Federated Learning on Non-IID Graphs via Structural Knowledge Sharing

Yue Tan[1*], **Yixin Liu**[2*], **Guodong Long**[1], **Jing Jiang**[1], **Qinghua Lu**[3], **Chengqi Zhang**[1]

[1] Australian Artificial Intelligence Institute, University of Technology Sydney, Australia
[2] Monash University, Australia
[3] Data61, CSIRO, Australia
yue.tan@student.uts.edu.au, yixin.liu@monash.edu, {guodong.long, jing.jiang}@uts.edu.au,
qinghua.lu@data61.csiro.au, chengqi.zhang@uts.edu.au

Abstract

Graph neural networks (GNNs) have shown their superiority in modeling graph data. Owing to the advantages of federated learning, federated graph learning (FGL) enables clients to train strong GNN models in a distributed manner without sharing their private data. A core challenge in federated systems is the non-IID problem, which also widely exists in real-world graph data. For example, local data of clients may come from diverse datasets or even domains, *e.g.*, social networks and molecules, increasing the difficulty for FGL methods to capture commonly shared knowledge and learn a generalized encoder. From real-world graph datasets, we observe that some structural properties are shared by various domains, presenting great potential for sharing structural knowledge in FGL. Inspired by this, we propose FedStar, an FGL framework that extracts and shares the common underlying structure information for inter-graph federated learning tasks. To explicitly extract the structure information rather than encoding them along with the node features, we define structure embeddings and encode them with an independent structure encoder. Then, the structure encoder is shared across clients while the feature-based knowledge is learned in a personalized way, making FedStar capable of capturing more structure-based domain-invariant information and avoiding feature misalignment issues. We perform extensive experiments over both cross-dataset and cross-domain non-IID FGL settings, demonstrating the superiority of FedStar.

Introduction

Graph neural networks (GNNs) have been widely used to model graph-structured data in a variety of scenarios and applications, such as recommender systems (Wu et al. 2020a), drug discovery (Gaudelet et al. 2021), and traffic (Bai et al. 2020). Most existing GNNs follow a centrally training principle where graph data need to be collected together before training (Kipf and Welling 2017; Xu et al. 2019). However, nowadays, a large number of graph data are generated from edge devices and may contain private data of users, hindering the traditional GNNs from training strong models by collective intelligence (Zhang et al. 2021a).

Federated learning (FL), as a new machine learning paradigm, allows clients to collaboratively train a globally
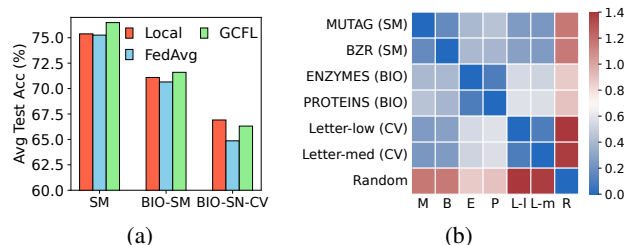

Figure 1: (a) Performance comparison of local training and FL/FGL methods (McMahan et al. 2017; Xie et al. 2021) in three cross-dataset/domain non-IID settings. (b) The heatmap of Jensen-Shannon divergence of degree distributions among six real-world graph datasets from different domains and random graphs. The random graphs are generated with Erdős–Rényi model (Erdős, Rényi et al. 1960).

shared model or personalized models in a decentralized manner while not sharing the local data of clients (Yang et al. 2019). Due to the advantages of FL, it is natural to apply FL to graph data to mitigate the data isolation and protect the security of graph data owned by end users, such as molecule graphs owned by pharmaceutical companies and social networks located in the social app of end users. Collaboratively training GNN models with an FL framework, federated graph learning (FGL) has emerged to be a promising direction to further explore the potential of GNNs on decentralized graph data.

An important challenge in FL is the non-IID data problem where local datasets of clients can be extremely different due to their diverse behaviors or preferences. This difference may lead to unstable training and performance drop in some extreme cases, *e.g.*, the data of clients are from different domains (Li et al. 2020c; Kairouz et al. 2021). The non-IID data problem, unfortunately, also naturally exists in FGL owing to the diversity of graph data. For example, different companies/platforms maintain rich social networks data, which are of great heterogeneity (*i.e.*, diverse feature spaces and connection rules) due to different data collection manners and purposes. It is beneficial to overcome the non-IID issue, enabling these companies to collaboratively provide better online social network service without sacrificing data security. Furthermore, it is of great potential to explore the underlying knowledge commonly shared by heterogeneous graph domains, *e.g.*, social networks and molecules.

An emerging branch of studies has attempted to alleviate the influence of non-IID graphs (Xie et al. 2021; Wang et al. 2020a). They take conventional GNNs (*e.g.*, GraphSAGE (Hamilton, Ying, and Leskovec 2017) and GIN (Xu et al. 2019)) as local encoders and introduce various techniques to local training or global aggregation, such as clustering and meta-learning. Based on the conventional GNNs, existing FGL methods still generate representations via a feature-based message passing scheme, deteriorated by the *feature heterogeneity* (*i.e.*, different feature spaces and dimensions) across graphs from multiple domains. In FGL procedures, aggregating GNN models trained upon diverse feature spaces may fail to capture useful underlying knowledge shared over non-IID graphs, which in turn unexpectedly hurts the personalized performance of local models. As shown in Fig. 1(a), in non-IID FGL settings, the performance of federated methods, *i.e.*, FedAvg and GCFL, is sometimes even worse than local training.

