓	✗	FS-G, OpenFGL
Topology Distribution Skew	Graph-FL	✗	✗	✓	OpenFGL
Metis-based Community Split	Subgraph-FL	✗	✗	✓	FS-G, OpenFGL
Louvain-based Community Split	Subgraph-FL	✗	✗	✓	FS-G, OpenFGL
Metis-based Label Imbalance Split	Subgraph-FL	✗	✓	✓	OpenFGL
Louvain-based Label Imbalance Split	Subgraph-FL	✗	✓	✓	OpenFGL

Cross Domain Data Skew is a fundamental challenge in FL, arising from the heterogeneous nature of data sources and collection methods across distributed databases [128]. In OpenFGL, we simulate this scenario by evenly distributing multiple datasets among a predefined number of clients, maintaining a diverse representation.

Topology Distribution Skew represents the strategy for partitioning graphs based on their topology properties in Graph-FL [105, 122]. This approach involves sorting the global graph according to specific characteristics, such as average node degree, and then distributing the resulting graphs to predefined clients. This ensures that the distribution of graph data among clients accurately reflects the underlying topological diversity of the original global dataset.

Metis-based Community Split is a widely adopted Subgraph-FL federated data simulation strategy that utilizes a multilevel recursive bisection and k-way partitioning technique. This method iteratively reduces the size of the graph and refines the partitioning. Notably, compared to the following the Louvain-based data simulation strategy, Metis can directly partition a graph into a predefined number of communities, aligning precisely with the number of clients and streamlining data allocation in the federated settings.

Louvain-based Community Split stands as the other prevalent federated data simulation strategy in Subgraph-FL, partitioning a graph into multiple communities (subgraphs) via modularity optimization. The number of communities is determined by the resolution parameter of the Louvain algorithm. However, the Louvain algorithm often generates more communities than the predefined number of clients. To resolve this, communities can be allocated among the clients by averaging node quantities.

Metis-based Label Imbalance Split is a new Subgraph-FL data simulation strategy introduced in this paper. The naive Metis-based Community Split lacks post-processing capabilities, leading to challenges in controlling subgraph heterogeneity among clients. In contrast, our approach enables predefined community partitioning, followed by clustering based on label distribution similarity, thereby consolidating similar communities under a single client.

Louvain-based Label Imbalance Split enhances the conventional Louvain method by allocating communities to clients based on similarities in label distributions rather than solely on node averages. This approach ensures that each client receives communities with consistent label characteristics, thereby mitigating label imbalance and promoting equitable model training across federated clients. By aligning label distributions, this strategy enhances the fairness and robustness of federated learning, reducing biases that may arise from heterogeneous label distributions among clients.

These 8 federated distributed data simulation strategies, meticulously developed based on the combination of features, labels, and topology, significantly enhance the robustness and generalization capabilities of FGL studies. We have crafted a comprehensive and realistic benchmark tailored for industrial applications, thereby fostering substantial progress and paving the way for future advancements in FGL research. Therefore, OpenFGL not only tests the effectiveness of existing FGL algorithms but also serves as a platform for developing new methodologies and approaches.

Downstream Tasks. Our proposed OpenFGL evaluates FGL studies across a range of downstream tasks, including graph classification and regression for Graph-FL, as well as node classification, clustering, and link prediction for Subgraph-FL. While acknowledging the traditional focus on graph and node classification, OpenFGL extends its scope to additional tasks, promoting broader advancements and greater flexibility in the FGL benchmark. Notably, to avoid complex presentation and ensure reader-friendly, we primarily focus on classification task to present experimental results.

3.2 Method-level FGL Algorithms

GNN Backbones. To provide a broader spectrum of learning paradigms on the client side, OpenFGL integrates a diverse range of local GNN backbones. Specifically, we implement various well-designed polling strategies (TopKPooling [26], SAGPooling [52], EdgePooling [19], and PANPooling [84]) based on the most representative GIN [124] with weight-free MeanPooling [124] for Graph-FL. As for Subgraph-FL, OpenFGL includes prevalent GCN [51], GAT [108], GraphSAGE [36], SGC [117] and GCNII [15]. The detailed descriptions of these backbones can be found in [1] (A.2).

FL/FGL Algorithms. To achieve federated training, multi-client collaboration algorithms are crucial (CV-based FL also can be applied to FGL). We follow the historical progression of FL to include a spectrum of algorithms from the most representative methods in CV to FGL: (1) CV-based FL algorithms: FedAvg [85], FedProx [63], Scaffold [48], MOON [61], FedDC [27], FedProto [106], FedNH [16] and FedTGP [137]; (2) all FGL algorithms possible: GCFL+ [122] and FedStar [105] in Graph-FL and FedSage+ [140], Fed-PUB [8], FedGTA [70], FGSSL [44], FedGL [13], AdaFGL [67], FGGP [110], FedDEP [138], and FedTAD [149] in Subgraph-FL. More details about these algorithms can be found in [1] (A.2). Notably, these algorithms are implemented with a unified API to facilitate future development in a user-friendly manner. For more details about our API design from the algorithm perspective, please refer to Sec. 4.5.

3.3 Experiment-level FGL Evaluations

Data Analysis. (1) Feature KL Divergence: It reveals feature skew among clients while the label domain remains consistent, which may arise due to the different geographical locations of clients, such as the characteristics of a certain disease may significantly differ across various regions. (2) Label Distribution: It is widely discussed in CV-based FL. However, in FGL, the relationship between labels and topology frequently reveals underlying connections, characterized as homophily. Thus, we further integrate multi-level homophily metrics [76, 80, 91–93] to offer comprehensive analysis. (3) Topology Statics: This perspective stems from the critical role of topology in GNNs, especially in distributed scenarios. This arises from the significant impact of diverse local topology statistics on the local model, causing complex cascading effects in collaboration. Therefore, we examine topological differences among clients (e.g., Degree, Centrality, Largest Component) to provide insights.

Effectiveness. Details of our evaluation metrics are as follows: graph/node classification (Accuracy, F1, Recall, Precision), graph regression (MSE, RMSE), link prediction (AP, AUC-ROC), and node clustering (Clustering-accuracy, NMI, ARI). More detailed descriptions of these metrics are presented in [1] (A.3).

Robustness. To evaluate the practical deployment of FGL, we examine its robustness from the following perspectives: (1) **Noise**: This corresponds to data quality issues resulting from data collection [58, 78]. (2) **Sparsity**: This reflects scenarios with incomplete features, labels, and topology due to data scarcity and high labor costs, and with a low rate of client participation [32, 56, 118]. (3) **Client Communication**: This simulates scenarios with network constraints or high communication costs [132, 134]. (4) **Generalization**: This evaluates the effectiveness of algorithms in various scenarios. (5) **Privacy Preserve**: This reflects the applicability of algorithms in privacy-sensitive scenarios, thereby we conduct an in-depth analysis from the perspective of Differential Privacy (DP). Please refer to [1] (A.4) for further details on the robustness settings.

Efficiency. To facilitate FGL deployment, we conduct an evaluation of current baselines regarding their efficiency. Specifically, we evaluate them from both theoretical (algorithm complexity) and experimental (communication cost and running time) perspectives.

4 EXPERIMENTS AND ANALYSIS

In this section, we systematically investigate FGL algorithms by answering the following questions: (1) For **Effectiveness**, **Q1**: What advantages does federated collaboration offer compared to training solely on local data? **Q2**: How do FGL algorithms and federated implementations of GNNs perform in two prevalent FGL scenarios? (2) For **Robustness**, **Q3**: How do FGL algorithms perform under local noise and sparsity settings (i.e., features, labels, and edges)? **Q4**: What are the performance of FGL algorithms under low client participation rates in communication? **Q5**: Can FGL algorithms maintain generalization across various graph-specific distributed scenarios? **Q6**: Do FGL algorithms support additional DP privacy protection? (3) For **Efficiency**, **Q7**: What are the theoretical algorithm complexity of FGL algorithms? **Q8**: What is the practical running efficiency of FGL algorithms? To maximize the usage for the constraint space, more results are shown in [1] (A.5-A.7).

4.1 Performance Comparison

To answer **Q1**, in addition to the federated multi-client collaboration, we introduce "Local" to represent solely local training for analyzing the advantages and potential limitations of FGL. Based on this, to answer **Q2**, we present the end-to-end performance in Table 6 and Table 7. The detailed analysis is presented as follows:

Graph-FL Scenario. Since the baselines for Graph-FL are scarce. Therefore, we expand pooling-based backbones and CV-based FL in Table 6. For **Q1**, we find that the benefits of federated collaboration are not significant for small-scale MUTAG, BZR, and COX2 due to limited data that can not support federated training and thus affect predictions. Subsequently, we conclude that *C1: Federated collaboration is more advantageous for larger-scale datasets, benefiting from abundant data sources [64, 98].* As for **Q2**, we observe that: (1) GCFL+ and FedStar concentrate on topology Non-iid within the cross-domain simulation, thereby not consistently achieving competitive performance in this single-source setting. Therefore, they perform less favorably than FL algorithms on ENZYMES, COLLAB, and MULTI. (2) Existing FGL algorithms heavily rely on node semantics. We observe significant improvements on datasets with abundant node descriptions like DD and PROTEINS, whereas limited performance on BINARY whose node representations are initialized with node degrees. Consequently, we conclude that *C2: Graph-FL algorithms still have improvement space, especially in the single-source domain and constrained data semantics [3, 57].*

Subgraph-FL Scenario. Experimental results are shown in Table 7. For **Q1**, although Subgraph-FL can benefit from abundant data, for Chameleon, Actor, and Ratings, the improvement from both FL and FGL is limited or even worse than solely local training in some cases. We attribute this to the heterophily, where differences in node connection rules across clients significantly affect local updates and server-side collaboration, deviating from the global optimum and resulting in sub-optimal performance. Although AdaFGL addresses this issue through personalized technology, there is still room for better performance. Based on this and *C1*, we conclude that *C3: The prerequisite for positive impacts of federated collaboration is the uniform distribution of node features, labels, and topology [79, 121].* Regarding **Q2**, we observe that: (1) Subgraph-FL is thriving, with numerous baselines vigorously competing for the best performance. Among them, FedTAD and AdaFGL stand out in most cases. (2) The outstanding performance of existing algorithms stems from the fine-grained exploration of the topology, but some methods lack scalability when dealing with large-scale ogbn-products, resulting in OOM (out-of-memory) errors. Consequently, we conclude that *C4: Subgraph-FL algorithms need to resolve the complexity in real-world deployments, especially focusing on large-scale scenarios and graph-specific federated heterogeneity challenges [68, 71].*

4.2 Robustness Analysis

Local Noise. To answer **Q3** from the perspective of noise, the experimental results are shown in the upper part of Fig. 1(a)-(c). For feature noise, we randomly select the channels of node features and inject Gaussian noise. To simulate edge noise in the Graph-FL without node labels, we utilize Metattack [150] to add noise edges, significantly perturbing the training gradients. As for label noise, we randomly reassign non-real labels to a certain proportion

Table 6: Graph-FL test accuracy (%). The best result is bold. The second result is underlined.

