Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 8545101, 13 pages
https://doi.org/10.1155/2023/8545101

WILEY | Hindawi

Research Article

Privacy-Preserving Federated Graph Neural Network Learning on Non-IID Graph Data

Kainan Zhang,[1] **Zhipeng Cai**,[1] **and Daehee Seo**[2]

[1]*Department of Computer Science, Georgia State University, Atlanta, GA 30303, USA*
[2]*Department of Computer Science, Sangmyung University, Seoul, Republic of Korea*

Correspondence should be addressed to Zhipeng Cai; zcai@gsu.edu

Received 3 August 2022; Revised 26 September 2022; Accepted 30 September 2022; Published 3 February 2023

Academic Editor: Yan Huo

Since the concept of federated learning (FL) was proposed by Google in 2017, many applications have been combined with FL technology due to its outstanding performance in data integration, computing performance, privacy protection, etc. However, most traditional federated learning-based applications focus on image processing and natural language processing with few achievements in graph neural networks due to the graph's nonindependent identically distributed (IID) nature. Representation learning on graph-structured data generates graph embedding, which helps machines understand graphs effectively. Meanwhile, privacy protection plays a more meaningful role in analyzing graph-structured data such as social networks. Hence, this paper proposes PPFL-GNN, a novel privacy-preserving federated graph neural network framework for node representation learning, which is a pioneer work for graph neural network-based federated learning. In PPFL-GNN, clients utilize a local graph dataset to generate graph embeddings and integrate information from other collaborative clients to utilize federated learning to produce more accurate representation results. More importantly, by integrating embedding alignment techniques in PPFL-GNN, we overcome the obstacles of federated learning on non-IID graph data and can further reduce privacy exposure by sharing preferred information.

1. Introduction

Data providers sometimes share their data to improve the analytical performance of all participants. However, the collaboration among data providers risks privacy leakage of data owners. Insecure data sharing coupled with poor deanonymization is the same as giving away the owner's information for free. Federated learning (FL) is a comparatively different learning strategy that eludes data collection in a centralized location [1], where a typical server model may reveal a user's sensitive data that he/she is not willing to share. Under this concern, FL is aimed at training deep neural networks on multiple local datasets present on local clients without explicitly exposing the data samples to either the central server or cooperating clients.

Graph data is helpful for processing tasks involving complex relationships and dynamic schemata, such as supply chain management and recommendation systems. Although graph neural network stands out by utilizing representation learning to accomplish graph analysis tasks such as node classification and link prediction in the current big data era [2], there are several reasons preventing FL from being widely applied in the domain of graph neural networks. Unlike most earlier federated learning researches with IID computer vision or language data underlying, the non-IID nature of graphs [3] may cause FL resulting in a worse accuracy than the centralized approach when the training dataset becomes large and noisy as real-world graphs and GNNs tend to overfit the training set if it is not properly regularized [4]. Worse, the aggregation mechanism of FL may fail on sparse graphs, where nodes within local neighborhoods provide more noise than useful information for feature aggregation [5]. More broadly, the diversity of the GNNs model makes the current definition of federated GNNs not uniform and unclear [6]. In addition, most of the existing FL algorithms, such as the naive FedAvg algorithm, are designed for the IID dataset, so it is difficult to effectively integrate the information between various clients in common federated

GNN setting [7]. Especially when clients have different sample nodes and cannot share the complete topology information for privacy concerns, applying the leading traditional averaging strategy to the federated process is not suitable because the input nodes of the graph neural networks are different.

Hence, to solve the aforementioned challenges, we propose a novel federated learning framework for graph neural networks with the embedding alignment technique. Because the framework only needs to integrate client-preferred public information, it can significantly reduce the risk of privacy disclosure during the learning process. The embedding alignment technique ensures that the clients holding non-IID data can change information. Furthermore, we find that injecting aligned information into the local model has regularization effects empirically and thus reduces the risk of overfitting. The main contributions of our work are summarized as follows:

(i) We investigate a general training scenario of the federated GNN setting in which multiple clients hold non-IID graph datasets sharing partial structural equivalence

(ii) We propose a novel framework to integrate federated learning and embedding alignment techniques into an end-to-end process flow to obtain accurate embedding results for individual clients

(iii) We conduct extensive experiments on ground truth datasets to prove the effectiveness of the proposed method with the embedding alignment technique and demonstrate the competitive performance of PPFL-GNN framework with respect to noise resistance

2. Related Works

2.1. Federated Learning with Non-IID Dataset. The non-IID local data usually brings statistical challenges for federated learning, which hurts training convergence and significantly reduces accuracy. To conquer the problem, Zhao et al. propose a strategy to improve the training of non-IID data by creating a small portion of data globally shared among all edge devices [8]. To offset the bias introduced by non-IID data and accelerate convergence, Wang et al. propose Favor [9], an experience-driven control framework, which can intelligently select client devices to participate in each round of federated learning. As another research direction, many FL algorithms are proposed to address the problem of learning efficiency under non-IID data settings. FedProx [10] is a generalization and reparametrization of FedAvg, which pioneers in tackling federated network heterogeneity. In FedPD [11], the authors also explore the nonconvex behavior of the FedAvg algorithm and propose a federated learning framework with optimal rates and adaptivity to non-IID data. Similarly, Li et al. propose FedBN [12], which uses local batch normalization to alleviate the feature shift before averaging models with the convergence rate speed-up. However, after conducting extensive experiments, Li et al. [13] find that the current state-of-the-art FL algorithm cannot outperform other algorithms in all cases with comprehensive data partitioning strategies that cover the typical non-IID data cases. Moreover, to achieve differential privacy in federated learning under a non-IID scenario, Xiong et al. design the 2DP-FL algorithm [14] that adds flexible noise to meet various privacy standards. Although these prior methods have been making progress in different fields, none of them consider using the graph with nature non-IID regarding characteristics as the experiment dataset.

2.2. Federated Learning on Graph Neural Networks. Compared with the voluminous progress made in the vision and language domains, researches about federated learning on graphs are still relatively lacking. For example, SGNN [15] uses a similarity-based graph neural network to capture the structural information of nodes, but it only borrows the thought of federated learning to hide the original information from different data sources. More like a variant of federated learning, Lalitha et al. propose a distributed learning algorithm in which nodes update their beliefs by aggregating information from neighbors and learn the most suitable model of the entire network [16]. Recently, the appearance of FedGraphNN [17] promotes federated learning research based on GNN as an open research federated learning system and benchmark. However, their experimental results pose significant challenges in federated GNN training. For example, federated GNNs perform worse in most datasets with a non-IID split than centralized GNNs, indicating that more research is necessary for this field.

