Graph Federated Learning with Center Moment Constraints for Node Classification

Bisheng Tang
School of Cyber Security, University of Chinese Academy of Sciences; Institute of Information Engineering, Chinese Academy of Sciences; Key Laboratory of Cyberspace Security Defense
Beijing, China
tangbisheng@iie.ac.cn

Xiaojun Chen*
School of Cyber Security, University of Chinese Academy of Sciences; Institute of Information Engineering, Chinese Academy of Sciences; Key Laboratory of Cyberspace Security Defense
Beijing, China
chenxiaojun@iie.ac.cn

Shaopu Wang
School of Cyber Security, University of Chinese Academy of Sciences; Institute of Information Engineering, Chinese Academy of Sciences; Key Laboratory of Cyberspace Security Defense
Beijing, China
wangshaopu@iie.ac.cn

Yuexin Xuan
School of Cyber Security, University of Chinese Academy of Sciences; Institute of Information Engineering, Chinese Academy of Sciences; Key Laboratory of Cyberspace Security Defense
Beijing, China
xuanyuexin@iie.ac.cn

Zhendong Zhao
School of Cyber Security, University of Chinese Academy of Sciences; Institute of Information Engineering, Chinese Academy of Sciences; Key Laboratory of Cyberspace Security Defense
Beijing, China
zhaozhendong@iie.ac.cn

ABSTRACT

Centralized learning graph-structured data representation commonly exists in various institutions, while it is challenging to learn other institutions' graph-structured data representations without data leakage. Federated learning (FL) is proposed to federally learn a global model while keeping data privacy and security. However, FL is notorious for the non-i.i.d data. The existing works aim to form an undivided graph by linking all party subgraphs while neglecting the local non-i.i.d feature in the training phase. To this end, we propose a novel graph FL framework called FedOMD to leverage global independent and identically distributed (i.i.d) hidden feature representation to guide the local graph model training. Specifically, We first model each local feature as a Gaussian distribution to decrease the representation discrepancy in different parties and then calculate a global Gaussian distribution in the server. Finally, we use central moment discrepancy to minimize the distance between the party local and the server global distribution. With such distribution constraints, all parties can train the graph model in a unified feature space. Our extensive experiments on five datasets have manifested the competitive effectiveness of FedOMD over the seven mentioned FL models. The relevant ablation and sensitivity analysis also verify the effectiveness of FedOMD.

*Corresponding author

ICPP Workshops '24, August 12–15, 2024, Gotland, Sweden

ACM ISBN 979-8-4007-1802-1/24/08
https://doi.org/10.1145/3677333.3678159

KEYWORDS

Graph, Federated Learning, Node, Non-i.i.d, Orthogonality, Center Moment

ACM Reference Format:
Bisheng Tang, Xiaojun Chen, Shaopu Wang, Yuexin Xuan, and Zhendong Zhao. 2024. Graph Federated Learning with Center Moment Constraints for Node Classification. In *The 53rd International Conference on Parallel Processing Workshops (ICPP Workshops '24), August 12–15, 2024, Gotland, Sweden.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3677333.3678159

1 INTRODUCTION

Graph-structured data is widely present in various domains such as medical institutions, banking, finance, and social networks and possesses substantial commercial value. Graph neural networks (GNNs) effectively deal with these graph-structured data relying on spectral graph theory and have demonstrated impressive success in recent years. However, several practical challenges hinder GNNs from learning a general and robust embedding representation. For instance, individuals and businesses are unlikely to provide their data to others due to security and privacy concerns. These will lead to each GNN learning a distinct embedding representation in different parties, thereby hindering effective classification. Instead, suppose the parties provide their private data to collaboratively learn a global model, a wide range of beneficial applications, such as epidemic disease prediction and bank money laundering detection, can be explored.

To mitigate the challenges posed by data security and privacy concerns, FL [20] has been proposed as a viable solution to train graph models without exposing private data. In this approach, parties upload their model parameters with encryption, and the

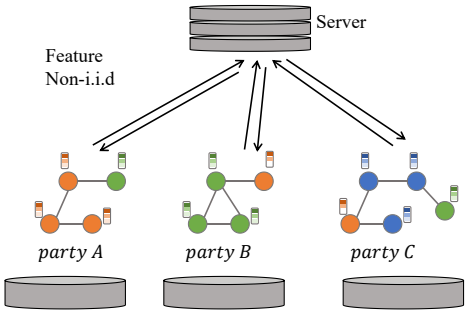


Figure 1: The feature space in each participant is not identically distributed.

server then aggregates the model parameters, further distributing them to participants. Such a widely applied algorithm in this paradigm is FedAVG [24]. Despite its great success, it is insufficient to aggregate only the model parameters when utilizing FL to learn graph-structured data. As figure 1 manifested, different local architectures with non-i.i.d data (i.e., heterogeneous data) [33] affect learning well-generalized global or personal models to varying degrees. Recently, the existing work [38] proposes the generation of cross-subgraph edges to build links between different parties, thereby enabling the federation of all subgraphs to learn a global i.i.d representation. [26] work extends this approach by incorporating SN-GAN to learn cross-subgraph edges, constraining the feature generation of various parties with the same distribution.

Despite their effectiveness in graph federated learning (GFL), the existing GFL suffers from, at least partially, the following problems. Firstly, the nodes and edges generated by the GFL approach may be imprecise or mismatched, as they may not exist in real-world scenarios or, if they do, may exhibit significant discrepancies. As a result, generating new nodes and edges is not always an effective way to build connections between parties. Secondly, another issue with GFL is that it does not consider the non-i.i.d feature distribution across various parties, which accompanies not effectively connecting each party. Building virtual links does not necessarily result in all features across parties becoming i.i.d data. Moreover, generated nodes' features derived from different distributions may cause a higher feature non-i.i.d problem, resulting in unstable performance. The non-i.i.d phenomenon widely exists. For instance, in the background of epidemic prediction, coronavirus presents discrepancies in symptoms due to various variance branches and kinds of environment. In some areas, the symptoms of coronavirus are similar to influenza virus. While in other areas, there may exist highly pathogenic and different severe symptoms. In other words, the features of coronavirus appear the non-i.i.d phenomenon in different regions, which is not beneficial to detect the coronavirus.

To mitigate the impact of the non-i.i.d feature generated by GFL, we propose a novel GFL model called FedOMD. FedOMD leverages global i.i.d hidden feature representation to guide local GNN training. The model consists of four main steps. First, FedOMD builds Gaussian distributions for all the parties' hidden features to reduce the distribution discrepancy of feature representation. Second, it forms a global feature distribution through the information exchange by the FL paradigm. Third, FedOMD collaboratively calculates the distance between the local non-i.i.d hidden feature and server i.i.d. hidden feature using the distribution metrics, central moment discrepancy (CMD) [36]. Finally, FedOMD simultaneously optimizes local model loss and CMD loss. By incorporating these steps, FedOMD addresses the issues mentioned earlier and provides a practical approach for learning graph-structured data in a federated setting. Our FedOMD model is conducive to learning all the non-i.i.d features into i.i.d representation, which promotes the detection of coronavirus in all regions.

