Fine-tuned Personality Federated Learning for Graph Data

Meiting Xue *, Zian Zhou * , Pengfei Jiao, *Member, IEEE* and Huijun Tang

Abstract—Federated Learning (FL) empowers multiple clients to collaboratively learn a global generalization model without the need to share their local data, thus reducing privacy risks and expanding the scope of AI applications. However, current works focus less on data in a highly nonidentically distributed manner such as graph data which are common in reality, and ignore the problem of model personalization between clients for graph data training in federated learning. In this paper, we propose a novel personality graph federated learning framework based on variational graph autoencoders that incorporates model contrastive learning and local fine-tuning to achieve personalized federated training on graph data for each client, which is called FedVGAE. Then we introduce an encoder-sharing strategy to the proposed framework that shares the parameters of the encoder layer to further improve personality performance. The node classification and link prediction experiments demonstrate that our method achieves better performance than other federated learning methods on most graph datasets in the non-iid setting. Finally, we conduct ablation experiments, the result demonstrates the effectiveness of our proposed method.

Index Terms—Federated Learning, Graph Neural Networks, Contrastive Learning, Graph Data.

1 Introduction

Federated learning (FL), which is proposed by McMahan et al [1], is an emerging distributed machine learning paradigm, that allows clients to collaboratively train shared global or personalized models with no need to share raw training data of the client. During the FL process, each client only uploads its own model parameters or model gradient to the server in each round and updates the local model after the server averages and aggregates, during which the original data will not be exchanged. These advantages make FL applicable to numerous scenarios with scattered data or high privacy requirements, such as healthcare [2], public transportation [3], and natural language processing [4].

The types of data processed in federated learning can be divided into Euclidean data and non-Euclidean data [5]. Euclidean data is organized in a structured manner, facilitating easy calculation of the distance between samples. Conversely, non-Euclidean data lacks a well-defined arrangement, and the determination of sample neighbors is uncertain. At present, it has achieved good performance in the scenarios of Euclidean data [6]–[8] such as images and voice. However, as typical non-Euclidean data, graph data is very different from images and speech in the way of processing, and a new method called graph neural network (GNN) has also been proposed. The GNN is an effective neural architecture for mining graph-structured data as it can capture high-order content and topological information by aggregating neighbors on graphs. Several GNN models have been proposed to process graph data, including GCN [9], GAT [10], GraphSage [11], and VGAE [12]. Overall, GNN has achieved advanced performance in many graph-based tasks, such as link prediction, node classification, community detection, and graph classification. However, most of the GNN methods are trained in a central server, which becomes impractical due to privacy and massive data collection issues. Recent studies have utilized federated learning on graph-structured data to share latent knowledge and address the issue of knowledge scarcity, called graph federated learning(FGL), such as FederatedScope-GNN [13], FedSage [14], and FedGRL [15]. The common workflow is shown in Fig. 1, which describes the collaborative training of a global generalization model by multiple clients, each with a part of the subgraph. In FGL, there are some works arguments that each client owns a part of the large graph, and tries to achieve the accuracy of the large graph as much as possible through joint training. Our work primarily

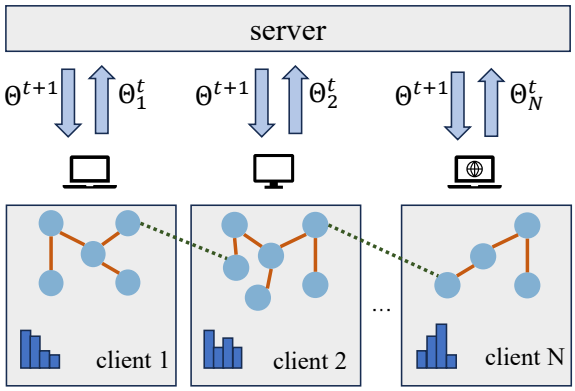


Fig. 1: In the workflow of graph federated learning, each client owns some subgraphs, and there are obvious structural and feature differences between the subgraphs. All clients jointly learn a global model.

*These authors contributed equally to this work.

- *M. Xue, Z. Zhou, P. Jiao and H. Tang are with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China. Email: {munuan, zza_1998, pjiao, tanghuijune}@hdu.edu.cn.*

(Corresponding author: Pengfei Jiao)

focuses on this.

However, there are still obstacles to the combined application of federated learning on graph data [16]. It is important to acknowledge that graphs inherently represent non-IID data [17]. The relationships between nodes within a graph are non-random and highly structured, adhering to specific real-world laws or influences. Nodes' attributes and their interconnections often offer abundant information about the nodes themselves. The intricate and diverse nature of this information renders graph data challenging to analyze, mine, and train in federated learning scenarios. For example, the noise information brought by the nodes of the sparse graph may not only not contribute to aggregation, but also degrade the performance of the global model.

