Virtual Nodes Can Help: Tackling Distribution Shifts in Federated Graph Learning

Xingbo Fu, Zihan Chen, Yinhan He, Song Wang, Binchi Zhang, Chen Chen, Jundong Li
University of Virginia
{xf3av, brf3rx, nee7ne, sw3wv, epb6gw, zrh6du, jundong}@virginia.edu

Abstract

Federated Graph Learning (FGL) enables multiple clients to jointly train powerful graph learning models, e.g., Graph Neural Networks (GNNs), without sharing their local graph data for graph-related downstream tasks, such as graph property prediction. In the real world, however, the graph data can suffer from significant distribution shifts across clients as the clients may collect their graph data for different purposes. In particular, graph properties are usually associated with invariant label-relevant substructures (i.e., subgraphs) across clients, while label-irrelevant substructures can appear in a client-specific manner. The issue of distribution shifts of graph data hinders the efficiency of GNN training and leads to serious performance degradation in FGL. To tackle the aforementioned issue, we propose a novel FGL framework entitled FedVN that eliminates distribution shifts through client-specific graph augmentation strategies with multiple learnable Virtual Nodes (VNs). Specifically, FedVN lets the clients jointly learn a set of shared VNs while training a global GNN model. To eliminate distribution shifts, each client trains a personalized edge generator that determines how the VNs connect local graphs in a client-specific manner. Furthermore, we provide theoretical analyses indicating that FedVN can eliminate distribution shifts of graph data across clients. Comprehensive experiments on four datasets under five settings demonstrate the superiority of our proposed FedVN over nine baselines.

Code — https://github.com/xbfu/FedVN

1 Introduction

Graphs are pervasive in a wide range of real-world scenarios, including bioinformatics (Yang et al. 2020; Yuan and Bar-Joseph 2020), healthcare systems (Cui et al. 2020; Fu et al. 2023), and fraud detection (Motie and Raahemi 2024; Zheng et al. 2024). As a dominant approach for modeling graph data, Graph Neural Networks (GNNs) (Hamilton, Ying, and Leskovec 2017; Kipf and Welling 2017; Veličković et al. 2018; Xu et al. 2019) have demonstrated great prowess in graph representation learning and have been widely adopted in various graph-level applications, such as molecular property regression (Li et al. 2022; Zhuang et al. 2023) and 3D shape classification (Wang et al. 2019; Zhou et al. 2021). Specifically, GNNs follow a message-passing mechanism that recursively aggregates neighboring information of each node to obtain its node embedding. The representation of an entire graph can be obtained through different graph pooling operations (Xu et al. 2019; Ying et al. 2018), e.g., by summing all node embeddings in the graph. The final graph representation can then be used for graph property prediction, including graph classification and graph regression. Despite their superior performance, most GNNs are trained in a centralized manner where graph data need to be collected in a single machine before training. In the real world, however, a great number of graph data are stored by different data owners and cannot be shared due to data privacy (Voigt and Von dem Bussche 2017; Wang et al. 2024), which hampers us in training powerful GNNs.

Federated Learning (FL) (McMahan et al. 2017) is a decentralized learning paradigm where multiple clients collaboratively train machine learning models over their local data while preserving privacy. In this study, we focus on Federated Graph Learning (FGL) (Fu et al. 2022) that aims to train GNNs over distributed graph data from multiple clients in a federated fashion. One critical challenge in the context of FGL is that graph data can encounter significant distribution shifts across clients. In graph property prediction, the property of a graph (e.g., a molecule) is usually determined by its causal substructure (e.g., functional groups) (Lucic et al. 2022; Luo et al. 2020; Yuan et al. 2021). Typically, such causality can be consistent across all clients. Nonetheless, the non-causal substructure in a graph can vary significantly across clients in that the clients may collect graph data for different purposes. Consider a medical system with two institutes as an example. Institute A studies benzene-based compounds so it mainly has molecules containing the phenyl group (e.g., benzoic acid molecules). In contrast, Institute B studies ester-based compounds so it mainly has molecules containing an ester (e.g., glyceride molecules). The goal of the two institutes is to jointly train a GNN model to predict the water solubility of molecules. Typically, the label (i.e., the water solubility) of each molecule is causally determined by the substructure hydroxy (i.e., -OH), which is invariant across the two institutes, while the function groups - the phenyl group in Institute A and the ester in Institute B - are non-causal substructures and quite different across the two clients. Such distribution shifts of graph data caused by distinct non-causal substructures incur divergent information embedded in graph representations, which results in severe performance degradation of GNNs and consequently hinders further FGL deployments in practice.

Although numerous recent efforts attempt to grapple with the distribution shift of graph data from multiple sources for graph property prediction (Chen et al. 2022; Gui et al. 2023; Liu et al. 2022; Sui et al. 2023; Zhuang et al. 2023), these approaches are inapplicable to FGL since they require the multi-source graph data collected centrally in a mini-batch when training GNNs. In the context of FGL, only a few studies investigate the problem of graph property prediction in FGL (Tan et al. 2023; Xie et al. 2021). These studies propose to leverage common graph properties from different datasets or even divergent domains and enhance collaborative training of GNNs by either client clustering (Xie et al. 2021) or sharing structural knowledge (Tan et al. 2023). Nevertheless, in-depth analyses of FGL with the distribution shift of graph data have not been fully explored yet.

Our goal in this study is to essentially grapple with the aforementioned challenge at the data level. More concretely, each client will learn to manipulate its original graph data so that the altered graphs can facilitate collaborative training of GNNs. Hence, we naturally ask a question: *How to properly augment local graphs in each client so that the distribution shift of graph data in FGL can be eliminated?* To answer this question, we first need to devise a suitable graph augmentation strategy for this setting. As a common trick for graph augmentation, adding a Virtual Node (VN) to graphs has been adopted by recent studies (Hu et al. 2021, 2020) for graph property prediction. Typically, the VN is designed to uniformly connect to all original graph nodes and consequently improve the capacity of GNNs for capturing long-range dependencies in a graph (Cai et al. 2023). While the problem investigated in this paper is essentially different from the above studies, the role of VNs in FGL remains an untouched area that is worthy of extensive exploration.