Moreover, federated GNN inherits the core problems from traditional federated settings including expensive communication, systems heterogeneity, statistical heterogeneity [18], and privacy concerns [10]. For instance, to handle the statistical heterogeneity of the data, He et al. propose SpreadGNN [19], a novel multitask federated training framework, which can run in the presence of partial labels and no central server by utilizing decentralized periodic averaging SGD to solve decentralized multitask learning problems. Aiming to moderate the privacy concerns, Sajadmanesh and Gatica-Perez develop a privacy-preserving, architecture-agnostic GNN learning algorithm with formal privacy guarantees based on local differential privacy [20], which also aggregates multihop nodes' features to denoise the noisy labels. In addition to general models and theoretical research, the application of federated GNN to practical problems is also worth studying. For example, FedGNN proposes a federated framework for the GNN-based recommendation system [21], which can collectively train GNN models from decentralized user data while using high-level user-item interaction information to preserve privacy. In the remainder of this article, we also address these four challenges in our work and discuss the framework's applicability.

2.3. Embedding Space Alignment. The embedding approach has become a primary topic in machine learning and graphical analysis [22, 23]. Naturally, the alignment of different embedding spaces plays an important role similar to the translation in the communication of different languages. As the pioneer of alignment technique, cross-lingual word

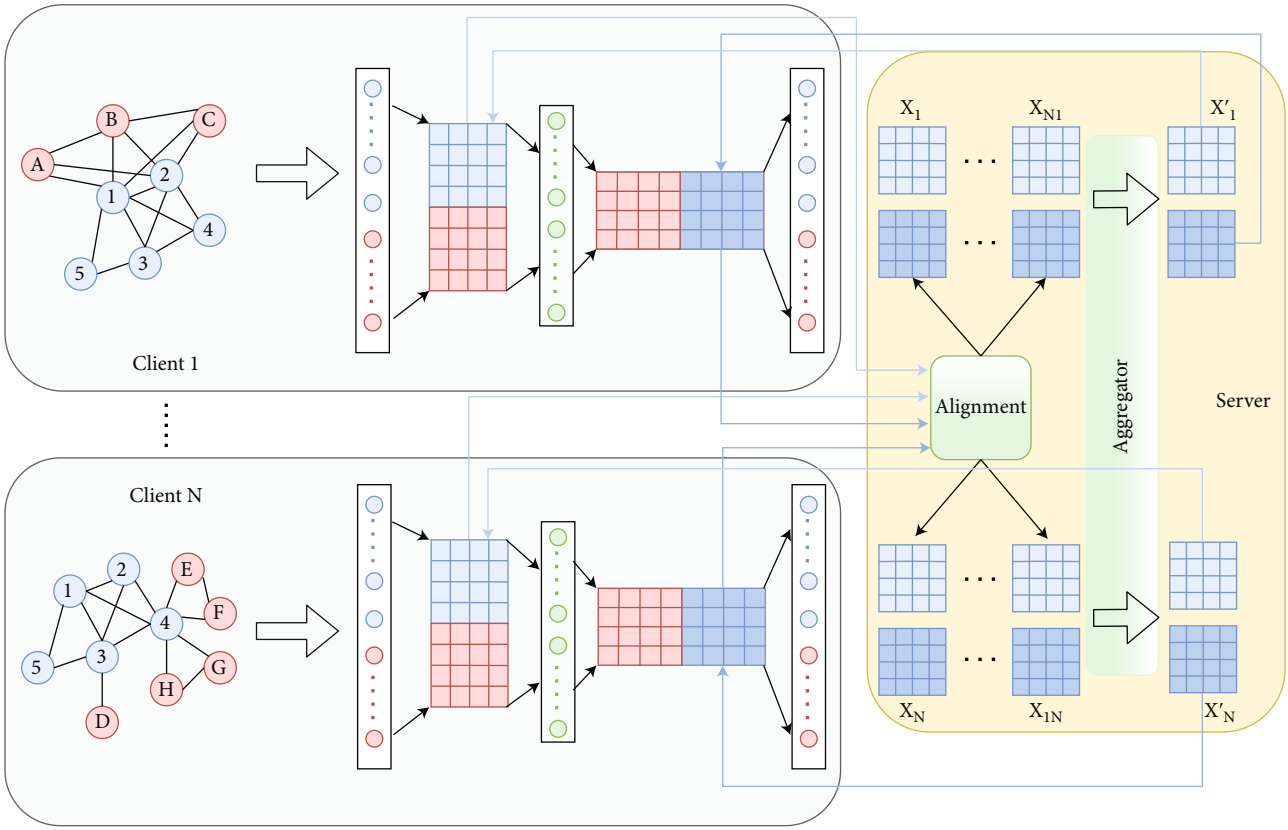

Figure 1: Overview of the federated DeepWalk framework. The red nodes are private, and the blue nodes are public. Local training is highlighted in grey, and server aggregation is highlighted in yellow.

embedding alignments have rapidly grown in the past few years [24]. Both MUSE [25] and VecMap [26] provide modern and oft-cited toolkits for bilingual lexical induction (BLI) datasets. With the development of knowledge-driven applications such as question answering and knowledge graph completion, substantial researches on knowledge graph embedding alignments have emerged recently [27, 28]. These thorough studies enlighten us to apply the existing alignment technique, instead of the training target, but as a tool of information extraction and data integration during the training process.

3. Proposed Work

In this section, we first introduce the problem formulation of our work and then explain the details of our approach to learning graph representation in a privacy-preserving way based on two state-of-the-art models.

3.1. Problem Formulation. Denote $\mathbf{C} = \{c_1, c_2, \cdots c_n\}$ as the sets of clients participating in federated learning, and client c_i holds a local undirected graph $G = (\mathbf{U}, \mathbf{E}, \mathbf{F})$ including node set $\mathbf{U}$, edge set $\mathbf{E}$, and node-feature set $\mathbf{F}$. We assume that all the local graphs share a certain amount of nodes defined as public node set $\mathbf{U}_k = \mathbf{U}_1 \cap \mathbf{U}_2 \cap \cdots \cap \mathbf{U}_n$. To protect privacy, each client saves the original data locally, including the edge and attribute information of nonpublic nodes. Only the processed public node information, which is generated as public node embedding by the client's local model, will be uploaded to the server. Our goal is to generate accurate node representation for each client by utilizing federated learning without building and storing the entire graph on the server or client.

