FedGraphNN: A Federated Learning Benchmark System for Graph Neural Networks

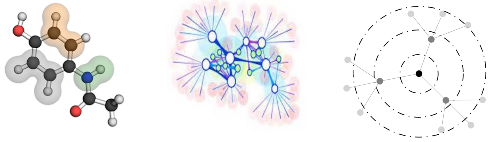
Chaoyang He[1*] , **Keshav Balasubramanian**[1*] , **Emir Ceyani**[1*] , **Carl Yang**[2] , **Han Xie**[2]
Lichao Sun[3] , **Lifang He**[3] , **Liangwei Yang**[4] , **Philip S. Yu**[4] , **Yu Rong**[5] , **Peilin Zhao**[5]
Junzhou Huang[5] , **Murali Annavaram**[1] and **Salman Avestimehr**[1]
[1]University of Southern California, [2]Emory University, [3]Lehigh University
[4]University of Illinois at Chicago, [5]Tencent AI Labs
{chaoyang.he, keshavba, ceyani}@usc.edu

Abstract

Graph Neural Networks (GNN) are state-of-the-art methods for learning distributed representations from graph-structured data. However, centralizing a massive amount of real-world graph data for GNN training is prohibitive due to privacy concerns, regulation restrictions, and commercial competitions. Federated learning (FL), a trending distributed learning paradigm, provides possibilities to solve this challenge while preserving data privacy. Despite recent advances in vision and language domains, there is no suitable platform for the FL of GNNs. Thus, we introduce FedGraphNN, an open-source FL benchmark system for federated GNNs. FedGraphNN is built on a unified formulation of graph FL and contains a wide range of datasets from different domains, popular GNN models, and FL algorithms, with secure and efficient system support. Our empirical analysis shows the utility, efficiency, and security of our benchmark system while exposing significant challenges in graph FL implying that more research efforts are needed to unravel the mystery behind federated GNNs.

1 Introduction

GNNs are state-of-the-art models that learn representations from complex graph-structured data in various domains such as drug discovery [Sun *et al.*, 2019], recommendation systems [Wu *et al.*, 2021], and traffic flow modeling [Cui *et al.*, 2019]. However, due to privacy concerns, regulatory restrictions, and commercial competition, there are widespread real-world cases in which graph data is decentralized. For example, in the AI-based drug discovery industry, pharmaceutical research institutions would significantly benefit from other institutions' data, but neither can afford to disclose their private data due to commercial reasons. FL is a distributed learning paradigm that addresses this data isolation problem. In FL, training is an act of collaboration between multiple clients without requiring centralized local data [McMahan *et al.*, 2017; Kairouz *et al.*, 2019]. Despite its successful application in domains like vision [He *et al.*, 2020a] and natural language [Hard *et al.*, 2018], FL has yet to be widely adopted for graph ML for multiple reasons:



(a) Graph-level (b) Subgraph-level (c) Node-level

Figure 1: Three settings of graph federated learning.

1. There is a lack of unified formulation over the various graph FL settings and tasks, making it difficult to understand essential challenges in federated GNNs;
2. Existing FL libraries, as summarized by [He *et al.*, 2020b], do not support diverse datasets and learning tasks to benchmark different models and training algorithms. Given the complexity of graph data, the dynamics of training federated GNNs may be different from training vision or language models [Zhang *et al.*, 2021; Xie *et al.*, 2021; He *et al.*, 2021]. A fair and easy-to-use benchmark with standardized open datasets and reference implementations is essential to the development of new graph FL models and algorithms;
3. The simulation-oriented federated training system is inefficient and unsecure for graph FL research on large-scale and private graph datasets in the cross-silo settings. Disruptive research ideas may be constrained by the lack of a modularized federated training system tailored for diverse GNN models and FL algorithms.

To address these issues, we present an open FL benchmark system for GNNs, called FedGraphNN, which contains a variety of graph datasets from different domains and eases the training and evaluation of various GNN models and FL algorithms. We first formulate graph FL to provide a unified framework for federated GNNs (Section 2). Under this formulation, we curate various graph datasets and contribute two new graph datasets, including *hERG* for drug discovery and *Tencent* for social networks, as well as their related partitioning algorithms according to real-world application scenarios (Section 3). An efficient and secure FL system is presented to support GNN models and FL algorithms and provide low-

[*]Contact Author, Equal contribution

level programmable APIs for research and industrial deployment (Section 4). Extensive empirical analysis demonstrates the utility and efficiency of our system and indicates the need of further research in graph FL (Section 5). Finally, we summarize the open challenges in graph FL based on emerging related works (Section 6) as well as future directions based on FedGraphNN (Section 7).

2 Federated Graph Neural Networks

We consider a *distributed graph scenario* in which a single graph is partitioned or multiple graphs are dispersed over multiple edge servers that cannot be centralized due to privacy or regulatory restrictions. However, collaborative training over the dispersed data can aid the formulation of more powerful and generalizable graph models. In this work, we focus on training GNNs using FL with a central-server.

In our unified framework of FedGraphNN, we assume that there are K clients in the distributed graph scenario, and the k^{th} client has its own dataset $\mathcal{D}^{(k)} := (\mathcal{G}^{(k)}, \mathbf{Y}^{(k)})$, where $\mathcal{G}^{(k)} = (\mathcal{V}^{(k)}, \mathcal{E}^{(k)})$ is the graph(s) in $\mathcal{D}^{(k)}$ with vertex and edge feature sets $\boldsymbol{X}^{(k)} = \{\boldsymbol{x}_m^{(k)}\}_{m\in\mathcal{V}^{(k)}}$ and $\boldsymbol{Z}^{(k)} = \{\boldsymbol{e}_{m,n}^{(k)}\}_{m,n\in\mathcal{V}^{(k)}}$, $\mathbf{Y}^{(k)}$ is the label set of $\mathcal{G}^{(k)}$. Each client owns a GNN model to learn graph representations and make predictions. Multiple clients are interested in collaborating through a server to improve their GNN models without necessarily revealing their graph datasets.

As illustrated in Figure 2, FedGraphNN is modeled with a Message Passing Neural Network (MPNN) framework [Gilmer *et al.*, 2017], where the forward pass has two phases: a message-passing phase and a readout phase.

GNN phase 1: Message-passing (same for all tasks). The message-passing phase contains two steps: (1) the model gathers and transforms the neighbors' messages, and (2) the model uses aggregated messages to update the nodes' hidden states. For client k and layer indices $\ell = 0, \ldots, L-1$, an L-layer MPNN is formalized as follows:

$$\boldsymbol{m}_i^{(k,\ell+1)} = \texttt{AGG}\left(\left\{\boldsymbol{M}_\theta^{(k,\ell+1)}\left(\boldsymbol{h}_i^{(k,\ell)}, \boldsymbol{h}_j^{(k,\ell)}, \boldsymbol{z}_{i,j}\right) \mid j \in \mathcal{N}_i\right\}\right),$$
$$\boldsymbol{h}_i^{(k,\ell+1)} = \boldsymbol{U}_\phi^{(k,\ell+1)}\left(\boldsymbol{h}_i^{(k,\ell)}, \boldsymbol{m}_i^{(k,\ell+1)}\right), \tag{1}$$

where $\boldsymbol{h}_i^{(k,0)} = \boldsymbol{x}_i^{(k)}$ is the k^{th} client's node features, ℓ is the layer index, $\texttt{AGG}$ is the aggregation function (e.g., in the GCN model, the aggregation function is a simple $\texttt{SUM}$ operation), $\mathcal{N}_i$ is the neighbor set of node i, and $\boldsymbol{M}_\theta^{(k,\ell+1)}(\cdot)$ is the message generation function which takes the hidden state of current node $\boldsymbol{h}_i$, the hidden state of the neighbor node $\boldsymbol{h}_j$ and the edge features $\boldsymbol{z}_{i,j}$ as inputs. $\boldsymbol{U}_\phi^{(k,\ell+1)}(\cdot)$ is the state update function receiving the aggregated feature $\boldsymbol{m}_i^{(k,\ell+1)}$.

GNN phase 2: Readout (different across tasks). After propagating through an L-layer MPNN, the readout phase computes feature vectors from the hidden states of the last MPNN layer and makes predictions for downstream tasks,

$$\hat{y}_S^{(k)} = \boldsymbol{R}_\delta\left(\left\{h_i^{(k,L)} \mid i \in \mathcal{V}_S^{(k)}\right\}\right). \tag{2}$$

Note that to handle different downstream tasks, S can be a single node (node classification), a node pair (link prediction), a node set (graph classification) and so forth, and $\boldsymbol{R}_\delta$ can be the concatenation function or a pooling function such as $\texttt{SUM}$ plus a single- or multi-layer perceptron.

GNN with FL. To formulate the FL setting, we define $\boldsymbol{W} = \{\boldsymbol{M}_\theta, \boldsymbol{U}_\phi, \boldsymbol{R}_\delta\}$ as the overall learnable weights in the GNN. Consequently, we formulate FedGraphNN as a distributed optimization problem as follows:

$$\min_{\boldsymbol{W}} F(\boldsymbol{W}) \overset{\text{def}}{=} \min_{\boldsymbol{W}} \sum_{k=1}^{K} \frac{N^{(k)}}{N} \cdot f^{(k)}(\boldsymbol{W}), \tag{3}$$

where $f^{(k)}(\boldsymbol{W}) = \frac{1}{N^{(k)}} \sum_{i=1}^{N^{(k)}} \mathcal{L}(\boldsymbol{W}; x_i^{(k)}, z_i^{(k)}, y_i^{(k)})$ is the k^{th} client's local objective function that measures the local empirical risk over the graph dataset $\mathcal{D}^{(k)}$ with $N^{(k)}$ data samples. $F(\boldsymbol{W})$ is the loss function of the global GNN model. To solve this problem, federated optimizers such as FedAvg [McMahan *et al.*, 2017] can be used. It is important to note here that in FedAvg, the aggregation function on the server merely averages model parameters. We use GNNs inductively. Thus, no topological information about graphs on any client is required on the server during parameter aggregation. We refer to Appendix A for more details on the supported GNN baselines.

Under the unified framework of FedGraphNN, we organize various distributed graph scenarios motivated by real-world applications into three settings based on how the graphs are distributed across silos, and provide support to the corresponding typical tasks in each setting.

- **Graph-level FedGraphNN:** Each client holds a set of graphs, where the typical task is graph classification. Real-world scenarios may include molecular trials [Rong *et al.*, 2020] and protein discovery [Yang *et al.*, 2018], where each institute holds a limited set of graphs with ground-truth labels due to expensive experiments.
- **Subgraph-level FedGraphNN:** Each client holds a subgraph of a larger global graph, where the typical task is node classification and link prediction. Real-world scenarios include recommendation systems [Yang *et al.*, 2021], knowledge graph completion [Chen *et al.*, 2020] and so forth, where each institute holds a subset of user-item interaction data or entity/relation data.
- **Node-level FedGraphNN:** Each client owns ego-networks of one or multiple nodes, where the task is node classification or link prediction. Real-world scenarios include social networks [Zhou *et al.*, 2008] where each node only sees its k-hop neighbors and their connections.

3 Datasets

FedGraphNN is centered around three federated GNN settings based on how the graph data is distributed in real-world scenarios, which covers a broad range of domains, tasks and challenges of graph FL. Specifically, it includes 36 datasets from 7 domains, such as molecules, proteins, knowledge graphs, recommendation systems, citation networks and social networks. Here, to facilitate clear understanding over the various graph FL settings, we introduce examples of real-world datasets in each of the three federated GNN settings.