In this study, we propose a novel FGL framework entitled FedVN to eliminate distribution shifts of graph data across clients. The intuition of FedVN is to let each client learn a client-specific graph augmentation strategy and enable global GNNs to be trained over identical augmented graphs across clients. Inspired by the idea of adding VNs, we propose to introduce multiple learnable VNs that are shared across clients in FedVN. To eliminate cross-client distribution shifts of graph data, each client augments its local graphs with these VNs by a personalized edge generator. To avoid collapsing to fewer VNs, we design a decoupling loss in FedVN that encourages VNs to diffuse across the feature space. Additionally, we propose a novel score-contrastive loss to guide the generated edges within a client following the same pattern. We provide theoretical analyses to show that our design in FedVN effectually eliminates the distribution shift issue of graph data in FGL. We conduct extensive empirical evaluations on four datasets under five settings. The results demonstrate that our proposed FedVN outshines nine SOTA baselines.

Our main contributions are summarized as follows:

- We take the first step towards investigating the distribution shift issue of graph data across clients for graph property prediction in FGL. In this work, we present a formal problem formulation of the studied issue.
- We propose a novel FGL framework FedVN to tackle the above problem by learning client-specific graph augmentation strategies with multiple VNs. We design a personalized edge generator that determines how the VNs connect local graphs so that the global GNN model can be trained over identical augmented graphs across clients.
- We provide theoretical analyses to support that our design in FedVN has the capability of eliminating the distribution shift issue in FGL.
- Extensive experiments are conducted on four datasets under five settings. The results validate the superiority of our proposed FedVN compared with other baselines.

2 Related Work

Federated graph learning. A plethora of recent efforts attempt to apply FL techniques to graph data and train powerful GNNs for various downstream tasks (Fu et al. 2024b,a). For example, FedLit (Xie, Xiong, and Yang 2023) investigates the issue of latent link-type heterogeneity for node classification. A few studies propose to recover cross-client missing information for training GNNs (Peng et al. 2022; Zhang et al. 2021). In terms of graph property prediction, the pioneering work (Xie et al. 2021) demonstrates that different graphs from different datasets or even different domains may share common properties. A subsequent work (Tan et al. 2023) proposes to design GNNs in a feature-structure decoupled manner and share the structure encoder across clients. However, the above two works do not perform in-depth analyses of the distribution shift issue with its impact on graph property prediction in FGL. One recent work (Wan, Huang, and Ye 2024) investigates FGL under domain shifts but only focuses on node-level tasks.

Virtual nodes in GNNs. A VN augments a graph by adding an extra node that is uniformly connected to all nodes in the original graphs (Gilmer et al. 2017). It is demonstrated that adding the VN is effective for graph property prediction in a handful of benchmarks (Hu et al. 2021, 2020) since it can help GNNs capture long-range dependencies in a graph (Cai et al. 2023). Additionally, a recent study (Hwang et al. 2022) investigates the role of VNs in link prediction and proposes to add multiple VNs based on node clusters.

Contrastive learning. As a representative scheme in self-supervised learning, contrastive learning has been widely adopted in a variety of domains, such as computer vision (Chen et al. 2020; Zbontar et al. 2021) and graph mining (Xu et al. 2024; Shen et al. 2023; Xu et al. 2023). The key idea of contrastive learning is to maximize the agreement between samples with shared semantic information (i.e., positive pairs) and minimize the agreement between irrelevant samples (i.e., negative pairs). In the context of FL, a few studies propose contrastive learning to enhance collaborative training based on different designs of sample pairs. For instance, FedPCL (Tan et al. 2022) performs contrastive learning between prototypes from different clients. FGSSL

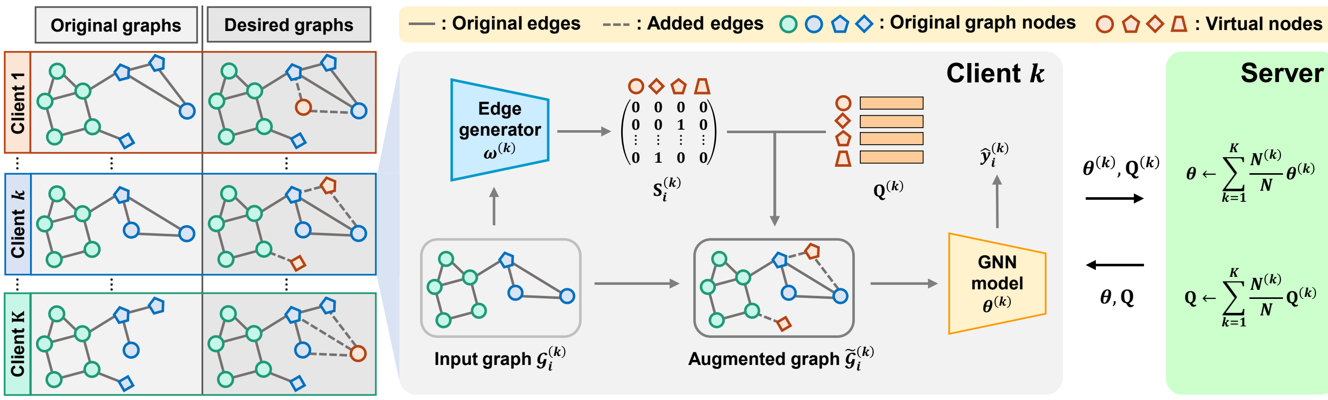

Figure 1: An overview of the proposed FedVN. FedVN aims to learn client-specific graph augmentation strategies by adding multiple virtual nodes through personalized edge generators so that the global GNN model can be trained over identical graphs.

(Huang et al. 2023) designs federated node semantic contrast from global and local views.