3.2. Federated DeepWalk. DeepWalk extends the idea of language modeling to network topology [29], which forms the embryo of graph embedding. Given a random walk sequence composed of network nodes,

$$V_1^n = (v_0, v_1, \cdots, v_n), \tag{1}$$

where $v_i \in \mathbf{U}$. The goal so far is to retrieve the likelihood of observing v_i given the previous $i-1$ nodes in the random walk:

$$\Pr(v_i | (v_1, v_2, \cdots, v_{i-1})). \tag{2}$$

To learn the latent representation, instead of only a probability distribution of node cooccurrence, DeepWalk introduces a mapping function $\Phi : v \in V \longrightarrow R_{|V| \times d}$, which actually is a $|V| \times d$ matrix of free weights serving as the low-dimensional representations of all network nodes in the graph.

Input: $\mathbf{C} = \{c_1, c_2, \cdots c_n\}$: the set of clients
G_i: the local subgraph hold by c_i
$\mathbf{U}_k$: the public nodes shared among $\mathbf{C}$
Output: the matrix of node representation $\Phi_i \in R^{|V|\times d}$ of G_i

LOCAL CLIENTS:
for each client $c_i \in \mathbf{C}$ **do**
  Compute the DeepWalk model weights Φ_i
  Generate the public nodes' embeddings X_i of U_k from Φ_i:
    $X_i = \{\Phi_i(u_1), \cdots, \Phi_i(uk)\}$
  Upload X_i to the server
end for

while not converge **do**
  SERVER:
  for each $i \in k$ **do**
    for each $j \in k(i \neq j)$ **do**
      Align X_j into c_i's space: $X_{ji} = W_{ji}X_j$
    end for
    Aggregate all the aligned embeddings with X_i
      $X_i' = 1/k(\sum_j^k X_{ji} + X_i)$
    distribute X_i' to client c_i for local update
  end for

  LOCAL CLIENTS:
  for each client $c \in \mathbf{C}$ **do**
    Substitute the public nodes' embeddings in Φ_i by X_i'
      $\Phi_i' \longleftarrow (\Phi_i, X_i')$
    Initial the DeepWalk model with Φ_i'
    Compute the model weights Φ_i
  end for
end while
return the matrix of node representation $\Phi_i \in R^{|V|\times d}$ of G*i*

Algorithm 1:The federated DeepWalk framework.

However, the computation is not efficient depending on the length of the random walks. Thus, the SkipGram method in Word2vec [30] is applied to solve the computational problem. Rather than predicting the occurrence of a missing node in the walk, we compute the likelihood of a node appearing as a neighbor in a given window, and the new optimization goal is summarized as follows:

$$\min_{\phi} -\log \Pr(\{(v_{i-w}, \cdots, v_{i-1}, v_{i+1}, \cdots, v_{i+w})\}|\Phi(v_i)), \quad (3)$$

where w is the window size for iterating the possible collocation of the given node v_i. Suppose we deploy DeepWalk as the neural network model in the federated learning setting, then the local client c_i can train a low-dimensional latent representation $R^{|V|\times d}$ of his local graph G_i. After all clients have generated their local graph embeddings, the challenge of a federated learning setting is how all clients collaborate to improve the training results with less disclosure of sensitive information.

In traditional federated learning, each client uploads all weights of the local model to a central server. The central server aggregates these weights to update the global model and then distributes the global model back to the clients. However, in our problem definition, we cannot aggregate all weights directly because each client holds a different subgraph of the global network, which means that the trained latent representations only share commonality on the public nodes partially. Because the potential relationship between public and private nodes is stored in the public nodes' latent representations, as shown in Figure 1, instead of uploading all weights (i.e., the latent representations of all the nodes in the local graph), a client can only upload the weights related to the public nodes (i.e., the latent representations of public nodes), which also carry some sensitive information of the private nodes.

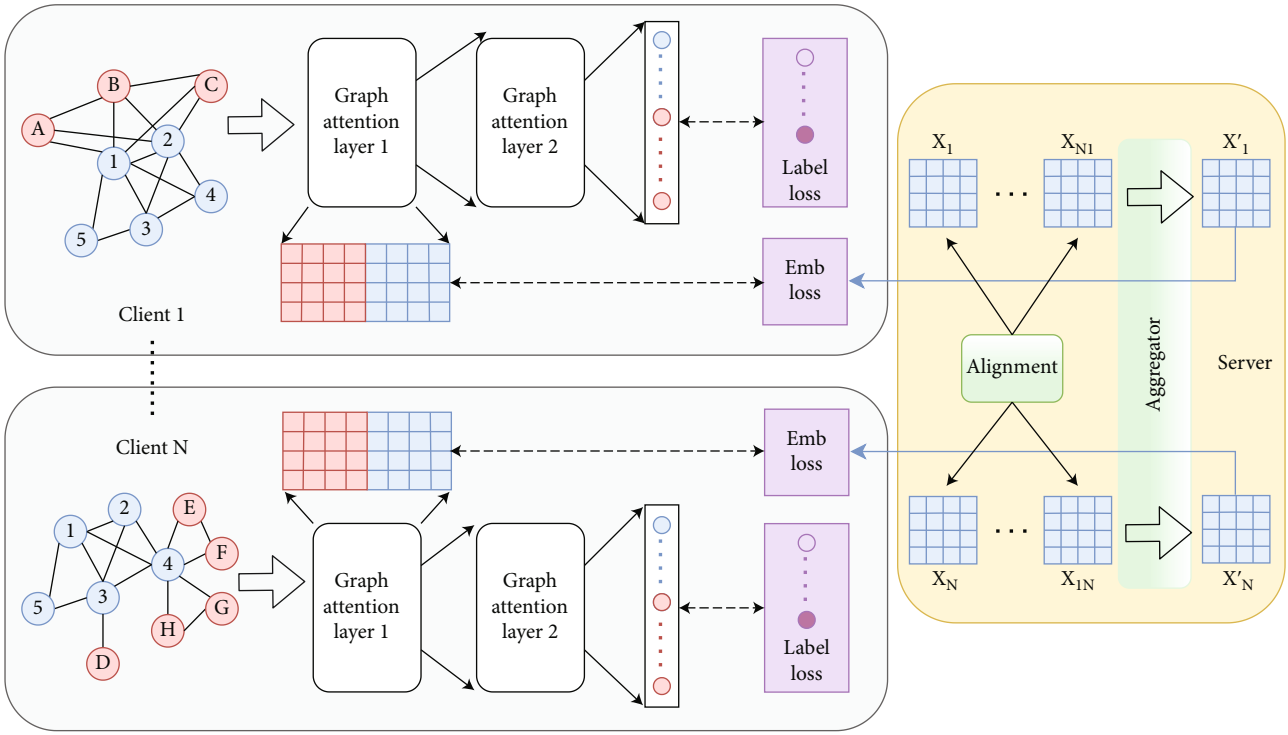

FIGURE 2: Overview of the federated GAT framework. The red nodes are private, and the blue nodes are public. Local training is highlighted in grey, and server aggregation is highlighted in yellow.