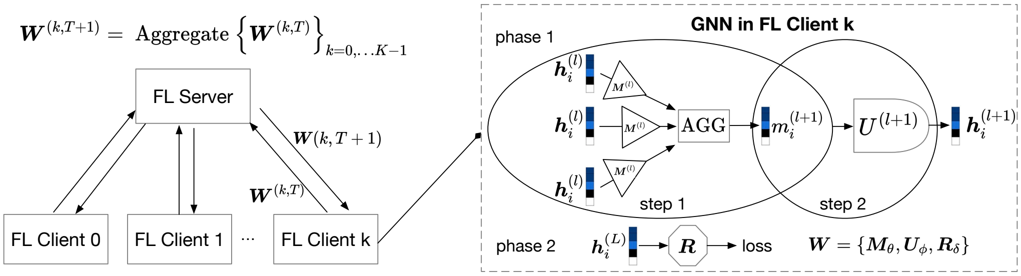


Figure 2: Formulation of FedGraphNN (Federated Graph Neural Network)

Exact sources and statistics are provided in Table 1, while more details and references are provided in Appendix B.

- **Graph-level Setting:** In the real world, biomedical institutions might hold their own set of graphs such as molecules and proteins, and social network companies might hold their own set of community graphs. Such graphs may constitute large and diverse datasets for GNN traning, but they cannot be directly shared across silos. To simulate such scenarios, we utilize datasets from the domains of molecular ML [Wu *et al.*, 2018], bioinformatics [Borgwardt *et al.*, 2005] and social computing [Yanardag and Vishwanathan, 2015], We also introduce a new large-scale dataset, called hERG [Gaulton *et al.*, 2016] for federated drug discovery.
- **Subgraph-level Setting:** The first realistic scenario of subgraph-level FL is recommendation systems, where the users can interact with items owned by different shops or sectors, which makes each data owner only holding a part of the global user-item graph. To simulate such scenarios, we use recommendation datasets from both publicly available sources [Tang *et al.*, 2012; Richardson *et al.*, 2003] and internal sources [He *et al.*, 2019], which have high-quality meta-data information. Another realistic scenario is knowledge graphs, where different organizations or departments might only have a subset of the entire knowledge, due to the focus in particular domains. We integrate the FB15k-237 [Dettmers *et al.*, 2018], WN18RR [Toutanova and Chen, 2015] and YAGO3-10 [Mahdisoltani *et al.*, 2013] datasets, where subgraphs can be build based on relation types to distinguish specialized fields or communities to distinguish the entities of focus.
- **Node-level Setting:** In social networks, each user's personal data can be sensitive and only visible to his/her k-hop neighbors (e.g., in Instagram, $k = 1$ for contents and $k = 2$ for links, of private accounts). Thus, it is natural to consider node-level FL in social networks with clients holding the user ego-networks. To simulate this scenario, we use the open social networks [Shchur *et al.*, 2019] and publication networks [McCallum *et al.*, 2000; Bojchevski and Günnemann, 2018; Giles *et al.*, 1998; Sen *et al.*, 2008; Tang *et al.*, 2008] and partition them into sets of ego-networks.

In terms of graph mining tasks, FedGraphNN supports all three common tasks of graph classification, node classification and link prediction. Some tasks are naturally important in certain graph FL settings while others are not, which we also clarify with real examples as follows:

- **Graph Classification:** This task is to categorize different types of graphs based on their structure and overall information. Unlike other tasks, this requires to characterize the property of the entire input graph. This task is naturally important in graph-level FL, with real examples such as molecule property prediction, protein function prediction, and social community classification.
- **Link Prediction:** This task is to estimate the probability of links between any two nodes in a graph. It is important in the subgraph-level FL, for example, in recommendation systems and knowledge graphs, where link probabilities are predicted in the former, and relation types are predicted in the latter. It is less likely but still viable in the node-level setting, where friend suggestion and social relation profiling can be attempted in users' ego-networks.
- **Node Classification:** This task is to predict the labels of individual nodes in graphs. It is more important in node-level FL, such as predicting the habits of a user based on his/her k-hop friends. It might also be important in subgraph-level FL, such as the collaborative prediction of disease infections based on the patient networks dispersed in multiple healthcare facilities.

Data sources. We have collected 36 datasets from 7 domains and planning to continually enrich the available datasets in the future. Among them, 34 are from publicly available sources such as MoleculeNet [Wu *et al.*, 2018] and graph kernels datasets [Borgwardt *et al.*, 2005]. In addition, we introduce two new de-identified datasets: *hERG* [Kim *et al.*, 2021; Gaulton *et al.*, 2017], a graph dataset for classifying protein molecules responsible for cardiac toxicity and *Tencent* [He *et al.*, 2019], a large bipartite graph representing the relationships between users and groups. More details(and their sources) can be found in Appendices B.1 & B.2.

3.1 Generating Federated Learning Datasets

Non-I.I.D-ness in FL is an astonishing challenge in realistic FL simulations. Coupled with the persistent structure and feature heterogeneity of graphs [Yang *et al.*, 2020; Xie *et al.*, 2021], distinguishing the sources of skewness in federated GNNs remains as an open problem. Here, we present two ways of injecting reproducible non-I.I.D.-ness (detailed in Appendix B.3): *Dirichlet Sampling*[1] for data-invariant splits via Dirichlet distribution [Hsu *et al.*, 2019] and *Meta-Data Based Sampling* [2] for meta-data specific splits.

[1] This method is utilized for graph-level and node-level tasks only.

[2] This type of sampling can be used for any FedGraphNN tasks.

Table 1: Summary of open graph datasets from various domains contained in FedGraphNN.

Task-Level	Category	Datasets	# Graphs	Avg. # Nodes	Avg. # Edges	Avg. Degree	# Classes
Graph-Level	Molecules	BACE [Subramanian *et al.*, 2016]	1513	34.12	36.89	2.16	2
		HIV [Riesen and Bunke, 2008]	41127	25.53	27.48	2.15	2
		MUV [Rohrer and Baumann, 2009]	93087	24.23	26.28	2.17	17
		Clintox [Gayvert *et al.*, 2016]	1478	26.13	27.86	2.13	2
		SIDER [Kuhn *et al.*, 2016]	1427	33.64	35.36	2.10	27
		Toxcast [Richard *et al.*, 2016]	8575	18.78	19.26	2.05	167
		Tox21 [tox, 2017]	7831	18.51	25.94	2.80	12
		BBBP [Martins *et al.*, 2012]	2039	24.05	25.94	2.16	2
		QM9 [Gaulton *et al.*, 2012]	133885	8.8	27.6	6.27	1
		ESOL [Delaney, 2004]	1128	13.29	40.65	6.11	1
		FreeSolv [Mobley and Guthrie, 2014]	642	8.72	25.6	5.87	1
		Lipophilicity [Gaulton *et al.*, 2012]	4200	27.04	86.04	6.36	1
		hERG [Gaulton *et al.*, 2016]	10572	29.39	94.09	6.40	1
		MUTAG [Debnath *et al.*, 1991]	188	17.93	19.79	2.21	2
		NCI1 [Wale *et al.*, 2008]	4110	29.87	32.3	2.16	2
	Proteins	PROTEINS [Borgwardt *et al.*, 2005]	1113	39.06	72.82	3.73	2
		DDI [Segura Bedmar *et al.*, 2013]	1178	284.32	715.66	5.03	2
		PPI [Hamilton *et al.*, 2017]	24	56,944	818,716	28.76	121
	Social networks	COLLAB [Yanardag and Vishwanathan, 2015]	5000	74.49	2457.78	65.99	3
		REDDIT-B [Yanardag and Vishwanathan, 2015]	2000	429.63	497.75	2.32	2
		REDDIT-M-5K [Yanardag and Vishwanathan, 2015]	4999	508.52	594.87	2.34	5
		IMDB-B [Yanardag and Vishwanathan, 2015]	1000	19.77	96.53	9.77	2
		IMDB-M [Yanardag and Vishwanathan, 2015]	1500	13	65.94	10.14	3
Subgraph-Level	Recomm. systems	Ciao [Tang *et al.*, 2012]	28	5150.93	19280.93	3.74	5
		Epinions [Richardson *et al.*, 2003]	27	15824.22	66420.52	4.20	5
		Tencent [He *et al.*, 2019]	1	709074	991713	2.80	2
	Knowledge graphs	FB15k-237 [Dettmers *et al.*, 2018]	1	14505	212110	14.62	237
		WN18RR [Toutanova and Chen, 2015]	1	40559	71839	1.77	11
		YAGO3-10 [Mahdisoltani *et al.*, 2013]	1	123143	774182	6.29	37
Node-level	Publication networks	CORA [McCallum *et al.*, 2000]	1	2708	5429	2.00	7
		CORA-full [Bojchevski and Günnemann, 2018]	1	19793	65311	3.30	70
		CITESEER [Giles *et al.*, 1998]	1	4230	5358	1.27	6
		PUBMED [Sen *et al.*, 2008]	1	19717	44338	2.25	3
		DBLP [Tang *et al.*, 2008]	1	17716	105734	5.97	4
	Social networks	CS [Shchur *et al.*, 2019]	1	18333	81894	4.47	15
		Physics [Shchur *et al.*, 2019]	1	34493	247962	7.19	5

Dirichlet Sampling: We generate a heterogeneous partition into J clients by sampling $p_k \sim \text{Dir}_J(\alpha)$ and allocating a $p_{k,j}$ proportion of the training instances of class k to a local client. The parameter α controls the I.I.D.'ness of the sample distribution. Lower the α, more non-I.I.D. the sample distribution is. Figures 5 & 6 depict several datasets' non-I.I.D. distributions generated this method for graph-level tasks. The alpha values for LDA for representative datasets can be found in Table 2 and 20 in the Appendix E.3.

Meta-Data Based Sampling: When users have enough information on how some features affect non-I.I.D.'ness of the data in a particular data domain, FedGraphNN allows data-partitioning based on meta-data information. For example, in recommendation systems, a user's behavior is different for items from different categories [Cho *et al.*, 2013]. Splitting the whole user-item bi-partite graph based on item categories captures the non-uniform behaviour difference among categories. Another example is for knowledge graphs with two possible settings: Building sub-graphs from different relation types and from node communities. Figure 8 in the Appendix shows the non-I.I.D. distribution of user's rating number on an item from different categories.

4 Efficient, Secure, and Modularized Benchmark System Design

FedGraphNN is tailored for benchmarking and developing graph FL models with three unique principles.

Enhancing realistic evaluation with efficient and deployable distributed system design. We design the training system to support realistic distributed computing in multiple edge servers, given that FedGraphNN is mainly executed in the cross-silo settings where each FL client represents an edge server belonging to an organization rather than smartphone or IoT devices. The system architecture, shown in Figure 3, is composed of three layers: `FedML-core` layer, `FedML-API` layer, and `Application` layer. `FedML-core` layer supports both RPC (remote procedure call) and MPI (message passing interface), which enable communication among edge servers located at different data centers. The communication primitives are wrapped as abstract communication APIs (i.e., `ComManager` in Figure 3) to simplify the message definition and passing requested by different FL algorithms in `FedML-API` layer (see details in Appendix C).

With this design, researchers can run realistic evaluations in a parallel computing environment where multiple CPU/GPU servers located in multiple organizations (e.g., edge servers in AWS EC2). The training can be finished in only a few minutes for medium and small-scale graph datasets. Scaling deployment to numerous FL clients with heterogeneous hardware and OS configuration is further simplified by Docker-based deployment.

Enabling secure benchmarking with lightweight secure aggregation. Industrial FL applications may require private customer datasets with other organizations. However, model weights from clients may still have the risk of privacy leakage [Zhu and Han, 2020]. As such, legal and regulatory departments normally do not permit FL research on private customer datasets when strong security is not guaranteed. To break this barrier, we integrate baseline secure aggregation algorithms, refer to Appendix D.2), within `FedML-core` and `FedML-API` in the system architecture, as shown in Figure 3.