Hence, to solve the aforementioned challenges, we should think about how to learn adaptive models for each client to fully cater to this non-iid feature under the federated learning scene and try to learn common knowledge. Personalized Federated Learning (PFL) is proposed to improve the performance of individual clients in the FL environment by learning knowledge suitable for its data distribution for each client from the global model or generating additional models suitable for itself. To overcome the non-IID challenges on Euclidean data, PFL has an excellent ability to correct global aggregation bias. However, previous PFL methods pay less attention to non-Euclidean data such as graphs that are common in reality. So we propose **F**ederated **V**ariational **G**raph **A**uto**E**ncoders (FedVGAE) framework to learn a better global model for graph data so that the client can better obtain useful information for itself. The contributions of this work are summarized as follows:

- We propose a novel personalized graph federated learning framework based on VGAE that incorporates contrastive learning and local fine-tuning to achieve fast convergence performance and personalized learning on graph data for each client in the FL environment.
- To further improve the performance in personalized learning, we introduce an encoder-sharing strategy that only shares the parameters of the encoder layer to retain the local decoder model of each client for personalized learning.
- We conduct extensive experiments on six datasets. Evaluation results demonstrate the proposed method can improve the average model accuracy by $2 \sim 8\%$ and can significantly reduce the performance degradation of clients caused by the global model.

The remainder of this paper is organized as follows. Related works are provided in Section 2. The system model and problem formulation are provided in Section 3. Extensive simulation experiments are conducted and discussed in Section 4. Section 5 concludes this paper's main work.

2 Related Works

In this section, we will provide a concise overview of the existing literature on federated learning, graph federated learning, and personalized federated learning.

2.1 Federated Learning

There has been growing attention toward privacy protection recently year, which has made federated learning increasingly popular. Federated learning is a paradigm where data is distributed across various remote terminals and is aggregated and trained cooperatively through a central server. The original FedAvg [1] proposed by A optimized the local SGD process and performed multiple iterations to reduce communication consumption. FedProx [18] adds an approximation term to the locally trained loss function to make the local model close to the global model. SCAFFOLD [19] introduces control variables to correct drift in local updates and speed up convergence. MOON [20] employs contrastive learning to enhance the model's expressive ability which primarily focuses on image data. GCFL [21] clusters different clients first and then aggregates them to reduce the problem of data heterogeneity. Other studies also aim to reduce the variance between clients. From the perspective of model aggregation, selecting clients that contribute more to the global model performance can speed up convergence and mitigate the impact of non-IID data.

2.2 Graph Federated Learning

Graph Neural Networks (GNNs) is a neural network model designed for processing graph-structured data, which has demonstrated remarkable performance on various graph tasks and therefore attracted significant research attention. Most GNN architecture algorithms are based on message passing, where the node representation is updated by aggregating information from neighboring nodes.

Graph federated learning combines graph machine learning and federated learning to address the limitations of centralized training for graph neural networks and the scattered distribution of real-world data while protecting users' privacy. In [22], graph federated learning is classified into two categories: structured data federation and structured federation. **Structured data federation** refers to client data as graph data, such as multiple graphs or subgraphs. FedGraph proposes a new sampling method and cross-client aggregation strategy to address the privacy issue in federated learning and the loss problem of graph training. FedSage [14] generates missing edges in the federated environment across clients using the generative model to improve the model convergence and performance capabilities of each client. VFGNN [23] considers the vertical federation scenario and uses differential privacy to protect the privacy of graph data for safe aggregation. ASFGNN [24] proposes a separate federated learning framework that can automatically learn hyperparameters and splits the process of training GNN into two steps: aggregation of neighbors and loss calculation. However, the previous graph federation method did not consider the heterogeneity between clients, so it is hard to learn a more generalized model under heterogeneous conditions. Our method falls into this category and can perform well. **Structured federation** means that each client constitutes a graph as a node. For example, SFL [25] constructs a graph of each traffic device to learn hidden relationships and then conducts GNN training. SpreadGNN [26] uses a decentralized approach for federated learning to solve the multi-task optimization problem of federated training.

2.3 Personalized Federated Learning

Due to the problems of model heterogeneity, data heterogeneity, and device heterogeneity in the current federated learning environment, personalized federated learning is proposed as an effective solution in this context. The purpose of the personalized federation is to allow every heterogeneous client to maintain good model performance while federating. Mansour et al [27] proposed a method based on user similarity clustering to improve the personalization of the model. Part of personalized federated learning is combined with transfer learning. Transfer learning [28] can efficiently transfer problem-solving knowledge to another problem. To alleviate the problem of large differences in data distribution, FedPer [29] adopts the concept of layer-level training, only the basic layer is shared based on FedAvg, and the top layers are used as personalized layers for local training. F. Hanz et. al [30] proposed to balance the local and global models by learning a mixture of the global and its local model. And based on a variant of the local gradient descent method called Loopless Local Gradient Descent (LLGD) to alleviate FedAvg's too-aggressive local aggregation. [31] proposed a method of knowledge distillation to effectively deal with the overfitting of small data sets in the process of personalized federated learning. However, previous works only considered constructing a global model, or only performing individualized federated learning for the client but lacked global consideration.