Although feature information tends to be heterogeneous across different graph domains, some structural properties underlying graphs can be universally shared across different domains (Qiu et al. 2020; Leskovec, Kleinberg, and Faloutsos 2005; Xie et al. 2021), inspiring us to exploit the potential of sharing structural knowledge with FGL. As shown in Fig. 1(b), compared with random graphs, real-world graphs share more inherent structural properties even if they are from different domains. Nevertheless, it is difficult for existing FGL frameworks to capture and communicate such structural knowledge. Firstly, these methods usually depend on conventional GNNs where the structure information is implicitly encoded into feature-based node representations; thus, heterogeneous features inevitably affect the quality of structural knowledge learning. Moreover, most FGL methods still perform parameter aggregation over all learnable parameters, making it hard to extract and share structure-specific knowledge across clients. Hence, a natural research question to ask is "*how to insulate structure information from the heterogeneous features during local training and how to perform federated knowledge sharing?*"

To answer the aforementioned question, in this work, we propose **Fed**erated graph learning via **St**ructure knowledge sh**ar**ing (**FedStar** for short), a framework that takes full advantage of natural structure information in graphs to deal with inter-graph FL tasks. Our theme is to capture and share the universal structural knowledge across multiple clients to boost the local performance of clients in FGL. Specifically, we first introduce a vectorial structure embedding that explicitly represents the inherent structural properties shared across different domains. Then, to avoid interference by feature heterogeneity, we design a feature-structure decoupled GNN to capture high-level structural knowledge with an independent channel while learning attributive knowledge with another one. Finally, we design an FL framework with structural knowledge sharing, where domain-agnostic structural information is shared across different clients and feature-based representations are learned locally. As a result, clients are capable of learning generalized structure encoders as well as personalized feature encoders simultaneously. The contributions of our work are three-fold:

- We study the non-IID problem in federated graph learning with tremendous heterogeneous graph data from a new perspective, *i.e.*, structural knowledge sharing, which sheds good light on the future research in this field.
- We propose a novel federated graph learning framework, namely FedStar, enabling clients to learn domain-invariant structural knowledge globally while capturing domain-specific feature-based knowledge locally.
- We conduct extensive experiments on four cross-dataset/domain non-IID settings and show that FedStar consistently outperforms baselines by a large margin.

Related Work

Graph Neural Network. Graph neural networks (GNNs) are a family of neural models for modeling graph-structured data (Wu et al. 2020b). Most GNNs follow a message passing scheme, where node representation is learned by aggregating and transforming the embeddings of its neighbors and itself (Hamilton, Ying, and Leskovec 2017; Xu et al. 2019). For instance, GCN (Kipf and Welling 2017) aggregates messages via averaging the neighboring representations, and GIN (Xu et al. 2019) leverages a summation function for aggregation. Thanks to such a message passing scheme, GNNs can achieve excellent performance when learning from a single graph (Veličković et al. 2018; Liu et al. 2022) or a set of graphs with shared feature space (Hu et al. 2020a,b). However, the feature-based message initialization mechanism makes the feature space and learnable parameters in GNNs tightly coupled, leading to the difficulty of training GNNs on graph data in multiple domains (Qiu et al. 2020; Zhu et al. 2021; Liu et al. 2023a). Besides, a branch of GNNs extracts structural/positional encoding to explicitly represent the structure information (Li et al. 2020a; Dwivedi et al. 2021), inspiring us to capture universal structural patterns across different domains by GNNs.

Federated Learning. Federated learning (FL) has attracted much attention recently owing to its potential to enable collaborative training while protecting data privacy (Kairouz et al. 2021; Yang et al. 2019; Lyu et al. 2022). The standard FL algorithm, namely FedAvg, iteratively conducts local training at clients and global parameter averaging at central server (McMahan et al. 2017). A number of methods are proposed to improve FedAvg in terms of communication efficiency (Hamer, Mohri, and Suresh 2020), generalization ability (Yuan et al. 2021; Qu et al. 2022), robustness to heterogeneity (Wang et al. 2020b; Tan et al. 2022a; Chen et al. 2022a), etc. One of the existing challenges in real-world FL is the data heterogeneity, also known as the non-IID problem, where clients may have diverse label and/or feature distributions due to their various behaviours and habits (Luo et al. 2021; Tan et al. 2022b; Chen et al. 2022b). To tackle this, clustering-based FL methods propose to divide clients according their similarity (Ghosh et al. 2020; Long et al. 2022; Ma et al. 2022). Multiple works leverage meta-learning to improve the personalized ability of the local model (Fallah, Mokhtari, and Ozdaglar 2020; Jiang et al. 2019) or introduce model decoupling scheme to enable better personalization (Chen and Chao 2022).

Federated Graph Learning. Federated graph learning (FGL) supports distributed GNN training, extending its original application scenarios. So far, existing FGL studies can be categorized into three types, *i.e.*, inter-graph, intra-graph, and graph-structured FGL (Zhang et al. 2021a). In inter-graph FGL, each client owns a set of graphs and participates in FL training to learn a better GNN to model local data (Xie et al. 2021), learn a generalizable model (Zhu, Luo, and White 2022), or to model spatial-temporal graph data (Jiang et al. 2022; Lou et al. 2021). In intra-graph FGL, instead of complete graphs, each client only owns a subgraph of the entire graph and the learning scheme is to deal with the missing links (Chen et al. 2021), *i.e.*, generating missing neighbors (Zhang et al. 2021b), community discover (Baek et al. 2022). Intra-graph FGL can be applied to financial crimes detection (Suzumura et al. 2019), In graph-structured FGL, graphs are used to model the inherent relationship among clients and can be applied to various data types, such as images (Chen et al. 2022c) and traffic data (Meng, Rambhatla, and Liu 2021). In this paper, we consider inter-graph FGL and focus on learning a better local model for each client rather than training a global GNN model.

Preliminaries

Graph Neural Networks

Let $G = (V, E)$ be a graph consisting of a set of nodes V and a set of edges E connecting these nodes. Each node $v \in V$ has a feature vector $\mathbf{x}_v$. Based on the graph structure and node features, GNNs can be used to learn the node-level representation vector $\mathbf{h}_v$ of node $v \in V$ and/or the graph-level representation vector $\mathbf{h}_G$ of graph G. Existing GNNs usually follow the message passing scheme where $\mathbf{h}_v$ is iteratively updated by aggregating the representations of node v's neighbors. Formally, for an L-layer GNN, its l-th layer can be formulated as

$$\mathbf{a}_v^{(l)} = \text{AGGREGATE}^{(l)}\left(\left\{\mathbf{h}_u^{(l-1)} : u \in \mathcal{N}(v)\right\}\right), \quad (1)$$

$$\mathbf{h}_v^{(l)} = \text{UPDATE}^{(l)}\left(\mathbf{h}_v^{(l-1)}, \mathbf{a}_v^{(l)}\right), \quad (2)$$

where $\mathbf{h}_v^{(l)}$ is the representation vector of node v output by the l-th layer, $\mathcal{N}(v)$ is the set of node v's neighbors. Different AGGREGATE and UPDATE strategies are adopted in different GNN variants and sometimes can be integrated together (Kipf and Welling 2017; Hamilton, Ying, and Leskovec 2017; Gilmer et al. 2017; Xu et al. 2018).