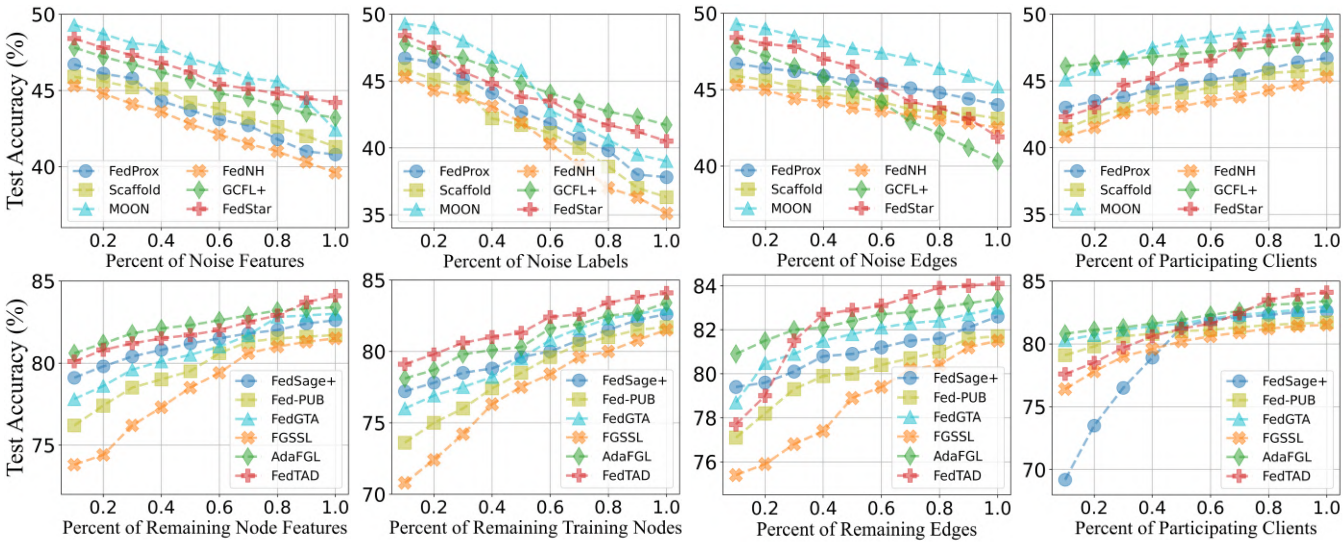
Graph-FL	MUTAG	BZR	COX2	ENZYMES	DD	PROTEINS	COLLAB	BINARY	MULTI
GIN-Local	84.2±2.5	84.3±1.6	81.6±2.9	40.7±1.1	82.7±2.0	81.8±1.1	75.4±1.9	76.3±2.8	47.1±1.8
SAG-Local	82.0±1.9	**89.5±1.9**	82.1±2.3	41.4±0.8	80.3±1.5	83.3±1.4	77.0±1.5	77.7±2.5	49.0±2.1
Edge-Local	80.8±1.9	86.7±1.8	78.6±2.4	42.1±0.7	81.5±2.4	82.7±0.9	76.2±1.7	78.9±1.9	48.0±2.0
PAN-Local	**86.1±2.7**	80.3±2.0	80.8±2.4	38.0±1.0	84.2±2.2	81.0±1.9	75.1±2.0	80.7±2.3	46.1±2.4
FedAvg	78.9±2.9	76.5±1.3	79.0±1.7	47.4±0.9	82.4±2.6	80.1±1.5	77.5±1.6	79.2±2.5	50.8±2.0
FedProx	76.5±2.4	81.8±1.7	77.2±1.6	46.7±1.4	83.1±1.5	77.4±1.7	77.9±1.9	81.9±2.0	51.6±2.2
Scaffold	75.4±2.9	82.3±1.8	82.0±1.4	40.9±1.5	84.5±2.4	79.9±1.1	76.4±1.8	80.3±1.7	**52.4±2.8**
MOON	80.5±2.9	82.6±1.8	78.8±1.5	**49.3±1.6**	79.8±2.1	80.0±2.0	**79.8±2.0**	81.1±2.0	51.4±2.2
FedProto	82.7±2.0	86.7±1.4	79.4±2.4	42.5±1.4	85.2±2.0	80.3±1.3	76.7±1.4	80.6±2.7	49.9±2.1
FedNH	84.3±2.2	85.2±1.6	81.2±2.4	45.3±1.5	84.9±2.0	81.2±1.8	75.3±1.2	79.4±2.0	50.4±2.6
FedTGP	83.8±2.8	84.6±1.0	81.8±1.6	43.0±1.0	87.3±2.8	80.9±2.0	77.2±2.1	78.6±2.5	50.9±2.5
GCFL+	82.6±2.6	87.8±1.9	**82.6±2.3**	47.8±1.3	85.2±2.5	83.6±1.3	77.5±1.1	80.4±2.3	51.8±2.5
FedStar	84.7±2.6	89.1±1.5	80.6±2.3	48.4±0.8	**88.4±2.3**	**84.5±1.7**	78.6±1.7	**82.7±2.3**	51.4±2.6

Table 7: Subgraph-FL test accuracy (%). The best result is bold. The second result is underlined.

Subgraph-FL	Cora	CiteSeer	PubMed	Photo	Computers	Products	Chameleon	Actor	Ratings
GCN-Local	77.9±0.3	64.3±0.8	84.6±0.3	88.8±0.4	87.7±0.6	79.4±0.5	64.7±0.6	29.4±1.4	45.5±0.8
GAT-Local	78.5±0.4	63.9±0.6	85.3±0.4	89.6±0.5	87.4±0.5	78.9±0.6	65.1±0.5	30.2±0.9	46.2±0.5
FedAvg	82.5±0.7	69.5±0.7	86.4±0.5	90.3±0.7	89.1±0.4	82.3±0.5	61.2±0.8	28.7±0.8	42.5±0.4
FedDC	81.4±0.8	70.4±0.5	87.9±0.4	91.2±0.6	88.4±0.5	81.9±0.3	58.6±1.2	26.9±1.2	41.2±0.4
FedProto	79.4±0.6	67.2±0.2	85.1±0.2	87.4±0.4	86.9±0.3	77.2±0.4	64.0±0.6	28.0±0.6	46.1±0.4
FedTGP	80.7±0.5	68.8±0.4	85.9±0.3	86.5±0.5	86.4±0.6	78.3±0.5	62.7±1.1	28.4±0.7	45.7±0.9
FedSage+	82.6±0.8	71.2±0.8	**88.2±0.7**	90.8±0.8	90.4±0.8	82.8±0.7	65.6±0.7	30.8±1.0	45.8±0.7
FedGTA	83.0±0.4	**72.4±0.4**	87.6±0.4	91.0±0.4	90.8±0.5	**83.2±0.4**	66.2±0.8	30.5±0.6	45.5±0.5
Fed-PUB	81.7±0.6	71.9±0.7	87.8±0.3	**91.5±0.6**	91.1±0.7	82.1±0.5	64.4±0.7	29.2±0.8	44.8±0.6
FGSSL	81.5±0.8	70.1±0.5	87.3±0.4	88.8±0.6	89.2±0.5	OOM	64.9±0.9	28.9±1.0	45.2±0.8
FedGL	82.5±0.7	71.5±0.7	87.1±0.5	89.7±0.5	90.7±0.5	OOM	65.4±0.8	29.4±1.3	46.0±0.7
AdaFGL	83.4±0.5	72.0±0.5	87.9±0.6	91.3±0.7	91.0±0.6	OOM	**67.2±1.0**	**31.2±1.2**	**46.9±0.5**
FGGP	81.4±0.5	69.1±0.7	87.0±0.4	88.6±0.6	88.3±0.4	OOM	65.3±0.7	28.3±0.4	45.6±0.5
FedTAD	**84.1±0.6**	71.8±0.8	88.0±0.6	91.4±0.5	**91.6±0.7**	81.9±0.3	65.9±1.3	30.7±1.0	46.1±0.8

of training node samples. Based on the experimental results, we observe that FGL algorithms are highly sensitive to edge noise compared to topology-agnostic FL algorithms. This inherent limitation directly disrupts the model optimization of GCFL+ and FedStar, misleading local updates and topology-driven collaboration, thereby resulting in sub-optimal predictive performance. However, GCFL+ and FedStar demonstrate superior robustness under feature and label noise, as they address client interference with each other using server-side clustering customization and additional local models maintained at client-side. Consequently, we can conclude that *C5: Noise scenarios determine the performance lower bound for FGL algorithms, where personalized strategies emerge as crucial technologies. However, they fall slightly short in addressing edge noise [77, 141].*

Local Sparsity. To answer **Q3** from the perspective of sparsity, the experimental results are shown in the lower part of Fig. 1(a)-(c). Regarding feature sparsity, we simulate partial feature absence for unlabeled nodes and randomly remove the original edges. As for label sparsity, we change the ratio of training nodes. Under feature sparsity, FGSSL and Fed-PUB exhibit significant performance fluctuations due to heavy reliance on high-quality features for node-wise contrastive learning and model-wise gradient matching. Conversely, FedSage+, AdaFGL, and FedTAD leverage multi-client collaboration, mitigating confusion in under-trained model collaboration, thus ensuring robustness. Similar analysis can extend to label sparsity, as both of them directly affect local updates. Regarding edge sparsity, FedTAD's vulnerability lies in its reliance on client-side



(a) Feature-level (b) Label-level (c) Edge-level (d) Round Communication

Figure 1: Robustness performance on Graph-FL ENZYMES (upper) and Subgraph-FL Cora (lower).