3 Proposed Methods

In this section, we propose a personality graph federated learning method, FedVGAE, to improve the personalization for clients and alleviate the problems caused by data heterogeneity. We first propose a problem formulation about graph federation, second, we design an architecture of FedVGAE and introduce its implementation details, and then we use a partial validation set to fine-tune client-side aggregation. All notations used in this paper are shown in Table 1.

3.1 Problem Formulation

GNN is divided into recurrent GNNs (RecGNNs), convolutional GNNs(ConvGNNs), graph autoencoders (GAEs), and spatial-temporal GNNs (STGNNs) [32]. Generally, a graph can be represented as G=(V, E, X), where V, E, and X denote the set of nodes, links, and node features respectively. $\mathcal{N}_v$ denotes a set of neighbors of $v \in V$, n = $|V|$ denotes the number of nodes. $A \in R^{n\times n}$is the adjacency matrix of G.

GNNs can make full use of the topology information on the graph to aggregate neighbors to generate node embedding. A typical GNN algorithm usually consists of a message-passing mechanism and neighbor aggregation. Typically, GCNs consist of multiple convolutional layers, each of layer used to aggregate the features of its neighboring nodes. Here the $H^{(l)}$ denotes the node embedding for each layer $l = 1, 2, ..., L$,and $H^{(0)}$=X. The propagation process of each layer of GCN can be described by Eq. 1:

$$H^{(l+1)} = \sigma(AH^{(l)}W^{(l)}) \quad \forall l \in L, \tag{1}$$

where $W^{(l)}$ are parameters for the l-th layer and need to learn, and $A \in R^{n\times n}$is the adjacency matrix of G. σ is an activation function and typically expressed as a softmax function in the last layer, while the previous layer is expressed as a relu function. In this work, we choose VGAE as our basic model.

TABLE 1: Notations and their definitions.

Notation	Representation
G	An undirected graph
$V; u, v$	The set of nodes; u and v are nodes
X	The feature matrix of G
A	The adjacency matrix
$\hat{A}$	The adjacency matrix after reconstruction
λ	The weight factor for contrastive loss
φ	The weight factor for model aggregation
μ	The mean vectors of nodes in VGAE
σ^2	The variance vectors of nodes in VGAE
Θ^t	Global model parameters during t-th round
Θ_i^t	Client i model parameters during t-th round
$D_i; \|D_i\|$	All data of client i; The number of nodes in client i
$KL(:)$	The Kullback-Leibler divergence operation
$sim(:)$	The cosine similarity of two vectors
L_o	The loss function of task loss
L_{con}	The loss function of model contract learning

In a standard federated learning setting. A client $i \in m$ holds D_i amount of private data. The server aggregates and updates each client model parameter and sends it back to prepare the next round until convergence. Specifically the aggregate as

$$\omega^{(t+1)} = \sum_{i=1}^{m} \frac{|D_i|}{|D|}\omega^{(t)}, \tag{2}$$

where $|D|$ is the total size of nodes over all clients, at t round, each client sends self parameters $\omega^{(t)}$ to the server and gets the new parameters $\omega^{(t+1)}$ for the next local train. However, for personalized learning, we keep the decoder as a personalization layer and only share the encoder part of the client model which is denoted by Θ.

3.2 System Architecture

Without loss of generality, our proposed architecture can be divided into a global stage and a local stage, where the global stage is similar to FedAvg, but the local stage is divided as follows parts. 1) The VGAE is used to generate node embeddings. 2) Model contract to restrict local model closer to global model and far away from the last local model. 3) Fine turn mechanism to adjust local aggregation. 4) Global aggregation and update.

3.2.1 Base Model

Real datasets often exhibit various characteristics, encompassing diverse types and fields, and contain large amounts of data. To address this challenge, VGAE [12] can reconstruct and generate new graphic data, achieving generalization and functioning across different fields. Furthermore, it utilizes a variational inference method that efficiently handles large graph datasets. Thus, we have chosen VGAE as the base model for this article. The encoder is constructed as