Our contributions can be summarized in three main parts:

(i) We deeply analyze the deficiencies of generating the cross-subgraph edges aiming at forming global i.i.d feature distribution when confronted with local non-i.i.d feature distribution. Furthermore, we directly form the i.i.d hidden feature, thus having achieved better results.

(ii) we propose a new GFL model, FedOMD, to address the non-i.i.d feature training problem, a significant issue in GFL. Besides, our model can implicitly calculate the IID distribution by only 2-round interaction, which avoids expensive communication costs.

(iii) we conduct extensive experiments on five graph datasets, demonstrating the competitive advantages of FedOMD over commonly used GFL models. Furthermore, the relevant ablation analyses also comprehensively verify the effectiveness of FedOMD.

2 RELATED WORKS

Graph Neural Networks. GNNs have demonstrated impressive classification capabilities in downstream tasks, e.g., node classification, edge prediction, and graph classification. The messaging passing mechanism has dominated GNN designations, including GCN [17], GAT [28], GraphSage [12], APPNP [18], GCNII [7], and Graphormer [37] in recent years. Meanwhile, to adapt various graph types, such as heterophily graphs, models like H2GCN [40] and BM-GCN [14] have been proposed. Other models like HAN [30] and HGNN [8] aim to handle heterogeneous graphs and hypergraphs, respectively. GNNs have also demonstrated remarkable performance in self-supervised tasks, such as NASA [3]. Recently, graph contrastive learning (GCL), as an augmentation technique in graph learning, has promoted the model performance, such as GraphCL [35]. Graph federated learning generally considers the low complexity GNN as the local basic model, such as GCN [17] and GraphSage [12]. Our work is most relevant to orthogonal GCN [11]. We leverage its orthogonalization to calculate Gaussian distribution in our proposed method.

Federated Learning. FL has addressed the problem of data silos, thus benefiting commercial applications, such as healthcare, edge computing, and recommendation systems. The most famous FL model is FedAVG [24], which averagely aggregates the isolated local model. To further explore the data heterogeneity, several works have been proposed, e.g., FedProx [21], SCAFFLOD [16], and FedBaBU [25]. The FL can be divided into Horizontal and Vertical FL according to the feature splitting. Our work is relevant to Horizontal FL, whose datasets differ in samples yet share the same feature space. The Vertical FL, where two data sets differ in feature space but have a common sample space, is very suited to cross-domain product recommendations.

Graph Federated Learning. GFL [9, 22], federally coping with graph-structured data, has gradually attracted researchers' attention. GFL can be classified into two categories. One is the horizontal graph FL, which focuses on graph level FL (e.g., FedGraphNN [13]), subgraph level FL (e.g., FedGNN [31], GraphFL [29], ASFGNN [39], FedSage [38], FedNI [26], FedCog [19]) and node level FL (e.g., [6]). Another is vertical graph FL, which researches the node feature vertically being separated to various parties, such as VFGNN [5]. Apart from the model discrepancy, there are also some applications, e.g., time series prediction [6] and health record representation [23]. GFL utilizes the graph data in different parties to learn a global graph representation or personalize their local task with the data augmentation views. Moreover, various parties can build connections with each other under the FL process, which means every party can be regarded as a node to participate in the message passing. Such learning methods generally can be chosen as auxiliary enhancement tasks, such as work [6]. The existing works [27, 33] mainly address the non-i.i.d problem in graph classification, and our work focuses on subgraph level FL, especially in node feature non-i.i.d scenarios.

3 PRELIMINARIES

The Training Process of Federated Learning. The training process of FL can be concluded with the following three phases.

Phase 1 (Hyper-parameter Initialization): The server initials its hyper-parameters according to the relevant data and task requirements and subsequently distributes global model W_0 to each participant.

Phase 2 (Local Model Training): Each participant receives the global model and updates its local model. Participants then continue to train the updated model with its local data $\mathbb{D}_i$ and negotiated epoch. After finishing the local training (minimizing loss $\mathcal{L}(\mathbb{D}_i, W_i)$), participant upload the trained model W_i^* to server.

$$W_i^* = \arg\min \mathcal{L}(\mathbb{D}_i, W_i). \tag{1}$$

Phase 3 (Global Model Update): After the server has received all uploaded models, it begins to aggregate the uploaded models. The global model loss function is minimized as follows:

$$\mathcal{L}(W_0) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(\mathbb{D}_i, W_i). \tag{2}$$

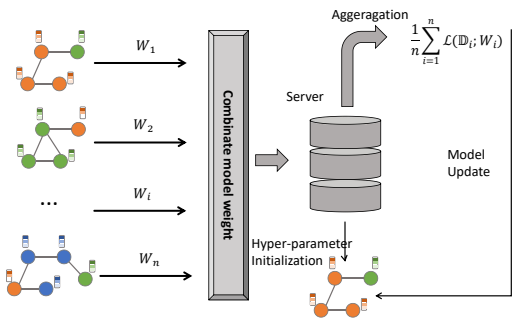


Figure 2: FL training process.

4 OUR PROPOSED METHOD: FEDOMD

This section commences by introducing the notations on GFL. It proceeds to provide an elaborate elucidation of the local orthogonal graph model and CMD. Finally, we present a novel orthogonal GFL paradigm dubbed FedOMD.

4.1 Notations

Each client, denoted by $\mathbb{C}_i$, possesses a graph $G_i = (X_i, V_i, E_i)$, where X_i denotes the feature matrix, and V_i and E_i refer to the node and edge sets, respectively. A_i represents the adjacency matrix and satisfies $(A_i)_{jk} = 1$ iff edge $e_{jk} \in E_i$. The feature vector of the m-th node $V_i(m) \in V_i$ is represented as $X_i(m)$ and labeled with $Y_i(m) \in Y_i$. The objective of a graph classifier is to utilize the information in A_i, X_i, and Y_i to learn a lower-dimensional representation Z^l in the l-th layer. One way to obtain Z^l is by applying the activation function $ReLU$ to $\widetilde{A}Z^{l-1}W^l$, where $\widetilde{A} = D^{-1/2}(A+I)D^{-1/2}$ subject to $D_{ii} = \sum_j (A+I)_{ij}$, and $Z^0 = X_i$. Here, D is a diagonal matrix, and I is the identity matrix. In the GFL scenario, each of the M clients represents an independent entity, such as a hospital. These clients collaborate to learn a global graph model while maintaining the privacy of their raw data, obtaining a more general feature embedding representation.

4.2 Motivations of Orthogonality

Directly utilizing a global model to constrain the local model in FL has achieved significant success in addressing data heterogeneity, such as FedProx [21] and SCAFFLOD [16]. Despite their effectiveness, those models still encounter several unresolved challenges. One is the distribution shifts, and another is global catastrophic forgetting. Distribution shifts in GFL represent that a graph model federally trained on several specific clients may not perform well when deployed on a target domain with a distribution of inputs that differs significantly from the learned global domain. Existing work [4] in the transfer learning field leverages pairwise orthogonality among all features to keep unique information not present in others to relieve the distribution shifts. Global catastrophic forgetting responds to the fact that when the global model is trained on new tasks, its performance on old tasks decreases. The existing FL method, Federated Orthogonal Training (FOT) [1], has relieved the dilemma of global catastrophic forgetting through orthogonal features and has achieved SOTA performance. In GFL, the problem of global catastrophic forgetting can be regarded as: a global model (new task) aggregated with various local models (old tasks) has a decreased performance in some local heterogeneity clients. Those two challenges exist in GFL, which have motivated us to leverage the layer-wise orthogonality feature embedding method to address the data heterogeneity in the GFL field.