3 Preliminaries

3.1 Graph Neural Networks

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ denote a graph where $\mathcal{V}$ is the set of $|\mathcal{V}|$ nodes, $\mathcal{E}$ is the edge set, and $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times d_x}$ is the node feature matrix. d_x is the number of node features. In this study, we consider graph property prediction, i.e., predicting the graph-specific target y of a graph $\mathcal{G}$. Generally, GNNs follow a message-passing mechanism (Hamilton, Ying, and Leskovec 2017; Kipf and Welling 2017; Veličković et al. 2018; Xu et al. 2019) where each node in a graph iteratively aggregates information from its neighboring graph nodes to update its node representation. More specifically, the l-th layer of an L-layer GNN updates the representation of node $v \in \mathcal{V}$ by

$$\mathbf{h}_v^{(l)} = \mathrm{CB}_{\mathrm{gn}}^{(l)}(\mathbf{h}_v^{(l-1)}, \mathrm{AGG}_{\mathrm{gn}}^{(l)}(\{\mathbf{h}_u^{(l-1)} : u \in \mathcal{N}(v)\})), \quad (1)$$

where $\mathrm{AGG}_{\mathrm{gn}}^{(l)}(\cdot)$ represents the aggregation operation extracting the neighboring information of node v, $\mathrm{CB}_{\mathrm{gn}}^{(l)}(\cdot)$ represents the combination operation integrating the previous representation of node v and its neighboring information, and $\mathcal{N}(v)$ denotes the neighbors of node v. $\mathbf{h}_v^{(0)}$ is initialized with node v's feature $\mathbf{x}_v \in \mathbf{X}$. After the L-layer propagation, the READOUT operation pools the final node representations in a graph $\mathcal{G}$ through a permutation-invariant pooling function (e.g., sum or $mean$) to obtain its representation $\mathbf{h}_{\mathcal{G}}$, and its prediction $\hat{y}$ is computed based on its representation through a multi-layer perceptron (MLP) by

$$\hat{y} = \mathrm{MLP}(\mathbf{h}_{\mathcal{G}}), \text{ where } \mathbf{h}_{\mathcal{G}} = \mathrm{READOUT}(\{\mathbf{h}_v^{(L)} : v \in \mathcal{V}\}). \quad (2)$$

In this study, we use a GNN model for graph property prediction. The GNN model typically consists of a GNN (e.g., GIN (Xu et al. 2019)) as the encoder and an MLP as the prediction head, with the READOUT operation between them.

3.2 Federated Learning

We consider an FL system with a set of K clients. Each client k has its local dataset $\mathcal{D}^{(k)} = \{(x_i^{(k)}, y_i^{(k)})\}_{i=1}^{N^{(k)}}$ where $x_i^{(k)}$ is the input and $y_i^{(k)}$ is its corresponding ground-truth label. $N^{(k)}$ is the number of data samples in client k. $N = \sum_{k=1}^{K} N^{(k)}$ is the total number of samples in all clients. The goal of the clients is to jointly learn a global model f with parameters θ orchestrated by a central server. The global objective function is defined by

$$\min_{\theta} \mathcal{L}(\theta) := \sum_{k=1}^{K} \frac{N^{(k)}}{N} \mathcal{L}^{(k)}(\theta), \quad (3)$$

where $\mathcal{L}^{(k)}(\theta) = \mathbb{E}_{(x_i^{(k)}, y_i^{(k)}) \sim \mathcal{D}^{(k)}}[\ell(f(x_i^{(k)}; \theta), y_i^{(k)})]$ is the local expected risk function in client k. Here, $\ell(\cdot, \cdot)$ denotes the loss function, such as the cross-entropy loss for the classification task and the mean-square error loss for the regression task. In practice, each client usually performs multiple local updates to optimize θ via Stochastic Gradient Descent (SGD) on the empirical version of $\mathcal{L}^{(k)}(\theta) := \frac{1}{N^{(k)}} \sum_{i=1}^{N^{(k)}} \ell(f(x_i^{(k)}; \theta), y_i^{(k)})$ during each round (McMahan et al. 2017).

4 Method

In this section, we present our proposed framework FedVN tailored for eliminating the distribution shift issue in FGL. We first formulate the problem setup of FGL with distribution shift. We then provide an overview of FedVN, followed by the details of its three main components: a GNN model, VNs, and an edge generator.

4.1 Problem Setup

In this work, we focus on the task of graph-level property prediction in FGL. Given a set of K clients, each client k has its local graph dataset $\mathcal{D}^{(k)} = \{(\mathcal{G}_i^{(k)}, y_i^{(k)})\}_{i=1}^{N^{(k)}}$ drawn from its own graph data distribution $P^{(k)}(\mathcal{G}, y)$. As the clients may collect their local graph data for different purposes, FGL can suffer from significant distribution shifts, i.e., $P^{(k)}(\mathcal{G}, y) \neq P^{(j)}(\mathcal{G}, y)$ for client k and j, while $P^{(k)}(y) = P^{(j)}(y)$. Following many studies in graph out-of-distribution generalization (Gui et al. 2023; Sui et al. 2023; Zhuang et al. 2023), we assume that a given graph $\mathcal{G}$'s label is invariantly determined by its causal substructure $\mathcal{G}_c$. The remaining part of $\mathcal{G}$ is the non-causal substructure denoted by $\mathcal{G}_n$. In our FGL setting, we assume that the non-causal substructure shares a similar pattern across graphs within a client but differs significantly among the clients, i.e., $P^{(k)}(\mathcal{G}_n) \neq P^{(j)}(\mathcal{G}_n)$. The goal of these clients is to jointly train a GNN model f with parameters θ for graph property prediction.

4.2 Proposed Method: FedVN

To deal with the above problem, we propose a novel FGL framework FedVN in this section. Figure 1 illustrates an overview of FedVN. The intuition of FedVN is to let the clients manipulate their local graph data through learnable graph augmentation strategies in order that the global GNN model can be trained over identical manipulated graph data without any distribution shift across clients. To achieve this, the key point is to design a proper scheme for graph augmentation. Inspired by recent studies about VNs (Gilmer et al. 2017; Hu et al. 2021, 2020) in graph learning, we propose to learn graph augmentation with extra VNs to eliminate distribution shifts in FGL. More specifically, the clients in FedVN collaboratively learn a set of shared VNs while training a global GNN model. Considering the cross-client distribution shift, FedVN enables each client to learn a personalized edge predictor that determines how the VNs connect its local graphs. In the following, we will introduce the three components of FedVN in detail.

GNN model. The goal of FedVN is to train a global GNN model over distributed graph data from multiple clients. The GNN model could employ common GNNs like GIN (Xu et al. 2019) as the encoder. Different from existing FGL studies that directly train GNN models over raw graph data, we propose to train a GNN model over the manipulated graph data identical across clients by client-specific graph augmentation strategies. More specifically, given each graph $\mathcal{G}_i^{(k)}$ with its corresponding label $y_i^{(k)}$ in client k, the personalized graph augmentation $\phi^{(k)}$ learns to manipulate $\mathcal{G}_i^{(k)}$ into an augmented graph $\tilde{\mathcal{G}}_i^{(k)}$. Therefore, the local empirical risk to optimize θ in client k can be rewritten as the average supervised loss over augmented graphs by

$$\mathcal{L}_S^{(k)} = \frac{1}{N^{(k)}} \sum_{i=1}^{N^{(k)}} \ell(f(\tilde{\mathcal{G}}_i^{(k)}; \theta), y_i^{(k)}), \quad \text{where } \tilde{\mathcal{G}}_i^{(k)} = \phi^{(k)}(\mathcal{G}_i^{(k)}). \tag{4}$$

In this study, we propose to realize graph augmentation by adding multiple VNs into local graphs to obtain desired graphs for training the global GNN model.