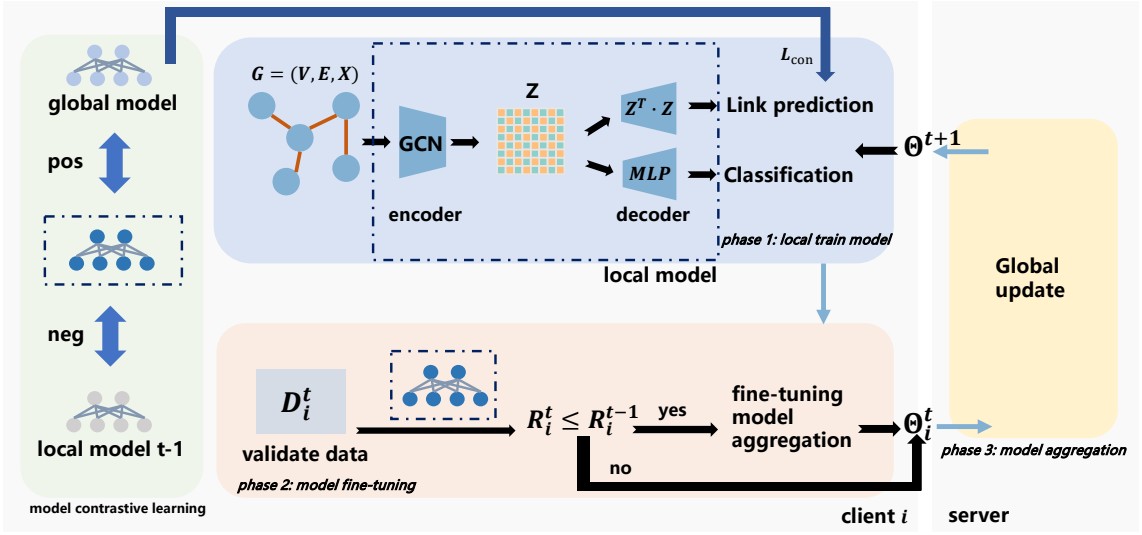


Fig. 2: The overall architecture of FedVGAE. The top part is basic train architecture. The left part is the contrast part to augment the model. The bottom part is a fine-tuning function. The right part is the central server.

a straightforward, three-layer graph convolutional network (GCN) to encode the topological structures A and node characteristics X. There, the GCN can be expressed as a spectral convolution function like Eq. 1.

Finding the probability distribution of a node's latent representation can often be challenging. Therefore, an approximation of the distribution is needed. Gaussian distribution, as the most common distribution, is often used to solve this problem [12].

$$\begin{aligned} q(\mathbf{Z}|X,A) &= \sum_{i=1}^{N} q_u(\mathbf{z}_u|X,A), \\ q_i(\mathbf{z}_u|X,A) &= \mathcal{N}(\mathbf{z}_u|\mu_u, diag(\sigma_u^2)), \end{aligned} \tag{3}$$

where μ_i and σ_i denote l-dimensional mean and standard deviation vectors corresponding to node u, respectively. The distribution parameters here can be learned from the two-layer GCN, as Eq. 4:

$$\begin{aligned} \mu &= GCN_\mu(GCN_1(A,X)),A), \\ \log_\sigma &= GCN_\sigma(GCN_1(A,X)),A), \\ Z^{n*d} &= reparametrize(\mu,\sigma), \end{aligned} \tag{4}$$

where GCN_μ and GCN_σ are different layer to generate matrices of μ_i and σ_i, respectively. GCN_1 is a common layer used to generate the lower embedding of nodes. Z is the node representation of the network sampling from the distribution and $reparametrize(\mu,\sigma) = \mu + \epsilon * \sigma, \epsilon \sim N(0,1)$. Given Z, the decoder dec(,) in VGAE is a simple inner product as:

$$\hat{A}_{uv} = dec(z_u, z_v) = \sigma(z_u^T z_v), \tag{5}$$

where $\hat{A}$ is the reconstructed adjacency matrix and z_v is embedding of node v. VGAE trains itself with optimizes the variational lower bound (ELBO):

$$L_{ELBO} = \mathbb{E}_{q(\mathbf{Z}|\gamma)}\left[p(\mathbf{A} \mid \mathbf{Z})\right] - \mathbf{KL}\left[q(\mathbf{Z} \mid X, A) \parallel (\mathbf{Z})\right], \tag{6}$$

The loss function of the link prediction task is L_{o1} as Eq.7.

$$\begin{aligned} L_{o1} = -\frac{1}{n}\sum_u\sum_v A_{uv} log(\hat{A}_{uv}) + (1-A_{uv}) log(1-\hat{A}_{uv}) \\ + \frac{1}{2}\sum_u (\mu_u^2 + \sigma_u^2 - log\sigma_u^2 - 1), \end{aligned} \tag{7}$$

where n denotes the number of nodes. We additionally use two layers of MLP for the readout function for node classification. The classification loss function denotes L_{o2} as Eq.8.

$$L_{o2} = \sum_k y_k log(\hat{y_k}) + \frac{1}{2}\sum_u (\mu_u^2 + \sigma_u^2 - log\sigma_u^2 - 1), \tag{8}$$

where y_k represents the probability that the node belongs to the k-th category and $\hat{y_k}$ is the predicted result. In our work, the loss function of the task is uniformly represented by L_o. Finally, the loss function is summarized as Eq. 9.

$$L_o = \begin{cases} L_{o1}, & \text{link prediction task,} \\ L_{o2}, & \text{classification task,} \end{cases} \tag{9}$$

3.2.2 Model Contrastive

We take advantage of the contrastive method in MOON [20] on the computer vision domain, which is based on the intuition that the global model can learn better than the local model. It approximates the local model and the global model, and rejects the history local model, to perform comparative learning. We use it for the GNNs model to further explore the feasibility of the comparative method on graph data. We use Θ_i to denote the local model parameters and Θ denotes the global model parameters. The nodes representing z, z_g, z_p correspond, respectively, to the current round of the global model, the current round of the local model, and the previous round of the local model that

passed through the encoder. So the model contrastive loss can be shown as follows:

$$L_{con} = -\log \frac{exp(sim(z, z_g)/\tau)}{exp(sim(z, z_g)/\tau) + exp(sim(z, z_p)/\tau)}, \tag{10}$$

where the sim(·) is a cosine similarity function, and τ is the temperature parameter. In general, the loss function of this paper is defined as follows.

$$L = L_o + \lambda L_{con}, \tag{11}$$

where λ is a hyperparameter that we set based on experience, and L_o is dependent on the task type.