Specifically, for graph classification, the graph representation $\mathbf{h}_G$ can be further obtained by aggregating all the node representations involved in graph G via various graph-level readout functions, such as summation and mean pooling.

Federated Learning

In a vanilla FL setting with M clients, the m-th client owns a private dataset $\mathcal{D}_m$. The global objective of the FL framework is

$$\min_{(w_1, w_2, \cdots, w_M)} \frac{1}{M} \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{N} \mathcal{L}_m\left(w_m; \mathcal{D}_m\right), \quad (3)$$

where N is the total number of instances over all clients, $\mathcal{L}_m$ and w_m are the loss function and model parameters of client m, respectively.

Standard FL methods aim to learn a globally shared model $w = w_1 = w_2 = \cdots = w_M$. The representative method is FedAvg (McMahan et al. 2017) which periodically aggregates the model parameters of all clients at the server by

$$\overline{w} \leftarrow \sum_{m=1}^{M} \frac{|\mathcal{D}_m|}{N} w_m \quad (4)$$

and return the averaged model back to clients. However, this kind of methods may have poor performance due to non-identical distributions across clients. Recent studies address this problem by applying personalized techniques to local training and/or global aggregation procedure, which allows w_m to perform better on the local data of client m.

In a federated graph learning (FGL) framework, w_m and $\mathcal{D}_m$ refer to the parameter set of GNN model and the graph dataset at client m, respectively. Since we focus on classification tasks over non-IID graph datasets in this paper, *e.g.*, protein function prediction, the loss function $\mathcal{L}_m$ in Eq. 3 is the commonly used cross-entropy loss.

Methodology

In this section, we present our FedStar framework in detail. Given a set of clients that own various graph datasets in the federated graph learning (FGL) framework, our goal is to enable each client to achieve higher performance on their own dataset by exploring universally shared underlying knowledge with other clients. To achieve the above goal by sharing structural knowledge across clients, we need to answer the following three questions:

- How to model structure information and capture universal structural patterns across different domains?

- How to design the GNN architecture to encode the feature and structure information, respectively?

- How to conduct structural knowledge sharing during the federated learning stage?

To answer the above questions, we first provide the definition of structure embedding and how it is initialized in our proposed method. Next, we illustrate the proposed feature-structure decoupled GNN architecture that learns attributive and structural knowledge with two channels, respectively. Finally, we discuss how to share structural knowledge with our proposed FedStar framework.

Structure Embedding Initialization

In most conventional GNNs, the structure information is *implicitly* encoded into node representations along with feature information through the procedure of feature aggregation. However, in non-IID FGL scenarios where features are usually heterogeneous across domains, it is difficult to uniformly encode general structure information (*e.g.*, node degree) into feature-based representations from different spaces. To bridge the gap, we introduce a novel type of node-level information carrier, namely **structure embedding**, to

(a) Feature-structure decoupled GNN
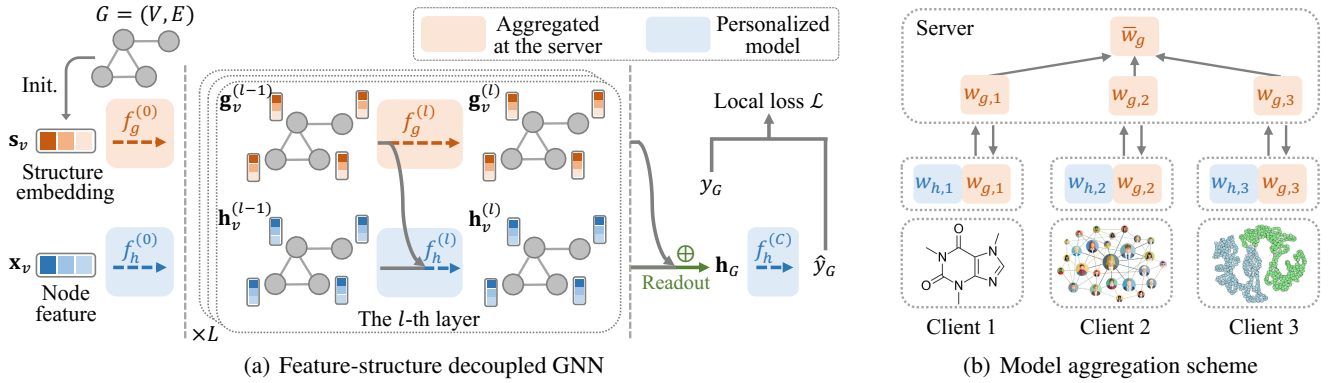
(b) Model aggregation scheme

Figure 2: (a) An overview of the local GNN architecture in our proposed FedStar. Blue boxes correspond to the models that are trained locally, and personalized for each client. Orange boxes correspond to the models that are aggregated at the server, with knowledge shared across clients. (b) An illustration of the data distribution and model aggregation scheme.

explicitly represent the universal structural knowledge in a vectorial form. The structure embedding can be initialized from the graph structure, and plays the role of "features" during message aggregation. In this case, the structure information is learned independently to feature information.

To capture general and comprehensive structure information, in FedStar, we construct structure embedding that incorporates both local and global structural patterns. In particular, to capture local structural knowledge, we introduce a degree-based structure embedding (DSE) which uses vertex degrees in the form of one-hot encoding. For node v in a graph, its DSE is denoted as

$$\mathbf{s}_v^{\text{DSE}} = [\mathbb{I}(d_v = 1), \mathbb{I}(d_v = 2), \cdots, \mathbb{I}(d_v \geq k_1)] \in \mathbb{R}^{k_1},$$
(5)

where d_v is the degree of node v, $\mathbb{I}$ is the identity function, and k_1 is the dimension of DSE. The benefit of using DSE to express local structure information is three-fold. First, degree is a basic geometric property that can be extracted from graphs of any domain. Second, the distributions of degree on different graph datasets share some similar properties, *e.g.*, the power law of degree distribution with different exponent values (Albert and Barabási 2002; Qiu et al. 2020). Last but not least, degree-based one-hot embedding is computationally friendly, which avoids high computational costs for embedding initialization.