4.3 Orthogonalized Node Representation Learning

The weights of neural networks are generally initial with Gaussian distribution, e.g., Xavier [10] and He initialization [15]. Therefore, the layer-wise feature distribution also can be considered with a Gaussian distribution due to the central limit theorem. The global

feature distribution comprises all the local graph feature distributions, which can be formalized with a Gaussian mixture model (GMM). Specifically, the GMM is defined as follows:

$$\mathbf{P}(y|\theta) = \sum_{i=1}^{M} \alpha_i \mathcal{P}(y|\theta_i). \tag{3}$$

α_i is the weight of the i-th client and is constrained to $\alpha_i \geq 0$. The Gaussian distribution with parameters θ_i is represented by $\mathcal{P}(y|\theta_i)$. Moreover, we define the local hidden feature as a multivariate Gaussian distribution, where the dimensionality depends on the hidden features. In the client, it aims to learn $\mathcal{P}(y|\theta_i)$ without considering other participants. However, in the federated scenario, the various clients collaborate to learn the global graph model by incorporating the local $\mathcal{P}(y|\theta_i)$ distributions, ultimately learning a global representation $\mathbf{P}(y|\theta)$ of hidden features.

Orthogonality in GFL. In the FL paradigm, the local graph data may exhibit an inconsistent feature distribution due to non-i.i.d samples. To mitigate this issue, all clients in terms of feature aspects can construct a global distribution. Specifically, the feature distribution in various clients $\mathbb{C}_i, i \in [M]$ can be formalized using a Gaussian distribution as follows:

$$\mathcal{P}(\boldsymbol{X_i}) = |2\pi\Sigma_i|^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{X_i} - \boldsymbol{\mu_i})^T \Sigma_i^{-1} (\boldsymbol{X_i} - \boldsymbol{\mu_i})\right) \tag{4}$$

where $\boldsymbol{X_i}$ is the node features in the i-th client, and $\boldsymbol{\mu_i}$ and Σ_i are the mean vector and covariance matrix, respectively. Assume $\boldsymbol{\mu_i} = \mathbf{0}$, the discrepancy between each client feature data becomes the covariance matrix. Since Σ_i is a constant in the i-client in general, the Gaussian distribution can be simplified to $\mathcal{P}(\boldsymbol{X_i}) \propto \exp\left(-\frac{1}{2}\boldsymbol{X_i}^T \boldsymbol{X_i}\right)$. Without considering the activation function in graph convolution networks as SGC [32] did, the l-layer representation can be linearly transformed to $Z_i^l = (\widetilde{A}_i)^l \boldsymbol{X_i} W^0 W^1 \cdots W^l$. Furthermore, if we set $\widetilde{A}_i \boldsymbol{X_i} W^l = Q_i \boldsymbol{X_i}$, then we can solve the matrix equations $({W^l}^T \otimes \widetilde{A}_i - I \otimes Q_i) Vec(\boldsymbol{X_i}) = \mathbf{0}, l \in [1, L-1]$ with Kronecker product $\otimes$ and vectorization operation $Vec(\cdot) : R^{N \times F} \longmapsto R^{NF}$, which equation futher approximately solved by Newton iteration as [11] did. Note that Newton's iteration approximate method will not largely increase computation costs over the basic model. We set $t = Q_i \boldsymbol{X_i}$. Then, we can calculate $\boldsymbol{X_i} = Q_i^- t$ and substitute it into $\mathcal{P}(\boldsymbol{X_i})$.

$$\begin{aligned} \mathcal{P}(\boldsymbol{t}) &\propto \exp\left(-\frac{1}{2}\left(Q_i^{-1}\boldsymbol{t}\right)^T \left(Q_i^{-1}\boldsymbol{t}\right)\right) \\ &= \exp\left(-\frac{1}{2}\boldsymbol{t}^T \left(Q_i Q_i^T\right)^{-1} \boldsymbol{t}\right). \end{aligned} \tag{5}$$

After conducting transformations in equation 5, the covariance matrix Σ_i can be expressed as $\Sigma_i = Q_i Q_i^T$, where Q_i is a factorization of the covariance matrix. Since the covariance matrix is symmetric, it can be decomposed into its eigenvalues and eigenvectors as $\Sigma_i = U \Lambda U^T$, which implies that $Q_i = U \Lambda^{1/2}$. Following spectral bounding normalization by setting $\widetilde{Q}_i = \frac{Q_i}{\|Q_i\|_F}$, the feature vector X_i can be orthogonally projected by U via graph convolution. Note that our mathematical derivations are limited to one local client and assume that all the clients are independent and have no information exchange. In FL, we cannot always assume the covariance matrix Σ_i equal to the Σ matrix with other clients, as each client owns a different feature distribution $N_i \sim (\boldsymbol{\mu_i}, \Sigma_i)$. To address this issue and integrate all feature distributions into a unified Gaussian distribution $N \sim (\boldsymbol{\mu}, \Sigma)$ in FL scenarios without disclosing private data, we employ CMD to implicitly calculate server feature distribution (i.e., we only need to upload the mean value and calculate CMD distance in figure 3) to constrain the local non-i.i.d feature distribution. In the k-layer feature orthogonal process illustrated in table 1, the reconstruction loss in client i is calculated as follows:

Table 1: Model structure.

order	Layer→Layer	Model	Dimension
1	0 → 1	GCNConv	$R^{d_i} \rightarrow R^{d_h}$
2	1 → 2	OrthoConv	$R^{d_h} \rightarrow R^{d_h}$
...	...	...	...
$l-1$	n-2 → n-1	OrthoConv	$R^{d_h} \rightarrow R^{d_h}$
l	n-1 → n	GCNConv	$R^{d_h} \rightarrow R^{d_o}$

$$\mathcal{L}_{ortho_i} = \sum_{k=2}^{l-1} \left\| W_i^k {W_i^k}^T - I \right\|_F. \tag{6}$$

Orthogonal Networks. In the client, we define a multi-hidden layer orthogonal network to learn a unified i.i.d hidden feature distribution. We calculate the first layer as:

$$Z^1 = \sigma(\widetilde{S} X W^0), \tag{7}$$

where σ is $ReLU$ function, and Z in the $l-1$ hidden layer is calculated as:

$$Z^{l-1} = \sigma(\widetilde{Q} Z^{l-2}). \tag{8}$$

The final layer is calculated as:

$$Z^l = softmax(\widetilde{S} Z^{l-1} W^{l-1}), \tag{9}$$

where $\widetilde{S}$ is the laplace normalised matrix $\widetilde{S} = D^{-1/2}(A+I)D^{-1/2}$ subject to $D_{ii} = \sum_j (A+I)_{ij}$, and $\widetilde{Q} = \frac{Q}{\|Q\|_F}$ is the normalised orthogonal projected matrix.