In our approach, the model comparison loss is jointly optimized with the training loss. After the first round of training is completed, the model of this training has been retained. We store the model in a queue. Each round of training will add a new model. If the upper limit of the number of comparison models is reached, it will poll out the earliest entry model. In each round of training, the global model obtained from the server and the locally saved model in this round are compared with the currently trained model for learning.

Algorithm 1 FedVGAE: Federated Variational Graph Auto-Encoders

Input:Global model Θ, the sizes of local data $|n_i|$, the number of local updating steps E, learning rate η_t, the sample percents p, the number of clients N.

```
Initialize the global model parameters Θ^0
Send the global model parameters Θ^0 to all client
for communication round t = 0, 1, ..., T − 1 do
    The server sample the clients C_t = p * N for training
    The server broadcasts the model Θ^t to clients in C_t
    for each client i ∈ C_t parallel do
        set Θ^t_{i,0} = Θ^t
        sample data D^t_i from validate data
        validate the model and get result R^t_i on D^t_i
        if R^t_i ≤ R^{t−1}_i then
            fine-tuning the aggregation percents
        end if
        for each iteration e = 0, 1, ..., E − 1 do
            Θ^t_{i,e+1} ← Θ^t_{i,e} − η_t ∇F_i(Θ^t_{i,e})
        end for
        Client i send the model parameters Θ^{t+1}_i = Θ^t_{i,E−1}
        to the server.
    end for
    The server aggregates the received local model parameters Θ^{t+1} = Σ_{i∈C_t} (|n_i| / Σ_{i∈C_t} |n_i|) Θ^{t+1}_i
end for
```

3.2.3 Adjust Local Aggregation

In the previous federated learning process, all clients directly updated the local parameters after receiving the gradient and model parameters from the server. To some extent, instead of benefiting from the global model, the quality of the local model drops a bit. To deal with this situation, we will make fine-tune during the aggregation process to increase the income and reduce the loss, so that each client model can achieve better performance. Formally,

$$\tilde{\Theta}_i^{t+1} = \Theta_i^t \cdot \varphi + \Theta^t \cdot (1 - \varphi). \tag{12}$$

where Θ_i^t is t rounds model parameters at i-th client, and $\tilde{\Theta}_i^{t+1}$ is new local parameters after local aggregation. φ is a hyperparameter for learning.

For client i, we sample a batch of data D_i at the beginning for validation, these data did not appear in the training set. During the model training, we first update the local model with the parameters obtained from the server, then the model would be verified by the sampled data D_i. Suppose the accuracy of the model drops to a certain extent, such as 1 percentage point. In that case, the adjustment condition is triggered, and we will re-aggregate the local model and adjust the proportion of the global model. In multiple aggregation adjustments, we can obtain global knowledge biased towards a specific client from the global model to better perform personalized learning.

3.2.4 Global Aggregation

In our model, we assume that the central server can be trusted. The global model will deliver the initially generated model to each client at the beginning of training. After the client training is completed, the global model will aggregate all parameters and re-deliver to continue the next round of training until the model converges. The detail is shown in Algorithm 1.

4 Experiments

TABLE 2: The common dataset used.

Dataset	$\|V\|$	$\|E\|$	Feature	Class
Cora [33]	2,708	10,566	1,433	7
CiteSeer [34]	3,327	9,104	3,704	6
PubMed [35]	19,717	88,648	500	3
CoraFull [36]	19,793	126,842	8,710	70
Ploblogs [37]	1,490	19,025	0	2
Flicker [38]	89,250	899,756	500	7

$|V|$ denotes the number of nodes. $|E|$ denotes the number of edges

4.1 Experiment Settings

Initially, the server distributes model parameters to each client. The client evaluates the performance of both the local and server models and adjusts the aggregation weight accordingly. Subsequently, the client trains the model, considering the model contrastive loss locally. After the training rounds, the client sends the updated model parameters to the server for aggregation. TABLE 2 shows the details of the datasets. Further baseline details and statistics of the datasets are summarized in the Appendix.