In the meantime, to capture global structural knowledge, we introduce a random walk-based structure embedding (RWSE) which is computed based on the random walk diffusion process (Tong, Faloutsos, and Pan 2006; Dwivedi et al. 2021; Liu et al. 2023b). Concretely, RWSE is denoted as

$$\mathbf{s}_v^{\text{RWSE}} = \left[\mathbf{T}_{ii}, \mathbf{T}_{ii}^2, \cdots, \mathbf{T}_{ii}^{k_2} \right] \in \mathbb{R}^{k_2}$$
(6)

where $\mathbf{T} = \mathbf{A}\mathbf{D}^{-1}$ is a random walk transition matrix computed by binary adjacency matrix $\mathbf{A}$ and diagonal degree matrix $\mathbf{D}$, i is the index of node v, and k_2 is the dimension of RWSE, denoting that neighbors within k_2 hops are involved during the diffusion process and contribute to the structure representation $\mathbf{s}_v^{\text{RWSE}}$ (Dwivedi et al. 2021). The k-th element of $\mathbf{s}_v^{\text{RWSE}}$ refers to the landing probability of node v to

itself at a k-step random walk. Different from DSE that focuses on local geometric property, RWSE characterizes the unique role (*e.g.*, tail node or central node) of nodes from a global perspective. Moreover, RWSE is capable of incorporating domain-invariant structure information, such as the distribution of neighborhoods at different hops.

Finally, the structure embedding can be obtained by concatenating both DSE and RWSE and represented as

$$\mathbf{s}_v = \text{concat}[\mathbf{s}_v^{\text{DSE}}, \mathbf{s}_v^{\text{RWSE}}].$$
(7)

It is worth noting that FedStar is agnostic to the construction of structure embedding, indicating that FedStar can be equipped with more types of structure-related embeddings, such as Laplacian eigenvectors (Dwivedi et al. 2021) and distance-based embeddings (You, Ying, and Leskovec 2019)]. We leave this exploration for future work.

Feature-Structure Decoupled GNN

In most existing federated graph learning (FGL) frameworks (Xie et al. 2021; Zhang et al. 2021b), the local encoders are usually defined as single-channel GNNs, such as GraphSAGE (Hamilton, Ying, and Leskovec 2017) and GIN (Xu et al. 2019), where the only pathway to generate node representations starts from the raw features. However, in FGL, especially the inter-graph task over non-IID graphs, the local datasets of clients are usually from different domains *e.g.*, molecules, protein, and social networks (Morris et al. 2020), resulting in their diverse feature distributions. In this case, if we directly build and share the feature-based encoders across heterogeneous clients, the personalized performance would be seriously hurt due to the misaligned feature/representation spaces. Moreover, although we have already obtained the structure embeddings containing shareable structure information, how to further extract universal high-level structural knowledge apart from domain-specific features still remains challenging for single-channel GNNs.

To address the above limitations, inspired by (Dwivedi et al. 2021), we propose a feature-structure decoupled GNN that learns attributive and structural knowledge separately with two parallel channels. In the feature-based channel, the

feature encoder learns from raw node features and generates hidden embeddings by aggregating neighboring attributive and structural information in each layer. Meanwhile, in the structure-based channel, the structure encoder learns on top of structure embeddings and propagates structure information over the graph. An overview of the proposed GNN is illustrated in Fig. 2(a). The GNN model is composed of three parts, (i) $f_h^{(0)}$ and $f_g^{(0)}$: linear layers with learnable parameters $w_h^{(0)}$ and $w_g^{(0)}$; (ii) $f_h^{(l)}$ and $f_g^{(l)}$, $l \in \{1, \cdots, L\}$: L stacked feature-structure decoupled GNN layers with each layer parameterized by $w_h^{(l)}$ and $w_g^{(l)}$; (iii) $f_h^{(C)}$: the classifier parameterized by $w_h^{(C)}$.

Given an input graph $G = (V, E)$, we take a node $v \in V$ with feature vector $\mathbf{x}_v$ and structure embedding $\mathbf{s}_v$ as an example. At the first step, we transform $\mathbf{x}_v$ and $\mathbf{s}_v$ into the corresponding input embeddings $\mathbf{h}_v^{(0)}$ and $\mathbf{g}_v^{(0)}$ by two parallel linear layers $f_h^{(0)}$ and $f_g^{(0)}$, respectively. With such linear layers, the inputs of two channels are unified into vectors with a consistent dimension.

Then, the feature and structure embedding $\mathbf{h}_v^{(0)}$ and $\mathbf{g}_v^{(0)}$ are fed into the L-layer GNN. At the l-th layer of structure encoder, denoted as $f_g^{(l)}$, the output hidden structure embedding $\mathbf{g}_v^l$ is generated by aggregating and transforming the hidden structure embeddings output by the previous layer, *i.e.*, $\mathbf{g}_v^{(l-1)}$ and $\{\mathbf{g}_u^{(l-1)} : u \in \mathcal{N}(v)\}$. Note that the structure encoder is only used to learn structural knowledge and hence would not take any signal from features as its input. Meanwhile, $f_h^{(l)}$, the l-th layer of feature encoder, takes the concatenation of hidden structure embeddings and hidden feature embeddings output by the previous layer as its input, and produces its output hidden feature embedding $\mathbf{h}_v^{(l)}$ with a message propagation-based GNN layer. In this way, the structural knowledge learned by the structure encoder can be further leveraged to generate feature embeddings, providing supplementary information for representation learning. Through our dual-channel GNN, the structure information can be learned independently, benefiting the feature learning procedure.