Table 8: Generalization performance (%). The best result is bold. The second result is underlined.

Graph-FL Simulation	DHFR Feature	DHFR Label (0.5)	DHFR Topology	DD Feature	DD Label (0.5)	DD Topology	COLLAB Feature	COLLAB Label (0.5)	COLLAB Topology	Multi-source Cross-domain
FedAvg	69.1±2.7	70.5±3.6	72.3±3.2	80.3±2.9	81.2±2.4	81.8±2.4	68.7±2.5	76.6±1.8	70.3±1.8	73.6±2.6
FedDC	70.3±3.0	68.2±4.2	71.6±3.5	78.6±4.2	83.9±3.2	80.7±3.0	66.9±3.9	78.2±2.2	69.8±2.7	71.3±3.2
FedProto	68.4±2.4	71.9±3.1	71.1±3.2	79.8±2.5	83.6±2.5	82.1±2.5	70.4±1.5	76.0±1.8	72.0±1.1	72.6±2.8
FedTGP	71.5±2.7	69.8±3.5	72.0±2.9	81.4±2.9	85.5±2.1	80.8±3.6	72.9±2.6	75.9±2.3	72.7±1.9	74.1±3.0
GCFL+	72.3±1.9	73.2±2.4	**74.5±2.4**	**83.1±2.2**	84.2±1.8	**83.9±2.7**	73.8±1.9	77.3±2.5	**75.4±1.5**	76.2±2.5
FedStar	**73.9±2.6**	**74.6±2.9**	73.4±3.1	82.5±2.9	**86.3±2.0**	82.2±3.2	**74.6±2.3**	**78.5±2.9**	73.8±2.6	**77.3±2.9**

Subgraph-FL Simulation	Photo Louvain	Photo Metis+	Photo Louvain+	Squirrel Louvain	Squirrel Metis+	Squirrel Louvain+	Questions Louvain	Questions Metis+	Questions Louvain+	Multi-source Cross-domain
FedGL	91.4±1.0	89.5±0.9	87.9±1.1	47.0±1.5	47.2±1.0	45.7±1.3	96.8±0.4	94.4±0.4	**93.5±0.3**	78.3±0.8
FedSage+	92.1±0.7	90.4±0.8	**88.9±1.0**	46.2±1.4	46.0±1.2	**46.4±1.1**	97.0±0.3	93.6±0.5	92.4±0.4	80.5±0.6
FGSSL	90.5±0.6	89.8±0.6	87.5±0.7	45.9±0.7	46.3±0.9	43.3±0.8	96.1±0.4	94.2±0.4	92.7±0.3	76.9±1.0
FedGTA	**92.4±0.4**	**91.0±0.5**	88.6±0.4	47.8±0.8	47.0±0.5	45.8±0.6	96.3±0.2	**95.0±0.3**	95.3±0.2	**82.4±0.4**
Fed-PUB	91.8±0.5	90.2±0.7	88.3±0.6	47.5±1.0	46.9±1.3	44.6±1.1	96.5±0.3	94.1±0.4	95.0±0.2	79.8±0.7
AdaFGL	91.7±0.8	90.9±0.7	88.2±0.8	**48.6±1.2**	**47.8±1.5**	46.0±0.9	**97.2±0.2**	94.7±0.3	95.2±0.3	81.6±0.9

topology embeddings to supervise server-sider pseudo-subgraph generator. However, AdaFGL and FedSage+, leveraging topology mining, adeptly handle this challenge. Based on this, we can conclude that: *C6: Sparsity scenarios determine the performance upper bound for FGL algorithms, where multi-client collaboration is the pivotal technology, particularly in synergy with topology mining [7, 119].*

Client Participation. To answer **Q4**, we present the experimental results in Fig. 1(d), where robust FGL algorithms with low client participation exhibit one of the following characteristics: (1) They rely less on messages received from the server and focus on local training; (2) They custom global messages for each participating client. For instance, in Graph-FL, the unstable performance of FedStar arises from its heavy reliance on specific topological properties. Conversely, GCFL+ ensures high-quality local updates by tailoring the most suitable messages for each client through server-side clustering. In Subgraph-FL, AdaFGL, Fed-PUB, and FedGTA rely on client-side personalized training, server-side pseudo-graph-driven clustering, or identification of subgraph statistics to ensure custom global messages for each participating client. Consequently, we can conclude that *C7: Low client participation underscores the emphasis of FGL algorithms on local updates, highlighting the importance of local data understanding and customizing messages for each client [23, 95].*

Table 9: Performance (%) on DP privacy preserve.

Graph-FL Simulation	AIDS Label	AIDS Topology	NCI1 Label	NCI1 Topology
FedAvg	94.2±0.7	93.6±1.1	82.7±0.6	79.5±0.4
ϵ(5)-DP	91.9±0.9	90.5±1.6	78.3±0.7	74.2±0.5
ϵ(20)-DP	93.5±0.6	92.8±0.9	80.4±0.8	76.8±0.5
GCFL	95.8±0.5	94.2±0.7	84.9±0.5	81.5±0.3
ϵ(5)-DP	92.3±0.7	90.8±0.9	81.1±0.8	75.1±0.4
ϵ(20)-DP	94.0±0.8	93.0±0.8	82.5±0.6	77.4±0.6
Subgraph-FL Simulation	**CS Louvain+**	**CS Metis+**	**Physics Louvain+**	**Physics Metis+**
FedAvg	83.2±0.9	84.0±0.7	91.6±0.3	91.0±0.4
ϵ(10)-DP	78.2±1.1	77.9±1.2	88.7±0.5	89.1±0.6
FedGTA	85.0±0.7	85.3±0.5	92.8±0.4	91.9±0.3
ϵ(10)-DP	78.8±0.8	79.2±0.9	89.4±0.7	89.2±0.8
FedSage+	84.2±1.1	85.6±0.7	92.1±0.9	91.4±0.5
FedDEP	84.5±0.9	84.9±0.6	91.7±1.1	91.7±0.4

Generalization. To answer **Q5**, the experimental results are shown in Table 8, where graph-level cross-domain setting includes MUTAG, COX2, PTC-MR, AIDS, ENZYMES, DD, PROTEINS, COLLAB, and IMDB-B/M and subgraph-level setting includes Cora, CiteSeer, Pubmed, CS, and Physics. We observe that current FGL algorithms exhibit inconsistent performance across data simulations. Specifically, in Graph-FL, tiny-scale datasets with abundant node descriptions such as DHFR and DD mitigate inherent differences among data simulations potentially due to over-fitting issues. As for the Subgraph-FL, data simulations present challenges for existing FGL algorithms. For instance, Fed-PUB and AdaFGL, incorporating personalized strategies, lose their previous advantages, whereas FedSage+ and FedGTA, emphasizing multi-client collaboration mechanisms, show significant potential. Therefore, we can conclude that *C8: In practical deployments aiming for generalization, client-specific design should be used cautiously, with an emphasis on discovering inherent global consensus [35, 88].*

DP-based Privacy Preserve. To answer **Q6**, the experimental results are shown in Table 9, where ϵ is the privacy budget. To implement DP in FGL, we introduce well-selected random noise in the local model gradients for server-side perturbed model aggregation. More technology details can be found in [1] (A.8). Meanwhile, we integrate FedDEP into OpenFGL, which introduces additional edge-level DP. Based on the results, we observe that a large ϵ enables privacy-preserving methods to match the accuracy of the non-private approach. However, reducing ϵ results in a notable performance drop, primarily due to the need for local models for excessive noise injection into gradients, leading to significant degradation. Regarding FedDEP, compared to FedSage+, it achieves edge-level differential privacy through random sampling and enhances model capacity with a deep neighbor generation module, striking a balance between performance and privacy preservation. Based on this, we conclude that *C9: FGL algorithms currently face a dilemma between predictive performance and privacy preservation [22, 144].*

4.3 FGL Algorithm Complexity Analysis

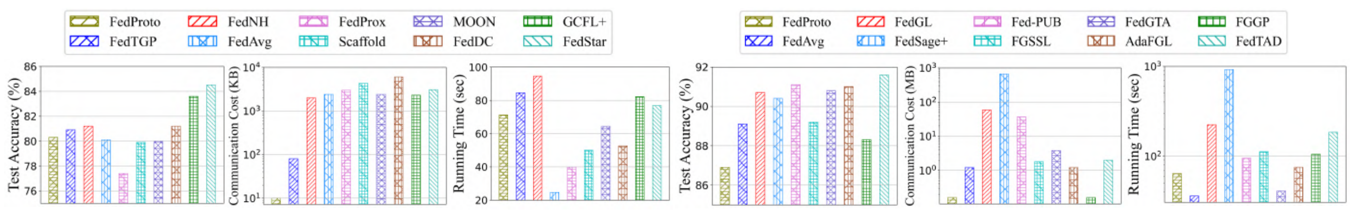
To answer **Q7**, we provide a theoretical algorithm complexity analysis of the prevalent FL and FGL baselines, as illustrated in Table 10, where n, m, c, and f are the number of nodes, edges, classes, and feature dimensions, respectively. s is the number of selected augmented nodes and g is the number of generated neighbors. b and T are the batch size and training rounds, respectively. k and K correspond to the number of times we aggregate features and moments order, respectively. N is the number of participating clients in each training round. t represents the number of clients exchanging information with the current client. ω represents the model-wise weight alignment loss term, Q denotes the size of the query set used for CL, E stands for the number of models for ensemble learning, M and p indicate the dimension of the trainable matrix used to mask trainable weights and the prototypes. Besides, P_g represents pseudo-graph data stored on the server side.