Virtual nodes. Adding a VN that uniformly connects to all input graph nodes is an effective graph augmentation technique for graph-level tasks (Cai et al. 2023; Gilmer et al. 2017). However, the role of VNs introduced in our proposed FedVN is essentially different. VNs in FedVN are designed to eliminate the distribution shift of graph data across clients, while the previous studies use a single VN for modeling long-range dependencies within a graph (Cai et al. 2023) since the VN serves as a shortcut linking every two nodes in the graph. In fact, simply adding a VN may not effectually bridge divergences between graphs due to the following two challenges. First, a single VN can be inadequate for different graphs. Considering the clients in Figure 1, we may need at least three types of VNs to obtain the desired identical graphs in the clients. Second, a VN, such as the square VN in Client k, is usually not supposed to equally connect all the nodes in an input graph, which is different from what the VN typically does. These two challenges motivate us to conceive a nontrivial mechanism for integrating VNs into our proposed framework.

To deal with the above two challenges, we propose to augment local graphs by learning multiple VNs in FedVN. More concretely, the clients will jointly learn M shared VNs with the feature matrix $\mathbf{Q} \in \mathbb{R}^{M \times d_x}$. Each VN m has a unique feature vector $\mathbf{q}_m \in \mathbf{Q}$ with the same dimension as the original graph nodes. Additionally, each client will determine how the M VNs connect their local graphs in a personalized manner. Specifically, for each graph $\mathcal{G}_i^{(k)}$ in client k, we inject each VN m into $\mathcal{G}_i^{(k)}$ by assigning a weighted edge with weight $s_{v,m} \in [0, 1]$ between m and each graph node v. $s_{v,m}$ measures how significantly m and v impact each other, and they are disconnected when $s_{v,m} = 0$. With the VNs, we can rewrite Equation (1) in a modified version for each node $v \in \mathcal{V}_i^{(k)}$ by

$$\mathbf{h}_v^{(l)} = \text{CB}_{\text{gn}}^{(l)}(\mathbf{h}_v^{(l-1)}, \text{AGG}_{\text{gn}}^{(l)}(\{\mathbf{h}_u^{(l-1)} : u \in \mathcal{N}(v)\})) + \sum_{m=1}^{M} s_{v,m} \cdot \mathbf{h}_m^{(l-1)}, \tag{5}$$

where $\mathbf{h}_m^0$ is the feature vector $\mathbf{q}_m$ of VN m, and $\mathcal{V}_i^{(k)}$ denotes the node set in $\mathcal{G}_i^{(k)}$. In the meantime, each VN m updates its representation by aggregating information from every graph node v through

$$\mathbf{h}_m^{(l)} = \text{CB}_{\text{vn}}^{(l)}(\mathbf{h}_m^{(l-1)}, \text{AGG}_{\text{vn}}^{(l)}(\{s_{v,m} \cdot \mathbf{h}_v^{(l-1)} : v \in \mathcal{V}_i^{(k)}\})), \tag{6}$$

where $\text{AGG}_{\text{vn}}^{(l)}(\cdot)$ and $\text{CB}_{\text{vn}}^{(l)}(\cdot)$ represent the aggregation and combination operations for VNs, respectively.

One potential issue when augmenting local graphs with the VNs is that they are indistinguishable without any prior knowledge. Although the clients are seemingly learning multiple VNs, the VNs are prone to collapse to a single or fewer points (we also observe this collapsing in our empirical experiments). Hence, multiple VNs will be essentially equivalent to fewer VNs and consequently fail to tackle distribution shifts in FGL. To avoid collapsing solutions, we propose to decouple VNs during local training in FedVN. The intuition here is to compel the VNs to reside in the whole feature space as much as possible so that they become dissimilar to each other. We achieve this in FedVN by diminishing the correlation between each pair of VNs. Specifically, we design a decoupling loss (Shi et al. 2022) during local training with the Frobenius norm of local $\mathbf{Q}$'s correlation matrix in each client k by

$$\mathcal{L}_V^{(k)} = \frac{1}{M^2}\|\mathbf{\Sigma}\|_F^2, \tag{7}$$

where $\mathbf{\Sigma}$ denotes the correlation matrix of $\mathbf{Q}$, and $\|\cdot\|_F$ represents the Frobenius norm.

Edge generation. Given the shared VNs, each client in FedVN is expected to learn a personalized graph augmentation strategy where the shared VNs connect their local graphs in a client-specific fashion so that the distribution shift can be eliminated. One straightforward approach is to construct edges based on the feature similarity between graph nodes and VNs. Although this feature similarity-based approach is widely adopted in graph structure learning (Chen, Wu, and Zaki 2020; Fatemi, El Asri, and Kazemi 2021), it is unsuitable for this scenario. In fact, edge patterns are significantly impacted by structure information as well. For instance, the house motif  in Figure 1 causally determines the label of *House*. Therefore, the circle nodes from the causal substructure and the non-causal part should have different patterns to connect VNs even though they have the same node type with identical node features.

To tackle the above issue, we propose to introduce an extra edge generator g with parameters ω in FedVN to construct edges between VNs and graph nodes. The edge generator is composed of a GNN encoder and an MLP projector. Given a graph $\mathcal{G}_i^{(k)}$ in client k, the output of the edge generator is a score matrix $\mathbf{S}_i^{(k)} \in \mathbb{R}^{|\mathcal{V}_i^{(k)}|\times M}$. Specifically, the GNN encoder first encodes each node v and its neighboring information to obtain its embedding $\mathbf{e}_v$ following Equation (1). The MLP projector then transforms $\mathbf{e}_v$ into an M-dimensional score vector $\mathbf{s}_v \in \mathbf{S}_i^{(k)}$ after a sigmoid function. Each score value $s_{v,m}$ in $\mathbf{s}_v$ is adopted as the edge weight between node v and VN m.