After the L stacked layers, all the node-level hidden feature/structure embeddings output by the last GNN layer, denoted by $\{\mathbf{h}_v^{(L)} : v \in V\}$ and $\{\mathbf{g}_v^{(L)} : v \in V\}$, are first concatenated together and then transformed to the graph-level embedding $\mathbf{h}_G$ of graph G with a readout function. In the final step, we optimize the local model with a cross-entropy loss $\mathcal{L}_m$ following the procedures of graph classification tasks. The set of learnable parameters at the m-th client is denoted by $w_m = \{w_{h,m}, w_{g,m}\}$ where $w_{h,m}$ refers to the parameters in both the feature encoder and the classifier and $w_{g,m}$ refers to those in the structure encoder. They are updated during the local training process as

$$w_{h,m}^*, w_{g,m}^* = \underset{w_{h,m}, w_{g,m}}{\arg\min} \mathcal{L}_m(w_m; \mathcal{D}_m). \qquad (8)$$

Discussion: Compared to conventional single-channel GNNs, the feature-structure decoupled GNN enjoys the following advantages in non-IID FGL scenarios. First, through

learning structure embeddings with an independent structure encoder, the structure information of data in various domains can be projected into the same representation space. In this case, the structural knowledge can be universally shared across different domains without the side-effect of unaligned feature space. Moreover, the feature encoder can focus on learning the domain-specific knowledge from the local feature space, and further benefit from structural knowledge.

Structural Knowledge Sharing

Based on the universal structure information captured by structure embeddings and structure encoders, FedStar aims to share the learned structural knowledge across clients with data from diverse domains. Fig. 2(b) illustrates the pipeline of our learning paradigm, where clients own graph datasets from different domains and the same model architecture. For the m-th client, the local model with parameters w_m can be explicitly decoupled as two submodels, *i.e.*, the feature encoder with parameters $w_{h,m}$ and the structure encoder with parameters $w_{g,m}$. Our core idea is to share $w_{g,m}$ with the FL framework while keeping $w_{h,m}$ being trained locally.

The overall training procedure of FedStar is similar to FedAvg (McMahan et al. 2017) where the local training at the client-side and central aggregation at the server are iteratively conducted. First, clients transmit their structure encoder parameters $\{w_{g,m}\}|_{m=1}^M$ to the server. Next, the server performs a weighted average of these local parameters to obtain the global structure encoder $\overline{w}_g$,

$$\overline{w}_g = \sum_{m=1}^M \frac{|\mathcal{D}_m|}{N} w_{g,m}, \qquad (9)$$

where $|\mathcal{D}_m|$ refers to the number of graphs in the local dataset of client m and N refers to the total number of graphs across all clients. Then, the server returns $\overline{w}_g$ to clients, and clients update their local structure encoder by $\overline{w}_g$ and start local training for the next round.

Such a structural knowledge sharing scheme helps clients to build a generalized structure encoder. Since the structural encoder is purely based on the structural information without any feature information, it is prone to capture more domain-invariant patterns behind the graphs. Then, with the aggregation at the server, FedStar tends to extract the common knowledge maintained by these structure encoders. Through iterative training and aggregating the structure-based encoder, the structural knowledge is generalized over non-IID graphs from various domains. Meanwhile, globally shared structural knowledge can further benefit the specific learning tasks at each client. With the shortcut that connects structure-based embedding to feature-based embedding at each layer, the underlying globally shared structural knowledge further guides the feature learning process as additional structure-aware information. To sum up, with our proposed structural knowledge sharing scheme, the FGL framework is capable of exploring the common structural knowledge as well as boosting the representation ability of domain-specific feature learning.

Setting (# domains)	SM(1)		BIO-SM(2)		BIO-SM-SN(3)		BIO-SN-CV(3)	
# datasets	7		10		13		9	
Accuracy	avg.	avg. gain	avg.	avg. gain	avg.	avg. gain	avg.	avg. gain
Local	75.38±2.26	-	71.09±1.21	-	69.37±3.05	-	66.91±2.84	-
FedAvg	75.26±2.00	-0.13	70.65±2.73	-0.44	68.92±2.12	-0.45	64.86±2.73	-2.05
FedProx	75.30±2.00	-0.08	70.75±2.26	-0.34	69.21±2.63	-0.16	65.18±2.01	-1.72
FedPer	77.09±3.36	1.70	71.97±1.97	0.88	69.37±2.92	-0.01	62.23±3.76	-4.67
FedSage	75.90±1.85	0.51	70.34±1.87	-0.74	69.55±2.15	0.18	67.95±1.87	1.04
GCFL	76.49±1.23	1.11	71.60±2.20	0.51	70.65±1.84	1.28	66.31±2.36	-0.60
FedStar (Ours)	**79.79±2.44**	**4.41**	**74.54±2.50**	**3.46**	**72.16±2.43**	**2.78**	**69.49±1.81**	**2.58**

Table 1: Performance on different federated graph classification tasks. In each task/setting, there are multiple datasets owned by different clients. These datasets come from one or multiple domains.

Experiments

Experimental Setup

Datasets. Following the settings in (Xie et al. 2021), we use 16 public graph classification datasets from four different domains, including Small Molecules (MUTAG, BZR, COX2, DHFR, PTC_MR, AIDS, NCI1), Bioinformatics (ENZYMES, DD, PROTEINS), Social Networks (COLLAB, IMDB-BINARY, IMDB-MULTI), and Computer Vision (Letter-low, Letter-high, Letter-med) (Morris et al. 2020). To simulate the data heterogeneity in FGL, we create four non-IID settings, *i.e.*, (1) cross-dataset setting based on seven small molecules datasets (SM); (2)-(4) cross-domain setting based on datasets from two or three domains (BIO-SM, BIO-SM-SN, BIO-SN-CV). In each of the settings, a client owns one of the corresponding datasets and randomly splits it into three parts: 80% for training, 10% for validation, and 10% for testing.

Baselines. We compare FedStar with six baselines including (1) **Local** where clients train their model locally; (2) **FedAvg** (McMahan et al. 2017), the standard FL algorithm; (3) **FedProx** (Li et al. 2020b) and (4) **FedPer** (Arivazhagan et al. 2019) that deal with heterogeneity issues in FL; (5) **FedSage** (Zhang et al. 2021b) and (6) **GCFL** (Xie et al. 2021), two state-of-the-art FGL methods.