4.4 Center Moment Optimization

In the server part, we need clients to upload their consensual iid sample representation. However, It is a significant challenge to identify i.i.d feature samples in the FL scenario. To this end, we propose to leverage the empirical global distribution for all features from various clients.

Communication Optimization. However, calculating the empirical global distribution needs all the features from different parties. Uploading all feature vectors into the server in an FL scenario can cause an enormous communication cost. To this end, we propose to upload statistical data instead. Specifically, as illustrated in figure 3, we propose a design that involves uploading $(\mathbf{E}(\boldsymbol{X_i}), n_i)$, which is instrumental in reducing communication costs and facilitating the CMD distance calculation. We calculate the mean of i.i.d

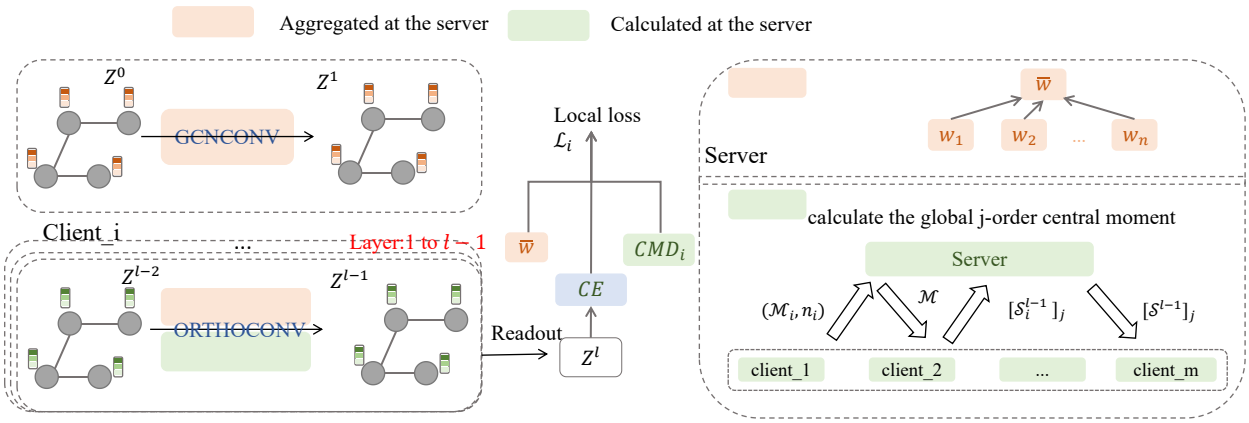


Figure 3: The whole framework of FedOMD. Each client model is set as an orthogonal graph network, and the server is to execute the FedAVG process and calculates the jth-order central moment.

features as follows:

$$\mathcal{M} = \frac{\sum_{i\in[M]} n_i \mathcal{M}_i}{\sum_{i\in[M]} n_i} \quad s.t. \quad \mathcal{M}_i = \mathbf{E}(Z_i). \tag{10}$$

Central Moment Discrepancy. After obtaining the global mean value of the hidden feature in sever, we calculate the jth-order central moment through the two times distribute-upload global mean value process, which depicts more concrete in algorithm 1. After finishing calculating the jth-order central moment, we minimize the distance of the jth-order central moment between the client and server. The CMD involves calculating the jth-order central moment for two types of distribution and can thus achieve distance constraints in the FL scenario. A smaller CMD value indicates a closer distribution and can be defined as follows:

$$\begin{aligned} d_{\mathrm{CMD}_i}\left(Z_i^{\mathrm{train}}, Z_{\mathrm{IID}}\right) &= \frac{1}{b-a}\left\|\mathbf{E}\left(Z_i^{\mathrm{train}}\right) - \mathbf{E}\left(Z_{\mathrm{IID}}\right)\right\|_2 \\ &+ \sum_{j=2}^{\infty} \frac{1}{|b-a|^j}\left\|[C_i]_j\left(Z_i^{\mathrm{train}}\right) - [\mathcal{S}]_j\left(Z_{\mathrm{IID}}\right)\right\|_2, \end{aligned} \tag{11}$$

where the elements of Z are limited to $[a, b]$, and $[C_i]_j, [\mathcal{S}]_j$ denotes the jth-order central moment. The jth-order central moment $[\mathcal{S}]_j$ in server is calculated as $[\mathcal{S}]_j = \mathbf{E}((Z - \mathbf{E}(Z))^j)$ and in client is calculated as $[C_i]_j = \mathbf{E}((Z_i - \mathbf{E}(Z_i))^j)$.

Motivations of CMD. We realize that node representations through neighbor aggregations are truthful and valuable in the real world, even if isolated from other parties. Though they produce personalized data bias, these do not prevent them from communicating with each other across space (e.g., through FL) and fusing to a new global data distribution, which motivates us to design IID-hidden representations in the server to guide federated training. An intuitive example is cultural exchange, which isolates and occasionally communicates with each other.

4.5 Overall Optimization Goal

The total optimization loss comprises three components: (1) the local graph model cross-entropy (CE) loss, (2) the local model orthogonalization loss, and (3) the CMD loss in the federated process. These three parts can be combined with hyper-parameters as follows:

$$\mathcal{L}_i = CE(Z_i^l, Y_i) + \alpha \times \mathcal{L}_{ortho_i} + \beta \times d_{CMD_i}, \tag{12}$$

where $\alpha = 0.0005$ and $\beta = 10$ are set in our experimental settings.

4.6 Algorithm

As algorithm 1 manifested, FedOMD collaboratively computes IID hidden samples representation between server and clients, which aims to guide the local GNN model training. We detail these processes in the following paragraph. In lines 3-7, parties first calculate the local hidden feature mean and subsequently leverage it to calculate the local jth-order central moment. All the participants are required to upload the mean results to sever, shown in line 8. After all the participants have uploaded their data, the server calculates the mean of all parties' data with line 25. After parties have received the server calculation results, they begin to calculate a part of the jth-order central moment (utilize the mean of all participants' hidden features) in lines 12-13 and then upload to the server for calculating the mean value of all participants' jth-order central moment in line 25. Through a 2-round interaction with mean value, all the parties will implicitly obtain the IID's hidden representation. Naturally, the parties calculate the CMD loss in lines 19-20 and leverage the FedAVG in lines 26-29 to update the local model weight.

5 EXPERIMENTS

This section provides detailed experimental settings, including the statistics of the dataset, the baselines and their respective achievements, and the achievement of our FedOMD. Subsequently, we conduct a semi-supervised node classification task on five datasets with various party settings. Lastly, we have conducted several extensive ablations of hyper-parameters. We aim to answer three research questions as follows. **RQ1:** Can FedOMD achieve competitive performance than FL and GFL models? **RQ2:** How do the orthogonalization and CMD cooperate during FedOMD's training? **RQ3:** How do the hyper-parameters affect FedOMD's training?

Algorithm 1 The whole algorithm of FedOMD.