For convenience, we choose SGC [117] as the local model (k-step feature propagation), otherwise, we adopt the model architecture (L-layer) used in their original paper. For the k-layer SGC model with batch size b, the $\mathbf{X}^{(k)}$ is the propagated feature bounded by a space complexity of $O((b+k)f)$. The overhead for linear regression by multiplying $\mathbf{W}$ is $O(f^2)$. In the training stage, the above procedure is repeated to iteratively update the model weights. For the server performing FedAvg, it needs to receive the model weights and the number of samples participating in this round. Its space complexity and time complexity are bounded as $O(Nf^2)$ and $O(N)$. As discovered by previous studies [14, 66, 142], the dominating term is $O(kmf)$ or $O(Lmf)$ when the graph is large since feature learning can be accelerated by parallel computation. Consequently, $O(Lmf)$ emerges as the dominating complexity term of linear transformation. Although FGL offers a new perspective for large-scale graph learning through a distributed paradigm, it still requires the deployment of suitable scalable GNNs on the local client.

The current mainstream trend in FL or FGL studies emphasizes the development of well-designed client-side updates to fit local data. For instance, FedProx introduces model weight alignment loss, resulting in complexities of $O(\omega f^2)$. Similarly, approaches like MOON, FGSSL, FGGP, FedStar, and AdaFGL employ CL loss and ensemble learning for local updates, introducing additional computational overhead upon the graph learning. Specifically, for the contrastive learning in MOON, FGSSL, and FGGP, the additional computational cost depends on the size and semantics of the query sample set, resulting in complexities of $O(Qf^2)$, $Q((b+k)f+f^2)$, $O(Qcp^2)$ respectively. This will lead to unacceptable computational overhead as the scale of local data increases. As for ensemble learning approaches like FedStar and AdaFGL, which maintain multiple models locally to extract private data semantics from various perspectives, they can be bounded by $O(E((b+k)f+f^2))$. Furthermore, FedSage+, Fed-PUB, and FedGTA exchange additional information during communication to improve federated training. Despite their inherent similarities, these methods exhibit significantly different time-space complexities due to variations in their design. Specifically, FedSage+ involves client-side data sharing for local subgraph data augmentation, leading to a complexity of $O(L((n+sg)f+f^2))$. Fed-PUB maintains a global pseudo-graph on the server side and utilizes locally uploaded weights to update

Table 10: Algorithm complexity analysis for existing prevalent FL and FGL studies.

Method	Client Mem.	Server Mem.	Inference Mem.	Client Time.	Server Time.	Inference Time
FedAvg	$O((b+k)f+f^2)$	$O(Nf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2)$	$O(Nf)$	$O(kmf+nf^2)$
FedProx	$O((b+k)f+\omega f^2)$	$O(Nf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+f^2)$	$O(Nf)$	$O(kmf+nf^2)$
Scaffold	$O((b+k)f+\omega f^2)$	$O(Nf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+f^2)$	$O(Nf+f^2)$	$O(kmf+nf^2)$
MOON	$O((b+k)f+Qf^2)$	$O(Nf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+Qnf)$	$O(Nf)$	$O(kmf+nf^2)$
FedProto	$O((b+k)f+cp)$	$O(Ncp)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+cp^2)$	$O(Ncp)$	$O(kmf+nf^2)$
FedNH	$O((b+k)f+cp)$	$O(N(f^2+cp)+c^2p^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+cp^2)$	$O(N(f+cp)+c^3p^3)$	$O(kmf+nf^2)$
FedTGP	$O((b+k)f+cp)$	$O((N+Q)cp+p^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2+cp^2)$	$O(Ncp+Qcp^2)$	$O(kmf+nf^2)$
GCFL+	$O((b+k)f+f^2)$	$O(TNf^2)$	$O((b+k)f+f^2)$	$O(kmf+nf^2)$	$O(N^2(\log(N)+T^2f^2))$	$O(kmf+nf^2)$
FedStar	$O(E((b+k)f+f^2))$	$O(Nf^2)$	$O(E((b+k)f+f^2))$	$O(E(kmf+nf^2))$	$O(Nf)$	$O(E(kmf+nf^2))$
FedGL	$O((b+k)f+f^2+c)$	$O(N(f^2+nc+n^2))$	$O((b+k)f+f^2)$	$O(kmf+nf^2+nc^2)$	$O(Nf+nc+n^2f^2)$	$O(kmf+nf^2)$
FedSage+	$O(L((n+sg)f+f^2))$	$O(LtNf^2)$	$O(L(n+sg)f+Lf^2)$	$O(L((m+sg)f+(n+sg)f^2))$	$O(Nf)$	$O(L((m+sg)f+(n+sg)f^2))$
FGSSL	$O(Q((b+k)f+f^2))$	$O(Nf^2)$	$O(L(b+k)f+Lf^2)$	$O(Qkmf+Qnf^2)$	$O(Nf)$	$O(kmf+nf^2)$
Fed-PUB	$O(M((b+k)f+f^2)+M^2)$	$O(N(f^2+M)+P_g)$	$O(M(b+k)f+Mf^2)$	$O(Mkmf+Mnf^2)$	$O(N^2(\log(N)+M^2))$	$O(Mkmf+Mnf^2)$
FGGP	$O((n+sg)f+f^2+Qcp)$	$O(Ncp)$	$O((b+k)f+f^2)$	$O((m+sg)f+(n+sg)f^2+Qcp^2)$	$O(N^2(\log(N)+c^2p^2)+Ncp)$	$O(kmf+nf^2)$
FedGTA	$O((b+k)f+f^2+kKc)$	$O(Nf^2+NkKc)$	$O((b+k)f+f^2)$	$O(km(f+knc)+n(f^2+c))$	$O(Nf+NkKc)$	$O(kmf+nf^2)$
AdaFGL	$O(E((b+k)f+f^2))$	$O(Nf^2)$	$O(E((b+k)f+f^2))$	$O(E(kmf+nf^2))$	$O(Nf)$	$O(E(kmf+nf^2))$
FedTAD	$O((b+k)f+f^2+nf)$	$O(Nf^2+sgf+Lf^2)$	$O((b+k)f+f^2)$	$O(knmf+nf^2)$	$O(Nf+(L+n)f^2+Lsgf)$	$O(kmf+nf^2)$



(a) PROTEINS in Graph-FL Scenario (b) Computers in Subgraph-FL Scenario

Figure 2: Practical efficiency in terms of performance, communication costs, and running time.

Table 11: Performance (%) on large-scale GraphFL.

Graph-FL Label (0.5)	molhiv (ROC-AUC)	molpcba (AP)	ppa (Acc)	code2 (F1 score)
FedAvg	72.2±2.4	20.4±0.4	65.8±1.6	29.8±0.4
FedProx	72.8±2.2	20.9±0.3	65.4±1.9	29.5±0.5
Scaffold	71.7±2.5	21.2±0.3	66.4±1.5	29.7±0.5
FedProto	70.5±1.8	19.3±0.2	63.7±1.2	28.6±0.3
FedNH	71.2±2.0	19.5±0.2	64.4±1.1	28.4±0.2
FedTGP	70.8±2.4	19.7±0.3	64.2±1.4	28.8±0.3
GCFL+	73.4±1.9	21.8±0.3	67.0±1.5	30.5±0.3
FedStar	73.8±1.7	21.5±0.2	67.2±1.8	31.2±0.4

Table 12: Efficiency on Louvain-based Physics (Subgraph-FL).

Method	Test Acc (%)	Memory	Com.	Running Time
Scaffold	90.6±0.5	538k	1076k	152.42s
FedTGP	87.4±0.3	539k	0.38k	76.87s
FedSage+	91.7±0.8	1296k	1784k	1517.96s
GCFL+	91.0±0.3	538k	538k	184.24s
Fed-PUB	91.6±0.5	1076k	1076k	391.60s
FedGTA	91.5±0.3	538k	539k	120.45s
FedStar	91.3±0.5	1076k	539k	268.12s
FGSSL	91.1±0.6	751k	540k	476.58s
AdaFGL	92.0±0.4	964k	538k	162.19s

trainable mask matrices for personalized learning, introducing a complexity of $O(N(f^2+M)+P_g)$. In contrast, FedGTA is a lightweight method that utilizes topology-aware soft labels to encode local data, enabling personalized model aggregation on the server. As a result, this approach possesses a complexity of $O(kKC)$.