4.2 Overall Perforamance

4.2.1 Link prediction

The link prediction task involves predicting the likelihood of an edge forming between two points. We utilized VGAE to reconstruct the adjacency matrix of the graph for this

TABLE 3: Link Prediction Result On Real Dataset

Dataset	CoreFull		Cora		Pubmed		Citeseer		Polblogs	
	auc	ap	auc	ap	auc	ap	auc	ap	auc	ap
Local Only	0.8949	0.8887	0.7479	0.7306	0.8583	0.8624	0.7546	0.7801	0.7092	0.7577
FedProx	0.8277	0.8303	0.7382	0.7178	0.8535	0.8508	0.8086	0.8273	0.7039	0.7503
FedAvg	0.8529	0.8581	0.7684	0.7585	0.8487	0.8661	0.7927	0.8121	0.7085	0.7526
GCFL	0.8480	0.8507	0.7406	0.7626	0.8525	0.8634	0.8438	**0.8741**	0.6813	0.7554
MOON	0.8443	0.8472	0.7598	0.7748	0.8629	0.8684	0.8194	0.8385	0.7208	0.7654
FedSage	0.8555	0.8591	0.7468	0.7640	0.7642	0.8044	0.8305	0.8546	0.7229	0.7603
FedPUB	0.8495	0.8552	0.7402	0.7487	0.8294	0.8394	0.8291	0.8444	0.7165	**0.7960**
FedVGAE(ours)	**0.8689**	**0.8704**	**0.7881**	**0.8043**	**0.8635**	**0.8815**	**0.8460**	0.8659	**0.7235**	0.7675

purpose. In our experiment, we divided the large graph dataset into 10 subgraphs and trained them for 100 rounds. Each round of local iterative training was repeated 5 times. Furthermore, we implemented early termination and evaluated the model performance of each client after training completion, calculating the average. The final results are presented in Table 3, We have bolded and underlined the optimal and suboptimal federated learning results respectively.

It can be seen from the experimental results that FedVGAE dominates the other methods on average test accuracy, which shows the effectiveness of the method proposed. In the Cora, Pubmed, and Ploblogs datasets, many federated learning methods have shown a tendency to deteriorate the aggregation of client-side models,but our method can effectively prevent such results from happening. We believe that this is due to the contrastive learning and Model-local fine-tuning making FedVGAE gain more beneficial knowledge from the global model. Although CoraFull is highly heterogeneous, FedVGAE also has competitive performance on the CoraFull dataset with the local-only train, outperforming other federated algorithms or graph federated algorithms.

4.2.2 Node Classification

We conduct experiments on node classification under the above settings. Each data is the average of five independent experiments, and the result is shown in Table 4. We indicate the optimal and suboptimal results in bold and underlined respectively. It can be seen intuitively that our proposed FedVGAE achieves the best performance on most of the datasets by a noticeable margin. The poor performance of the flicker is caused by huge non-iid data. Even so, our method outperforms other federated learning baselines. FedVGAE greatly improves the amount of information contained in the node embedding, alleviating the performance degradation caused by heterogeneity, thus improving the final performance.

4.3 Ablation Study

To further verify the effectiveness of the encoder-sharing-only strategy, we conducted an ablation study by sharing all model layers during the training. In this experiment. We use the encoder to indicate that the model only shares his encoder layer. The result in Table 5 shows that keeping an appropriate amount of layers locally enhances the localization of the local models, thereby improving the expressiveness of the individual local models overall.

TABLE 4: Node Classification Result On Real Dataset

DataSet	Cora	Pubmed	Citeseer	Flicker
Local Only	0.8149	0.8093	0.6268	0.4920
FedProx	0.7249	0.8701	0.6903	0.3753
FedAvg	0.7986	0.8721	0.6346	0.3350
GCFL	0.8177	0.8593	0.6893	0.4545
MOON	0.7949	0.8728	0.7169	0.4267
FedSage	0.8318	0.8705	0.7060	0.4802
FedPUB	0.8023	0.8601	0.7032	0.4192
FedVGAE(ours)	**0.8428**	**0.8812**	**0.7193**	**0.4810**

TABLE 5: Ablation Study on Share Strategy

Dataset	CiteSeer		Cora		PubMed	
Shared part	encoder	all	encoder	all	encoder	all
FedAvg	**0.7747**	0.7060	**0.7635**	0.7187	**0.8713**	0.8606
FedProx	**0.7716**	0.7059	**0.7086**	0.6631	0.8601	**0.8606**
FedVGAE	**0.7927**	0.7149	**0.8306**	0.7112	**0.8682**	0.8612

An additional ablation study focuses on the fine-tuning and contraction phases of the model. We fix all other parameters and evaluate the impact of sharing the whole model on the experimental results. Table 6 displays the results, where FedVGAE represents the method proposed in this paper, "con" represents the model comparison phase, and "ft" represents the fine-tuning phase of the model.

As the result in Table 6 indicates, The efficacy of model comparison and fine-tuning varies depending on the dataset. Specifically, for the pubmed and citeseer datasets, the model comparison is more effective than fine-tuning, as it can reduce the discrepancy between the local and global models and leverage the implicit knowledge of the global model to improve performance. In contrast, fine-tuning is preferable for the cora dataset, as it maintains the original generalization capacity of the local model while gradually enhancing its performance without compromising its own effectiveness.

TABLE 6: Ablation Study on Model Component

		FedVGAE	w/o con	w/o ft	w/o con + ft
Cora	auc	**0.7831**	0.7793	0.7704	0.7684
	Δ auc	-	-0.0038	-0.0127	-0.0147
PubMed	auc	**0.8635**	0.85	0.8607	0.8487
	Δ auc	-	-0.0135	-0.0028	-0.0148
CiteSeer	auc	**0.8103**	0.7775	0.8038	0.7726
	Δ auc	-	-0.0328	-0.0065	-0.0377

4.4 Parameter Study

To have a better understanding of the hyperparameter λ and φ, we estimate the weight divergence of each client with the varying hyperparameter values.