Input: i-th Local Adjacency Matrix A_i, Feature Matrix X_i, and label Y_i.
Parameter:W_i, where $i \in [M]$.
Output: Trained client-side model weights W_i.
ClientExecute:

for client round $1, 2, \cdots, epoch$ **do**
  for each client $i \in [M]$ in parallel **do**
    Z_i^l = **OrthoGCN**(X_i)
    $[\mathcal{M}_i^1, \cdots, \mathcal{M}_i^{l-1}]$ = **CalculateMean**$([Z_i^1, \cdots, Z_i^{l-1}])$ ▷ Calculate local hidden feature mean.
    for $j \in [2, 3, 4, 5]$ **do**
      $[C_i^1, \cdots, C_i^{l-1}]_j = \frac{1}{n_i}\sum_{m=1}^{n_i}([Z_i^1(m), \cdots, Z_i^{l-1}(m)] - [\mathcal{M}_i^1, \cdots, \mathcal{M}_i^{l-1}])^j$ ▷ Calculate local jth-order central moment.
    end for
    Send2Sever$([\mathcal{M}_i^1, \cdots, \mathcal{M}_i^{l-1}], n_i)$ ▷ Upload to server.
    for all clients have executed **Send2Sever do**
      ReciveFromServer$([\mathcal{M}^1, \cdots, \mathcal{M}^{l-1}])$ ▷ Obtain the global hidden feature mean.
    end for
    for $j \in [2, 3, 4, 5]$ **do**
      $[\mathcal{S}_i^1, \cdots, \mathcal{S}_i^{l-1}]_j = \frac{1}{n_i}\sum_{m=1}^{n_i}([Z_i^1(m), \cdots, Z_i^{l-1}(m)] - [\mathcal{M}^1, \cdots, \mathcal{M}^{l-1}])^j$ ▷ Calculate jth-order central moment.
    end for
    Send2Sever$([\mathcal{S}_i^1, \cdots, \mathcal{S}_i^{l-1}]_j, n_i)$
    for all clients have executed **Send2Sever do**
      ReciveFromServer$([\mathcal{S}^1, \cdots, \mathcal{S}^{l-1}]_j)$ ▷ Obtain the global jth-order central moment.
    end for
    $d_{CMD_i} = \sum_{i=1}^{[M]}\sum_{l=1}^{L-1}(\frac{1}{b-a}\|\mathcal{M}_i^l - \mathcal{M}^l\|_2 + \sum_{j=2}^{5}\frac{1}{(b-a)^j}\|[C_i^l]_j - [\mathcal{S}^l]_j\|_2)$
    $\mathcal{L}_i = CrossEntropy(softmax(Z_i^l), Y_i) + \alpha \times \mathcal{L}_{ortho_i} + \beta \times d_{CMD_i}$
    $W_i \leftarrow$ **ServerUpdate**$(W_i(\nabla loss_i))$ ▷ Use FedAVG to aggregate weight.
  end for
end for
Return W
ServerExecute(Z_i):
Return $\mathcal{M}_i = \frac{\sum_{i\in[M]} n_i \mathbf{E}(Z_i)}{\sum_{i\in[M]} n_i}$ ▷ Send2Client and ReciveFromClient.
ServerUpdate:
for $i \in [M]$ **do**
  $\overline{W} = \sum_{i=0}^{[M]} \lambda_i \cdot W_i$ ▷ Use FedAVG to aggregate the weight collected from clients.
end for
return ∇

5.1 Settings

Table 2: The datasets for the statistics graph in our experiments are split into train, validation, and test sets at a ratio of 1%, 20%, and 20% respectively.

Dataset	#Nodes	#Edges	#Classes	#Features
Cora	2,708	5,429	7	1,433
Citeseer	3,312	4,732	6	3,703
Computer	13,381	245,778	10	767
Photo	7,487	119,043	8	745
Coauthor-cs	18,333	182,121	15	6805

Datasets. We assess the efficacy of FL for node prediction on five datasets: Cora, Citeseer, Amazon Electronics Computers (Computer), Amazon Electronics Photo (Photo), and Coauthor Microsoft Academic Computing Science (Coauthor-cs). The pertinent statistics of the datasets are presented in table 2.

Baselines. We selected seven models as our baselines: FedMLP, FedProx [21], SCAFFOLD [16], LocGCN, FedGCN, and FedSage+. FedMLP is a 2-layer multi-layer perception model with a hidden dimension of 64. FedProx calculates a proximal term to the local subproblem based on FedMLP. SCAFFOLD uses control variates to correct for the 'client-drift' in its local updates based on FedMLP. LocGCN employs a two-layer GCN model to train the local graph data and subsequently averages the accuracy across various parties. FedGCN incorporates additional federally learned GCN parameters based on LocGCN. FedSage+ and FedLIT are adapted from the work of [38] and [34], respectively. We propose FedOMD based on the Ortho-GCN model [11] to an FL scenario.

Implementation Details. We utilized the Louvain-cut algorithm to partition a global graph into several local graphs. The hyperparameter "resolution" was set to the default value in the Cora and Citeseer and 20 in the Computer and Photo datasets. We have set the communication interval to 1 and limited the number of parties to 3, 5, 7, 9. The alpha and beta are set to 0.0005 and 10, respectively. We have set the maximum epoch to 1000, the patience to 200, and fixed weight decay to 1e-4. We calculate the average accuracy five times. We conduct experiments on an Intel(R) Xeon(R) Silver 4314 CPU @ 2.40GHz, A100 Nvidia GPU, and system Ubuntu 20.04.6 LTS.

Visualization. To evaluate the effectiveness of our method in a non-i.i.d setting, we split the global graph into several subgraphs, each assigned to different parties, as shown in the figure 4. The subgraphs exhibit high non-i.i.d properties, as evidenced by the distribution of labels. Additionally, the feature distribution varies significantly across different parties, further highlighting the non-i.i.d properties of the dataset.

5.2 Results on Node Classification

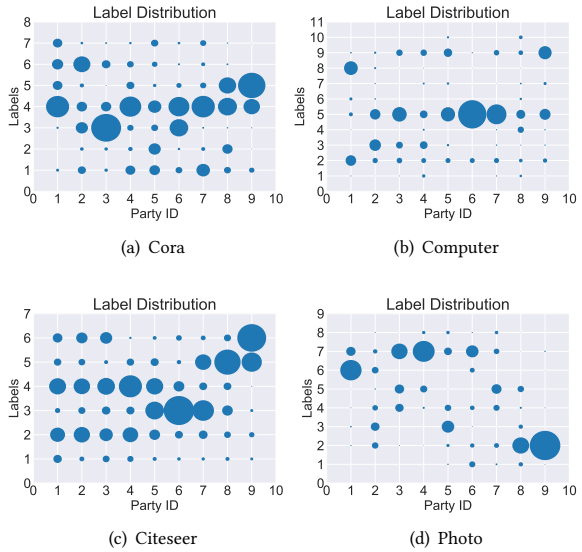
The results from table 4 demonstrate that our proposed FedOMD model has achieved competitive accuracy in node classification tasks. Compared with traditional FL, i.e., FedMLP, FedProx, and SCAFFOLD, our FedOMD has presented significant performance

Table 3: n, m, c, and f are the number of nodes, edges, classes, and feature dimensions, respectively. s is the number of selected augmented nodes, and g is the number of generated neighbors. k corresponds to the number of times we aggregate features. Besides, N is the number of participating clients in each training round. L is the number of model layers in their original paper.