Implementation Details. We use a three-layer GCN (Kipf and Welling 2017) as the structure encoder and a three-layer GIN (Xu et al. 2019) as the feature encoder, both with the hidden size of 64. The dimension of DSE and RWSE, denoted as k_1 and k_2, are both set to be 16. The local epoch number and batch size are 1 and 128, respectively. We use an Adam (Kingma and Ba 2014) optimizer with weight decay 5e-4 and learning rate 0.001. The number of communication rounds is 200 for all FL methods. We report the results with the average over 5 runs of different random seeds. We implement all the methods using PyTorch and conduct all experiments on one NVIDIA Tesla V100 GPU. More implementation details about the model architecture, datasets, hyper-parameters, and baselines can be found in Appendix A. The code of FedStar is available at https://github.com/yuetan031/FedStar.

Sharing	DC	Setting (# domains)		
		BIO-SM	BIO-SM-SN	BIO-SN-CV
All	-	70.86±2.25	69.32±2.42	65.23±2.52
None	-	71.59±1.93	69.42±3.06	68.17±3.04
All	✓	71.97±2.14	69.85±2.43	65.78±4.25
None	✓	74.08±2.45	71.30±1.89	68.76±2.24
FE	✓	71.00±3.51	68.53±2.74	64.14±2.73
SE(Ours)	✓	**74.54±2.50**	**72.15±2.43**	**69.49±1.81**

Table 2: Ablation studies on the effects of decoupling and sharing mechanisms. FE and SE stand for feature encoder and structure encoder, respectively. FE: feature encoder; All: all learnable parameters; DC: decoupling.

Experimental Results

Performance Comparison. We show the federated graph classification results of all methods under four non-IID settings, including one cross-dataset setting (SM) and three cross-domain settings (BIO-SM, BIO-SM-SN, BIO-SN-CV). We summarize the final average test accuracy and its average gain compared with Local in Table 1. It suggests that FedStar outperforms all the baselines by a notable margin. Conventional FL methods such as FedAvg and FedProx fail to surpass Local due to their inevitable performance deterioration in non-IID settings, while personalized FL algorithm FedPer achieves better performance because only partial learnable parameters are aggregated at the server, alleviating the deterioration issue. Two FGL methods, *i.e.*, FedSage and GCFL, perform better than Local in most cases, owing to their unique designs for FGL tasks. For example, GCFL introduces the clustering scheme to aggregate within similar clients.

Convergence Analysis. Fig. 3 shows the curves of the average test accuracy with standard deviation during the training process across five random runs, including the results of all the baselines. It can be observed that in all the four non-IID settings, FedStar dominates the other methods on the average test accuracy and achieves a faster convergence, suggesting that it is much easier to align the structure-based encoder compared with aligning the feature-based encoder.

Effects of Decoupling and Sharing Mechanisms. We compare several variants of FedStar by sharing different components of the local GNN (*i.e.*, all parameters, fea-

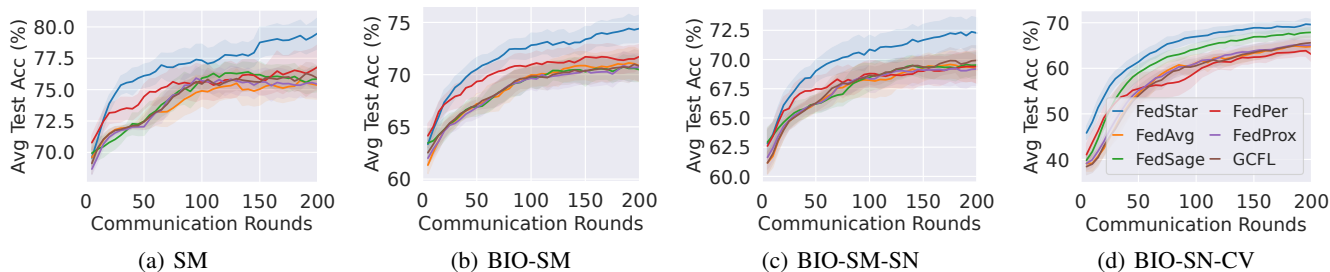(a) SM　　　　　(b) BIO-SM　　　　　(c) BIO-SM-SN　　　　　(d) BIO-SN-CV

Figure 3: Test accuracy curves of our proposed FedStar and five FL/FGL methods along the communication rounds. Each subfigure corresponds to a specific cross-dataset (a) or cross-domain (b,c,d) non-IID settings.

DSE	RWSE	Setting (# domains)		
		BIO-SM	BIO-SM-SN	BIO-SN-CV
-	-	69.51±2.25	69.64±1.92	66.05±2.92
✓	-	74.42±3.15	72.05±2.82	69.25±2.41
-	✓	72.74±3.44	70.48±3.37	67.23±2.74
✓	✓	**74.54±2.50**	**72.15±2.43**	**69.49±1.81**

Table 3: An ablation study on varying structure embeddings.

Epochs	1	2	3	4
FedAvg	75.26	**76.59**	76.96	76.36
GCFL	**76.49**	76.23	75.88	76.27
FedStar	79.79	80.26	79.90	**80.58**

Table 4: The performance on cross-dataset non-IID setting (SM), with varying local epochs.

ture encoder, structure encoder, and none) and/or involving feature-structure decoupled scheme or not. For the variants without decoupling scheme, we alternatively remove the structure encoder and concatenate the structure embedding with the raw feature for once. In Table 2, we observe that the feature-structure decoupling scheme brings about 0.5%-2.4% improvement to the cases without it, indicating that independently learning structural and attributive knowledge can improve FGL performance in general. Moreover, compared with pure local training, sharing all parameters or the feature encoder results in performance degradation. Such an observation illustrates that sharing feature information across heterogeneous datasets perturbs the FGL procedure, supporting our interpretation in Fig. 1(a). Finally, we find that only sharing the structure encoder leads to the best result, validating the effectiveness of the structural knowledge sharing paradigm.