Facilitating algorithmic innovations with diverse datasets, GNN models, and FL algorithms. FedGraphNN provides flexible customizations thanks to its modularity. The method of defining the model and related trainer is kept the same as in

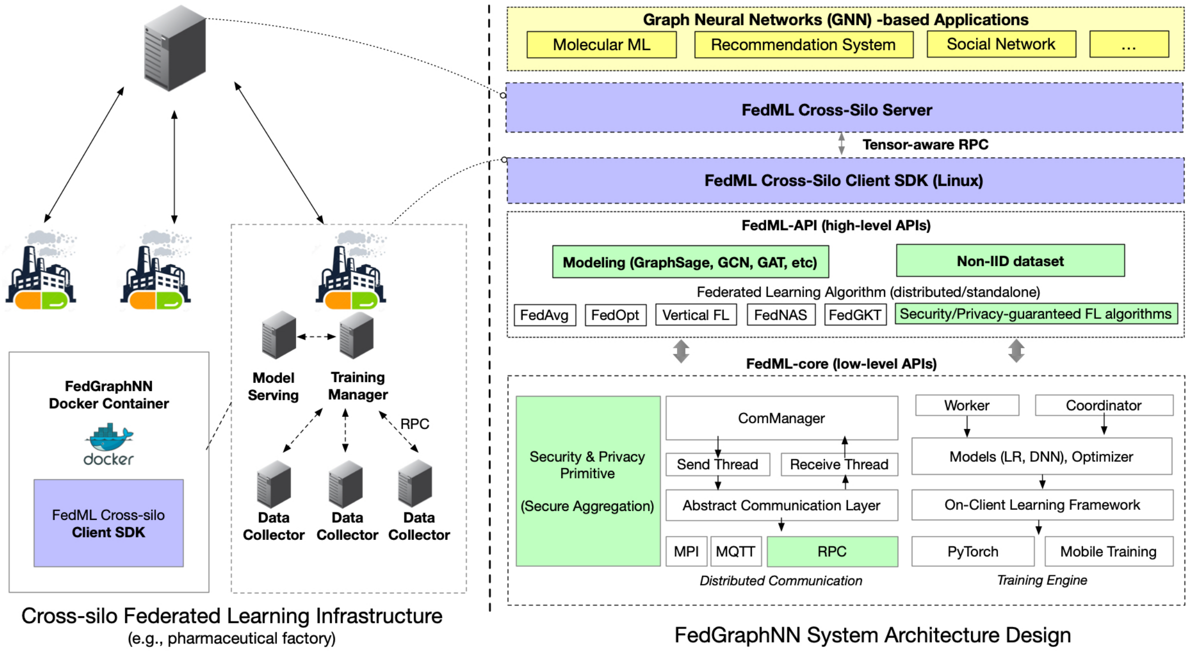

Graph Neural Networks (GNN) -based Applications
Molecular ML
Recommendation System
Social Network
...
FedML Cross-Silo Server
Tensor-aware RPC
FedML Cross-Silo Client SDK (Linux)
FedML-API (high-level APIs)
Modeling (GraphSage, GCN, GAT, etc)
Non-IID dataset
Federated Learning Algorithm (distributed/standalone)
FedAvg
FedOpt
Vertical FL
FedNAS
FedGKT
Security/Privacy-guaranteed FL algorithms
FedML-core (low-level APIs)
Security & Privacy Primitive
(Secure Aggregation)
ComManager
Send Thread
Receive Thread
Abstract Communication Layer
MPI
MQTT
RPC
Distributed Communication
Worker
Coordinator
Models (LR, DNN), Optimizer
On-Client Learning Framework
PyTorch
Mobile Training
Training Engine
FedGraphNN Docker Container
docker
FedML Cross-silo Client SDK
Model Serving
Training Manager
RPC
Data Collector
Data Collector
Data Collector
Cross-silo Federated Learning Infrastructure
(e.g., pharmaceutical factory)
FedGraphNN System Architecture Design

Figure 3: Overview of FedGraphNN System Architecture Design

centralized training to reduce the difficulty of developing the distributed training framework. For new FL algorithm development, worker-oriented programming reduces the difficulty of message passing and definition (details can be found in the Appendix C). The user-oriented interface (main training script) is simplified as the example code shown in Figure 4 where a few lines of code can launch a federated training in a cross-silo cloud environment.

5 Empirical Analysis

5.1 Experimental Setup

Our experiments are conducted on multiple GPU servers each equipped with 8 NVIDIA Quadro RTX 5000 (16GB GPU memory). The hyper-parameters are selected via our built-in efficient parameter sweeping functionalities from the ranges listed in Appendix E.1. We present results on the ROC-AUC metric for graph classification and RMSE & MAE for graph regression, MAE, MSE, and RMSE for link prediction, and micro-F1 for node classification. More evaluation metrics are presented in Appendix E.2.

5.2 Baseline Performance Analysis

We report results of several popular GNN models trained with FedAvg, to examplify the utility of FedGraphNN. More results with varying baselines, hyper-parameters, evaluation metrics and visualizations are presented in Appendix E.3. After hyper-parameter tuning, we present the main performance results as well as runtimes in Tables 2, 3 and 4.

Besides showcasing the utility of FedGraphNN, there are multiple takeaways from these results:

1. When the graph datasets are small, FL accuracy is often on par with centralized learning.
2. When dataset sizes and numbers of clients grow, GNN accuracy in the FL setting becomes significantly worse

Table 2: Performance of graph classification in the graph-level FL setting (#clients=4).

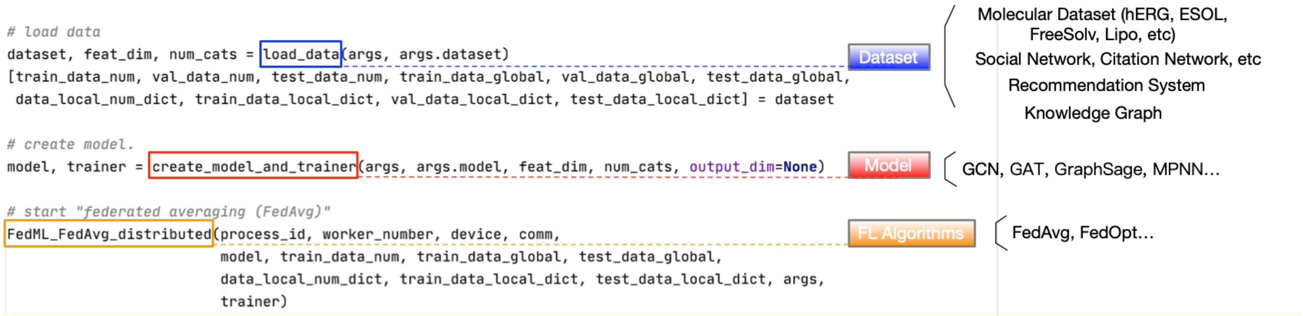
Metric	ROC-AUC					Training Time (sec.)				
Method	SIDER $\alpha = 0.2$	BACE $\alpha = 0.5$	CLINTOX $\alpha = 0.5$	BBBP $\alpha = 2$	TOX21 $\alpha = 3$	SIDER	BACE	CLINTOX	BBBP	TOX21
MoleculeNet Results	0.6380	0.8060	0.8320	0.6900	0.8290	Not Published				
GCN (Centralized)	0.6476	0.7657	0.8914	0.8705	0.7800	458	545	686	532	1034
GCN (FedAvg)	0.6266	0.6594	0.8784	0.7629	0.7128	358	297	280	253	903
GAT (Centralized)	0.6639	0.9221	0.9573	0.8824	0.8144	739	603	678	533	2045
GAT (FedAvg)	0.6591	0.7714	0.9129	0.8746	0.7186	528	327	457	328	1549
GraphSAGE (Centralized)	0.6669	0.9266	0.9716	0.8930	0.8317	193	327	403	312	1132
GraphSAGE (FedAvg)	0.6700	0.8604	0.9246	0.8935	0.7801	127	238	282	206	771

Table 3: Performance of link prediction in the subgraph-level FL setting (#clients = 8).

Metric	MAE		MSE		RMSE		Training Time (sec.)	
DataSet	CIAO	EPINIONS	CIAO	EPINIONS	CIAO	EPINIONS	CIAO	EPINIONS
GCN (Centralized)	0.8167	0.8847	1.1184	1.3733	1.0575	1.1718	268	650
GCN (FedAvg)	0.7995	0.9033	1.0667	1.4378	1.0293	1.1924	352	717
GAT (Centralized)	0.8214	0.8934	1.1318	1.3873	1.0639	1.1767	329	720
GAT (FedAvg)	0.7987	0.9032	1.0682	1.4248	1.0311	1.1882	350	749
GraphSAGE (Centralized)	0.8231	1.0436	1.1541	1.8454	1.0742	1.3554	353	721
GraphSAGE (FedAvg)	0.8290	0.9816	1.1320	1.6136	1.0626	1.2625	551	810

Table 4: Performance of Node classification in the node-level FL setting (#clients = 10).

Metric	micro F1				Training Time (sec.)			
Method	CORA	CITESEER	PUBMED	DBLP	CORA	CITESEER	PUBMED	DBLP
GCN (Centralized)	0.8622	0.9820	0.9268	0.9294	1456	742	1071	1116
GCN (FedAvg)	0.8549	0.9743	0.9128	0.9088	833	622	654	653
GAT (Centralized)	diverge	0.9653	0.8621	0.8308	1206	1765	1305	957
GAT (FedAvg)		0.9610	0.8557	0.8201	871	652	682	712
GraphSAGE (Centralized)	0.9692	0.9897	0.9724	0.9798	1348	934	692	993
GraphSAGE (FedAvg)	0.9749	0.9854	0.9761	0.9749	774	562	622	592



```
# load data
dataset, feat_dim, num_cats = load_data(args, args.dataset)
[train_data_num, val_data_num, test_data_num, train_data_global, val_data_global, test_data_global,
 data_local_num_dict, train_data_local_dict, val_data_local_dict, test_data_local_dict] = dataset

# create model.
model, trainer = create_model_and_trainer(args, args.model, feat_dim, num_cats, output_dim=None)

# start "federated averaging (FedAvg)"
FedML_FedAvg_distributed(process_id, worker_number, device, comm,
                         model, train_data_num, train_data_global, test_data_global,
                         data_local_num_dict, train_data_local_dict, test_data_local_dict, args,
                         trainer)
```

Dataset
Molecular Dataset (hERG, ESOL, FreeSolv, Lipo, etc)
Social Network, Citation Network, etc
Recommendation System
Knowledge Graph
Model
GCN, GAT, GraphSage, MPNN...
FL Algorithms
FedAvg, FedOpt...

Figure 4: Example code for benchmark evaluation with FedGraphNN

than centralized learning. We conjecture that such accuracy drop is because the basic GNN models and FL algorithms cannot properly quantify multiple sources of non-I.I.D.ness in graphs properly.

3. The dynamics of training federated GNNs are different from training federated vision or language models. Our findings show that the best model in the centralized setting may not necessarily be the best model in the FL setting.

4. Counterintuitive phenomenons (highlights in above tables) further add to the mystery of federated graph neural networks: in graph-level experiments, GAT suffers the most performance compromise on 5 out of 9 datasets; in both subgraph-level and node-level FL, results on some datasets (CIAO, CORA, PubMed) may even have slightly higher performance than centralized training; GAT cannot achieve reasonable accuracy in node-level FL (e.g., in CORA dataset), etc.

These results indicate the limitations of the baselines in FedGraphNN and motivate further research in understanding the nuances of graph FL. For evaluations on efficiency and security, we refer to Appendices D.1 & D.2.