Model	Client Time	Server Time	Inference Time
FedMLP	$O(nf^2)$	$O(N)$	$O(nf^2)$
FedProx	$O(nf^2 + f^2)$	$O(N)$	$O(nf^2)$
SCAFFOLD	$O(nf^2 + f^2)$	$O(N + Nf^2 + f^2)$	$O(nf^2)$
FedGCN	$O(kmf + nf^2)$	$O(N)$	$O(kmf + nf^2)$
FedLIT	$O(kmf + nf^2)$	$O(N + Nf^2 + f)$	$O(kmf + nf^2)$
FedSage+	$O(L(m + sg)f + L(n + sg)f^2)$	$O(N)$	$O(L(m + sg)f + L(n + sg)f^2)$
FedOMD	$O(kmf + nf^2 + f^2 + n^2f)$	$O(N + N^2f^2 + Nf)$	$O(kmf + nf^2)$

Table 4: We contrast seven baselines in four graph datasets with our FedOMD. The party number is set in $\{3, 5, 7, 9\}$. The best test results (accuracy ± std) are bold.

Acc(%)	**Cora**				**Citeseer**			
Model	**M=3**	**M=5**	**M=7**	**M=9**	**M=3**	**M=5**	**M=7**	**M=9**
FedMLP	33.37 (±3.32)	29.03 (±2.37)	28.12 (±2.90)	30.69 (±1.89)	31.29 (±2.59)	29.77 (±4.17)	28.69 (±3.91)	26.96 (±4.13)
SCAFFOLD	34.01 (±4.19)	31.88 (±0.98)	30.38 (±3.74)	32.89 (±1.34)	35.49 (±5.00)	32.08 (±4.15)	31.18 (±2.41)	30.53 (±3.17)
FedProx	32.84 (±6.20)	38.09 (±5.28)	33.54 (±2.43)	36.96 (±2.39)	35.96 (±6.33)	34.10 (±5.69)	32.23 (±3.64)	34.00 (±1.68)
LocGCN	42.73 (±4.52)	38.31 (±2.25)	34.05 (±4.16)	36.08 (±4.68)	39.87 (±3.44)	32.86 (±1.29)	33.56 (±3.74)	29.59 (±6.74)
FedGCN	47.12 (±7.07)	44.27 (±2.73)	38.45 (±3.76)	**41.22 (±3.03)**	42.98 (±2.58)	36.59 (±4.09)	36.89 (±3.06)	33.87 (±6.83)
FedLIT	33.33 (±1.91)	42.05 (±5.10)	34.67 (±1.03)	27.67 (±3.84)	36.24 (±2.01)	36.71 (±1.77)	33.26 (±1.03)	32.23 (±1.61)
FedSage+	40.72 (±3.52)	39.73 (±3.58)	37.43 (±4.11)	38.74 (±2.49)	40.59 (±3.75)	37.78 (±3.83)	37.01 (±4.05)	35.40 (±1.65)
FedOMD	**54.35 (±5.86)**	**50.10 (±7.49)**	**41.90 (±4.71)**	38.19 (±5.25)	**53.08 (±3.67)**	**39.38 (±1.85)**	**40.46 (±3.07)**	**35.67 (±3.62)**
Acc(%)	**Computer**				**Photo**			
Model	**M=3**	**M=5**	**M=7**	**M=9**	**M=3**	**M=5**	**M=7**	**M=9**
FedMLP	41.22 (±3.42)	41.13 (±2.08)	41.05 (±1.79)	40.09 (±3.04)	35.92 (±9.09)	46.87 (±3.40)	39.77 (±7.58)	40.22 (±1.20)
SCAFFOLD	41.82 (±3.52)	40.45 (±2.47)	41.72 (±1.05)	42.01 (±1.54)	34.40 (±4.50)	39.41 (±5.37)	45.84 (±3.26)	43.17 (±1.86)
FedProx	41.29 (±3.66)	44.21 (±4.04)	41.61 (±4.28)	45.34 (±4.12)	40.14 (±7.48)	46.05 (±6.48)	48.84 (±3.71)	49.00 (±8.87)
LocGCN	55.33 (±5.77)	54.57 (±6.18)	54.98 (±3.67)	54.09 (±2.19)	65.93 (±4.20)	62.84 (±5.05)	62.53 (±4.00)	66.76 (±3.24)
FedGCN	42.42 (±2.29)	42.07 (±3.71)	39.57 (±1.22)	40.12 (±2.54)	57.82 (±5.47)	53.65 (±2.46)	51.69 (±3.66)	49.04 (±3.01)
FedLIT	49.69 (±4.12)	36.34 (±0.34)	53.07 (±4.53)	51.12 (±5.31)	73.44 (±2.67)	44.11 (±1.07)	39.28 (±6.89)	38.45 (±5.94)
FedSage+	59.46 (±2.40)	42.68 (±7.26)	47.02 (±7.12)	47.66 (±6.74)	63.23 (±3.24)	59.28 (±3.06)	56.43 (±2.63)	58.09 (±6.50)
FedOMD	**68.08 (±3.68)**	**71.40 (±2.59)**	**71.01 (±2.77)**	**68.71 (±1.96)**	**77.71 (±2.68)**	**75.98 (±2.09)**	**76.05 (±3.52)**	**78.30 (±2.36)**



(a) Cora (b) Computer

(c) Citeseer (d) Photo

Figure 4: The distribution of labels is non-i.i.d across different parties. This figure illustrates the number of labels in various parties using blue circles, with the area of the circles corresponding to the number of labels. A larger area indicates more label counts. The non-i.i.d label distribution approximately displays the non-i.i.d feature distribution in our experimental graph datasets, in that we cut a whole graph into several subgraphs while keeping the consistency of feature space.

promotion. Notably, when training with our label rate, the performances of FedLIT and FedSage+ have comparatively decreased over their oracle federated models. We attribute the reason to the demand of FedLIT and FedSage+ for massive samples to cluster latent link types and maintain sampling effectiveness, respectively. However, when the node samples are not randomly and uniformly acquired, the sample distribution can harm the learning of a global model. Compared to other GFL models, FedSage+ may underperform or even perform worse than it (e.g., FedGCN). Our FedOMD model aims to address the non-i.i.d effects in federated aggregation by imposing a global i.i.d sample constraint using the CMD distance, which presents high effectiveness in node classification. Notably, only a few statistical data of local features are required to calculate the CMD distance, causing negligible communication costs in algorithm 1, and we show the relevant calculation time in table 3. We tested our model with various party numbers 3, 5, 7, and 9 and have achieved almost the best performance. As table 5 shows, we also conduct experiments on more party numbers, i.e., 20 and 50, and manifest the superior advantages of our model in 20 and 50 parties. All those mentioned analyses have fully answered the **RQ1**.

Convergence Analysis. As figure 5 illustrates, our FedOMD model has manifested stable performance along with the increasing communication round, which verifies the convergence of FedOMD. Besides, the baseline model's performance also converges, which

Table 5: The more numbers of parties are analyzed. We utilize {20, 50} parties as our experiment settings.