Despite the prowess of the edge generator in learning edge patterns regarding graph structures, the learned edge patterns cannot be guaranteed to follow a client-specific manner without explicit guidance. Instead, the edge generator may connect VNs and graph nodes within a client very diversely. Note that we assume graphs usually share similar non-causal substructures within a client but have diverse ones across clients (e.g., the function groups - the phenyl group in Institute A and the ester in Institute B - in our previous example). Therefore, the learned edge patterns on the graphs are expected to have similar distributions within a client and differ significantly across clients. Motivated by this, we propose a novel score-contrastive loss that encourages the total score vector of a graph to get close to the expected one of the graphs within the client and keep away from the expected one of the graphs from other clients. Let $\tilde{\mathbf{s}}_i^{(k)} = \sum_{v\in\mathcal{V}_i^{(k)}} \mathbf{s}_v$ denote the sum of score vectors in graph $\mathcal{G}_i^{(k)}$. $\mathbf{s}_{local} = \frac{1}{N^{(k)}}\sum_{i=1}^{N^{(k)}} \tilde{\mathbf{s}}_i^{(k)}$ is the average of $\tilde{\mathbf{s}}_i^{(k)}$ in client k, and $\mathbf{s}_{global}$ denotes the global expected sum of score vectors. Inspired by NT-Xent loss (Chen et al. 2020; Sohn 2016), we define $\tilde{\mathbf{s}}_i^{(k)}$ and $\mathbf{s}_{local}$ as a positive pair while $\tilde{\mathbf{s}}_i^{(k)}$ and $\mathbf{s}_{global}$ are defined as a negative pair. Therefore, our score-contrastive loss can be formulated as

$$\mathcal{L}_E^{(k)} = -\frac{1}{N^{(k)}}\sum_{i=1}^{N^{(k)}} \log \frac{e^{\mathrm{sim}(\tilde{\mathbf{s}}_i^{(k)},\mathbf{s}_{local})/\tau}}{e^{\mathrm{sim}(\tilde{\mathbf{s}}_i^{(k)},\mathbf{s}_{local})/\tau} + e^{\mathrm{sim}(\tilde{\mathbf{s}}_i^{(k)},\mathbf{s}_{global})/\tau}}, \tag{8}$$

where $\mathrm{sim}(\cdot,\cdot)$ represents the formula of cosine similarity, and τ is the temperature. Empirically, $\mathbf{s}_{global}$ could be simply substituted by a positive constant vector.

Finally, we obtain the local objective function of FedVN in client k formulated by

$$\min_{\theta,\mathbf{Q},\omega^{(k)}} \mathcal{L}_S^{(k)} + \lambda_1\mathcal{L}_V^{(k)} + \lambda_2\mathcal{L}_E^{(k)}, \tag{9}$$

where λ_1 and λ_2 are two hyperparameters to balance the impact of loss terms.

4.3 Algorithmic Design

Like other FGL frameworks, the proposed FedVN consists of two stages: local training and global update. Each client updates the GNN model, VNs, and the edge generator during local training. Then, the local GNN model and VNs are uploaded to the server for global update. The overall algorithm of FedVN can be found in our technical appendix.

Local training. During local training, we first optimize the local edge generator based on the global GNN model and VNs. We then optimize the GNN model and VNs with the updated edge generator.

► **Step 1: Fix θ and $\mathbf{Q}$, update $\omega^{(k)}$.** Train $\omega^{(k)}$ with the global θ and $\mathbf{Q}$ by

$$\omega^{(k)} \leftarrow \omega^{(k)} - \eta_\omega\nabla_\omega(\mathcal{L}_S^{(k)} + \lambda_2\mathcal{L}_E^{(k)}). \tag{10}$$

► **Step 2: Fix $\omega^{(k)}$, update $\theta^{(k)}$ and $\mathbf{Q}^{(k)}$.** After updating $\omega^{(k)}$, train $\theta^{(k)}$ and $\mathbf{Q}^{(k)}$ by

$$\theta^{(k)} \leftarrow \theta^{(k)} - \eta_\theta\nabla_\theta\mathcal{L}_S^{(k)}, \tag{11}$$

$$\mathbf{Q}^{(k)} \leftarrow \mathbf{Q}^{(k)} - \eta_Q\nabla_{\mathbf{Q}}(\mathcal{L}_S^{(k)} + \lambda_1\mathcal{L}_V^{(k)}). \tag{12}$$

η_ω, η_θ, and η_Q are their corresponding learning rates.

Global update. The server updates the GNN model and VNs following FedAvg (McMahan et al. 2017). Specifically, the global GNN model and VNs are updated as the weighted average of their local version by

$$\theta = \sum_{k=1}^{K}\frac{N^{(k)}}{N}\theta^{(k)}, \quad \mathbf{Q} = \sum_{k=1}^{K}\frac{N^{(k)}}{N}\mathbf{Q}^{(k)} \tag{13}$$

The complexity analysis of FedVN is provided in our technical appendix due to the page limit.

Dataset	Motif		CMNIST	ZINC	SST2
Metric	Accuracy ↑		Accuracy ↑	MAE ↓	Accuracy ↑
Partition setting	Basis	Size	Color	Scaffold	Length
Self-training	67.12±0.89	47.60±2.32	39.38±0.90	0.5442±0.0146	80.54±0.67
FedAvg	58.70±2.39	47.82±3.16	39.18±0.92	0.6235±0.0158	81.79±0.27
FedProx	57.90±1.36	47.88±4.08	39.78±0.68	0.6235±0.0165	81.74±0.33
FedBN	58.44±1.33	47.54±2.66	39.26±0.76	0.5129±0.0119	81.73±0.35
Ditto	63.38±0.89	47.48±3.20	39.00±0.94	0.5471±0.0146	81.69±0.67
FedRep	59.20±2.83	45.48±0.86	36.78±0.67	0.5220±0.0110	74.77±2.84
FedALA	59.92±1.14	48.52±3.34	39.22±1.12	0.5837±0.0159	81.77±0.61
GCFL+	57.36±2.00	49.34±2.70	38.82±1.11	0.6224±0.0147	81.39±0.45
FedStar	63.62±4.85	45.68±2.11	28.10±1.17	0.5963±0.0163	58.57±1.25
FedVN (Ours)	**75.72±1.85**	**50.41±1.17**	**43.67±1.25**	**0.4947±0.0174**	**83.13±0.79**

Table 1: Performance of FedVN and other baselines over four datasets under five settings.