Effects of Different Structure Embeddings. To understand how the explicit structure embedding contributes to the final performance, we carry out an ablation study by varying the components of structure embeddings. For the variant without DSE and RWSE, we take all-ones vectors as the structure embedding. As shown in Table 3, by concatenating DSE and RWSE, the best performance is achieved in all the three non-IID settings. It also suggests that, DSE plays a more important role than RWSE when considering only one structure embedding, which means that local structural information is more prone to be globally shared by different domains than global structural information. Moreover, the contribution made by RWSE is still not negligible and can benefit the structure learning.
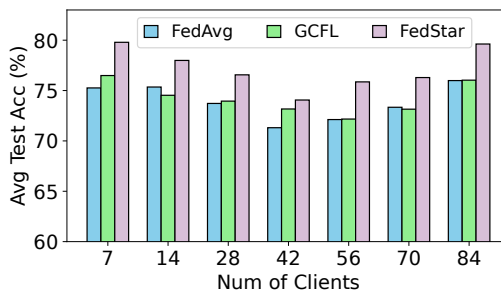

Figure 4: The performance on cross-dataset non-IID setting (SM), with varying numbers of clients.

Varying Local Epochs. In FL, clients can perform multiple local training epochs before global aggregation to reduce the communication costs. In Table 4, we provide the results of FedAvg, GCFL, and FedStar on the non-IID setting SM where there are seven clients in total, each of which owns a unique small molecule dataset. The results show that FedStar benefits more from the increasing local epochs and thus more communication-efficient.

Varying Numbers of Clients. To test the performance of FedStar in a larger FGL framework, we increase the number of clients from 7 to 84 under the SM non-IID setting where each of the seven small molecules datasets are split to 1-12 shards and then each shard is assigned to one client. In Fig. 4, as the number of clients increases, the performance of all the three algorithms first drops because of the further divergence when there are more clients participating training, and then improves because the local task in each client becomes easier when the local dataset is small enough. It suggests that FedStar outperforms the other two algorithms in both small- and large-scale FGL systems.

Conclusion

In this paper, we propose a novel federated graph learning (FGL) framework, namely FedStar, that solves the non-IID issues via structural knowledge sharing. Based on structure embeddings and feature-structure decoupled GNN, structural knowledge is explicitly represented and well captured without being influenced by the feature-based information. Through sharing the structure encoder within the FL framework, our proposed framework enables clients from different domains to learn domain-agnostic structural knowledge globally while keeping their feature-based node representation learning personalized. Experimental results illustrate that FedStar consistently outperforms the state-of-the-art methods in various non-IID FGL scenarios.

References

Albert, R.; and Barabási, A.-L. 2002. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1): 47.

Arivazhagan, M. G.; Aggarwal, V.; Singh, A. K.; and Choudhary, S. 2019. Federated Learning with Personalization Layers. *arXiv:1912.00818*.

Baek, J.; Jeong, W.; Jin, J.; Yoon, J.; and Hwang, S. J. 2022. Personalized Subgraph Federated Learning. *arXiv preprint arXiv:2206.10206*.

Bai, L.; Yao, L.; Li, C.; Wang, X.; and Wang, C. 2020. Adaptive graph convolutional recurrent network for traffic forecasting. In *NeruIPS*, volume 33, 17804–17815.

Chen, C.; Liu, Y.; Ma, X.; and Lyu, L. 2022a. CalFAT: Calibrated Federated Adversarial Training with Label Skewness. In *NeurIPS*.

Chen, C.; Lyu, L.; Yu, H.; and Chen, G. 2022b. Practical Attribute Reconstruction Attack Against Federated Learning. *IEEE Transactions on Big Data*.

Chen, F.; Long, G.; Wu, Z.; Zhou, T.; and Jiang, J. 2022c. Personalized Federated Learning With Graph. In *IJCAI*.

Chen, H.-Y.; and Chao, W.-L. 2022. On Bridging Generic and Personalized Federated Learning for Image Classification. In *ICLR*.

Chen, M.; Zhang, W.; Yuan, Z.; Jia, Y.; and Chen, H. 2021. Fede: Embedding knowledge graphs in federated setting. In *The 10th International Joint Conference on Knowledge Graphs*, 80–88.

Dwivedi, V. P.; Luu, A. T.; Laurent, T.; Bengio, Y.; and Bresson, X. 2021. Graph Neural Networks with Learnable Structural and Positional Representations. In *ICLR*.

Erdős, P.; Rényi, A.; et al. 1960. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1): 17–60.

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

Gaudelet, T.; Day, B.; Jamasb, A. R.; Soman, J.; Regep, C.; Liu, G.; Hayter, J. B.; Vickers, R.; Roberts, C.; Tang, J.; et al. 2021. Utilizing graph machine learning within drug discovery and development. *Briefings in bioinformatics*, 22(6): bbab159.

Ghosh, A.; Chung, J.; Yin, D.; and Ramchandran, K. 2020. An Efficient Framework for Clustered Federated Learning. In *NeurIPS*.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *ICML*, 1263–1272. PMLR.

Hamer, J.; Mohri, M.; and Suresh, A. T. 2020. Fedboost: A communication-efficient algorithm for federated learning. In *ICML*, 3973–3983. PMLR.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *NeruIPS*.

Hu, W.; Liu, B.; Gomes, J.; Zitnik, M.; Liang, P.; Pande, V.; and Leskovec, J. 2020a. Strategies for Pre-training Graph Neural Networks. In *ICLR*.

Hu, Z.; Dong, Y.; Wang, K.; Chang, K.-W.; and Sun, Y. 2020b. Gpt-gnn: Generative pre-training of graph neural networks. In *SIGKDD*, 1857–1867.

Jiang, M.; Jung, T.; Karl, R.; and Zhao, T. 2022. Federated Dynamic Graph Neural Networks with Secure Aggregation for Video-based Distributed Surveillance. *ACM TIST*, 13(4): 1–23.

Jiang, Y.; Konečný, J.; Rush, K.; and Kannan, S. 2019. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*.