6 Related Works and Open Challenges

FedGraphNN lies at the intersection of GNNs and FL. We first discuss related works under the umbrella of three different graph FL settings. (1) *Graph-level* : we believe molecular ML is a paramount application in this setting, where many small graphs are distributed between multiple institutions, as demonstrated in [He *et al.*, 2021; Xie *et al.*, 2021]. [Xie *et al.*, 2021] proposes a clustered FL framework specifically for GNNs to deal with feature and structure heterogeneity. [He *et al.*, 2021] develops a multi-task learning framework suitable to train federated graph-level GNNs without the need for a central server. (2) *Subgraph-level*: this scenario typically pertains to the entire social networks, recommender networks or knowledge graphs that need to be partitioned into many smaller subgraphs due to data barriers between different departments in a giant company or data platforms with different domain focuses as demonstrated in [Wu *et al.*, 2021; Zhang *et al.*, 2021]. [Wu *et al.*, 2021] proposes a federated recommendation system with GNNs, whereas [Zhang *et al.*, 2021] proposes FedSage, a subgraph-level federated GNN generating psuedo-neighbors utilizing variational graph autoencoder. (3) *Node-level*: when the privacy of specific nodes in a graph is important, node-level graph FL is useful in practice. The IoT setting is a good example [Zheng *et al.*, 2020]; [Wang *et al.*, 2020] uses a hybrid method of FL and meta-learning to solve the semi-supervised graph node classification problem in decentralized social network datasets; [Meng *et al.*, 2021] attempts to protect the node-level privacy using an edge-cloud partitioned GNN model for spatio-temporal forecasting tasks using node-level traffic sensor datasets.

Before FedGraphNN, there was a serious lack of standardized datasets and baselines for training federated GNNs. Previous platforms like LEAF [Caldas *et al.*, 2019], TFF, and PySyft [Ryffel *et al.*, 2018] have no support on GNNs. Beyond the direct goals of FedGraphNN, many open algorithmic challenges in graph FL remain be studied. First, the partitioning of a large graph into sub-graphs or ego-networks into local clients introduce dataset bias and information loss in terms of missing cross-subgraph links, which can impede the GNN performance in the FL setting (as shown in Tables 3 & 4), and motivates us to study the proper recovery of such missing cross-subgraph links [Zhang *et al.*, 2021]. Second, as observed from Tables 2–4, the non-I.I.D.ness in distributed graph datasets can impact the gap between federated and centralized GNNs, which motivates us to study the deep decoupling and quantification of the multiple sources of non-I.I.D.-ness in graphs are towards the appropriate design of graph FL algorithms [Xie *et al.*, 2021]. Third, integrating both graph topology of GNNs and network topology of FL in a principled and efficient way is of great interest when the federated GNNs can be trained in an asynchronous fashion. Finally, a universally useful framework for secure graph FL, despite various privacy-preserving methods [Zheng *et al.*, 2020] in literature, is still missing.

7 Conclusion and Future Works

In this work, we design an FL system and benchmark for GNNs, named FedGraphNN, which includes open datasets, baseline implementations, programmable APIs, all integrated in a robust and an affordable system. We hope FedGraphNN can serve as an easy-to-use research platform for researchers to explore vital problems at the intersection of FL and GNNs.

Here we highlight some future improvements and research directions based on our FedGraphNN system: 1. supporting more graph datasets and GNN models for diverse applications. 2. optimizing the system to further accelerate the training speed for large graphs; 3. designing advanced graph FL algorithms to mitigate the accuracy gap on datasets with non-

I.I.D.ness; 4. exploring label-efficient GNN models based on concepts such as meta-learning and self-supervision to exploit the graphs in each client and their collaboration; 5. addressing challenges in security and privacy under the setting of graph FL 6. proposing efficient compression algorithms that adapt to the level of compression to the available bandwidth of the users while preserving the privacy of users' local data; 7. organizing data competitions, themed workshops, special issues, etc., on the dissemination of FedGraphNN; 8. actively discussing ethics and societal impacts to avoid unwanted negative effects.

References

[Caldas *et al.*, 2019] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings, 2019.

[Chen *et al.*, 2020] Chaochao Chen, Jamie Cui, Guanfeng Liu, Jia Wu, and Li Wang. Survey and open problems in privacy preserving knowledge graph: Merging, query, representation, completion and applications. *arXiv preprint arXiv:2011.10180*, 2020.

[Cui *et al.*, 2019] Zhiyong Cui, Kristian Henrickson, Ruimin Ke, Ziyuan Pu, and Yinhai Wang. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting, 2019.

[Gilmer *et al.*, 2017] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*, pages 1263–1272. PMLR, 2017.

[Hard *et al.*, 2018] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

[He *et al.*, 2020a] Chaoyang He, Murali Annavaram, and Salman Avestimehr. Group knowledge transfer: Federated learning of large cnns at the edge. *Advances in Neural Information Processing Systems*, 33, 2020.

[He *et al.*, 2020b] Chaoyang He, Songze Li, Jinhyun So, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, Li Shen, Peilin Zhao, Yan Kang, Yang Liu, Ramesh Raskar, Qiang Yang, Murali Annavaram, and Salman Avestimehr. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

[He *et al.*, 2021] Chaoyang He, Emir Ceyani, Keshav Balasubramanian, Murali Annavaram, and Salman Avestimehr. Spreadgnn: Serverless multi-task federated learning for graph neural networks, 2021.

[Hsu *et al.*, 2019] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[Kairouz *et al.*, 2019] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

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

[Meng *et al.*, 2021] Chuizheng Meng, Sirisha Rambhatla, and Yan Liu. Cross-node federated graph neural network for spatio-temporal data modeling, 2021.

[Rong *et al.*, 2020] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data, 2020.

[Ryffel *et al.*, 2018] Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. A generic framework for privacy preserving deep learning, 2018.

[Sun *et al.*, 2019] Mengying Sun, Sendong Zhao, Coryandar Gilvary, Olivier Elemento, Jiayu Zhou, and Fei Wang. Graph convolutional networks for computational drug development and discovery. *Briefings in Bioinformatics*, 21(3):919–935, 06 2019.

[Wang *et al.*, 2020] Binghui Wang, Ang Li, Hai Li, and Yiran Chen. Graphfl: A federated learning framework for semi-supervised node classification on graphs. *arXiv preprint arXiv:2012.04187*, 2020.

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

[Xie *et al.*, 2021] Han Xie, Jing Ma, Li Xiong, and Carl Yang. Federated graph classification over non-iid graphs, 2021.

[Yang *et al.*, 2018] Kevin K Yang, Zachary Wu, Claire N Bedbrook, and Frances H Arnold. Learned protein embeddings for machine learning. *Bioinformatics*, 34(15):2642–2648, 2018.

[Yang *et al.*, 2020] Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. Heterogeneous network representation learning: A unified framework with survey and benchmark. In *TKDE*, 2020.

[Yang *et al.*, 2021] Liangwei Yang, Zhiwei Liu, Yingtong Dou, Jing Ma, and Philip S Yu. Consisrec: Enhancing gnn for social recommendation via consistent neighbor aggregation. *arXiv preprint arXiv:2105.02254*, 2021.

[Zhang *et al.*, 2021] Ke Zhang, Carl Yang, Xiaoxiao Li, Lichao Sun, and Siu Ming Yiu. Subgraph federated learning with missing neighbor generation, 2021.

[Zheng *et al.*, 2020] Longfei Zheng, Jun Zhou, Chaochao Chen, Bingzhe Wu, Li Wang, and Benyu Zhang. Asfgnn: Automated separated-federated graph neural network. *arXiv preprint arXiv:2011.03248*, 2020.

[Zhou *et al.*, 2008] Bin Zhou, Jian Pei, and WoShun Luk. A brief survey on anonymization techniques for privacy preserving publishing of social network data. *ACM Sigkdd Explorations Newsletter*, 10(2):12–22, 2008.

[Zhu and Han, 2020] Ligeng Zhu and Song Han. Deep leakage from gradients. In *Federated learning*, pages 17–31. Springer, 2020.

A More Details of the Supported Graph Neural Network Architectures

- **Graph Convolutional Networks** [Kipf and Welling, 2016] is a GNN model which is a 1^{st} order approximation to spectral GNN models. [Markowitz *et al.*, 2021]
- **GraphSAGE** [Hamilton *et al.*, 2017] is a general inductive GNN framework capable of generating node-level representations for unseen data.
- **Graph Attention Networks** [Veličković *et al.*, 2018] is the first attention-based GNN model. Attention is computed in a message-passing fashion.
- **Simplifying Graph Convolutional Networks** [Wu *et al.*, 2019] is the first attention-based GNN model. Attention is computed in a message-passing fashion.
- **Graph Isomorphism Networks** [Xu *et al.*, 2019] is the SotA method on graph classification showing that GNNs are strong at most a 1-Weisfeiler Lehman test.

B More Details of the Open Datasets

B.1 Data Sources

The details of each dataset are listed below:

Datasets for Graph-level FedGraphNN

- `BBBP` [Martins *et al.*, 2012] involves records of whether a compound carries the permeability property of penetrating the blood-brain barrier.
- `SIDER` [Kuhn *et al.*, 2016], or Side Effect Resource, the dataset consists of marketed drugs with their adverse drug reactions. The available
- `ClinTox` [Gayvert *et al.*, 2016] includes qualitative data of drugs both approved by the FDA and rejected due to the toxicity shown during clinical trials.
- `BACE` [Subramanian *et al.*, 2016] is collected for recording compounds that could act as the inhibitors of human β-secretase 1 (BACE-1) in the past few years.
- `Tox21`[tox, 2017] is a dataset which records the toxicity of compounds.
- `hERG`[Kim *et al.*, 2021; Gaulton *et al.*, 2017] is a dataset that records the gene (KCNH2) that codes for a protein known as Kv11.1 responsible for its contribution to the electrical activity of the heart to help the coordination of the heart's beating.
- `QM9` [Ramakrishnan *et al.*, 2014] is a subset of GDB-13, which records the computed atomization energies of stable and synthetically accessible organic molecules, such as HOMO/LUMO, atomization energy, etc. It contains various molecular structures such as triple bonds, cycles, amide, and epoxy.
- `ESOL` [Delaney, 2004] is a small dataset documenting the water solubility(log solubility in mols per litre) for common organic small molecules.
- `Lipophilicity` [Gaulton *et al.*, 2012] which records the experimental results of octanol/water distribution coefficient for compounds.
- `FreeSolv` [Mobley and Guthrie, 2014] contains the experimental results of hydration-free energy of small molecules in water.
- `MUTAG` [Debnath *et al.*, 1991] is a collection of nitroaromatic molecules and the aim is to classify their mutagenicity on Salmonella typhimurium. Input graphs are used to represent chemical compounds, where vertices stand for atoms and are labeled by the atom type (represented by one-hot encoding), while edges between vertices represent bonds between the corresponding atoms. It includes 188 samples of chemical compounds with 7 discrete node labels.
- `PROTEINS`[Borgwardt *et al.*, 2005] is a dataset of protein molecules and goal is to classify whether a protein is an enzyme or not. Nodes represent the amino acids and two nodes are connected by an edge if they are less than 6 Angstroms apart.
- `NCI1`[Wale *et al.*, 2008] is a dataset where each input graph represents a chemical compound in which each vertex is an atom of the molecule(encoded via 1-hot vector), and edges between vertices represent bonds between atoms. This dataset is used to detect whether a chemical is responsible for cell lung cancer or not.
- `DDI`[Segura Bedmar *et al.*, 2013] is a drug-drug interactions as well as documents describing drug-drug interactions from the DrugBank database.
- `COLLAB`[Yanardag and Vishwanathan, 2015] is a scientific collaboration dataset consisting of researchers' ego networks, the graph representation of a researcher and his/her collaborators' collaborations. These graphs have three possible labels to distinguish researchers' field: High Energy Physics, Condensed Matter Physics, and Astro Physics.