5 Theoretical Analysis

In this section, we provide theoretical analyses for our proposed FedVN. As discussed above, FedVN enables each client to eliminate the distribution shift through a personalized edge generator determining how the VNs connect local graphs in a client-specific manner. It means that we can obtain identical representations of augmented graphs simply by desired score matrices. To simplify expression, we consider a part θ_e in f to generate graph embeddings (i.e., a GNN model except for its prediction head).

Theorem 1. *Consider a pair of graphs $\mathcal{G}$ and $\mathcal{G}'$. For an arbitrary feature matrix $\mathbf{Q} \in \mathbb{R}^{M \times d_x}$ with $M = d_x$ and any given GNN model f's component θ_e, when $\mathbf{Q}$ has full rank, there exists a pair of score matrices $\mathbf{S}$ and $\mathbf{S}'$ that satisfies:*

$$f(\tilde{\mathcal{G}};\theta_e) = f(\tilde{\mathcal{G}}';\theta_e),\quad \text{where } \tilde{\mathcal{G}} = \phi(\mathcal{G},\mathbf{S},\mathbf{Q}) \text{ and } \tilde{\mathcal{G}}' = \phi(\mathcal{G}',\mathbf{S}',\mathbf{Q}). \tag{14}$$

The complete proof of Theorem 1 is provided in our technical appendix. According to Theorem 1, we can conclude that FedVN can obtain identical graph embedding of $\mathcal{G}$ and $\mathcal{G}'$ given any $\mathbf{Q}$ and θ_e. Therefore, FedVN has the capability of eliminating the distribution shift issue in FGL.

Note that we assume $\mathbf{Q}$ has full rank in Theorem 1. This requires that multiple VNs cannot collapse to fewer nodes. To avoid collapsing VNs, we design a decoupling loss in FedVN. Formally, we can conclude the following theorem.

Theorem 2. *Minimizing $\mathcal{L}_V^{(k)}$ drives $\mathbf{Q}$ to have full rank.*

The complete proof of Theorem 2 is provided in our technical appendix. As a result, the learned VNs are distinct from each other, avoiding VN collapse.

6 Experiments

6.1 Experimental Setup

Datasets. We adopt graph datasets in (Gui et al. 2022) to simulate distributed graph data in multiple clients. Specifically, we use four datasets, including Motif, CMNIST, ZINC, and SST2. We split each graph dataset into multiple clients according to its environment settings so that every client has local graphs from one environment. More details about these datasets can be found in our technical appendix.

GNN backbones. We follow previous studies (Xie et al. 2021; Tan et al. 2023) to choose GNN backbones. Specifically, we adopt a three-layer GIN as the encoder and a two-layer MLP as the prediction head in the GNN model, with *mean* pooling as the READOUT operation. The edge predictor similarly uses a three-layer GIN as the encoder, followed by a two-layer MLP as the projector.

Baselines. We adopt nine baselines in our experiments. Among them, FedAvg (McMahan et al. 2017) and FedProx (Li et al. 2020) are two typical FL frameworks for learning a global model. As for personalized FL frameworks, we use FedBN (Li et al. 2021b), Ditto (Li et al. 2021a), FedRep (Collins et al. 2021), FedALA (Zhang et al. 2023). Furthermore, we also include two recent approaches for graph property prediction in FGL, i.e., GCFL+ (Xie et al. 2021) and FedStar (Tan et al. 2023). More details of the baselines are provided in our technical appendix.

Implementation details. We run our experiment for five random repetitions and report the average results in the experiments. Each repetition runs 100 epochs. The local epoch is set to 1, and the batch size is 32. The hidden size of the GNN model and the edge predictor is set to 100. We use SGD as the optimizer for local updates with a learning rate set to 0.01 for CMNIST and 0.001 for others. The temperature τ is set to 0.1. More information about hyperparameter settings can be found in our technical appendix.

6.2 Comparative Results

Effectiveness of FedVN. We first evaluate the overall performance of FedVN and other baselines. Table 1 shows the results of FedVN and baselines over the datasets. From Table 1, we can observe that our proposed FedVN can consistently outshine the baselines on the five tasks. Specifically, FedVN achieves about accuracy improvement of 8% on Motif/basis. Since Motif/basis is a synthetic dataset where graphs in a client all have the same label-irrelevant base

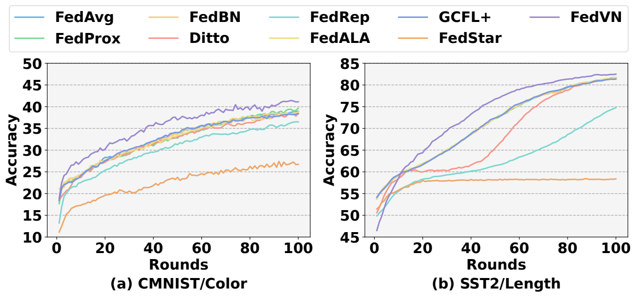

Figure 2: Convergence curves of FedVN and other baselines on CMNIST/Color and SST2/Length.

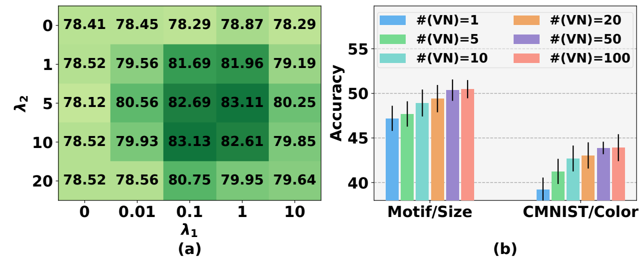

Figure 3: (a) Performance of FedVN on SST2/Length with different values of λ_1 and λ_2. (b) Performance of FedVN with different numbers of VNs.

subgraph, FedVN can eliminate such distribution shifts by adding VNs to local graphs. As for another semi-synthetic dataset, FedVN can perform well on CMNIST/Color. Considering the distribution shift in CMNIST/Color mainly on node features, the results indicate that FedVN can still overcome distribution shifts in terms of node features. Furthermore, FedVN can also outperform other baselines on the two real-world datasets ZINC/Scaffold and SST2/Length.

Convergence speed. Figure 2 illustrates the accuracy curves of FedVN and other baselines on CMNIST/Color and SST2/Length. We can observe that our method can outperform other baselines during the training process. Besides, we notice that FedStar suffers from significant performance degradation on CMNIST/Color and SST/Length. We argue that graphs in the two datasets are mainly labeled according to their node features. In this case, sharing structure encoders may not help enhance model utility.