Since the latent representations of public nodes are generated from different training graphs, simple aggregation and distribution will break their connections with the unprocessed latent representations of private nodes on the local client. Thus, we apply an embedding alignment technique in the weight aggregation on the central server to convert the latent representations from other clients into a form that the local client understands. For example, there are two local clients c_x and c_y sharing k nodes in the graph. Let $X = \{\Phi_x(u_1), \cdots, \Phi_x(u_k)\}$ and $Y = \{\Phi_y(u_1), \cdots, \Phi_y(u_k)\}$, $u_k \in \mathbf{U}_k$ be two sets of k public node embeddings coming from c_x and c_y, respectively. For c_y to understand the information of X, we need to align/translate X into the space of c_y, which technically is using a linear mapping matrix W that maps X from the source space c_x to the target space c_y. Furthermore, we can encapsulate the problem to the Procrustes problem [31] and solve it via the singular value decomposition (SVD) of YX^T:

$$W^* = \operatorname{argmin}_{W \in M_d(\mathbb{R})} \|WX - Y\|_F = UV^T, \quad \text{with } U\sum V^T = \operatorname{SVD}(YX^T) \tag{4}$$

where $M_d(\mathrm{R})$ is the $d \times d$ matrix space of real numbers. We denote $X_y = WX$ as the aligned embeddings from source space c_x to target space c_y, and Y_x in the opposite way. The server aggregates X_y and Y to obtain a merged weight Y' and returns Y' to c_y for substituting the current public node embedding vector $\Phi(y_k)$. For multiple clients $\mathbf{C} = \{c_1, c_2, \cdots c_n\}$, the server aligns the embed dings from any pair of clients $\forall c_i, c_j \in \mathbf{C}$ and applies the average aggregation on all the aligned embeddings in the same client's space to get the returning updates for each client. The local clients use the updates as the initial weights to train in a new round. Algorithm 1 summarizes the complete training procedure.

3.3. Federated GAT Framework. GAT [32] introduces an attention mechanism to replace the statically normalized convolution operation in GCN [33]. The input to a single attentional layer is a set of node features, $h = \{\vec{h_1}, \vec{h_2}, \cdots, \vec{h_n}\}$, $\vec{h_i} \in F$, where n is the number of nodes and F is the node feature set. A linear transformation is firstly applied to every node feature for higher-level expression:

$$z_i^{(l)} = W^{(l)} h_i^{(l)}, \tag{5}$$

where $W^{(l)}$ is a learn-able weight matrix.

Different from the dot product attention mechanism in GCN, GAT applies the additive attention mechanism, which concatenates the z embeddings of two neighbors i and j to compute a pairwise unnormalized attention score $e^{(l)}$ between them. The additive attention mechanism takes the dot product of the concatenation and a weight vector $\vec{a}$ and then applies a LeakyReLU activation function. In order to compare the attention scores with different nodes, a

Input: $\mathbf{C} = \{c_1, c_2, \cdots c_n\}$: the set of clients
G_i: the local subgraph hold by c_i
$\mathbf{U}_k$: the public nodes shared among **C**
Output: the node embeddings H_i' of G_i
1: LOCAL CLIENTS:
2: **for** each client $c_i \in \mathbf{C}$ **do**
3: Compute the GAT model embedding H_i'
4: Generate the public nodes' embeddings X_i of U_k from the intermediate H_i:
5: $X_i = \{H_i(u_1), \cdots, H_i(uk)\}$
6: Upload X_i to the server
7: **end for**
8:
9: **while** not converge **do**
10: SERVER:
11: **for** each $i \in k$ **do**
12: **for** each $j \in k(i \neq j)$ **do**
13: Align X_j into c_i's space: $X_{ji} = W_{ji}X_j$
14: **end for**
15: Aggregate all the aligned embeddings with X_i
16: $X_i' = 1/k(\sum_j^k X_{ji} + X_i)$
17: distribute X_i' to client c_i for local update
18: **end for**
19:
20: LOCAL CLIENTS:
21: **for** each client $c_i \in \mathbf{C}$ **do**
22: Take X_i' as new input weights
23: Compute the GAT-model embedding H_i' with loss L_{new}
24: **end for**
25: **end while**
26: **return** the node embeddings H_i' of G_i

Algorithm 2: The federated GAT framework.

Table 1: The results of the federated DeepWalk framework. LOC indicates the result of local training, FED indicates the result of federated learning, and GLOBAL indicates the cumulative improvement of all clients.

Dataset (classifier)	Client 1		Client 2		Client 3		Client 4		GLOBAL
	LOC	FED	LOC	FED	LOC	FED	LOC	FED	
Cora (MLP)	79.1	80.1	76.0	76.0	74.3	76.1	75.0	76.6	+4.41
Cora (SVC)	79.3	81.4	77.5	78.7	75.5	77.9	75.0	78.0	+8.72
CiteSeer (MLP)	48.1	51.4	46.4	50.4	48.9	51.6	49.2	51.7	+12.5
CiteSeer (SVC)	56.3	60.6	53.5	58.6	57.3	61.6	55.3	60.4	+18.8
Cora (full)	MLP		79.9		SVC		82.0		
CiteSeer (full)	MLP		58.6		SVC		65.4		

normalized coefficient $\alpha^{(l)}$ is computed by the softmax function in the end:

$$\alpha_{ij}^{(l)} = softmax_j\left(e_{ij}^{(l)}\right) = \frac{\exp\left(\text{LeakyReLU}\left(\vec{a}^T\left[z_i^{(l)}\middle\|z_j^{(l)}\right]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}\left(\vec{a}^T\left[z_i^{(l)}\middle\|z_k^{(l)}\right]\right)\right)} \quad (6)$$

where N_i is some neighbor of node i in the graph, $\|$ denotes the concatenation operation, and T represents transposition.

Having the normalized attention coefficients calculated, GAT generates the next-level embed-scaled by the attention coefficients.

$$h_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}_i}\alpha_{ij}^{(l)}z_j^{(l)}\right) \quad (7)$$

Once we obtain the local embeddings, we have to face the similar challenge of collaborating with different clients in federated learning as the federated DeepWalk framework. Although in DeepWalk model we can extract the

Table 2: The results of the federated GAT framework. Cora_noise is the Cora dataset with noisy labels.