- IMDB-B & IMDB-M[Yanardag and Vishwanathan, 2015] are movie collaboration datasets consisting of ego-networks of 1,000 actors/actresses featured in IMDB. In each graph, nodes represent actors/actresses, and there is an edge between them if they appear in the same movie. The main difference is that the latter one has more than 2 categories.
- REDDIT-B & REDDIT-M-5K[Yanardag and Vishwanathan, 2015] are relational datasets of graphs describing online discussions on Reddit where nodes represent users, and there is an edge between them if at least one of them respond to the other's comment. The only difference is that graphs in REDDIT-B labeled according to whether it belongs to a question/answer-based community or not. There are multiple categories inside REDDIT-B.

Datasets for Subgraph-level FedGraphNN Our focus is for recommendation systems. A node in the graph represent a user or an item while the edge weight represents the rating score from user to item. Items are also assigned to different categories based on their characteristics. Each item belongs to at least one category.

- Ciao [Tang *et al.*, 2012] dataset contains rating information of users given to items, and also contain item category information.
- Epinions [Richardson *et al.*, 2003] dataset is trust network dataset containing profile, ratings and trust relations of a user as triplet for each user in the network. For each rating, it has the product name and its category, the rating score, the time point when the rating is created, and the helpfulness of this rating.
- Tencent [He *et al.*, 2019] dataset is a large bipartite graph representing the relation between users and groups. An edge indicates the user belongs to the connected group. The data also contains all the node features and node labels.
- FB15k-237 [Dettmers *et al.*, 2018] contains knowledge base relation triples and textual mentions of Freebase entity pairs.
- WN18RR [Toutanova and Chen, 2015] is a link prediction dataset created from WordNet containing 93,003 triplets with 11 different relations of 40,943 entries.
- YAGO3-10 [Mahdisoltani *et al.*, 2013] or known as Yet Another Great Ontology 3-10, is a subset of YAGO, well-known benchmark dataset for knowledge base completion. Contains entities related to at least ten different relations. Triplets describe atributes like citizenship, profession, salary.

Datasets for Node-level FedGraphNN

- CORA[McCallum *et al.*, 2000] dataset is a citation network formed of 2708 scientific publications with 5429 links classified into one of seven classes. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 1433 unique words.
- CORA-full[Bojchevski and Günnemann, 2018] dataset is an additional version of cora [McCallum *et al.*, 2000] which is extracted from the original entire network. It consists of 19,793 scientific publications with 65,311 links classified into one of 70 classes. The node features are represented by one-hot vectors indicating the absence/presence of words.
- CITESEER[Giles *et al.*, 1998] is a citation network formed from 3312 scientific publications with 4732 links classified into one of six classes. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 3703 unique words.
- PUBMED[Sen *et al.*, 2008] dataset consists of 19717 scientific publications with 44338 links from PubMed database pertaining to diabetes classified into one of three classes. Each publication in the dataset is described by a TF/IDF weighted word vector from a dictionary which consists of 500 unique words.
- DBLP[Tang *et al.*, 2008] is a citation network dataset where it is extracted from DBLP, ACM, MAG, and other sources. The first version contains 629,814 papers and 632,752 citations. Each paper is associated with abstract, authors, year, venue, and title. The data set can be used for clustering with network and side information, studying influence in the citation network, finding the most influential papers, topic modeling analysis, etc.
- Coauthor-CS[Shchur *et al.*, 2019] is a coauthor network focusing on the area of computer science, in which nodes (18,333) represent authors and links (81, 894) indicate that the connected authors present on the same paper. The features of nodes represent paper keywords for each author's papers, and the labels of nodes (15) indicate the most active study area of authors.
- Coauthor-Physics[Shchur *et al.*, 2019] is a coauthor network focusing on the area of physics which has the same representations of nodes, links, and labels as Coauthor-CS. It consists of 34,493 nodes and 247,962 links, and nodes are classified into one of 5 classes.

B.2 Data Preprocessing

Dataset Splitting. Before generating non-I.I.D.'ness for our datasets, we partition all datasets such that 80% training, 10% validation, and 10% test. This ratio can be modified and as future work, we plan to support domain-specific splits such as scaffold splitting [Bemis and Murcko, 1996] for molecular machine learning datasets.

Molecular Datasets The feature extraction process is in two steps:

1. Atom-level feature extraction and Molecule object construction using RDKit [Landrum, 2006].
2. Constructing graphs from molecule objects using NetworkX [Hagberg *et al.*, 2008].

Atom features, shown in Table 5, are the atom features we used exactly the same as in [Rong *et al.*, 2020].

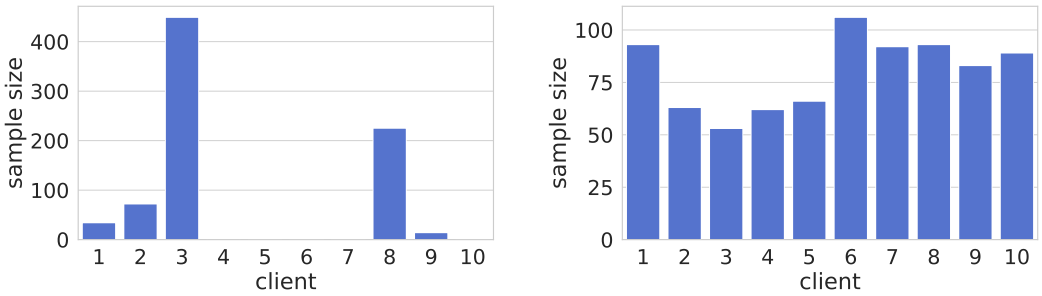
Table 5: Atom features

Features	**Size**	**Description**
atom type	100	Representation of atom (e.g., C, N, O), by its atomic number
formal charge	5	An integer electronic charge assigned to atom
number of bonds	6	Number of bonds the atom is involved in
chirality	5	Number of bonded hydrogen atoms
number of H	5	Number of bonded hydrogen atoms
atomic mass	1	Mass of the atom, divided by 100
aromaticity	1	Whether this atom is part of an aromatic system
hybridization	5	SP, SP2, SP3, SP3D, or SP3D2

Recommendation Systems/Knowledge Graph Datasets For recommendation systems, the data pre-processing step is partitioning the bipartite graph into subgraphs by item categories. Items belong to the same category and related users form a subgraph. Subgraphs are combined if one client holds more than one category. In knowledge graph, the pre-processing includes two steps. We first build subgraphs by relation types or node community, and then partition the data to multiple clients by a specific manner (uniform or non-I.I.D. partitioning with LDA).

Citation/Coauthor Datasets The main data pre-processing step for citation/coauthor networks in the node-level setting is to sample central nodes (egos) and build the k-hop neighborhoods correspondingly (k-hop ego-networks). For each dataset with a big entire graph, we randomly sample a number of egos (e.g. 1000) and construct k (e.g. 2)-hop egonetworks for them. These ego-networks are then partitioned by a specific manner (e.g. LDA non-IID partition) and distributed to multiple clients.

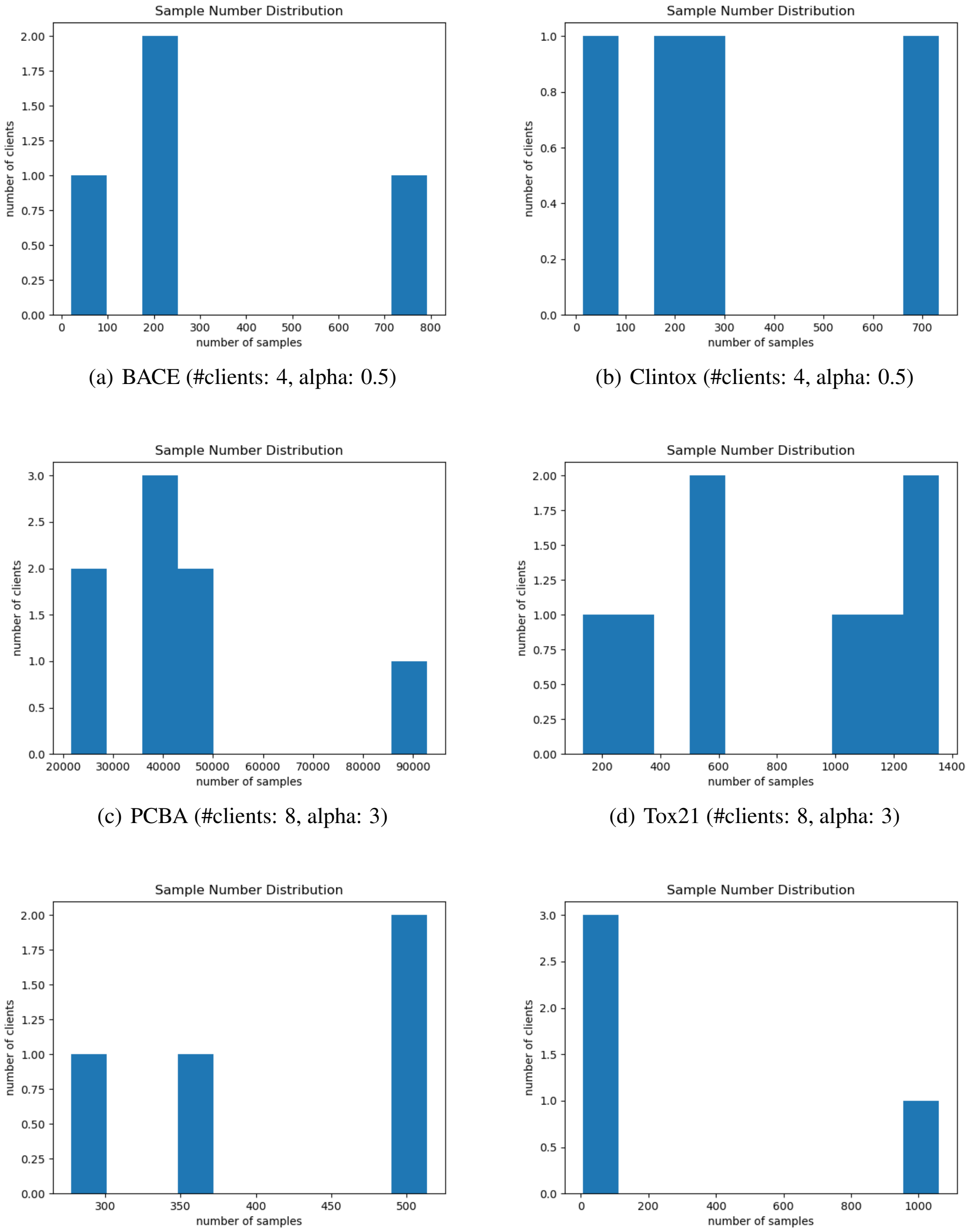
B.3 Non-I.I.D. Partitioning Distributions



(a) $\alpha = 0.1$

(b) $\alpha = 10.0$

Figure 5: Unbalanced sample distributions for citation networks (node-level tasks).