6.3 Analysis of FedVN

Influence of λ_1 and λ_2. We conduct experiments to explore the sensitivities of two hyperparameters λ_1 and λ_2. Figure 3(a) presents the accuracy heatmap of FedVN on SST2/Length with different values of λ_1 and λ_2. We observe that FedVN achieves good performance when λ_1 is 0.1$\sim$1 and λ_2 is 1$\sim$10. The performance of FedVN will degrade apparently if either λ_1 or λ_2 is 0. The above observation demonstrates that the two loss terms $\mathcal{L}_V^{(k)}$ and $\mathcal{L}_E^{(k)}$ are necessary in FedVN.

Influence of VN numbers. As discussed in Section 4.2, too few VNs are usually insufficient to eliminate cross-client distribution shifts. In contrast, a large number of VNs can bring extra communication and computational costs. We investigate the performance of fedVN with different numbers of VNs and report the results on Motif/Size and CMNIST/Color in Figure 3(b). From the figure, we can observe that FedVN can achieve good performance consistently when equipped with enough VNs. When there are only a few VNs (e.g., 1 or 5), the performance of FedVN is limited. This observation validates our motivation for involving multiple VNs in FedVN. In addition, since too many VNs (e.g., when $M = d_x = 100$ in our experiments) will be very hard to train, we may consider using fewer VNs in practice.

Dataset	Motif/Basis ↑	CMNIST/Color ↑
FedVN	75.72±1.85	43.67±1.25
FedVN w/o g	60.20±1.75	40.15±1.27

Table 2: Performance of FedVN and its variant without edge generators on Motif/Basis and CMNIST/Color.

Effectiveness of edge generators. One insight in our study is personalized edge generators to learn client-specific graph augmentation strategies for eliminating distribution shifts of graph data. Without edge generators, multiple VNs will uniformly connect graph nodes in local graphs. In this case, these VNs can be merged into a single VN, which can be regarded as an equivalent scheme in previous studies (Hu et al. 2020, 2021). To validate the effectiveness of edge generators in FedVN, we compare the performance of FedVN and its degraded version without edge generators. We report the results on Motif/Basis and CMNIST/Color in Table 2. We observe that FedVN without edge generators encounters apparent performance degradation, which validates the necessity of edge generators in FedVN.

More experimental results. Due to the page limit, more experimental results are provided in our technical appendix, including visualizations of VN collapse, visualizations of distribution shifts in FedAvg and FedVN.

7 Conclusion

In this study, we take the first step towards investigating the distribution shift issue of graph data across clients for graph property prediction in FGL. We grapple with this issue by designing personalized graph augmentation to eliminate cross-client distribution shifts in FGL. We propose FedVN, a novel FGL framework that integrates multiple learnable VNs into local graphs in a client-specific fashion. To achieve this, each client is equipped with a personalized edge generator which determines how the VNs connect its local graphs. In this way, the clients can jointly train a powerful global GNN model over augmented graph data identical across clients. Furthermore, we provide theoretical analyses to validate that our design in FedVN has the capability of eliminating distribution shifts in FGL. Our extensive experiments show that FedVN outshines nine baselines on five tasks.

Acknowledgments

This work is supported in part by the National Science Foundation under grants IIS-2006844, IIS-2144209, IIS-2223769, IIS-2331315, CNS-2154962, BCS-2228534, and CMMI-2411248 and the Commonwealth Cyber Initiative Awards under grants VV-1Q24-011, VV-1Q25-004.

References

Cai, C.; Hy, T. S.; Yu, R.; and Wang, Y. 2023. On the connection between mpnn and graph transformer. In *International Conference on Machine Learning*.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*.

Chen, Y.; Wu, L.; and Zaki, M. 2020. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in neural information processing systems*.

Chen, Y.; Zhang, Y.; Bian, Y.; Yang, H.; Kaili, M.; Xie, B.; Liu, T.; Han, B.; and Cheng, J. 2022. Learning causally invariant representations for out-of-distribution generalization on graphs. *Advances in Neural Information Processing Systems*.

Collins, L.; Hassani, H.; Mokhtari, A.; and Shakkottai, S. 2021. Exploiting shared representations for personalized federated learning. In *International conference on machine learning*.

Cui, L.; Seo, H.; Tabar, M.; Ma, F.; Wang, S.; and Lee, D. 2020. Deterrent: Knowledge guided graph attention network for detecting healthcare misinformation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*.

Fatemi, B.; El Asri, L.; and Kazemi, S. M. 2021. Slaps: Self-supervision improves structure learning for graph neural networks. *Advances in Neural Information Processing Systems*.

Fu, X.; Chen, C.; Dong, Y.; Vullikanti, A.; Klein, E.; Madden, G.; and Li, J. 2023. Spatial-Temporal Networks for Antibiogram Pattern Prediction. In *2023 IEEE 11th International Conference on Healthcare Informatics (ICHI)*.

Fu, X.; Chen, Z.; Zhang, B.; Chen, C.; and Li, J. 2024a. Federated Graph Learning with Structure Proxy Alignment. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Fu, X.; Wang, S.; Dong, Y.; Zhang, B.; Chen, C.; and Li, J. 2024b. Federated Graph Learning with Graphless Clients. *Transactions on Machine Learning Research*.

Fu, X.; Zhang, B.; Dong, Y.; Chen, C.; and Li, J. 2022. Federated graph machine learning: A survey of concepts, techniques, and applications. *ACM SIGKDD Explorations Newsletter*.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*.

Gui, S.; Li, X.; Wang, L.; and Ji, S. 2022. Good: A graph out-of-distribution benchmark. *Advances in Neural Information Processing Systems*, 35: 2059–2073.

Gui, S.; Liu, M.; Li, X.; Luo, Y.; and Ji, S. 2023. Joint learning of label and environment causal independence for graph out-of-distribution generalization. *Advances in Neural Information Processing Systems*.

Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Advances in neural information processing systems*.

Hu, W.; Fey, M.; Ren, H.; Nakata, M.; Dong, Y.; and Leskovec, J. 2021. Ogb-lsc: A large-scale challenge for machine learning on graphs. *arXiv preprint arXiv:2103.09430*.