Acc(%) Models	Coauthor-cs M=20	Coauthor-cs M=50
FedMLP	32.61 (±3.96)	32.65 (±4.62)
SCAFFOLD	40.45 (±5.79)	33.54 (±1.48)
FedProx	39.68 (±2.07)	36.81 (±1.48)
LocGCN	42.24 (±4.33)	37.93 (±3.40)
FedGCN	42.49 (±7.49)	39.49 (±2.75)
FedLIT	52.10 (±3.44)	34.58 (±1.46)
FedSage+	64.80 (±0.11)	44.23 (±0.13)
FedOMD	**70.35 (±0.85)**	**48.47 (±5.13)**

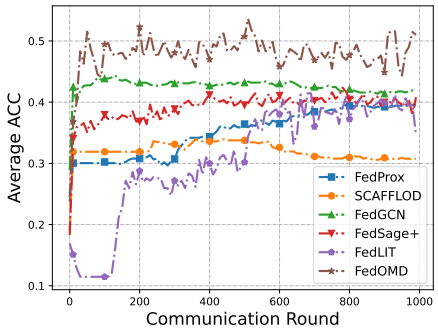


Figure 5: The average test accuracy with 5 parties (Cora).

shows the model's learning ability. Our model has also exhibited the advantage of convergence speed than those federated models.

5.3 Ablation Analysis

Effects of Key Components Mechanism. To answer **RQ2**, we conduct ablation experiments to various modules, i.e., orthogonalization and CMD. As shown in table 6, the combination of feature orthogonalization and CMD distance constraint has demonstrated the best performance improvement. The CMD distance plays a more critical role in enhancing the performance of FL than orthogonal modules by leveraging i.i.d samples to learn an i.i.d feature representation across all parties. The orthogonal constraint has also promoted the model performance, which manifests that orthogonal constraint can reduce the effect of feature distribution shifts and global catastrophic forgetting problems in GFL to some extent.

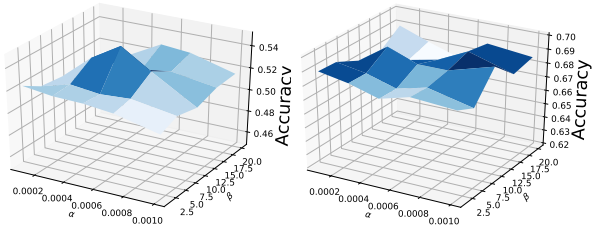
5.4 Sensitivity Analysis

Hyper-parameters in Loss Function. As shown in figure 6, the hyperparameters α and β exhibit different effects in the Cora and Computer datasets, suggesting that the performance of our FedOMD model is somewhat sensitive to the choice of hyperparameter values. This sensitivity arises from differences in the magnitude

Table 6: The average test accuracy with different combinations of augmentation strategies.

Ortho	CMD	Cora			
		M=3	M=5	M=7	M=9
✔	✗	52.13 (±9.13)	46.26 (±9.96)	33.95 (±8.21)	36.14 (±0.70)
✗	✔	54.28 (±5.61)	49.48 (±5.02)	41.12 (±4.48)	37.49 (±2.02)
✔	✔	54.35 (±5.86)	50.10 (±7.49)	41.90 (±4.71)	38.19 (±5.25)
Ortho	**CMD**	**CiteSeer**			
		M=3	**M=5**	**M=7**	**M=9**
✔	✗	51.58 (±2.65)	37.31 (±3.40)	35.85 (±3.45)	32.68 (±3.52)
✗	✔	52.46 (±4.37)	39.03 (±2.05)	40.43 (±2.90)	35.03 (±2.03)
✔	✔	53.08 (±3.67)	39.38 (±1.85)	40.46 (±3.07)	35.67 (±3.62)

of the effect produced by changes to α and β. To simplify the experimental setup, we fixed α at 0.0005 and β at 10, achieving competitive performance across both datasets.



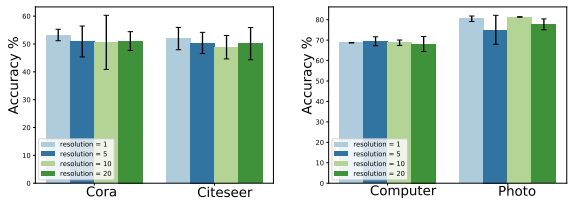
(a) Cora (b) Computer

Figure 6: The average test accuracy with different combinations of hyper-parameters (α, β) to the FedOMD model with three parties.

The Number of Orthogonal Layer. Table 7 has shown the effect of the number of hidden layers on FedOMD. Too many hidden layers will cause the over-smoothing problem, thus leading to performing worse in FL scenarios, while our FedOMD has successfully relieved this over-smoothing tendency. As table 7 manifested, our model with a 10-hidden layer still presents better performance than a total 2-layer FedGCN model. The main reasons we include as two aspects. One is to use orthogonalization to keep a distinguishable vector, and another is to utilize CMD distance to relieve the non-i.i.d phenomena.

Resolution. In figure 7, we also explore the impact of the Louvain-cut algorithm [2] with the resolution parameter, which determines the threshold for cutting the graph into various scale subgraphs. A larger resolution value will produce multiple smaller subgraphs. From the effect of the Louvain-cut algorithm on the accuracy, we analyze that it is easier to obtain higher accuracy with a small parameter value since it has cut a global graph into several large connected graphs from the observation of Cora and Citeseer datasets. As our experiential cognition, if cutting a large graph into several connected graphs to various participants, the local model also can perform well based on the observation of the Computer and Photo datasets.

Different hyper-parameters have significantly distinguished effects on FedOMD in the training process. Choosing proper hyper-parameters can effectively promote the performance of FedOMD, which can answer **RQ3**.



(a) Cora and Citeseer (b) Computer and Photo

Figure 7: This table illustrates the impact of hyper-parameter resolution for four datasets with three parties.

6 CONCLUSIONS

This paper proposes a novel framework for GFL, called FedOMD, which addresses the challenge of non-i.i.d features across different parties. FedOMD constructs a mixed multivariate Gaussian distribution in the server by integrating all the parties' local multivariate Gaussian distributions. The CMD distance is minimized between the server and the parties' distributions to learn a global i.i.d feature representation. The extensive experiments on various graph datasets have demonstrated the obvious advantages over other GFL approaches, and the ablation and sensitivity analyses also show the effectiveness of FedOMD.

ACKNOWLEDGMENTS

This work is supported by Beijing Municipal Science & Technology Commission New generation of information and communication technology innovation Research and demonstration application of key technologies for privacy protection of massive data for large model training and application(Z231100005923047).

REFERENCES

[1] Yavuz Faruk Bakman, Duygu Nur Yaldiz, Yahya H Ezzeldin, and Salman Avestimehr. 2023. Federated orthogonal training: Mitigating global catastrophic forgetting in continual federated learning. *arXiv preprint arXiv:2309.01289* (2023).
[2] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.
[3] Deyu Bo, BinBin Hu, Xiao Wang, Zhiqiang Zhang, Chuan Shi, and Jun Zhou. 2022. Regularizing Graph Neural Networks via Consistency-Diversity Graph Augmentations. (2022).
[4] Annie S Chen, Yoonho Lee, Amrith Setlur, Sergey Levine, and Chelsea Finn. 2023. Project and Probe: Sample-Efficient Adaptation by Interpolating Orthogonal Features. In *The Twelfth International Conference on Learning Representations*.
[5] Chaochao Chen, Jun Zhou, Longfei Zheng, Huiwen Wu, Lingjuan Lyu, Jia Wu, Bingzhe Wu, Ziqi Liu, Li Wang, and Xiaolin Zheng. 2020. Vertically federated graph neural network for privacy-preserving node classification. *arXiv preprint arXiv:2005.11903* (2020).
[6] Fengwen Chen, Guodong Long, Zonghan Wu, Tianyi Zhou, and Jing Jiang. 2022. Personalized federated learning with graph. *arXiv preprint arXiv:2203.00829* (2022).