Dataset (classifier)	Client 1		Client 2		Client 3		Client 4		GLOBAL
	LOC	FED	LOC	FED	LOC	FED	LOC	FED	
Cora (MLP)	84.4	86.3	85.7	85.8	83.1	83.8	85.7	86.0	+3.01
Cora (SVC)	86.1	86.7	86.5	86.0	85.6	85.0	85.4	86.0	+0.1
Cora_noise (MLP)	81.4	81.0	79.0	79.5	72.2	75.9	70.6	77.3	+10.4
Cora_noise (SVC)	82.0	82.4	80.0	80.8	74.5	77.4	72.3	78.3	+10.1
CiteSeer (MLP)	72.3	74.7	72.8	74.1	72.8	74.3	72.7	75.0	+7.51
CiteSeer (SVC)	72.8	74.5	72.3	74.3	72.1	74.3	72.1	74.1	+7.82
Cora (full)	MLP		86.2		SVC		86.1		
CiteSeer (full)	MLP		74.7		SVC		74.8		

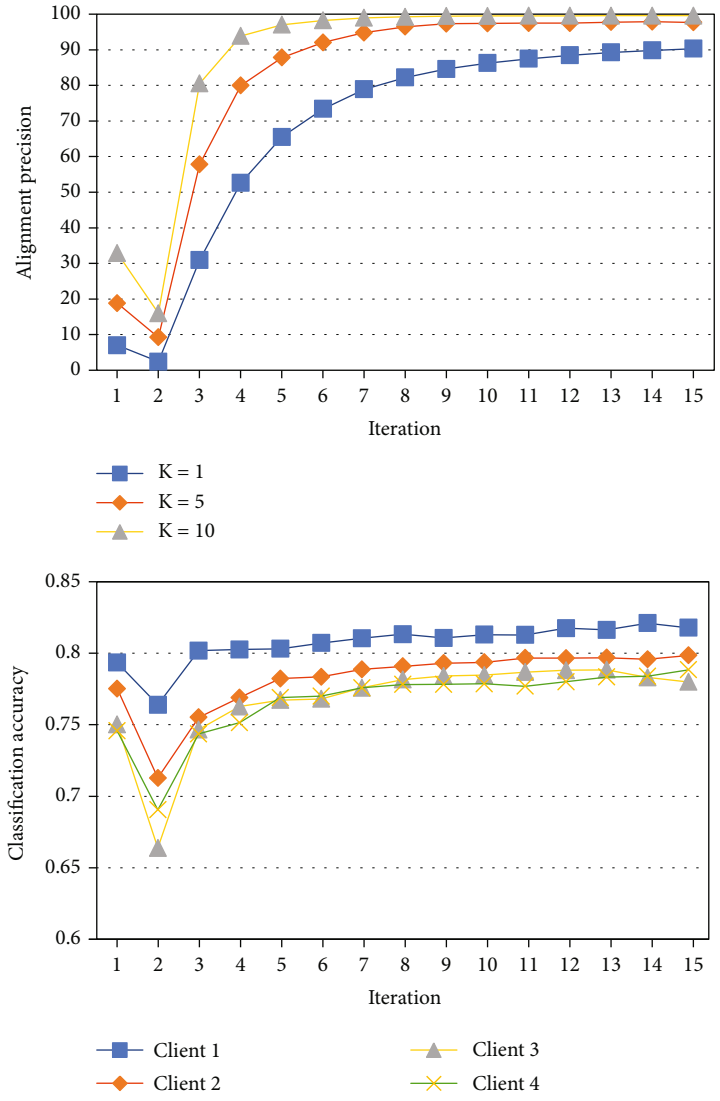

Figure 3: The KNN alignment precision and SVC classification accuracy corresponding to each iteration of the federated DeepWalk framework on the Cora dataset.

public nodes' embeddings from the model weights directly, the weights of GAT integrate both public and private information and cannot be split directly by node's category. Therefore, as shown in Figure 2, we upload the public nodes' embeddings X coming from the model's intermediate layer (i.e., $h_i^{(l)}, i \in \mathbf{U}_k$) to the server without exposing the model weights. Then, the server executes the same processes in the federated DeepWalk framework to align, aggregate, and

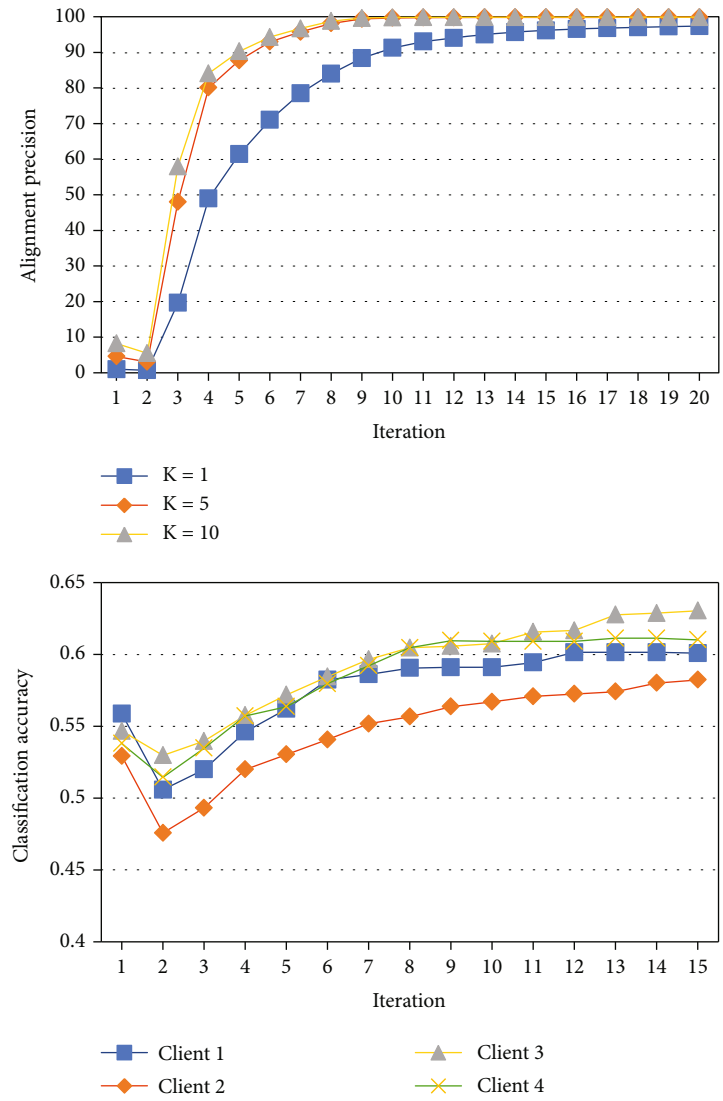

Figure 4: The KNN alignment precision and SVC classification accuracy corresponding to each iteration of the federated DeepWalk framework on the CiteSeer dataset.

distribute the updates X' to clients. As we cannot use the aligned embedding to manipulate GAT's model weights directly, another cosine-embedding loss L_{emb} is added beside the original cross-entropy loss L_{label} to integrate the information of X' back into the model.