While client-side training has proven effective, an increasing number of methods have recently recognized the significant potential of optimizing server-side model aggregation for federated training. For example, FedGL empowers local training by executing global pseudo-labeling and topological mining on the server side, which can be bounded by $O(Nf+nc+n^2f^2)$. FedTAD, on the other hand, meticulously adjusts global aggregation models to enhance the initialization of local models for the next communication round through graph-specific data-free knowledge distillation, albeit incurring additional overhead of $O(Nf+(L+n)f^2+Lsgf)$.

Moreover, FedProto, FedNH, and FedTGP propose prototype-based FL. They exchange class-specific embeddings between participating clients and servers in each communication round, reducing the complexity from $O(Nf^2)$ to $O(Ncp)$. Additionally, FedNH optimizes global prototype initialization using interior point methods, which, while effective, poses an out-of-time risk of $O(c^3p^3)$ when datasets comprise multiple categories. In comparison, FedTGP introduces independent neural architectures on the server side to adjust global prototypes, providing relative flexibility.

Table 13: A summary and selection suggestions for current prevalent FGL studies.

Methods	Effectiveness	Robustness					Efficiency		
	Statistic Heterogeneity	Noise	Sparsity	Low Client	Generalization	Privacy	Communication	Scalability	Parallelism
GCFL+ [122]	✗	✓	✗	✓	✗	✗	✓	✓	✓
FedStar [105]	✓	✗	✓	✗	✓	✗	✓	✗	✗
FedSage+ [140]	✗	✗	✓	✗	✗	✗	✗	✗	✗
Fed-PUB [8]	✓	✓	✗	✓	✗	✗	✗	✗	✗
FedGTA [70]	✓	✓	✗	✓	✗	✗	✓	✗	✓
FGSSL [44]	✗	✗	✓	✓	✗	✗	✓	✗	✗
FedGL [13]	✗	✗	✓	✗	✓	✗	✗	✗	✗
AdaFGL [67]	✓	✓	✓	✗	✓	✗	✓	✗	✓
FGGP [110]	✓	✗	✗	✓	✗	✗	✓	✓	✗
FedDEP [138]	✗	✗	✓	✗	✗	✓	✓	✗	✗
FedTAD [149]	✓	✗	✓	✗	✓	✗	✓	✗	✗

4.4 Efficiency Evaluation

To answer **Q8**, we provide the efficiency reports in Fig. 2, Table 11, and Table 12, and obtain the following observations: (1) Prototype-based FedProto, FedTGP, and FGGP reduce communication costs but require extra computation, offsetting their runtime advantage. (2) Cross-client FedGL and FedSage+ suffer reduced efficiency due to delays from inter-client communication. (3) Decoupled AdaFGL maximizes local computation and minimizes communication costs, providing efficiency advantages. (4) In Graph-FL, models treat data samples independently, so even with large-scale graph samples, the limited size of each graph minimizes OOM or OOT issues. However, no single approach consistently achieves superior performance. (5) In our experiments, Param. refers to trainable parameters, Time to total runtime, and Com. to communication costs. FedSage+ and AdaFGL perform better but incur high Com. and computational complexity. Consequently, we conclude that *C10: FGL algorithms leveraging prototypes and decoupled techniques (i.e., multi-client collaboration then local updates) demonstrate substantial potential in applications with stringent efficiency requirements [39, 69].*

4.5 FGL Guidance and OpenFGL Tutorial

In this section, we provide a detailed overview of OpenFGL: (1) Comprehensive evaluation of FGL methods to guide deployment (FGL Guidance); (2) User-friendly APIs to facilitate the reproduction and further development of FGL algorithms (OpenFGL Tutorial).

FGL Guidance. In Sec.4.1-Sec.4.4, we conduct a comprehensive empirical investigation of effectiveness, robustness, and efficiency, drawing 10 conclusions. These insights are crucial for selecting appropriate FGL algorithms in real-world applications. To provide a clearer presentation, we summarize the prevalent FGL algorithms in Table 13. Notably, scalability refers to the ability of a method to handle large-scale graphs without OOM or OOT. Regarding parallelism, we evaluate whether the method relies heavily on server-broadcasted information for local training. Minimal dependency allows for fewer communication and enables more independent parallel client training. Based on this, we observe that current FGL algorithms struggle to maintain consistent competitiveness across various requirements, highlighting that the field is still in its early stages with significant potential for future development.

Algorithm 1: OpenFGL-*FGLTrainer*.*Pytorch style.*

```
class FGLTrainer:
    def __init__(self, args):
        self.args = args
        self.message_pool = {}
        self.clients = load_client(args,...)

    def train(self):
        for round_id in range(self.args.num_rounds):
            self.message_pool["round"] = round_id
            self.message_pool["sampled_clients"] = sampled_clients
            self.server.send_message()
            for client_id in sampled_clients:
                self.clients[client_id].execute()
                self.clients[client_id].send_message()
            self.server.execute()
            self.evaluate()
```

OpenFGL Tutorial. We now present the algorithm design principles, which offer a unified API. It uses the *FGLTrainer* to manage client-server communication during each training round, aggregating messages via the *message_pool* variable. Users can customize the *FGLClient* and *FGLServer* to adjust message content and rules for specific algorithms. For clarity, we provide *PyTorch-style* implementations, exemplified by the *FedAvg* algorithm:

(a) FGLTrainer. This class manages message and command flows between clients and a central server. In each training round, the trainer selects clients, updates the *message_pool*, and dispatches server messages. Clients process tasks locally and send updates to the server for global aggregation. Each round includes an evaluation phase. The implementation of *FGLTrainer* is shown in Algorithm 1.

(b) FedAvgClient. Users only need to customize: (1) Local execution, where FedAvg downloads the global model and performs local training; (2) Client-to-server messages, including the local sample count and model weights in FedAvg. The *Pytorch-style* implementation of *FedAvgClient* is provided in Algorithm 2.

(c) FedAvgServer. Similar to *FedAvgClient*, users customize two modules: (1) Global execution (e.g., in FedAvg: weighted model averaging based on sample size); (2) Server-to-client messages (e.g., in FedAvg: global model weights). The *Pytorch-style* implementation of *FedAvgServer* is illustrated in Algorithm 3.

Algorithm 2: OpenFGL-*FGLClient*.*Pytorch style.*

```
class FedAvgClient(BaseClient):
    def __init__(self, args, client_id...):

    def execute(self):
        # Receive messages (global model weights) from the server
              stored in the message pool to update the local model
        with torch.no_grad():
            for (l_param, g_param) in zip(self.task.model.parameters
                  (), self.message_pool[server][weight]):
                local_param.data.copy_(global_param)
        self.task.train()

    def send_message(self):
        self.message_pool[f"client_{self.client_id}"] = {
                num_samples: self.task.num_samples,
                weight: list(self.task.model.parameters())}
```

Algorithm 3: OpenFGL-*FGLServer*: *Pytorch style.*

```
class FedAvgServer(BaseServer):
    def __init__(self, args, message_pool...):

    def execute(self):
        # Receive messages (number of samples, local model weight)
              from each client stored in the message pool
        with torch.no_grad():
            for it, client_id in enumerate(sampled_clients):
                weight = self.message_pool[client_id][num_samples]/
                      num_tot_samples (FedAvg_model_aggregation...)

    def send_message(self):
        self.message_pool["server"] = {
            weight: list(self.task.model.parameters())}
```

5 CONCLUSION AND FUTURE DIRECTIONS

In this paper, we first present a comprehensive overview of the current research progress in the FGL field and the significant potential of this technology for deployment in graph-based database applications. Subsequently, we propose OpenFGL, a comprehensive FGL benchmark, which encompasses 18 recently proposed SOTA FGL algorithms and 42 datasets from 18 domains for 5 downstream tasks across 2 prevalent FGL scenarios. The goal of our work is to fairly examine the current state of FGL development and offer key insights for future research endeavors. Although FGL primarily serves downstream tasks in graph-based ML, it also holds significant potential for database applications. Specifically, it enables each client to generate high-quality graph embeddings (nodes, edges, and subgraphs) in a privacy-preserve and distributed manner. These embeddings provide rich semantic representations, which can be leveraged for efficient retrieval in vector databases. This perspective introduces new opportunities for applying FGL techniques to enhance the performance and scalability of graph-based databases.

Subsequently, we conduct extensive experiments aimed at unveiling the performance of FGL algorithms from 3 orthogonal perspectives: effectiveness, robustness, and efficiency. Our investigations reveal promising advancements achieved by FGL studies but also highlight their limitations, such as vulnerability in inadequate node descriptions, robustness, and scalability. To inspire future research, we combine experimental ***Conclusions*** to present the following significant challenges and promising directions.

For **Effectiveness**, (1) **Quantify Distributed Graphs** (*C1*, *C3*). Essentially, the potential benefits of FGL in real-world deployments stem from the uniform graph distribution. However, the entanglement of node features, labels, and topology poses a challenge in explicitly quantifying statistics within distributed graphs, resulting in coarse descriptions. This constraint sharply contrasts with the intuitive semantic feature and label distribution skew observed in CV-based FL. Therefore, quantifying the statistics of multi-source graphs is crucial. (2) **FGL Heterogeneity** (*C2*, *C4*). Although some FGL studies attempt to define graph-based heterogeneity challenges, these definitions are often insufficient due to the complexity of graph characteristics and the diversity of applications. Furthermore, the advantages of these FGL algorithms (e.g., AdaFGL) lack significant impact. Consequently, there is still a necessary effort to be made in addressing FGL heterogeneity.