Table 7: The various number of hidden layers is analyzed. We utilize two hidden layers as our experiment settings.

Models	Layer	Computer				Photo			
		M=3	M=5	M=7	M=9	M=3	M=5	M=7	M=9
FedOMD	2-hidden	**68.08 (±3.68)**	**70.40 (±2.59)**	**71.01 (±2.77)**	**68.71 (±1.96)**	**77.71 (±2.68)**	**75.98 (±2.09)**	**76.05 (±3.52)**	**78.30 (±2.36)**
	4-hidden	63.44 (±1.98)	66.14 (±3.82)	67.04 (±2.88)	68.34 (±3.71)	73.97 (±2.83)	71.27 (±3.79)	69.54 (±5.35)	73.82 (±2.48)
	6-hidden	54.62 (±5.06)	58.19 (±3.27)	63.35 (±3.90)	64.05 (±1.69)	71.25 (±3.30)	67.46 (±4.46)	66.75 (±5.14)	68.47 (±0.37)
	8-hidden	51.48 (±3.28)	52.20 (±3.37)	57.78 (±4.56)	59.08 (±2.64)	61.81 (±7.37)	62.10 (±5.48)	62.35 (±3.50)	62.38 (±2.73)
	10-hidden	46.23 (±3.98)	51.90 (±3.13)	51.84 (±4.29)	50.19 (±4.42)	58.26 (±5.65)	59.43 (±3.16)	59.60 (±3.72)	57.38 (±3.11)
FedGCN	2-GCNConv	42.42 (±2.29)	42.07 (±3.71)	39.57 (±1.22)	40.12 (±2.54)	57.82 (±5.47)	53.65 (±2.46)	51.69 (±3.66)	49.04 (±3.01)

[7] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and deep graph convolutional networks. In *International Conference on Machine Learning*. PMLR, 1725–1735.

[8] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 3558–3565.

[9] Xingbo Fu, Binchi Zhang, Yushun Dong, Chen Chen, and Jundong Li. 2022. Federated graph machine learning: A survey of concepts, techniques, and applications. *ACM SIGKDD Explorations Newsletter* 24, 2 (2022), 32–47.

[10] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 249–256.

[11] Kai Guo, Kaixiong Zhou, Xia Hu, Yu Li, Yi Chang, and Xin Wang. 2022. Orthogonal graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 3996–4004.

[12] William L Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Neural Information Processing Systems* (2017), 1025–1035.

[13] Chaoyang He, Keshav Balasubramanian, Emir Ceyani, Carl Yang, Han Xie, Lichao Sun, Lifang He, Liangwei Yang, Philip S Yu, Yu Rong, et al. 2021. Fedgraphnn: A federated learning system and benchmark for graph neural networks. *arXiv preprint arXiv:2104.07145* (2021).

[14] Dongxiao He, Chundong Liang, Huixin Liu, Mingxiang Wen, Pengfei Jiao, and Zhiyong Feng. 2022. Block modeling-guided graph convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 4022–4029.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*. 1026–1034.

[16] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*. PMLR, 5132–5143.

[17] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations* (2017).

[18] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997* (2018).

[19] Runze Lei, Pinghui Wang, Junzhou Zhao, Lin Lan, Jing Tao, Chao Deng, Junlan Feng, Xidian Wang, and Xiaohong Guan. 2023. Federated Learning over Coupled Graphs. *IEEE Transactions on Parallel and Distributed Systems* 34, 4 (2023), 1159–1172.

[20] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine* 37, 3 (2020), 50–60.

[21] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems* 2 (2020), 429–450.

[22] Rui Liu, Pengwei Xing, Zichao Deng, Anran Li, Cuntai Guan, and Han Yu. 2022. Federated graph neural networks: Overview, techniques and challenges. *arXiv preprint arXiv:2202.07256* (2022).

[23] Songtao Lu, Yawen Zhang, and Yunlong Wang. 2020. Decentralized federated learning for electronic health records. In *2020 54th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 1–5.

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

[25] Jaehoon Oh, Sangmook Kim, and Se-Young Yun. 2021. Fedbabu: Towards enhanced representation for federated image classification. *arXiv preprint arXiv:2106.06042* (2021).

[26] Liang Peng, Nan Wang, Nicha Dvornek, Xiaofeng Zhu, and Xiaoxiao Li. 2022. Fedni: Federated graph learning with network inpainting for population-based disease prediction. *IEEE Transactions on Medical Imaging* (2022).

[27] Yue Tan, Yixin Liu, Guodong Long, Jing Jiang, Qinghua Lu, and Chengqi Zhang. 2023. Federated learning on non-iid graphs via structural knowledge sharing. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 37. 9953–9961.

[28] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. *International Conference on Learning Representations* (2018).

[29] Binghui Wang, Ang Li, Meng Pang, Hai Li, and Yiran Chen. 2022. Graphfl: A federated learning framework for semi-supervised node classification on graphs. In *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 498–507.

[30] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *The world wide web conference*. 2022–2032.

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

[32] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*. PMLR, 6861–6871.

[33] Han Xie, Jing Ma, Li Xiong, and Carl Yang. 2021. Federated graph classification over non-iid graphs. *Advances in Neural Information Processing Systems* 34 (2021), 18839–18852.

[34] Han Xie, Li Xiong, and Carl Yang. 2023. Federated node classification over graphs with latent link-type heterogeneity. In *Proceedings of the ACM Web Conference 2023*. 556–566.

[35] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems* 33 (2020), 5812–5823.

[36] Werner Zellinger, Thomas Grubinger, Edwin Lughofer, Thomas Natschläger, and Susanne Saminger-Platz. 2017. Central moment discrepancy (cmd) for domain-invariant representation learning. *arXiv preprint arXiv:1702.08811* (2017).

[37] Bohang Zhang, Shengjie Luo, Liwei Wang, and Di He. 2023. Rethinking the expressive power of gnns via graph biconnectivity. *arXiv preprint arXiv:2301.09505* (2023).

[38] Ke Zhang, Carl Yang, Xiaoxiao Li, Lichao Sun, and Siu Ming Yiu. 2021. Subgraph federated learning with missing neighbor generation. *Advances in Neural Information Processing Systems* 34 (2021), 6671–6682.

[39] Longfei Zheng, Jun Zhou, Chaochao Chen, Bingzhe Wu, Li Wang, and Benyu Zhang. 2021. Asfgnn: Automated separated-federated graph neural network. *Peer-to-Peer Networking and Applications* 14, 3 (2021), 1692–1704.

[40] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in Neural Information Processing Systems* 33 (2020), 7793–7804.