$$L_{emb} = 1 - \cos\left(X, X'\right)$$
$$L_{label} = -\frac{1}{N}\sum_{i=1}^{n} y_i \log\left(\hat{y}_i\right) \tag{8}$$

where y_i is the one-hot label of each node and $\hat{y}_i = \text{soft max}\left(h_i^{(l+1)}\right)$. Thus, the new loss of local training is the combination of the cosine-embedding loss and the cross-entropy loss:

$$L_{\text{new}} = L_{\text{label}} + \beta L_{\text{emb}}, \tag{9}$$

where β is a model hyperparameter to balance the local information preservation and the external information integration. During the experiment, we observe that the last attention layer is so powerful that it overwhelms the cosine-embedding loss of the final output $h^{(l+1)}$. Hence, based on two-stage CNN training [34] and federated split learning [35], we inject the external information via the intermediate layer $h^{(l)}$ at an earlier stage. Algorithm 2 summarizes the complete procedure of the federated GAT framework.

4. Experiments

In this section, we present the experiments developed by PyTorch and conducted on a workstation with an Intel Core i7 2.80 GHz CPU and a NVIDIA GeForce GTX 1070 GPU.

4.1. Datasets. In our experiments, we employ two datasets, Cora [36] and CiteSeer [37], which are commonly used in the GNN research. The Cora dataset includes 2,708

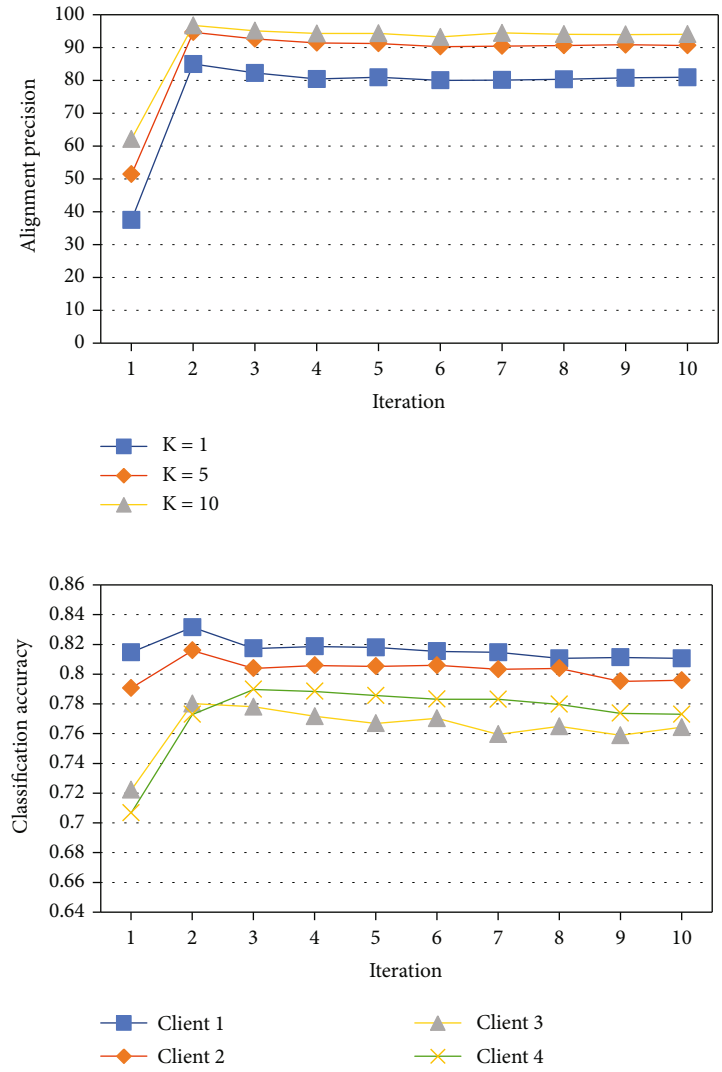

FIGURE 5: The KNN alignment precision and SVC classification accuracy corresponding to each iteration of the federated GAT framework on the Cora_noise dataset.

publications as nodes classified into seven classes, and its citation network consists of 5,429 links. The CiteSeer dataset includes 3,327 nodes classified into six classes, and its citation network consists of 4,732 links. Both datasets use unique words in each document as the node features. We set up four clients participating in the federated learning, so each dataset is split into four subgraphs with an equal number of nodes and assigned to each client. By default, each Cora subgraph has 1,489 (55%) nodes in total with 1,083 (40%) public nodes and 406 (15%) private nodes, while each CiteSeer subgraph has 1,829 (55%) nodes in total with 1,330 (40%) public nodes and 499 (15%) private nodes. (%) shows the percentage of the nodes in the original graph.

4.2. Baselines and Metrics. We use the client's local training as the baseline to verify whether our framework can improve each client's graph embedding result through collaborative training. In the DeepWalk-based training, all clients use the same model architecture with randomly initialized weights for local training or federated training, while in the GAT-based training, clients use the original GAT model for the local training and the modified GAT model with embedding loss for the federated training. Other architecture and parameters are fixed in a controlled experiment.

For all the implementations, we embed each graph into a 16-dimensional space and run the experiments on the classification tasks to evaluate the quality of the embedding by applying one multilayer perceptron (MLP) classifier and another support vector classifier (SVC) implemented in the Python module scikit-learn to predict the label of a node. For all the experiments, we use 5-fold cross-validation to ensure models' reliability and effectiveness, and the classification results of the two frameworks are given as micro F1-scores.

4.3. Performance Evaluation. The federated DeepWalk framework results are presented in Table 1. We can observe that in contrast to the embeddings generated by limited local information, both classifiers can achieve higher classification