(a) BACE (#clients: 4, alpha: 0.5)

(b) Clintox (#clients: 4, alpha: 0.5)

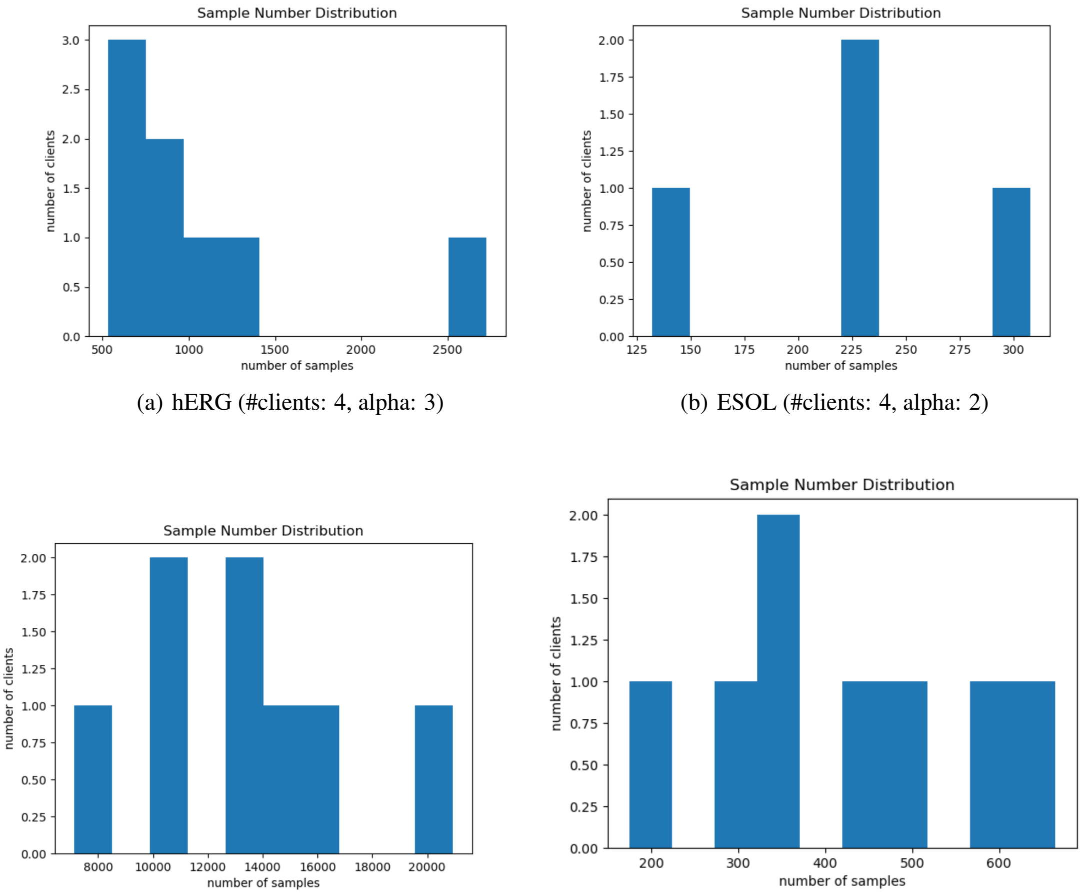
(c) PCBA (#clients: 8, alpha: 3)

(d) Tox21 (#clients: 8, alpha: 3)

(e) BBBP (#clients: 4, alpha: 2)

(f) SIDER (#clients: 4, alpha: 0.2)

Figure 6: Unbalanced Sample Distribution (Non-I.I.D.) for Molecular Graph Classification Datasets



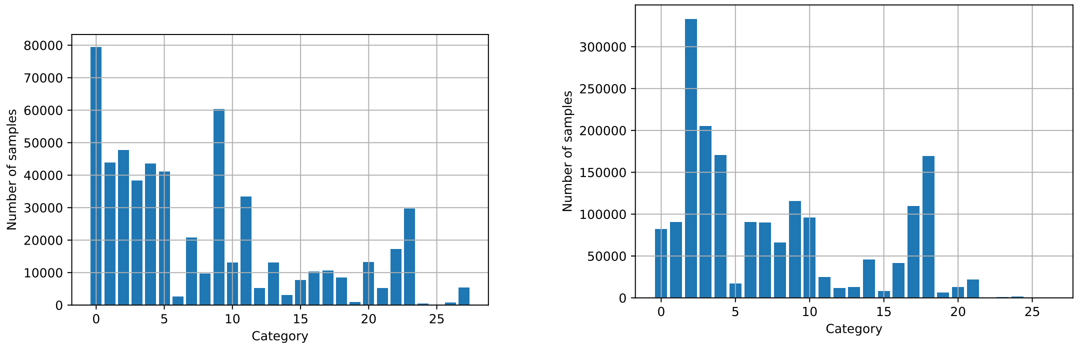
(a) hERG (#clients: 4, alpha: 3)

(b) ESOL (#clients: 4, alpha: 2)

(c) QM9 (#clients: 8, alpha: 3)

(d) LIPO (#clients: 8, alpha: 2)

Figure 7: Example (Non-I.I.D.) Sample Distributions for Molecular Property Prediction Datasets



(a) Ciao

(b) Epinions

Figure 8: Non-I.I.D. Sample Distributions for Recommendation Systems Datasets

C More Details of FedGraphNN System Design

`Data Collector and Manager` is a distributed computing system that can collect scattered datasets or features from multiple servers to `Training Manager`. Such collection can also keep the raw data in the original server with RPCs, which can only access the data during training. After obtaining all necessary datasets for federated training, `Training Manager` will start federated training using algorithms supported by `FedML-API`. Once training has been completed, `Model Serving` can request the trained model to deploy for inference. Under this SDK abstraction, we plan to address the challenges mentioned above (1) and (2) within the `Data Collector and Manager`. As for challenge (3), we plan to make `FedML Client SDK` compatible with any operating systems (Linux, Android, iOS) with a cross-platform abstraction interface design. In essence, the three modules inside `FedML Client SDK` builds up a pipeline that manages a model's life cycle, from federated training to personalized model serving (inference). Unifying three modules of a pipeline into a single SDK can simplify the system design. Any subsystem in an institute can integrate `FedML Client SDK` with a host process, which can be the backend service or desktop application. Overall, we hope `FedML Client SDK` could be a lightweight and easy-to-use SDK for federated learning among diverse cross-silo institutes.

D More Results of System Efficiency and Security

D.1 Evaluation on System Efficiency

Table 6: System-Level Performance Metrics for Graph-Level FedGraphNN tasks with FedAvg (Hardware: 8 x NVIDIA Quadro RTX 5000 GPU (16GB/GPU); RAM: 512G; CPU: Intel Xeon Gold 5220R 2.20GHz).

		SIDER	**BACE**	**Clintox**	**BBBP**	**Tox21**	**FreeSolv**	**ESOL**	**Lipo**	**hERG**	**QM9**
Wall-clock Time	GCN	5m 58s	4m 57s	4m 40s	4m 13s	15m 3s	4m 12s	5m 25s	16m 14s	35m 30s	6h 48m
	GAT	8m 48s	5m 27s	7m 37s	5m 28s	25m 49s	6m 24s	8m 36s	25m 28s	58m 14s	9h 21m
	GraphSAGE	2m 7s	3m 58s	4m 42s	3m 26s	14m 31s	5m 53s	6m 54s	15m 28s	32m 57s	5h 33m
Average FLOP	GCN	697.3K	605.1K	466.2K	427.2K	345.8K	142.6K	231.6K	480.6K	516.6K	153.9K
	GAT	703.4K	612.1K	470.2K	431K	347.8K	142.5K	232.6K	485K	521.3K	154.3K
	GraphSAGE	846K	758.6K	1.1M	980K	760.6K	326.9K	531.1K	1.5M	1.184M	338.2K
Parameters	GCN	15.1K	13.5K	13.6K	13.5K	14.2K	13.5K	13.5K	13.5K	13.5K	14.2K
	GAT	20.2K	18.5K	18.6K	18.5K	19.2K	18.5K	18.5K	18.5K	18.5K	19.2K
	GraphSAGE	10.6K	8.9K	18.2K	18.1K	18.8K	18.1K	18.1K	269K	18.1K	18.8K

*Note that we use the distributed training paradigm where each client's local training uses one GPU. Please refer to our code for details.

Table 7: System-level Performance Metrics for Subgraph-Level FedGraphNN tasks with FedAvg (Hardware: 8 x NVIDIA Quadro RTX 5000 GPU (16GB/GPU); RAM: 512G; CPU: Intel Xeon Gold 5220R 2.20GHz).

		Ciao	**Epinions**
Wall-clock Time	GCN	352s	717s
	GAT	350s	749s
	GraphSAGE	551s	810s
Average FLOP	GCN	697.3K	605.1K
	GAT	703.4K	612.1K
	GraphSAGE	846K	758.6K
Parameters	GCN	15.1K	13.5K
	GAT	20.2K	18.5K
	GraphSAGE	10.6K	8.9K

*Note that we use the distributed training paradigm where each client's local training uses one GPU. Please refer to our code for details.

The training time using RPC is also evaluated; and results are similar to that of using MPI. Note that RPC is useful for realistic deployment when GPU/CPU-based edge devices can only be accessed via public IP addresses due to locating in different data centers. We will provide detailed test results in such a scenario in our future work.

D.2 Evaluation on Security (`LightSecAgg`)

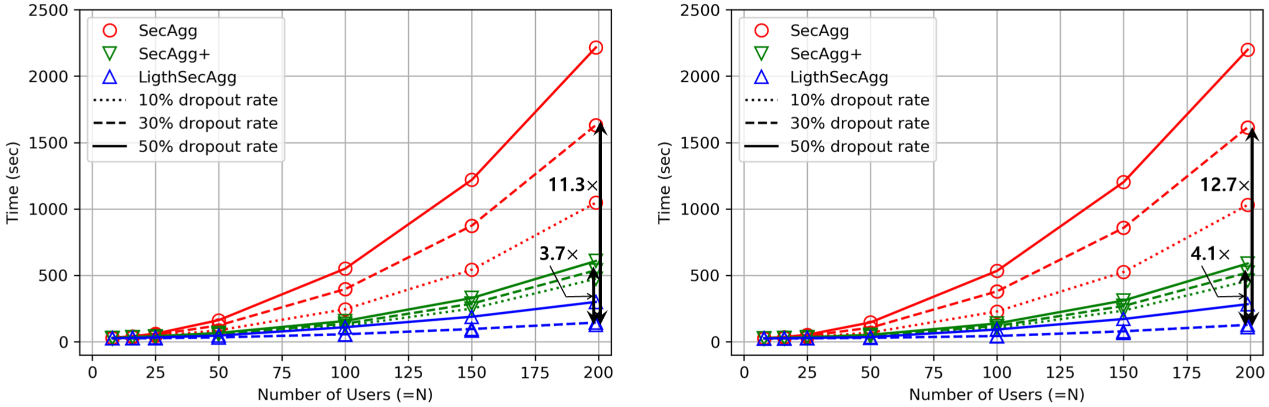
`LightSecAgg`, a FL security algorithm, providing model privacy guarantees as the state-of-the-art (SecAgg [Bonawitz *et al.*, 2017] and SecAgg+ [Bell *et al.*, 2020]) while substantially reducing the aggregation (hence run-time) complexity (Figure 9). The main idea of `LightSecAgg` is that each user protects its local model using a locally generated random mask. This mask is then encoded and shared to other users, in such a way that the aggregate mask of any sufficiently large set of surviving users

Table 8: System-level Performance Metrics for Node-Level FedGraphNN tasks with FedAvg (Hardware: 8 x NVIDIA Quadro RTX 5000 GPU (16GB/GPU); RAM: 512G; CPU: Intel Xeon Gold 5220R 2.20GHz).

		CORA	**Citeseer**	**DBLP**	**PubMed**
Wall-clock Time	GCN	833s	622s	654s	653s
	GAT	871s	652s	682s	712s
	GraphSAGE	774s	562s	622s	592s
Average FLOP	GCN	44.9M	739.4K	8.8M	1.9M
	GAT	47.8M	845.3K	9.5M	2.3M
	GraphSAGE	45.3M	817K	9.2M	2.1M
Parameters	GCN	282.1K	20.5K	53.7K	17.2K
	GAT	285.1K	23.7K	56.5K	19.3K
	GraphSAGE	283K	21.6K	54.7K	18.2K

*Note that we use the distributed training paradigm where each client's local training uses one GPU. Please refer to our code for details.

can be directly reconstructed at the server. Our main effort in FedGraphNN is integrating `LightSecAgg`, optimizing its system performance, and designing user-friendly APIs to make it compatible with various models and FL algorithms.



(a) Non-overlapped

(b) Overlapped

Figure 9: LightSecAgg: Total running time of LightSecAgg versus the state-of-the-art protocols (SecAgg[Bonawitz *et al.*, 2017] and SecAgg+ [Bell *et al.*, 2020]) to train neural networks with 1.2 M parameters (more than all GNN models used in this work), as the number of users increases, for various dropout rates.