For **Robustness**, (3) **Personalized FGL** (*C5*, *C7*). In the real world, the robustness of FGL against client-specific noise and low client participation in communication is essential. During federated training, these factors significantly impact the attainment of global consensus, thereby hindering high-quality supervision provided for local training. Fortunately, personalized techniques can leverage local knowledge to establish unbiased global consensus for local updates. (4) **Multi-client Collaboration FGL** (*C6*, *C8*). During our investigation, we found that promoting server-side multi-client collaboration can extract global insights from sparse data. Additionally, this collaborative approach can capture shared semantic knowledge across data domains to facilitate robust generalization. (5) **Privacy-preserve FGL** (*C9*). The goal of FGL is to achieve multi-client collaborative training in a privacy-preserving manner without direct data sharing. However, current FGL algorithms, in pursuit of superior performance, increasingly share local information, raising potential concerns. Therefore, the development of FGL algorithms with strict privacy requirements is imperative.

For **Efficiency**, (6) **Decoupled and Scalable FGL** (*C10*). Existing FGL algorithms face challenges in practical deployment due to communication delay and topology mining, making it difficult to handle large-scale datasets. Therefore, developing new federated collaboration paradigms such as decoupled mechanisms and focusing on algorithm design scalability is crucial.

FGL should establish federated collaboration standards for various graph types (e.g., directed, signed, hypergraphs, heterogeneous) and learning paradigms (e.g., unsupervised, few-shot, continual, unlearning) based on the data systems [9, 107, 146]. However, FGL remains a burgeoning field with numerous research gaps. Nevertheless, we are committed to continually enhancing OpenFGL to support future research endeavors. For instance, we are progressively refining the execution standards for federated heterogeneous graph learning. Notably, considering the space constraints and the need for a clear and reader-friendly presentation, we provide an overview and corresponding experimental results in [1] (A.9).

ACKNOWLEDGMENTS

This work was partially supported by Key Program of the National Natural Science Joint Foundation of China, U2241211 and U24A20255. Rong-Hua Li is the corresponding author of this paper.

REFERENCES

[1] 2024. OpenFGL Technical Report. In https://github.com/xkLi-Allen/OpenFGL.

[2] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In Proceedings of ACM SIGSAC Conference on Computer and Communications Security, CCS.

[3] Mohamad Elhadi Abushofa, Amir Atapour Abarghouei, Matthew Forshaw, and Andrew Stephen Mcgough. 2023. FEGR: Feature Enhanced Graph Representation Method for Graph Classification. In Proceedings of the International Conference on Advances in Social Networks Analysis and Mining.

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

[5] Rodolfo Stoffel Antunes, Cristiano André da Costa, Arne Küderle, Imrana Abdullahi Yari, and Björn Eskofier. 2022. Federated learning for healthcare: Systematic review and architecture proposal. ACM Transactions on Intelligent Systems and Technology (TIST) 13, 4 (2022), 1–23.

[6] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 2016. 3d semantic parsing of large-scale indoor spaces. In Proceedings of the IEEE conference on computer vision and pattern recognition. 1534–1543.

[7] Sara Babakniya, Souvik Kundu, Saurav Prakash, Yue Niu, and Salman Avestimehr. 2023. Revisiting Sparsity Hunting in Federated Learning: Why does Sparsity Consensus Matter? Transactions on Machine Learning Research (2023).

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

[9] Maciej Besta, Patrick Iff, Florian Scheidl, Kazuki Osawa, Nikoli Dryden, Michal Podstawski, Tiancheng Chen, and Torsten Hoefler. 2022. Neural graph databases. In Learning on Graphs Conference, LoG.

[10] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. Journal of machine Learning research 3, Jan (2003), 993–1022.

[11] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment 2008, 10 (2008), P10008.

[12] Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. 2005. Protein function prediction via graph kernels. Bioinformatics 21, suppl_1 (2005), i47–i56.

[13] Chuan Chen, Ziyue Xu, Weibo Hu, Zibin Zheng, and Jie Zhang. 2024. Fedgl: Federated graph learning framework with global self-supervision. Information Sciences 657 (2024), 119976.

[14] Ming Chen, Zhewei Wei, Bolin Ding, Yaliang Li, Ye Yuan, Xiaoyong Du, and Ji-Rong Wen. 2020. Scalable graph neural networks via bidirectional propagation. Advances in Neural Information Processing Systems, NeurIPS (2020).

[15] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and deep graph convolutional networks. In International Conference on Machine Learning, ICML.

[16] Yutong Dai, Zeyuan Chen, Junnan Li, Shelby Heinecke, Lichao Sun, and Ran Xu. 2023. Tackling data heterogeneity in federated learning with class prototypes. In Proceedings of the AAAI Conference on Artificial Intelligence, AAAI.

[17] Ameya Daigavane, Gagan Madan, Aditya Sinha, Abhradeep Guha Thakurta, Gaurav Aggarwal, and Prateek Jain. 2021. Node-level differentially private graph neural networks. In arXiv preprint arXiv:2111.15521.

[18] Asim Kumar Debnath, Rosa L Lopez de Compadre, Gargi Debnath, Alan J Shusterman, and Corwin Hansch. 1991. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. Journal of medicinal chemistry 34, 2 (1991), 786–797.

[19] Frederik Diehl, Thomas Brunner, Michael Truong Le, and Alois Knoll. 2019. Towards graph pooling by edge contraction. In ICML 2019 Workshop on Learning and Reasoning with Graph-Structured Data.

[20] Paul D Dobson and Andrew J Doig. 2003. Distinguishing enzyme structures from non-enzymes without alignments. Journal of molecular biology 330, 4 (2003), 771–783.

[21] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In Theory of Cryptography: Third Theory of Cryptography Conference, TCC. Springer.

[22] Jie Fu, Yuan Hong, Xinpeng Ling, Leixia Wang, Xun Ran, Zhiyu Sun, Wendy Hui Wang, Zhili Chen, and Yang Cao. 2024. Differentially private federated learning: A systematic review. arXiv preprint arXiv:2405.08299 (2024).

[23] Lei Fu, Huanle Zhang, Ge Gao, Mi Zhang, and Xin Liu. 2023. Client selection in federated learning: Principles, challenges, and opportunities. IEEE Internet of Things Journal (2023).

[24] Xinyu Fu and Irwin King. 2023. FedHGN: A Federated Framework for Heterogeneous Graph Neural Networks. In Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI.

[25] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. MAGNN: Metapath Aggregated Graph Neural Network for Heterogeneous Graph Embedding. In Proceedings of the ACM Web Conference, WWW.

[26] Hongyang Gao and Shuiwang Ji. 2019. Graph u-nets. In International Conference on Machine Learning, ICML.

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

[28] Zihao Gao, Huifang Ma, Xiaohui Zhang, Yike Wang, and Zheyu Wu. 2023. Similarity measures-based graph co-contrastive learning for drug–disease association prediction. Bioinformatics 39, 6 (2023), btad357.

[29] Anna Gaulton, Anne Hersey, Michał Nowotka, A Patricia Bento, Jon Chambers, David Mendez, Prudence Mutowo, Francis Atkinson, Louisa J Bellis, Elena Cibrián-Uhalte, et al. 2017. The ChEMBL database in 2017. Nucleic acids research 45, D1 (2017), D945–D954.

[30] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. 2020. Inverting gradients-how easy is it to break privacy in federated learning?. In Advances in Neural Information Processing Systems, NeurIPS.

[31] Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. arXiv preprint arXiv:1802.06893 (2018).

[32] Zishan Gu, Ke Zhang, Guangji Bai, Liang Chen, Liang Zhao, and Carl Yang. 2023. Dynamic activation of clients and parameters for federated learning over heterogeneous graphs. In International Conference on Data Engineering, ICDE. IEEE.

[33] Zishan Gu, Ke Zhang, Guangji Bai, Liang Chen, Liang Zhao, and Carl Yang. 2023. Dynamic Activation of Clients and Parameters for Federated Learning over Heterogeneous Graphs. In International Conference on Data Engineering, ICDE.

[34] Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J Mitra. 2018. Pcpnet learning local shape properties from raw point clouds. In Computer graphics forum, Vol. 37. Wiley Online Library, 75–85.

[35] Yaming Guo, Kai Guo, Xiaofeng Cao, Tieru Wu, and Yi Chang. 2023. Out-of-distribution generalization of federated learning via implicit invariant relationships. In International Conference on Machine Learning, ICML.

[36] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. Advances in Neural Information Processing Systems, NeurIPS (2017).

[37] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In Advances in Neural Information Processing Systems, NeurIPS.

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

[39] Clare Elizabeth Heinbaugh, Emilio Luz-Ricca, and Huajie Shao. 2023. Data-free one-shot federated learning under very high statistical heterogeneity. In The Eleventh International Conference on Learning Representations.

[40] Christoph Helma, Ross D. King, Stefan Kramer, and Ashwin Srinivasan. 2001. The predictive toxicology challenge 2000–2001. Bioinformatics 17, 1 (2001), 107–108.

[41] David K Hsiao. 1992. Federated databases and systems: part i—a tutorial on their data sharing. The VLDB Journal 1 (1992), 127–179.

[42] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. Advances in Neural Information Processing Systems, NeurIPS (2020).

[43] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous Graph Transformer. In Proceedings of the ACM Web Conference, WWW.

[44] Wenke Huang, Guancheng Wan, Mang Ye, and Bo Du. 2023. Federated graph semantic and structural learning. In Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI.

[45] Woochang Hyun, Jaehong Lee, and Bongwon Suh. 2023. Anti-Money Laundering in Cryptocurrency via Multi-Relational Graph Neural Network. In Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, 118–130.

[46] Zhida Jiang, Yang Xu, Hongli Xu, Zhiyuan Wang, and Chunming Qiao. 2023. Clients Help Clients: Alternating Collaboration for Semi-Supervised Federated Learning. In International Conference on Data Engineering, ICDE. IEEE.

[47] Magdi N Kamel and Nabil N Kamel. 1992. Federated database management system: Requirements, issues and solutions. Computer Communications 15, 4 (1992), 270–278.