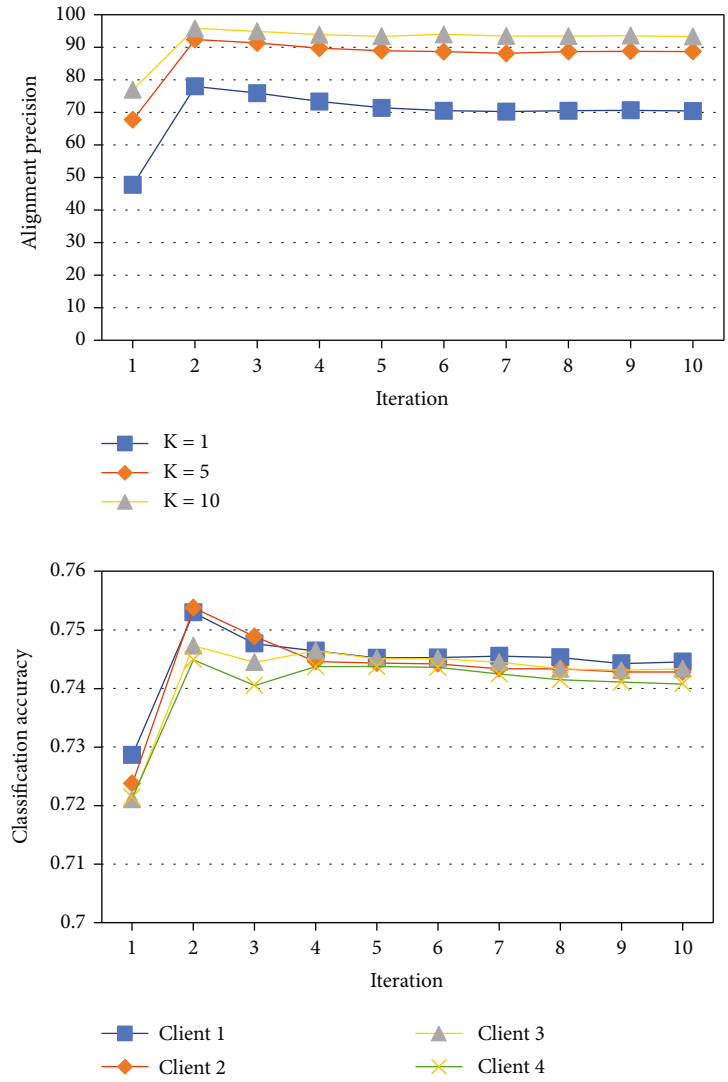

Figure 6: The KNN alignment precision and SVC classification accuracy corresponding to each iteration of the federated GAT framework on the CiteSeer dataset.

accuracy on the embeddings trained by full use of the graph data. Under this precondition, the proposed FL methodology can improve the global results by 4.41% (MLP) and 8.72% (SVC) on the Cora dataset, while 12.5% (MLP) and 18.8% (SVC) on the CiteSeer dataset, respectively. Specifically, every client in the CiteSeer experiment receives steady improvement compared with the local baseline.

For the federated GAT framework, we set the hyperparameter $\beta = 1$ for the Cora dataset and $\beta = 0.75$ for the CiteSeer dataset. As shown in Table 2, we obtain similar results on the CiteSeer dataset with 7.51% (MLP) and 7.82% (SVC) accuracy improvements. Because GAT uses weighting neighbor features with feature-dependent and structure-free normalization, which does not rely on knowing the entire graph structure in advance, the local client can generate favorable embedding by partial information of a denser Cora dataset. Thus, our method is subject to further refining the embedding in this case. In addition to cleaning the Cora dataset, we randomly modify 15% labels as noisy (incorrect) labels during the training, leading to a considerable performance loss for clients such as client 3 and client 4. However, our proposed method can effectively mitigate the influence of noisy labels by integrating information from other clients. Consequently, the poor performance of client 3 or client 4 receives a significant improvement.

4.4. Impact of Alignment on Performance. To demonstrate the effectiveness of applying alignment during the FL aggregation, we plot the alignment precision of the public latent representations and the SVC classification accuracy of the graph embeddings corresponding to each training iteration of both frameworks in Figures 3–6. We use k-nearest neighbors with $k = 1$, 5, and 10 to measure the alignment precision between any pair of the public latent representations. Because we need to align each local representation to the dimension of the other clients, there are 12 pairs in the four clients' settings, and we only show the average value of 12 alignments in the figures as the variance is slight.

Table 3: Results of the different shared public nodes.

Percent	DeepWalk			GAT		
	LOC	FED	Diff	LOC	FED	Diff
5%	57.1	61.8	+4.7	74.4	75.1	+0.7
10%	57.3	62.3	+5.0	73.9	75.1	+1.2
20%	61.8	65.0	+3.2	71.8	73.5	+1.9
30%	63.4	65.8	+2.4	70.5	72.6	+2.1
40%	67.8	69.2	+1.4	70.5	73.0	+2.5

For the federated DeepWalk framework in Figures 3 and 4, although the classification accuracy of locally trained graph embedding is acceptable in the initial iteration, their alignment results are inferior because of the random initialization. Consequently, we cannot integrate the information of different clients effectively, which leads to the performance diving in the second iteration. However, the rough integration in the first two iterations helps in the united initialization by setting the tone for the subsequent training. Thus, we observe that the quality of graph embedding improves with the promotion of the alignment effect, which can achieve above 90% precision of $k = 1$ at the convergence stage. For the federated GAT framework in Figures 5 and 6, the initial representation alignment results are satisfactory with a fair classification accuracy of graph embedding. Moreover, we observe both alignment precision and classification accuracy surge in the second iteration after the federated learning process. Nevertheless, as we only use cosine-embedding loss at the intermediate layer, partial integrated information is squeezed out when the federated procedure converges within ten iterations. In general, there is a positive correlation between the alignment precision and classification accuracy, which confirms the effectiveness of our method.

5. Discussion

Through previous experiments, we find that our method performs better when applied to the CiteSeer dataset, which is more sparse than the Cora dataset relatively. Denser subgraphs mean the local clients have more information, limiting the improvement effect of federated learning. However, if the degrees of the shared nodes are low, they cannot comprehensively transmit the local information during the integration. Therefore, we design the supplemental experiments to further study the suitable application scenarios. Instead of randomly generating the subgraphs and selecting 40% public nodes to share, we compose the subgraphs with different percentages of top high-degree nodes from the original graph as the public nodes. We conduct the same embedding classification experiment and render the average accuracy of four clients in Table 3.

Under the DeepWalk framework, the classification accuracy of locally generated graph embeddings increases as the degree of nodes in the subgraph increases. Although federated learning can still improve the overall classification effect, the magnitude of improvement diminishes. With a simpler model DeepWalk, the local clients are more likely to get an underfit model with inferior prediction accuracy below 70%. Federated learning tackles the underfitting issue more by sharing public information between clients and indirectly increasing the local training dataset's size. In the experimental group of GAT, we notice that the higher subgraph density reduces the accuracy of local graph embedding. One reason is that the subgraphs generated by our method are disassortative, and the local aggregation mechanism of GAT may fail on disassortative graphs, where nodes within local neighborhoods provide more noise than helpful feature information. Another reason is that the local model is overfitting the denser training subgraph. However, the federated learning setting prevents the local model from focusing on the training data, and the embedding alignment technique has regularization effects empirically to avoid overfitting. Overall, our approach is suitable for general application scenarios, and the improvement effect is more prominent when the local embedding effect is unsatisfied.

