J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

FedGCD: Federated Learning Algorithm with GNN based Community Detection for Heterogeneous Data☆

Wooseok Shin Jitae Shin*

ABSTRACT

Federated learning (FL) is a ground breaking machine learning paradigm that allow smultiple participants to collaboratively train models in a cloud environment, all while maintaining the privacy of their raw data. This approach is in valuable in applications involving sensitive or geographically distributed data. However, one of the challenges in FL is dealing with heterogeneous and non-independent and identically distributed (non-IID) data across participants, which can result in suboptimal model performance compared to traditionalmachine learning methods. To tackle this, we introduce FedGCD, a novel FL algorithm that employs Graph Neural Network (GNN)-based community detection to enhance model convergence in federated settings. In our experiments, FedGCD consistently outperformed existing FL algorithms in various scenarios: for instance, in a non-IID environment, it achieved an accuracy of 0.9113, a precision of 0.8798,and an F1-Score of 0.8972. In a semi-IID setting, it demonstrated the highest accuracy at 0.9315 and an impressive F1-Score of 0.9312. We also introduce a new metric, nonIIDness, to quantitatively measure the degree of data heterogeneity. Our results indicate that FedGCD not only addresses the challenges of data heterogeneity and non-IIDness but also sets new benchmarks for FL algorithms. The community detection approach adopted in FedGCD has broader implications, suggesting that it could be adapted for other distributed machine learning scenarios, thereby improving model performance and convergence across a range of applications.

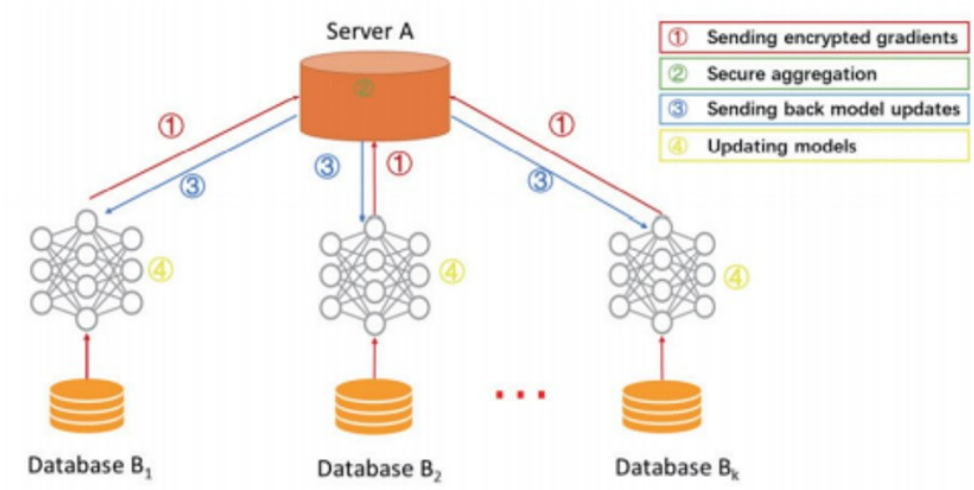
☞ keyword : Federated Learning, Non-IID, data heterogeneity, Community Detection, Graph Neural Networks

1. Introduction

Federated Learning (FL)[1] has emerged as a powerful machine learning technology in cloud computing environments, where servers and clients coexist. It allows multiple participants to jointly train models without sharing raw data, as shown in Fig. 1. This is particularly useful in scenarios where data is sensitive or distributed, and data sharing is not feasible or desirable. However, FL faces several challenges [2], such as ensuring model convergence and quality in the presence of heterogeneous and non-IID (non-independent and identically distributed) data across participants. As depicted in Fig.2, if one client possesses 100GB of food photos while the other client has only 10GB of person photos, it is not possible to ensure satisfactory outcome if both clients are trained using the same weight. In FL settings, data is distributed across multiple devices or nodes, and each node can have a slightly different data distribution due to various factors such as user behavior, device type, and location. In addition, data between nodes can be non-IID, which means that the distribution of data can vary significantly among nodes. In this case, it often falls short of the performance of data-centric learning used in machine learning in cloud computing. To address these challenges, several FL algorithms have been proposed in recent years. Many of these algorithms utilize clustering-based techniques for model aggregation and communication. However, recent research trends [3] show that few studies have applied graph neural networks (GNNs), which are frequently used for clustering, to federated learning. GNNs are effective in modeling complex relationships that can be used to form communities. Thus, clustering based on GNN-based community detection can be particularly useful in real-world FL scenarios.