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

[49] George Karypis and Vipin Kumar. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM Journal on Scientific

Computing 20, 1 (1998), 359–392.
[50] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In International Conference on Learning Representations, ICLR.
[51] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In International Conference on Learning Representations, ICLR.
[52] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-attention graph pooling. In International Conference on Machine Learning, ICML.
[53] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. 2005. Graphs over time: densification laws, shrinking diameters and possible explanations. In Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD.
[54] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford large network dataset collection. (2014).
[55] Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. arXiv preprint arXiv:2109.02846 (2021).
[56] Anran Li, Yuanyuan Chen, Jian Zhang, Mingfei Cheng, Yihao Huang, Yueming Wu, Anh Tuan Luu, and Han Yu. 2024. Historical Embedding-Guided Efficient Large-Scale Federated Graph Learning. Proceedings of the ACM on Management of Data, SIGMOD (2024).
[57] Baiqi Li, Yedi Ma, Yufei Liu, Hongyan Gu, Zhenghan Chen, and Xinli Huang. 2024. Federated Learning on Distributed Graphs Considering Multiple Heterogeneities. In IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP.
[58] Li Li, Yuxi Fan, Mike Tse, and Kuo-Yi Lin. 2020. A review of applications in federated learning. Computers & Industrial Engineering 149 (2020), 106854.
[59] Nianzhe Li, Hanwen Liu, Shunmei Meng, and Qianmu Li. 2023. FDRS: Federated Diversified Recommender System Based on Heterogeneous Graph Convolutional Network. In International Conferences on High Performance Computing and Communications, HPCC.
[60] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. 2022. Federated learning on non-iid data silos: An experimental study. In International Conference on Data Engineering, ICDE.
[61] Qinbin Li, Bingsheng He, and Dawn Song. 2021. Model-contrastive federated learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR.
[62] Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. 2021. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. IEEE Transactions on Knowledge and Data Engineering 35, 4 (2021), 3347–3366.
[63] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. Proceedings of Machine Learning and Systems, MLSys (2020).
[64] Wenqian Li, Shuran Fu, Fengrui Zhang, and Yan Pang. 2024. Data Valuation and Detections in Federated Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR.
[65] Xunkai Li, Ronghui Guo, Jianwen Chen, Youpeng Hu, Meixia Qu, and Bin Jiang. 2022. Effective Hybrid Graph and Hypergraph Convolution Network for Collaborative Filtering. Neural Comput. Appl. 35, 3 (sep 2022), 2633–2646.
[66] Xunkai Li, Meihao Liao, Zhengyu Wu, Daohan Su, Wentao Zhang, Rong-Hua Li, and Guoren Wang. 2024. LightDiC: A Simple Yet Effective Approach for Large-Scale Digraph Representation Learning. Proceedings of the VLDB Endowment (2024).
[67] Xunkai Li, Zhengyu Wu, Wentao Zhang, Henan Sun, Rong-Hua Li, and Guoren Wang. 2024. AdaFGL: A New Paradigm for Federated Node Classification with Topology Heterogeneity. arXiv preprint arXiv:2401.11750 (2024).
[68] Xunkai Li, Zhengyu Wu, Wentao Zhang, Henan Sun, Rong-Hua Li, and Guoren Wang. 2024. AdaFGL: A New Paradigm for Federated Node Classification with Topology Heterogeneity. arXiv preprint arXiv:2401.11750 (2024).
[69] Xunkai Li, Zhengyu Wu, Wentao Zhang, Henan Sun, Rong-Hua Li, and Guoren Wang. 2024. AdaFGL: A New Paradigm for Federated Node Classification with Topology Heterogeneity. arXiv preprint arXiv:2401.11750 (2024).
[70] Xunkai Li, Zhengyu Wu, Wentao Zhang, Yinlin Zhu, Rong-Hua Li, and Guoren Wang. 2023. FedGTA: Topology-Aware Averaging for Federated Graph Learning. Proceedings of the VLDB Endowment (2023).
[71] Xunkai Li, Zhengyu Wu, Wentao Zhang, Yinlin Zhu, Rong-Hua Li, and Guoren Wang. 2023. FedGTA: Topology-Aware Averaging for Federated Graph Learning. Proceedings of the VLDB Endowment (2023).
[72] Yipeng Li and Xinchen Lyu. 2024. Convergence Analysis of Sequential Federated Learning on Heterogeneous Data. In Advances in Neural Information Processing Systems, NeurIPS.
[73] Zhize Li, Haoyu Zhao, Boyue Li, and Yuejie Chi. 2022. SoteriaFL: A unified framework for private federated learning with communication compression. In Advances in Neural Information Processing Systems, NeurIPS.
[74] Ling Liang, Jilan Lin, Zheng Qu, Ishtiyaque Ahmad, Fengbin Tu, Trinabh Gupta, Yufei Ding, and Yuan Xie. 2023. Spg: Structure-private graph database via squeezepir. Proceedings of the VLDB Endowment (2023).
[75] Daniil Likhobaba, Nikita Pavlichenko, and Dmitry Ustalov. 2023. Toloker Graph: Interaction of Crowd Annotators. (2023). https://doi.org/10.5281/zenodo.7620795
[76] D. Lim, F. Hohne, X. Li, S. L. Huang, V. Gupta, O. Bhalerao, and S. N. Lim. 2021. Large Scale Learning on Non-Homophilous Graphs: New Benchmarks and Strong Simple Methods. In Advances in Neural Information Processing Systems, NeurIPS.
[77] Shiyi Lin, Deming Zhai, Feilong Zhang, Junjun Jiang, Xianming Liu, and Xiangyang Ji. 2024. Overhead-free noise-tolerant federated learning: A new baseline. Machine Intelligence Research 21, 3 (2024), 526–537.
[78] Ji Liu, Jizhou Huang, Yang Zhou, Xuhong Li, Shilei Ji, Haoyi Xiong, and Dejing Dou. 2022. From distributed machine learning to federated learning: A survey. Knowledge and Information Systems 64, 4 (2022), 885–917.
[79] Rui Liu, Pengwei Xing, Zichao Deng, Anran Li, Cuntai Guan, and Han Yu. 2024. Federated graph neural networks: Overview, techniques, and challenges. IEEE Transactions on Neural Networks and Learning Systems (2024).
[80] Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. 2022. Revisiting heterophily for graph neural networks. Advances in Neural Information Processing Systems, NeurIPS (2022).
[81] Xinjian Luo, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. 2021. Feature inference attack on model predictions in vertical federated learning. In International Conference on Data Engineering, ICDE.
[82] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. Are we really making much progress?: Revisiting, benchmarking and refining heterogeneous graph neural networks. In Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD.
[83] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. 2021. Is homophily a necessity for graph neural networks? International Conference on Learning Representations, ICLR (2021).
[84] Zheng Ma, Junyu Xuan, Yu Guang Wang, Ming Li, and Pietro Liò. 2020. Path integral based convolution and pooling for graph neural networks. Advances in Neural Information Processing Systems, NeurIPS (2020).
[85] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. Artificial Intelligence and Statistics (2017).
[86] Ilya Mironov. 2017. Rényi differential privacy. In IEEE Computer Security Foundations Symposium, CSF.
[87] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. 2017. Geometric deep learning on graphs and manifolds using mixture model cnns. In Proceedings of the IEEE conference on computer vision and pattern recognition. 5115–5124.
[88] Alessio Mora, Armir Bujari, and Paolo Bellavista. 2024. Enhancing generalization in federated learning with heterogeneous data: A comparative literature review. Future Generation Computer Systems (2024).
[89] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2019. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In IEEE Symposium on Security and Privacy, SP.
[90] Qiying Pan, Yifei Zhu, and Lingyang Chu. 2023. Lumos: Heterogeneity-aware federated graph learning over decentralized devices. In International Conference on Data Engineering, ICDE. IEEE.
[91] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-gcn: Geometric graph convolutional networks. In International Conference on Learning Representations, ICLR.
[92] Oleg Platonov, Denis Kuznedelev, Artem Babenko, and Liudmila Prokhorenkova. 2023. Characterizing graph datasets for node classification: Beyond homophily-heterophily dichotomy. Advances in Neural Information Processing Systems, NeurIPS (2023).
[93] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. 2023. A critical look at the evaluation of GNNs under heterophily: are we really making progress? International Conference on Learning Representations, ICLR (2023).
[94] Yuxin Qiu. 2023. Default Risk Assessment of Internet Financial Enterprises Based on Graph Neural Network. In IEEE Information TechnoLoGy, Networking, Electronic and Automation Control Conference, Vol. 6. IEEE, 592–596.
[95] Zhe Qu, Xingyu Li, Xiao Han, Rui Duan, Chengchao Shen, and Lixing Chen. 2023. How to prevent the poor performance clients for personalized federated learning?. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR.
[96] Zongshuai Qu, Tao Yao, Xinghui Liu, and Gang Wang. 2023. A Graph Convolutional Network Based on Univariate Neurodegeneration Biomarker for Alzheimer's Disease Diagnosis. IEEE Journal of Translational Engineering in Health and Medicine (2023).

[97] Kaspar Riesen and Horst Bunke. 2008. IAM graph database repository for graph based pattern recognition and machine learning. In Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshop, SSPR & SPR 2008, Orlando, USA, December 4-6, 2008. Proceedings. Springer, 287–297.

[98] Yichen Ruan, Xiaoxi Zhang, and Carlee Joe-Wong. 2024. How valuable is your data? optimizing client recruitment in federated learning. IEEE/ACM Transactions on Networking (2024).

[99] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In European Semantic Web Conference, ESWC.

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

[101] Amit P Sheth and James A Larson. 1990. Federated database systems for managing distributed, heterogeneous, and autonomous databases. ACM Computing Surveys (CSUR) 22, 3 (1990), 183–236.