E More Details of the Empirical Analysis

E.1 Hyper-parameters

For each task, we utilize grid search to find the best results. Table 9 & **??** list all the hyper-parameters ranges used in our experiments. All hyper-parameter tuning is run on a single GPU. The best hyperparameters for each dataset and model are listed in Table 13,14,15, & 16 For molecule tasks ,batch-size is kept fixed since the molecule-level task requires us to have mini-batch is equal to 1. Also, number of GNN layers were fixed to 2 because having too many GNN layers result in over-smoothing phenomenon For all experiments, we used Adam optimizer.

Table 9: Hyper-parameter Range for Graph-Level Centralized Training(classification & regression)

hyper-parameter	Description	Range
learning rate	Rate of speed at which the model learns.	$[0.00015, 0.0015, 0.015, 0.15]$
dropout rate	Dropout ratio	$[0.2, 0.3, 0.5, 0.6]$
node embedding dimension	Dimensionality of the node embedding	$[16, 32, 64, 128, 256]$
hidden layer dimension	Hidden layer dimensionality	$[16, 32, 64, 128, 256]$
readout embedding dimension	Dimensionality of the readout embedding	$[16, 32, 64, 128256]$
graph embedding dimension	Dimensionality of the graph embedding	$[16, 32, 64, 128, 256]$
attention heads	Number of attention heads required for GAT	1-7
alpha	LeakyRELU parameter used in GAT model	0.2

Table 10: Hyper-parameter Range for Graph-Level Federated Learning(classification & regression)

hyper-parameter	Description	Range
learning rate	Rate of speed at which the model learns.	$[0.00015, 0.0015, 0.015, 0.15]$
dropout rate	Dropout ratio	$[0.3, 0.5, 0.6]$
node embedding dimension	Dimensionality of the node embedding	64
hidden layer dimension	Hidden layer dimensionality	64
readout embedding dimension	Dimensionality of the readout embedding	64
graph embedding dimension	Dimensionality of the graph embedding	64
attention heads	Number of attention heads required for GAT	1-7
alpha	LeakyRELU parameter used in GAT model	0.2
rounds	Number of federating learning rounds	[10, 50, 100]
epoch	Epoch of clients	1
number of clients	Number of users in a federated learning round	4-10

Table 11: Hyper-parameter Range for Subgraph-Level Federated Learning

hyper-parameter	Description	Range
learning rate	Rate of speed at which the model learns.	$[0.0001, 0.001, 0.01, 0.1]$
node embedding dimension	Dimensionality of the node embedding	64
hidden layer dimension	Hidden layer dimensionality	$[64]$
rounds	Number of federating learning rounds	$[1, 10, 20, 50, 100]$
local epoch	Epoch of clients	$[1, 2, 5]$
number of clients	Number of users in a federated learning round	4-10

Table 12: Hyper-parameter Range for Node-Level Federated Learning

hyper-parameter	Description	Range
learning rate	Rate of speed at which the model learns.	$[0.1, 0.01, 0.001, 0.0001]$
dropout rate	Dropout ratio	$[0.3, 0.5, 0.6]$
hidden layer dimension	Hidden layer dimensionality	[32, 64, 128]
alpha	LeakyRELU parameter used in GAT model	[0.1, 10]
rounds	Number of federating learning rounds	100
epoch	Epoch of clients	[1, 3, 5]
number of clients	Number of users in a federated learning round	10

Table 13: Hyperparameters for Graph-Level Molecular Classification Task

Dataset	**Score & Parameters**	**GCN**	**GAT**	**GraphSAGE**
BBBP	ROC-AUC Score	0.8705	0.8824	**0.8930**
	learning rate	0.0015	0.015	0.01
	dropout rate	0.2	0.5	0.2
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
BACE	ROC-AUC Score	0.9221	0.7657	**0.9266**
	learning rate	0.0015	0.001	0.0015
	dropout rate	0.3	0.3	0.3
	node embedding dimension	64	64	16
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
Tox21	ROC-AUC Score	0.7800	0.8144	**0.8317**
	learning rate	0.0015	0.00015	0.00015
	dropout rate	0.4	0.3	0.3
	node embedding dimension	64	128	256
	hidden layer dimension	64	64	128
	readout embedding dimension	64	128	256
	graph embedding dimension	64	64	128
	attention heads	None	2	None
	alpha	None	0.2	None
SIDER	ROC-AUC Score	0.6476	0.6639	**0.6669**
	learning rate	0.0015	0.0015	0.0015
	dropout rate	0.3	0.3	0.6
	node embedding dimension	64	64	16
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
ClinTox	ROC-AUC Score	0.8914	0.9573	**0.9716**
	learning rate	0.0015	0.0015	0.0015
	dropout rate	0.3	0.3	0.3
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None

Table 14: Hyperparameters for Graph-Level Federated Molecular Classification Task

Dataset	**Score & Parameters**	**GCN + FedAvg**	**GAT + FedAvg**	**GraphSAGE + FedAvg**
BBBP	ROC-AUC Score	0.7629	0.8746	**0.8935**
	number of clients	4	4	4
	learning rate	0.0015	0.0015	0.015
	dropout rate	0.3	0.3	0.6
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
BACE	ROC-AUC Score	0.6594	0.7714	**0.8604**
	number of clients	4	4	4
	learning rate	0.0015	0.0015	0.0015
	dropout rate	0.5	0.3	0.5
	node embedding dimension	64	64	16
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
Tox21	ROC-AUC Score	0.7128	0.7171	**0.7801**
	number of clients	4	4	4
	learning rate	0.0015	0.0015	0.00015
	dropout rate	0.6	0.3	0.3
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
SIDER	ROC-AUC Score	0.6266	0.6591	**0.67**
	number of clients	4	4	4
	learning rate	0.0015	0.0015	0.0015
	dropout rate	0.6	0.3	0.6
	node embedding dimension	64	64	16
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
ClinTox	ROC-AUC Score	0.8784	0.9160	**0.9246**
	number of clients	4	4	4
	learning rate	0.0015	0.0015	0.015
	dropout rate	0.5	0.6	0.3
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None

Table 15: Hyperparameters for Graph-Level Molecular Regression Task

Dataset	**Score &Parameters**	**GCN**	**GAT**	**GraphSAGE**
Freesolv	RMSE Score	0.8705	0.8824	**0.8930**
	learning rate	0.0015	0.015	0.01
	dropout rate	0.2	0.5	0.2
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
ESOL	RMSE Score	0.8705	0.8824	**0.8930**
	learning rate	0.0015	0.015	0.01
	dropout rate	0.2	0.5	0.2
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
Lipophilicity	RMSE Score	0.8521	0.7415	**0.7078**
	learning rate	0.0015	0.001	0.001
	dropout rate	0.3	0.3	0.3
	node embedding dimension	128	128	128
	hidden layer dimension	64	64	64
	readout embedding dimension	128	128	128
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
hERG	RMSE Score	0.7257	**0.6271**	0.7132
	learning rate	0.001	0.001	0.005
	dropout rate	0.3	0.5	0.3
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
QM9	RMSE Score	14.78	**12.44**	13.06
	learning rate	0.0015	0.015	0.01
	dropout rate	0.2	0.5	0.2
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None

Table 16: Hyperparameters for Graph-Level Federated Molecular Regression Task

Dataset	**Parameters**	**GCN + FedAvg**	**GAT + FedAvg**	**GraphSAGE + FedAvg**
FreeSolv	RMSE Score	2.747	3.108	**1.641**
	number of clients	4	8	4
	learning rate	0.0015	0.00015	0.015
	dropout rate	0.6	0.5	0.6
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
ESOL	RMSE Score	1.435	**1.028**	1.185
	number of clients	4	4	4
	learning rate	0.0015	0.0015	0.0015
	dropout rate	0.5	0.3	0.3
	node embedding dimension	64	256	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
Lipophilicity	RMSE Score	1.146	1.004	**0.7788**
	number of clients	4	4	4
	learning rate	0.0015	0.0015	0.0015
	dropout rate	0.3	0.3	0.3
	node embedding dimension	64	64	256
	hidden layer dimension	64	64	256
	readout embedding dimension	64	64	256
	graph embedding dimension	64	64	256
	attention heads	None	2	None
	alpha	None	0.2	None
hERG	RMSE Score	0.7944	0.7322	**0.7265**
	number of clients	8	8	8
	learning rate	0.0015	0.0015	0.0015
	dropout rate	0.3	0.3	0.6
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64
	readout embedding dimension	64	64	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None
QM9	MAE Score	21.075	23.173	**19.167**
	number of clients	8	8	8
	learning rate	0.0015	0.00015	0.15
	dropout rate	0.2	0.5	0.3
	node embedding dimension	64	256	64
	hidden layer dimension	64	128	64
	readout embedding dimension	64	256	64
	graph embedding dimension	64	64	64
	attention heads	None	2	None
	alpha	None	0.2	None

Table 17: Hyperparameters for Subgraph-Level Centralized Link Prediction Task

Dataset	Score & Parameters	GCN	GAT	GraphSAGE
Ciao	mean absolute error	**0.8167**	0.8214	0.8231
	mean squared error	**1.1184**	1.1318	1.1541
	root mean squared error	**1.0575**	1.0639	1.0742
	communication round	100	100	50
	learning rate	0.01	0.001	0.01
	node embedding dimension	64	64	64
	hidden layer dimension	32	32	16
Epinions	mean absolute error	**0.8847**	0.8934	1.0436
	mean squared error	**1.3733**	1.3873	1.8454
	root mean squared error	**1.1718**	1.1767	1.3554
	communication round	50	50	100
	learning rate	0.01	0.01	0.01
	node embedding dimension	64	64	64
	hidden layer dimension	32	32	64

Table 18: Hyperparameters for Subgraph-Level Federated Link Prediction Task

Dataset	Score & Parameters	GCN + FedAvg	GAT + FedAvg	GraphSAGE + FedAvg
Ciao	mean absolute error	0.7995	**0.7987**	0.8290
	mean squared error	**1.0667**	1.0682	1.1320
	root mean squared error	**1.0293**	1.0311	1.0626
	communication round	100	100	50
	local epochs	5	2	5
	number of clients	8	8	8
	learning rate	0.01	0.001	0.01
	node embedding dimension	64	64	64
	hidden layer dimension	32	32	32
Epinions	mean absolute error	0.9033	**0.9032**	0.9816
	mean squared error	1.4378	**1.4248**	1.6136
	root mean squared error	1.1924	**1.1882**	1.2625
	communication round	100	50	100
	local epochs	2	1	2
	number of clients	8	8	8
	learning rate	0.01	0.001	0.01
	node embedding dimension	64	64	64
	hidden layer dimension	64	64	64

Table 19: Hyperparameters for Node-Level Federated Node Classification Task

Dataset	**Parameters**	**GCN + FedAvg**	**GAT + FedAvg**	**GraphSAGE + FedAvg**
CORA	Micro F1 score	0.8549	diverge	**0.9746**
	number of clients	10	10	10
	learning rate	0.001	0.001	0.001
	dropout rate	0.5	0.5	0.5
	local epochs	5	5	5
	hidden layer dimension	128	128	128
Citeseer	Micro F1 score	0.9743	0.9610	**0.9854**
	learning rate	0.001	0.01	0.001
	dropout rate	0.5	0.5	0.5
	local epochs	5	3	5
	hidden layer dimension	128	128	128
PubMed	Micro F1 score	0.9191	0.8557	**0.9761**
	number of clients	10	10	10
	learning rate	0.001	0.001	0.001
	dropout rate	0.5	0.5	0.5
	local epochs	5	5	5
	hidden layer dimension	128	128	128
DBLP	Micro F1 score	0.9088	0.8201	**0.9749**
	number of clients	10	10	10
	learning rate	0.001	0.001	0.001
	dropout rate	0.5	0.5	0.5
	local epochs	5	5	5
	hidden layer dimension	128	128	128

E.2 Evaluation Metrics

Current metrics supported in FedGraphNN include:

- **Graph Classification**: ROC-AUC (Area Under the Curve - Receiver Operating Characteristics) is a well-used classification metric to evaluate the performance at various thresholds.
- **Graph Regression**: For this task, we chose RMSE (Root Mean Squared Error). However, when train and test distributions differ, it is more suitable to use metrics such as MAPE (Mean Absolute Percentage Error).
- **Node Classification**: In ego-network node-level FL task, we use F1 score as the metric because F1 score is better than the accuracy metric when imbalanced class distribution exists.
- **Link prediction (Recommendation Systems)**: Specifically for recommendation systems, we can treat it as a regression problem. Thus, it is possible to use well-known metrics such as MAE (Mean Absolute Error), MSE (Mean Squared Error), RMSE (Root Mean Squared Error). Widely used ranking based metrics can also be applied such as DCG (Discounted Cumulative Gain) and NDCG (Normalized Discounted Cumulative Gain),.
- **Relation Prediction (Knowledge Graphs)**: Besides accuracy based metrics such as precision, recall and f1 score, ranking based metrics are also applied to relation type prediction on knowledge graphs such as MRR (Mean Reciprocal Rank)[Radev *et al.*, 2002] and HR (Hit Ratio).