Hu, W.; Fey, M.; Zitnik, M.; Dong, Y.; Ren, H.; Liu, B.; Catasta, M.; and Leskovec, J. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*.

Huang, W.; Wan, G.; Ye, M.; and Du, B. 2023. Federated graph semantic and structural learning. In *Proc. Int. Joint Conf. Artif. Intell*.

Hwang, E.; Thost, V.; Dasgupta, S. S.; and Ma, T. 2022. An analysis of virtual nodes in graph neural networks for link prediction. In *The First Learning on Graphs Conference*.

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Li, S.; Zhou, J.; Xu, T.; Dou, D.; and Xiong, H. 2022. Geomgcl: Geometric graph contrastive learning for molecular property prediction. In *Proceedings of the AAAI conference on artificial intelligence*.

Li, T.; Hu, S.; Beirami, A.; and Smith, V. 2021a. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*.

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

Li, X.; JIANG, M.; Zhang, X.; Kamp, M.; and Dou, Q. 2021b. FedBN: Federated Learning on Non-IID Features via Local Batch Normalization. In *International Conference on Learning Representations*.

Liu, G.; Zhao, T.; Xu, J.; Luo, T.; and Jiang, M. 2022. Graph rationalization with environment-based augmentations. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Lucic, A.; Ter Hoeve, M. A.; Tolomei, G.; De Rijke, M.; and Silvestri, F. 2022. Cf-gnnexplainer: Counterfactual explanations for graph neural networks. In *International Conference on Artificial Intelligence and Statistics*.

Luo, D.; Cheng, W.; Xu, D.; Yu, W.; Zong, B.; Chen, H.; and Zhang, X. 2020. Parameterized explainer for graph neural network. *Advances in neural information processing systems*.

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

Motie, S.; and Raahemi, B. 2024. Financial fraud detection using graph neural networks: A systematic review. *Expert Systems with Applications*.

Peng, L.; Wang, N.; Dvornek, N.; Zhu, X.; and Li, X. 2022. Fedni: Federated graph learning with network inpainting for population-based disease prediction. *IEEE Transactions on Medical Imaging*.

Shen, X.; Sun, D.; Pan, S.; Zhou, X.; and Yang, L. T. 2023. Neighbor contrastive learning on learnable graph augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Shi, Y.; Liang, J.; Zhang, W.; Tan, V. Y.; and Bai, S. 2022. Towards understanding and mitigating dimensional collapse in heterogeneous federated learning. *arXiv preprint arXiv:2210.00226*.

Sohn, K. 2016. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*.

Sui, Y.; Wu, Q.; Wu, J.; Cui, Q.; Li, L.; Zhou, J.; Wang, X.; and He, X. 2023. Unleashing the power of graph data augmentation on covariate distribution shift. *Advances in Neural Information Processing Systems*.

Tan, Y.; Liu, Y.; Long, G.; Jiang, J.; Lu, Q.; and Zhang, C. 2023. Federated learning on non-iid graphs via structural knowledge sharing. In *Proceedings of the AAAI conference on artificial intelligence*.

Tan, Y.; Long, G.; Ma, J.; Liu, L.; Zhou, T.; and Jiang, J. 2022. Federated learning from pre-trained models: A contrastive learning approach. *Advances in neural information processing systems*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Voigt, P.; and Von dem Bussche, A. 2017. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*.

Wan, G.; Huang, W.; and Ye, M. 2024. Federated Graph Learning under Domain Shift with Generalizable Prototypes. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Wang, S.; Dong, Y.; Zhang, B.; Chen, Z.; Fu, X.; He, Y.; Shen, C.; Zhang, C.; Chawla, N. V.; and Li, J. 2024. Safety in Graph Machine Learning: Threats and Safeguards. *arXiv preprint arXiv:2405.11034*.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*.

Xie, H.; Ma, J.; Xiong, L.; and Yang, C. 2021. Federated graph classification over non-iid graphs. In *Advances in neural information processing systems*.

Xie, H.; Xiong, L.; and Yang, C. 2023. Federated node classification over graphs with latent link-type heterogeneity. In *Proceedings of the ACM Web Conference 2023*.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? In *International Conference on Learning Representations*.

Xu, Y.; Peng, Z.; Shi, B.; Hua, X.; and Dong, B. 2024. Learning dynamic graph representations through timespan view contrasts. *Neural Networks*.

Xu, Y.; Shi, B.; Ma, T.; Dong, B.; Zhou, H.; and Zheng, Q. 2023. CLDG: Contrastive learning on dynamic graphs. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*.

Yang, F.; Fan, K.; Song, D.; and Lin, H. 2020. Graph-based prediction of protein-protein interactions with attributed signed graph embedding. *BMC bioinformatics*.

Ying, Z.; You, J.; Morris, C.; Ren, X.; Hamilton, W.; and Leskovec, J. 2018. Hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31.

Yuan, H.; Yu, H.; Wang, J.; Li, K.; and Ji, S. 2021. On explainability of graph neural networks via subgraph explorations. In *International conference on machine learning*.

Yuan, Y.; and Bar-Joseph, Z. 2020. GCNG: graph convolutional networks for inferring gene interaction from spatial transcriptomics data. *Genome biology*.

Zbontar, J.; Jing, L.; Misra, I.; LeCun, Y.; and Deny, S. 2021. Barlow twins: Self-supervised learning via redundancy reduction. In *International conference on machine learning*.

Zhang, J.; Hua, Y.; Wang, H.; Song, T.; Xue, Z.; Ma, R.; and Guan, H. 2023. Fedala: Adaptive local aggregation for personalized federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Zhang, K.; Yang, C.; Li, X.; Sun, L.; and Yiu, S. M. 2021. Subgraph federated learning with missing neighbor generation. In *Advances in neural information processing systems*.

Zheng, Q.; Xu, Y.; Liu, H.; Shi, B.; Wang, J.; and Dong, B. 2024. A survey of tax risk detection using data mining techniques. *Engineering*.

Zhou, H.; Feng, Y.; Fang, M.; Wei, M.; Qin, J.; and Lu, T. 2021. Adaptive graph convolution for point cloud analysis. In *Proceedings of the IEEE/CVF international conference on computer vision*.

Zhuang, X.; Zhang, Q.; Ding, K.; Bian, Y.; Wang, X.; Lv, J.; Chen, H.; and Chen, H. 2023. Learning invariant molecular representation in latent discrete space. *Advances in Neural Information Processing Systems*.