Kairouz, P.; McMahan, H. B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A. N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. 2021. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2): 1–210.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

Leskovec, J.; Kleinberg, J.; and Faloutsos, C. 2005. Graphs over time: densification laws, shrinking diameters and possible explanations. In *SIGKDD*, 177–187.

Li, P.; Wang, Y.; Wang, H.; and Leskovec, J. 2020a. Distance encoding: Design provably more powerful neural networks for graph representation learning. In *NeruIPS*, volume 33, 4465–4478.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2020b. Federated optimization in heterogeneous networks. *MLSys*.

Li, X.; Jiang, M.; Zhang, X.; Kamp, M.; and Dou, Q. 2020c. FedBN: Federated Learning on Non-IID Features via Local Batch Normalization. In *ICLR*.

Liu, Y.; Ding, K.; Liu, H.; and Pan, S. 2023a. GOOD-D: On Unsupervised Graph Out-Of-Distribution Detection. In *WSDM*.

Liu, Y.; Zheng, Y.; Zhang, D.; Chen, H.; Peng, H.; and Pan, S. 2022. Towards unsupervised deep graph structure learning. In *WWW*, 1392–1403.

Liu, Y.; Zheng, Y.; Zhang, D.; Lee, V.; and Pan, S. 2023b. Beyond Smoothing: Unsupervised Graph Representation Learning with Edge Heterophily Discriminating. In *AAAI*.

Long, G.; Xie, M.; Shen, T.; Zhou, T.; Wang, X.; and Jiang, J. 2022. Multi-center federated learning: clients clustering for better personalization. *World Wide Web*, 1–20.

Lou, G.; Liu, Y.; Zhang, T.; and Zheng, X. 2021. STFL: A temporal-spatial federated learning framework for graph neural networks. *arXiv preprint arXiv:2111.06750*.

Luo, M.; Chen, F.; Hu, D.; Zhang, Y.; Liang, J.; and Feng, J. 2021. No Fear of Heterogeneity: Classifier Calibration for Federated Learning with Non-IID Data. In *NeurIPS*.

Lyu, L.; Yu, H.; Ma, X.; Chen, C.; Sun, L.; Zhao, J.; Yang, Q.; and Philip, S. Y. 2022. Privacy and robustness in federated learning: Attacks and defenses. *IEEE TNNLS*.

Ma, J.; Long, G.; Zhou, T.; Jiang, J.; and Zhang, C. 2022. On the convergence of clustered federated learning. *arXiv preprint arXiv:2202.06187*.

McMahan, H. B.; Moore, E.; Ramage, D.; et al. 2017. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*.

Meng, C.; Rambhatla, S.; and Liu, Y. 2021. Cross-node federated graph neural network for spatio-temporal data modeling. In *SIGKDD*, 1202–1211.

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

Qiu, J.; Chen, Q.; Dong, Y.; Zhang, J.; Yang, H.; Ding, M.; Wang, K.; and Tang, J. 2020. GCC: Graph contrastive coding for graph neural network pre-training. In *SIGKDD*, 1150–1160.

Qu, Z.; Li, X.; Duan, R.; Liu, Y.; Tang, B.; and Lu, Z. 2022. Generalized Federated Learning via Sharpness Aware Minimization. *arXiv preprint arXiv:2206.02618*.

Suzumura, T.; Zhou, Y.; Baracaldo, N.; Ye, G.; Houck, K.; Kawahara, R.; Anwar, A.; Stavarache, L. L.; Watanabe, Y.; Loyola, P.; et al. 2019. Towards federated graph learning for collaborative financial crimes detection. *arXiv preprint arXiv:1909.12946*.

Tan, Y.; Long, G.; Liu, L.; Zhou, T.; Lu, Q.; Jiang, J.; and Zhang, C. 2022a. FedProto: Federated prototype learning over heterogeneous devices. In *AAAI*.

Tan, Y.; Long, G.; Ma, J.; Liu, L.; Zhou, T.; and Jiang, J. 2022b. Federated Learning from Pre-Trained Models: A Contrastive Learning Approach. In *NeurIPS*.

Tong, H.; Faloutsos, C.; and Pan, J.-Y. 2006. Fast random walk with restart and its applications. In *ICDM*, 613–622. IEEE.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *ICLR*.

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

Wang, J.; Liu, Q.; Liang, H.; Joshi, G.; and Poor, H. V. 2020b. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. *NeruIPS*.

Wu, S.; Sun, F.; Zhang, W.; Xie, X.; and Cui, B. 2020a. Graph neural networks in recommender systems: a survey. *ACM CSUR*.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020b. A comprehensive survey on graph neural networks. *IEEE TNNLS*, 32(1): 4–24.

Xie, H.; Ma, J.; Xiong, L.; and Yang, C. 2021. Federated graph classification over non-iid graphs. *NeruIPS*, 34.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? In *ICLR*.

Xu, K.; Li, C.; Tian, Y.; Sonobe, T.; Kawarabayashi, K.-i.; and Jegelka, S. 2018. Representation learning on graphs with jumping knowledge networks. In *ICML*, 5453–5462. PMLR.

Yang, Q.; Liu, Y.; Chen, T.; and Tong, Y. 2019. Federated machine learning: Concept and applications. *ACM TIST*, 10(2): 1–19.

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

Yuan, H.; Morningstar, W. R.; Ning, L.; and Singhal, K. 2021. What Do We Mean by Generalization in Federated Learning? In *ICLR*.

Zhang, H.; Shen, T.; Wu, F.; Yin, M.; Yang, H.; and Wu, C. 2021a. Federated Graph Learning–A Position Paper. *arXiv preprint arXiv:2105.11099*.

Zhang, K.; Yang, C.; Li, X.; Sun, L.; and Yiu, S. M. 2021b. Subgraph federated learning with missing neighbor generation. In *NeruIPS*, volume 34, 6671–6682.

Zhu, Q.; Yang, C.; Xu, Y.; Wang, H.; Zhang, C.; and Han, J. 2021. Transfer learning of graph neural networks with ego-graph information maximization. In *NeruIPS*, volume 34, 1766–1779.

Zhu, W.; Luo, J.; and White, A. D. 2022. Federated learning of molecular properties with graph neural networks in a heterogeneous setting. *Patterns*, 100521.