[102] Kuangchi Sun and Aijun Yin. 2024. Multisensor Graph Adaptive Federated Generalization for Helicopter Transmission System Fault Diagnosis. IEEE Transactions on Instrumentation and Measurement 73 (2024), 1–11.

[103] Lichao Sun and Lingjuan Lyu. 2021. Federated model distillation with noise-free differential privacy. In Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI.

[104] Jeffrey J Sutherland, Lee A O'brien, and Donald F Weaver. 2003. Spline-fitting with a genetic algorithm: A method for developing classification structure-activity relationships. Journal of chemical information and computer sciences 43, 6 (2003), 1906–1915.

[105] Yue Tan, Yixin Liu, Guodong Long, Jing Jiang, Qinghua Lu, and Chengqi Zhang. 2023. Federated learning on non-iid graphs via structural knowledge sharing. In Proceedings of the AAAI Conference on Artificial Intelligence, AAAI.

[106] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. 2022. Fedproto: Federated prototype learning across heterogeneous clients. In Proceedings of the AAAI Conference on Artificial Intelligence, AAAI.

[107] Yuanyuan Tian. 2023. The world of graph databases from an industry perspective. ACM SIGMOD Record 51, 4 (2023), 60–67.

[108] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In International Conference on Learning Representations, ICLR.

[109] Nikil Wale, Ian A Watson, and George Karypis. 2008. Comparison of descriptor spaces for chemical compound retrieval and classification. Knowledge and Information Systems 14 (2008), 347–375.

[110] Guancheng Wan, Wenke Huang, and Mang Ye. 2024. Federated Graph Learning under Domain Shift with Generalizable Prototypes. In Proceedings of the AAAI Conference on Artificial Intelligence, AAAI.

[111] Junfu Wang, Yawen Li, Meiyu Liang, and Ang Li. 2022. Embedding Representation of Academic Heterogeneous Information Networks Based on Federated Learning. In IEEE International Conference on Cloud Computing and Intelligent Systems, CCIS.

[112] Kuansan Wang, Zhihong Shen, Chiyuan Huang, Chieh-Han Wu, Yuxiao Dong, and Anshul Kanakia. 2020. Microsoft academic graph: When experts are not enough. Quantitative Science Studies 1, 1 (2020), 396–413.

[113] Qinyong Wang, Hongzhi Yin, Tong Chen, Junliang Yu, Alexander Zhou, and Xiangliang Zhang. 2022. Fast-adapting and privacy-preserving federated recommender system. The VLDB Journal 31, 5 (2022), 877–896.

[114] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In Proceedings of the ACM Web Conference, WWW.

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

[116] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. 2020. Federated learning with differential privacy: Algorithms and performance analysis. IEEE Transactions on Information Forensics and Security, TIFS 15 (2020), 3454–3469.

[117] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In International Conference on Machine Learning, ICML.

[118] Honghu Wu, Xiangrong Zhu, and Wei Hu. 2024. A Blockchain System for Clustered Federated Learning with Peer-to-Peer Knowledge Transfer. Proceedings of the VLDB Endowment (2024).

[119] Leijie Wu, Song Guo, Yaohong Ding, Junxiao Wang, Wenchao Xu, Yufeng Zhan, and Anne-Marie Kermarrec. 2024. Rethinking Personalized Client Collaboration in Federated Learning. IEEE Transactions on Mobile Computing (2024).

[120] Yuncheng Wu, Naili Xing, Gang Chen, Tien Tuan Anh Dinh, Zhaojing Luo, Beng Chin Ooi, Xiaokui Xiao, and Meihui Zhang. 2023. Falcon: A privacy-preserving and interpretable vertical federated learning system. Proceedings of the VLDB Endowment (2023).

[121] Zhichang Xia, Xinglin Zhang, Lingyu Liang, Yun Li, and Yuejiao Gong. 2024. Federated Graph Augmentation for Semisupervised Node Classification. IEEE Transactions on Computational Social Systems (2024).

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

[123] Han Xie, Li Xiong, and Carl Yang. 2023. Federated Node Classification over Graphs with Latent Link-type Heterogeneity. In Proceedings of the ACM Web Conference, WWW.

[124] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? (2019).

[125] Bo Yan, Yang Cao, Haoyu Wang, Wenchuan Yang, Junping Du, and Chuan Shi. 2024. Federated Heterogeneous Graph Neural Network for Privacy preserving Recommendation. In Proceedings of the ACM Web Conference, WWW.

[126] Pinar Yanardag and SVN Vishwanathan. 2015. Deep graph kernels. In Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD.

[127] Liangwei Yang, Shengjie Wang, Yunzhe Tao, Jiankai Sun, Xiaolong Liu, Philip S Yu, and Taiqing Wang. 2023. DGRec: Graph Neural Network for Recommendation with Diversified Embedding Generation. In Proceedings of the ACM International Conference on Web Search and Data Mining, WSDM.

[128] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST) 10, 2 (2019), 1–19.

[129] Qiang Yang, Changsheng Ma, Qiannan Zhang, Xin Gao, Chuxu Zhang, and Xiangliang Zhang. 2023. Counterfactual Learning on Heterogeneous Graphs with Greedy Perturbation. In Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD.

[130] Xiaocheng Yang, Mingyu Yan, Shirui Pan, Xiaochun Ye, and Dongrui Fan. 2023. Simple and Efficient Heterogeneous Graph Neural Network. In Proceedings of the Association for the Advancement of Artificial Intelligence, AAAI.

[131] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In International Conference on Machine Learning, ICML.

[132] Feng Yao, Qian Tao, Wenyuan Yu, Yanfeng Zhang, Shufeng Gong, Qiange Wang, Ge Yu, and Jingren Zhou. 2023. RAGraph: A Region-Aware Framework for Geo-Distributed Graph Processing. Proceedings of the VLDB Endowment (2023).

[133] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. 2016. A scalable active framework for region annotation in 3d shape collections. ACM Transactions on Graphics (ToG) 35, 6 (2016), 1–12.

[134] Wei Yuan, Liang Qu, Lizhen Cui, Yongxin Tong, Xiaofang Zhou, and Hongzhi Yin. 2024. Hetefedrec: Federated recommender systems with model heterogeneity. In International Conference on Data Engineering, ICDE. IEEE.

[135] Xiaoming Yuan, Jiahui Chen, Jiayu Yang, Ning Zhang, Tingting Yang, Tao Han, and Amir Taherkordi. 2022. Fedstn: Graph representation driven federated learning for edge computing enabled urban traffic flow prediction. IEEE Transactions on Intelligent Transportation Systems 24, 8 (2022), 8738–8748.

[136] Ye Yuan, Delong Ma, Zhenyu Wen, Zhiwei Zhang, and Guoren Wang. 2021. Subgraph matching over graph federation. Proceedings of the VLDB Endowment (2021).

[137] Jianqing Zhang, Yang Liu, Yang Hua, and Jian Cao. 2024. FedTGP: Trainable Global Prototypes with Adaptive-Margin-Enhanced Contrastive Learning for Data and Model Heterogeneity in Federated Learning. In Proceedings of the AAAI Conference on Artificial Intelligence, AAAI.

[138] Ke Zhang, Lichao Sun, Bolin Ding, Siu Ming Yiu, and Carl Yang. 2024. Deep efficient private neighbor generation for subgraph federated learning. In Proceedings of SIAM International Conference on Data Mining, SDM.

[139] Ke Zhang, Lichao Sun, Bolin Ding, Siu Ming Yiu, and Carl Yang. 2024. Deep efficient private neighbor generation for subgraph federated learning. In SIAM International Conference on Data Mining, SDM.

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

[141] Rongyu Zhang, Yun Chen, Chenrui Wu, and Fangxin Wang. 2024. Multi-level Personalized Federated Learning on Heterogeneous and Long-Tailed Data. IEEE Transactions on Mobile Computing (2024).

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

[143] Xinyi Zhang, Qichen Wang, Cheng Xu, Yun Peng, and Jianliang Xu. 2024. FedKNN: Secure Federated k-Nearest Neighbor Search. Proceedings of the ACM on Management of Data, SIGMOD (2024).

[144] Yi Zhang, Yuying Zhao, Zhaoqing Li, Xueqi Cheng, Yu Wang, Olivera Kotevska, S Yu Philip, and Tyler Derr. 2024. A survey on privacy in graph neural networks: Attacks, preservation, and applications. IEEE Transactions on Knowledge and

Data Engineering (2024).
[145] Xin Zheng, Yixin Liu, Shirui Pan, Miao Zhang, Di Jin, and Philip S Yu. 2022. Graph neural networks for graphs with heterophily: A survey. arXiv preprint arXiv:2202.07082 (2022).
[146] Xuanhe Zhou, Guoliang Li, Jianhua Feng, Luyang Liu, and Wei Guo. 2023. Grep: A graph learning based database partitioning system. Proceedings of the ACM on Management of Data, SIGMOD (2023).
[147] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. 2021. Federated learning on non-IID data: A survey. Neurocomputing 465 (2021), 371–390.
[148] Ligeng Zhu, Zhijian Liu, and Song Han. 2019. Deep leakage from gradients. In Advances in Neural Information Processing Systems, NeurIPS.
[149] Yinlin Zhu, Xunkai Li, Zhengyu Wu, Di Wu, Miao Hu, and Rong-Hua Li. 2024. FedTAD: Topology-aware Data-free Knowledge Distillation for Subgraph Federated Learning. arXiv preprint arXiv:2404.14061 (2024).
[150] Daniel Zügner and Stephan Günnemann. 2019. Adversarial Attacks on Graph Neural Networks via Meta Learning. In International Conference on Learning Representations, ICLR.