6. Conclusions

This paper investigates a practical problem of federated graph neural networks with non-IID datasets and proposes a novel federated learning framework. Through embedding alignment, we can normalize the common latent representation of each client as uniformly as possible and enable information integration in a federated setting. The experimental results demonstrate that our framework can achieve higher data usability than local training with privacy preservation. Other advanced embedding alignment technologies can be explored in future work for more accurate information integration. Investigation of the shared public nodes is still worthwhile. For future expansion, discovering an optional composition of public nodes to reduce the number of shares can better balance privacy protection and data availability.

Data Availability

The Cora dataset consists of 2,708 scientific publications classified into one of seven classes. The citation network consists of 5,429 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 1,433 unique words. (original source: http://web.archive.org). The CiteSeer dataset consists of 3,312 scientific publications classified into one of six classes. The citation network consists of 4,732 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 3,703 unique words (introduced by C. Lee Giles et al. in CiteSeer: An Automatic Citation Indexing System).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean Government under grant 20ZR1300 (Core Technology Research on Trust Data Connectome). This work was also supported by the National Science Foundation (No. 1704287).

References

[1] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: strategies for improving communication efficiency," 2016, https://arxiv.org/abs/1610.05492.

[2] 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, 2021.

[3] W. Zhang, J. C. Weiss, S. Zhou, and T. Walsh, "Fairness amidst non-IID graph data: a literature review," 2022, https://arxiv.org/abs/2202.07170.

[4] K. Zhou, Y. Dong, W. Lee, B. Hooi, H. Xu, and J. Feng, "Effective training strategies for deep graph neural networks," 2020, https://arxiv.org/abs/2006.07107.

[5] Y. Ye and S. Ji, "Sparse graph attention networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 905–916, 2021.

[6] P. Kairouz, H. B. McMahan, B. Avent et al., "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1-2, pp. 1–210, 2021.

[7] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," 2019, https://arxiv.org/abs/1907.02189.

[8] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, https://arxiv.org/abs/1806.00582.

[9] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-IID data with reinforcement learning," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pp. 1698–1707, Toronto, ON, Canada, 2020.

[10] 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.

[11] X. Zhang, M. Hong, S. Dhople, W. Yin, and Y. Liu, "FedPD: a federated learning framework with optimal rates and adaptivity to non-IID data," 2020, https://arxiv.org/abs/2005.11418.

[12] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "FedBN: federated learning on non-IID features via local batch normalization," 2021, https://arxiv.org/abs/2102.07623.

[13] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-IID data silos: an experimental study," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pp. 965–978, Kuala Lumpur, Malaysia, 2022.

[14] Z. Xiong, Z. Cai, D. Takabi, and W. Li, "Privacy threat and defense for federated learning with non-i.i.d. data in AIoT," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, 2022.

[15] G. Mei, Z. Guo, S. Liu, and L. Pan, "SGNN: a graph neural network based federated learning approach by hiding structure," in *2019 IEEE International Conference on Big Data (Big Data)*, pp. 2560–2568, Los Angeles, CA, USA, 2019.

[16] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar, "Peer-to-peer federated learning on graphs," 2019, https://arxiv.org/abs/1901.11173.

[17] C. He, K. Balasubramanian, E. Ceyani et al., "FedGraphNN: a federated learning system and benchmark for graph neural networks," 2021, https://arxiv.org/abs/2104.07145.

[18] J. Pang, Y. Huang, Z. Xie, Q. Han, and Z. Cai, "Realizing the heterogeneity: a self-organized federated learning framework for IoT," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3088–3098, 2021.

[19] C. He, E. Ceyani, K. Balasubramanian, M. Annavaram, and S. Avestimehr, "SpreadGNN: serverless multi-task federated learning for graph neural networks," 2021, https://arxiv.org/abs/2106.02743.

[20] S. Sajadmanesh and D. Gatica-Perez, "Locally private graph neural networks," in *CCS '21: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2130–2145, Seoul, South Korea, 2021.

[21] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie, "FedGNN: federated graph neural network for privacy-preserving recommendation," 2021, https://arxiv.org/abs/2102.04925.

[22] K. Li, G. Lu, G. Luo, and Z. Cai, "Seed-Free Graph De-Anonymiztiation with Adversarial Learning," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM '20)*, pp. 745–754, New York, NY, USA, 2020.

[23] K. Li, G. Lu, G. Luo, and Z. Cai, "Adversarial privacy-preserving graph embedding against inference attack," *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 8, pp. 6904–6915, 2021.

[24] S. Ruder, I. Vulic, and A. Søgaard, "A survey of cross-lingual word embedding models," *Journal of Artificial Intelligence Research*, vol. 65, pp. 569–631, 2019.

[25] A. Conneau, G. Lample, M.'. A. Ranzato, L. Denoyer, and H. Jégou, "Word translation without parallel data," 2018, https://arxiv.org/abs/1710.04087.

[26] G. Dinu, A. Lazaridou, and M. Baroni, "Improvng zero-shot learning by mitigating the hubness problem," 2014, https://arxiv.org/abs/1412.6568.

[27] Z. Sun, W. Hu, Q. Zhang, and Y. Qu, "Bootstrapping entity alignment with knowledge graph embedding," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)*, pp. 4396–4402, Stockholm, Sweden, 2018.

[28] Q. Zhang, Z. Sun, W. Hu, M. Chen, L. Guo, and Y. Qu, "Multi-view knowledge graph embedding for entity alignment," 2019, https://arxiv.org/abs/1906.02390.

[29] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710, New York, USA, 2014.

[30] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, https://arxiv.org/abs/1301.3781.

[31] C. Wang and S. Mahadevan, "Manifold alignment using Procrustes analysis," in *Proceedings of the 25th international conference on Machine learning - ICML '08*, pp. 1120–1127, Madison, Wisconsin, USA, 2008.

[32] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph Attention Networks," in *International Conference on Learning Representations*, Vancouver, BC, Canada, February 2018.

[33] M. Welling and T. N. Kipf, "Semisupervised classification with graph convolutional networks," in *J. International Conference on Learning Representations (ICLR 2017)*, Toulon, France, 2016.

[34] J. Pang, W. Sun, J. S. Ren, C. Yang, and Q. Yan, "Cascade residual learning: a two-stage convolutional neural network for stereo matching," in *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 887–895, Venice, Italy, 2017.

[35] C. Thapa, P. C. M. Arachchige, S. Camtepe, and L. Sun, "Splitfed: when federated learning meets split learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, pp. 8485–8493, 2022.

[36] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. EliassiRad, "Collective classification in network data," *AI Magazine*, vol. 29, no. 3, pp. 93–93, 2008.

[37] 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 - DL '98*, pp. 89–98, Pittsburgh, Pennsylvania, USA, 1998.