In addition to the metrics described, FedGraphNN users can add their custom metrics as well. As a future work, we plan to analyze these metrics' representative capacity on the FL performance.

E.3 More Experimental Results

Aside from additional results for graph-level FL, we will further provide more live results and reports in our project website https://FedML.ai/FedGraphNN. We hope these visualized training results can be a useful reference for future research exploration.

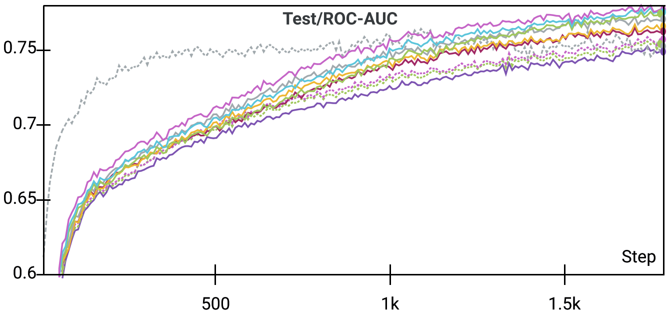


Figure 10: Tox21: test score during sweeping

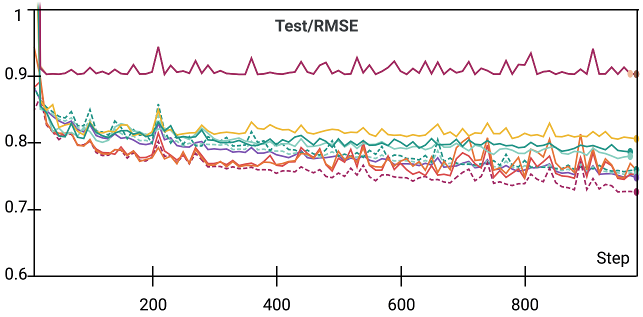


Figure 11: hERG: test score during sweeping

Table 20: Performance of graph regression in the graph-level FL setting (#clients=4, MAE for QM9).

Metric	RMSE				
Method	FREESOLV $\alpha = 0.2$	ESOL $\alpha = 0.5$	LIPO $\alpha = 0.5$	hERG $\alpha = 3$	QM9 $\alpha = 3$
MoleculeNet Results	1.40±0.16	0.97±0.01	0.655±0.036	DNE	2.35
GCN (centralized)	1.5787	1.0190	0.8518	0.7257	14.78
GCN (FedAvg)	2.7470	1.4350	1.1460	0.7944	21.075
GAT (Centralized)	1.2175	0.9358	0.7465	0.6271	12.44
GAT (FedAvg)	1.3130	0.9643	0.8537	0.7322	23.173
GraphSAGE (centralized)	1.3630	0.8890	0.7078	0.7132	13.06
GraphSAGE (FedAvg)	1.6410	1.1860	0.7788	0.7265	19.167

Appendix References

[Bell *et al.*, 2020] James Henry Bell, Kallista A Bonawitz, Adrià Gascón, Tancrède Lepoint, and Mariana Raykova. Secure single-server aggregation with (poly) logarithmic overhead. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 1253–1269, 2020.

[Bemis and Murcko, 1996] Guy W Bemis and Mark A Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of medicinal chemistry*, 39(15):2887–2893, 1996.

[Bojchevski and Günnemann, 2018] Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. In *International Conference on Learning Representations*, 2018.

[Bonawitz *et al.*, 2017] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191, 2017.

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

[Cho *et al.*, 2013] Young Sung Cho, Song Chul Moon, Seon-phil Jeong, In-Bae Oh, and Keun Ho Ryu. Clustering method using item preference based on rfm for recommendation system in u-commerce. In *Ubiquitous information technologies and applications*, pages 353–362. Springer, 2013.

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

[Delaney, 2004] John S Delaney. Esol: estimating aqueous solubility directly from molecular structure. *Journal of chemical information and computer sciences*, 44(3):1000–1005, 2004.

[Dettmers *et al.*, 2018] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *AAAI*, pages 1811–1818, 2018.

[Gaulton *et al.*, 2012] Anna Gaulton, Louisa J Bellis, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, et al. Chembl: a large-scale bioactivity database for drug discovery. *Nucleic acids research*, 40(D1):D1100–D1107, 2012.

[Gaulton *et al.*, 2016] Anna Gaulton, Anne Hersey, Michał Nowotka, A. Patrícia Bento, Jon Chambers, David Mendez, Prudence Mutowo, Francis Atkinson, Louisa J. Bellis, Elena Cibrián-Uhalte, Mark Davies, Nathan Dedman, Anneli Karlsson, María Paula Magariños, John P. Overington, George Papadatos, Ines Smit, and Andrew R. Leach. The ChEMBL database in 2017. *Nucleic Acids Research*, 45(D1):D945–D954, 11 2016.

[Gaulton *et al.*, 2017] 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. The chembl database in 2017. *Nucleic acids research*, 45(D1):D945–D954, 2017.

[Gayvert *et al.*, 2016] Kaitlyn M Gayvert, Neel S Madhukar, and Olivier Elemento. A data-driven approach to predicting successes and failures of clinical trials. *Cell chemical biology*, 23(10):1294–1301, 2016.

[Giles *et al.*, 1998] C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. Citeseer: An automatic citation indexing system. In *Proceedings of the Third ACM Conference on Digital Libraries*, DL '98, page 89–98, New York, NY, USA, 1998. Association for Computing Machinery.

[Hagberg *et al.*, 2008] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

[Hamilton *et al.*, 2017] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *CoRR*, abs/1706.02216, 2017.

[He *et al.*, 2019] Chaoyang He, Tian Xie, Yu Rong, Wenbing Huang, Junzhou Huang, Xiang Ren, and Cyrus Shahabi. Cascade-bgnn: Toward efficient self-supervised representation learning on large-scale bipartite graphs. *arXiv preprint arXiv:1906.11994*, 2019.

[Kim *et al.*, 2021] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, et al. Pubchem in 2021: new data content and improved web interfaces. *Nucleic Acids Research*, 49(D1):D1388–D1395, 2021.

[Kipf and Welling, 2016] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016.

[Kuhn *et al.*, 2016] Michael Kuhn, Ivica Letunic, Lars Juhl Jensen, and Peer Bork. The sider database of drugs and side effects. *Nucleic acids research*, 44(D1):D1075–D1079, 2016.

[Landrum, 2006] Greg Landrum. Rdkit: Open-source cheminformatics, 2006.

[Mahdisoltani *et al.*, 2013] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. Yago3: A knowledge base from multilingual wikipedias. In *CIDR*, Asilomar, United States, January 2013.

[Markowitz *et al.*, 2021] Elan Markowitz, Keshav Balasubramanian, Mehrnoosh Mirtaheri, Sami Abu-El-Haija, Bryan Perozzi, Greg Ver Steeg, and Aram Galstyan. Graph traversal with tensor functionals: A meta-algorithm for scalable learning, 2021.

[Martins *et al.*, 2012] Ines Filipa Martins, Ana L Teixeira, Luis Pinheiro, and Andre O Falcao. A bayesian approach to in silico blood-brain barrier penetration modeling. *Journal of chemical information and modeling*, 52(6):1686–1697, 2012.

[McCallum *et al.*, 2000] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

[Mobley and Guthrie, 2014] David L Mobley and J Peter Guthrie. Freesolv: a database of experimental and calculated hydration free energies, with input files. *Journal of computer-aided molecular design*, 28(7):711–720, 2014.

[Radev *et al.*, 2002] Dragomir R Radev, Hong Qi, Harris Wu, and Weiguo Fan. Evaluating web-based question answering systems. In *LREC*. Citeseer, 2002.

[Ramakrishnan *et al.*, 2014] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 2014.

[Richard *et al.*, 2016] Ann M Richard, Richard S Judson, Keith A Houck, Christopher M Grulke, Patra Volarath, Inthirany Thillainadarajah, Chihae Yang, James Rathman, Matthew T Martin, John F Wambaugh, et al. Toxcast chemical landscape: paving the road to 21st century toxicology. *Chemical research in toxicology*, 29(8):1225–1251, 2016.

[Richardson *et al.*, 2003] Matthew Richardson, Rakesh Agrawal, and Pedro Domingos. Trust management for the semantic web. In *International semantic Web conference*, pages 351–368. Springer, 2003.

[Riesen and Bunke, 2008] Kaspar Riesen and Horst Bunke. Iam graph database repository for graph based pattern recognition and machine learning. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 287–297. Springer, 2008.

[Rohrer and Baumann, 2009] Sebastian G Rohrer and Knut Baumann. Maximum unbiased validation (muv) data sets for virtual screening based on pubchem bioactivity data. *Journal of chemical information and modeling*, 49(2):169–184, 2009.

[Rong *et al.*, 2020] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data, 2020.

[Segura Bedmar *et al.*, 2013] Isabel Segura Bedmar, Paloma Martínez, and María Herrero Zazo. Semeval-2013 task 9: Extraction of drug-drug interactions from biomedical texts (ddiextraction 2013). Association for Computational Linguistics, 2013.

[Sen *et al.*, 2008] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[Shchur *et al.*, 2019] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation, 2019.

[Subramanian *et al.*, 2016] Govindan Subramanian, Bharath Ramsundar, Vijay Pande, and Rajiah Aldrin Denny. Computational modeling of β-secretase 1 (bace-1) inhibitors using ligand based approaches. *Journal of chemical information and modeling*, 56(10):1936–1949, 2016.

[Tang *et al.*, 2008] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998, 2008.

[Tang *et al.*, 2012] Jiliang Tang, Huiji Gao, and Huan Liu. mtrust: Discerning multi-faceted trust in a connected world. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 93–102, 2012.

[Toutanova and Chen, 2015] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *CVSCW*, 07 2015.

[tox, 2017] Tox21 challenge. https://tripod.nih.gov/tox21/challenge/, 2017.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[Wale *et al.*, 2008] Nikil Wale, Ian A Watson, and George Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowledge and Information Systems*, 14(3):347–375, 2008.

[Wu *et al.*, 2018] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[Wu *et al.*, 2019] Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr. au2, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. Simplifying graph convolutional networks, 2019.

[Xu *et al.*, 2019] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks?, 2019.

[Yanardag and Vishwanathan, 2015] Pinar Yanardag and S.V.N. Vishwanathan. *Deep Graph Kernels*, page 1365–1374. Association for Computing Machinery, New York, NY, USA, 2015.