TABLE 7: AUC values on Pubmed with different λ

λ	0.1	0.5	0.8	1	2	5	10
Cora	0.761	0.780	**0.787**	0.764	0.779	0.772	0.779
PubMed	0.858	0.857	0.870	**0.871**	0.864	0.863	0.815
CiteSeer	0.783	0.790	**0.808**	0.801	0.778	0.804	0.798

In Table 7, we present AUC scores obtained with different λ values. It's important to note that, due to the trade-off between statistical heterogeneity, generalization, and personalization, the optimal value of λ may vary. As we can observe, the performance of the final link forecast first increased and then decreased. By fine-tuning λ, adaptive strategies can be used to identify the best λ value for each customer. Ultimately, we can improve the model's effectiveness by setting λ around 0.8. Therefore, in this article, we have set the parameter to the default value in our experiments.

TABLE 8: AUC values on Pubmed with different φ

φ	0.1	0.2	0.3	0.5	0.8	0.9
Cora	0.7743	0.7574	0.7840	**0.8097**	0.7736	0.7895
PubMed	0.8563	0.8713	0.8662	**0.8726**	0.8585	0.8500
CiteSeer	0.7902	0.7352	0.7955	**0.8073**	0.7778	0.7966

Additionally, we conduct the link prediction task by adjusting φ, where 0 indicates full usage of the global model, and 1 indicates full usage of the local model. While φ does have a certain impact on the model's performance, the results in Table 8 show that it is not overly significant. The model achieves better performance when φ is around 0.5, but this value may vary due to the heterogeneity of data and its distribution.

4.5 Efective Study

We perform an effective experiment and got results as Fig. 3. After the experiment, we can observe that our method has excellent convergence ability. Compared with FedAvg and FedProx, in the CoraFull data set, FedVGAE has converged in about 60 rounds, and the final accuracy is also higher than other methods.

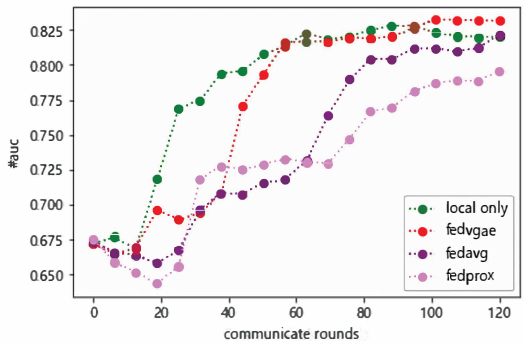


Fig. 3: Test result of the link task on CoraFull

5 Conclution

In this paper, to address the challenge of model deterioration in non-IID environments, we propose FedVGAE, which combines comparative learning and personalized federated learning methods to simultaneously train global and local personalized models, and ultimately improve the performance of the GNN model on the client. We conduct sufficient experiments, and the experimental results show that our method has a certain improvement effect on this scenario, and surpasses other federated learning methods.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62372146, in part by the Zhejiang Laboratory Open Research Project under Grant K2022QA0AB01, in part by the Fundamental Research Funds for the Provincial Universities of Zhejiang Grant GK229909299001-008.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[2] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein *et al.*, "The future of digital health with federated learning," *NPJ digital medicine*, vol. 3, no. 1, p. 119, 2020.

[3] Y. Liu, J. James, J. Kang, D. Niyato, and S. Zhang, "Privacy-preserving traffic flow prediction: A federated learning approach," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7751–7763, 2020.

[4] M. Liu, S. Ho, M. Wang, L. Gao, Y. Jin, and H. Zhang, "Federated learning meets natural language processing: a survey," *arXiv preprint arXiv:2107.12603*, 2021.

[5] J. Laub, V. Roth, J. M. Buhmann, and K.-R. Müller, "On the information and representation of non-euclidean pairwise data," *Pattern Recognition*, vol. 39, no. 10, pp. 1815–1826, 2006.

[6] W. Huang, M. Ye, and B. Du, "Learn from others and be yourself in heterogeneous federated learning," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, Jun 2022, p. 10133–10143.

[7] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "Fedbn: Federated learning on non-iid features via local batch normalization," no. arXiv:2102.07623, May 2021, arXiv:2102.07623 [cs].

[8] T. Zhou and E. Konukoglu, "Fedfa: Federated feature augmentation," no. arXiv:2301.12995, Jan 2023, arXiv:2301.12995 [cs].

[9] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

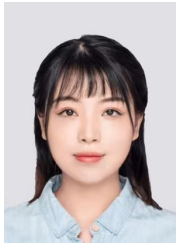
[10] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio *et al.*, "Graph attention networks," *stat*, vol. 1050, no. 20, pp. 10–48 550, 2017.

[11] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.
[12] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.
[13] Z. Wang, W. Kuang, Y. Xie, L. Yao, Y. Li, B. Ding, and J. Zhou, "Federatedscope-gnn: Towards a unified, comprehensive and efficient package for federated graph learning," no. arXiv:2204.05562, Aug 2022, arXiv:2204.05562 [cs].
[14] K. Zhang, C. Yang, X. Li, L. Sun, and S. M. Yiu, "Subgraph federated learning with missing neighbor generation," *Advances in Neural Information Processing Systems*, vol. 34, pp. 6671–6682, 2021.
[15] S. Suresh, D. Godbout, A. Mukherjee, M. Shrivastava, J. Neville, and P. Li, "Federated graph representation learning using self-supervision," *arXiv preprint arXiv:2210.15120*, 2022.
[16] W. Zhang, J. C. Weiss, S. Zhou, and T. Walsh, "Fairness amidst non-iid graph data: A literature review," *arXiv preprint arXiv:2202.07170*, 2022.
[17] K. Zhang, Z. Cai, D. Seo *et al.*, "Privacy-preserving federated graph neural network learning on non-iid graph data," *Wireless Communications and Mobile Computing*, vol. 2023, 2023.
[18] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.
[19] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5132–5143.
[20] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 713–10 722.
[21] H. Xie, J. Ma, L. Xiong, and C. Yang, "Federated graph classification over non-iid graphs," *Advances in neural information processing systems*, vol. 34, pp. 18 839–18 852, 2021.
[22] X. Fu, B. Zhang, Y. Dong, C. Chen, and J. Li, "Federated graph machine learning: A survey of concepts, techniques, and applications," *ACM SIGKDD Explorations Newsletter*, vol. 24, no. 2, pp. 32–47, 2022.
[23] C. Chen, J. Zhou, L. Zheng, H. Wu, L. Lyu, J. Wu, B. Wu, Z. Liu, L. Wang, and X. Zheng, "Vertically federated graph neural network for privacy-preserving node classification," *arXiv preprint arXiv:2005.11903*, 2020.
[24] L. Zheng, J. Zhou, C. Chen, B. Wu, L. Wang, and B. Zhang, "Asfgnn: Automated separated-federated graph neural network," *Peer-to-Peer Networking and Applications*, vol. 14, no. 3, pp. 1692–1704, 2021.
[25] F. Chen, G. Long, Z. Wu, T. Zhou, and J. Jiang, "Personalized federated learning with graph," *arXiv preprint arXiv:2203.00829*, 2022.
[26] C. He, E. Ceyani, K. Balasubramanian, M. Annavaram, and S. Avestimehr, "Spreadgnn: Serverless multi-task federated learning for graph neural networks," *arXiv preprint arXiv:2106.02743*, 2021.
[27] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," *arXiv preprint arXiv:2002.10619*, 2020.
[28] L. Y. Pratt, "Discriminability-based transfer between neural networks," *Advances in neural information processing systems*, vol. 5, 1992.
[29] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, "Federated learning with personalization layers," *arXiv preprint arXiv:1912.00818*, 2019.
[30] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," *arXiv preprint arXiv:2002.05516*, 2020.
[31] D. Li and J. Wang, "Fedmd: Heterogenous federated learning via model distillation," *arXiv preprint arXiv:1910.03581*, 2019.
[32] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.
[33] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the construction of internet portals with machine learning," *Information Retrieval*, vol. 3, pp. 127–163, 2000.
[34] C. L. Giles, K. D. Bollacker, and S. Lawrence, "Citeseer: An automatic citation indexing system," in *Proceedings of the third ACM conference on Digital libraries*, 1998, pp. 89–98.
[35] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, pp. 93–93, 2008.
[36] A. Bojchevski and S. Günnemann, "Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking," 2017.
[37] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 u.s. election: Divided they blog," in *Proceedings of the 3rd International Workshop on Link Discovery*, ser. LinkKDD '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 36–43.
[38] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. Prasanna, "Graphsaint: Graph sampling based inductive learning method," 2019.

Meiting Xue received the PhD degree in instrument science and technology from Zhejiang University, China, in 2020. She is an instructor with the Department of Cyberspace, Hangzhou Dianzi University. Her research interests include database acceleration, embedded computer systems and federated learning.

Zian Zhou received the Bachelor degree from Wenzhou Medicine University in 2020. He is currently pursuing a master's degree at the School of Cyberspace, Hangzhou Dianzi University. His current research interests include federated learning and complex network analysis.

Pengfei Jiao received the Ph.D. degree in computer science from Tianjin University, Tianjin, China, in 2018. From 2018 to 2021, he was a lecturer with the Center of Biosafety Research and Strategy of Tianjin University. He is currently a Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His current research interests include complex network analysis and its applications.



Huijun Tang received the BSc degree from Jinan University, China in 2016 and the M.S. and Ph.D. degree from Tianjin University, China in 2018 and 2022, respectively. She is currently a Lecturer with the School of Cyberspace, Hangzhou Dianzi University. Her research interests include internet of things, mobile edge computing and deep learning.