In this paper, we propose FedGCD, a novel FL algorithm that integrates GNN-based community detection to enhance

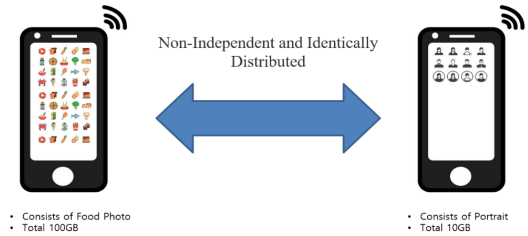
1 Dept. Of Electrical and Computer Eng, Sungkyunkwan University, Suwon, 16419, Korea.
* Corresponding author (jtshin@skku.edu)
☆ This work was supported by the BK21 FOUR Project.
[Received 15 June 2023, Reviewed 4 July 2023 (R2 8 August 2023, R3 22 September 2023), Accepted 18 October 2023]



(Fig. 1) Schematic Diagram of FL

model convergence and quality in federated settings. The mainidea of FedGCD is to identify participant clusters with similar data distributions and train a separate GNN model for each cluster. We then combine the GNN models to produce a more robust and accurate final model than previous FL algorithms.

To identify participant clusters, FedGCD utilizes a community detection approach based on graph clustering. The nodes in the graph represent participants, and the edges represent similarities between the data distributions of participants. FedGCD uses clustering algorithms to group nodes into communities based on similarity and then trains a separate GNN model for each community. If there is an overlapping community [4], the client defines an indicator called a membership score and uses a weighted average based on the membership score. We also create a new non-IID indicator that can be easily measured using the distribution of the norm of the weight tensor of the learning results for efficient experiments. This indicator reveals a positive correlation with the level of heterogeneity present in a federated learning-only dataset. Furthermore, it demonstrates that this indicator can be measured accurately and simply in an actual dataset. We also evaluate FedGCD by varying the degree of data heterogeneity in the Synthetic Dataset [5] and demonstrate that we outperform state-of-the-art FL algorithms [6] in terms of model accuracy. Furthermore, extensive experiments are conducted to demonstrate the effectiveness and heterogeneity of community detection approaches, as well as the robustness of the algorithm on non-IID data.



(Fig. 2) Non-IID Data Environment

Overall, our results suggest that FedGCD has the potential to significantly advance the field of federated learning and enable a wide range of applications in sensitive and distributed domains. The proposed method provides an effective solution to the heterogeneity and non-IID data problem of FL, and the community detection approach has the potential to be applied to other distributed machine learning scenarios.

The average accuracy increased by 1.2% to 2.1% in all situations compared to FedProx, which was the most widely

used method. Additionally, the number of communities was not significantly different from the number of clusters in the clustering-based method, which is the most similar method to this study. Recall, precision, and F1 scores were measured as 0.9717, 0.7128, and 0.8224, respectively. AUC and ROC analysis was also performed, and the value of AUC was 0.7606. The proposed GNN-based community detection algorithm, based on the FL method, can provide valuable insights into the structure of the data and the relationships between clients, leading to a more accurate and efficient FL model.

Section 2 provides related research and insights on federated learning and GNN-based community detection algorithms. Section 3 describes the detailed process of the proposed method, including the community detection method used and the approach for determining the optimal number of communities. Section 4 provides the experimental results, and Section 5 provides a review and conclusion of the experimental results.

2. Related Works

The related works can be categorized into five main groups: federated learning algorithms, graph-based methods for federated learning, clustering-based techniques for federated learning, community detection in graphs, and GNNs for clustering. The federated learning algorithms category includes some of the most commonly used algorithms in FL, while the graph-based methods and clustering-based techniques categories highlight the use of graph-based and clustering-based techniques in FL. The community detection in graphs and GNNs for clustering categories specifically focus on the use of these techniques in the context of clustering and community detection in graphs. Overall, these related works provide a comprehensive overview of the state-of-the-art in FL, graph-based methods, clustering-based techniques, and community detection in graphs, which helps to motivate and contextualize theproposed FedGCD algorithm

2.1 Federated Learning Algorithms with non-IID situations

FedAvg is one ofthe earliest and most widely used FL algorithms. The key idea behind FedAvg isto perform model updates by averaging the weights of the local models trained on each participant's data instead of sharing the raw data. The algorithm consists of several rounds, where in each round, the participants train their local models on their respective data, and the weights are then averaged to obtain a global model. FedAvg employs a weighted averaging scheme, where each participant's contribution to the global model is proportional to its data size. FedAvg has been shown to achieve high model accuracy while maintaining privacy and security.

FedMA [7] is an FL algorithm designed to handle multi-attribute data, where participants have different subsets of features and labels. FedMA uses a meta-learning approach to learn ashared feature representation that is independent of the specific attributes ofeach participant. The algorithm consists of two phases: a meta-learning phase, where a shared feature representation is learned using a small subset of participants, and a FL phase, where the learned representation is used to train a global model on the remaining participants' data. FedMA has been shown to achieve high model accuracy while maintaining privacy and robustness in FL scenarios with multi-attribute data.

FedProx [8] is a FL algorithm that aims to improve the robustness of FL in the presence of non-IID data and slow or unreliable participants. The main idea behind FedProx is to introduce a proximal term into the objective function to penalize the deviation of the local models from the global model. The proximal term is designed to encourage the local models to stay close to the global model and to promote convergence, even in the presence of non-IID data. FedProx also introduces a weighting scheme to give more importance to participants with higher data quality and reliability. The algorithm has been shown to improve the convergence rate and reduce communication overhead in FL scenarios.

FedVar [9] is a FL algorithm that addresses the challenges of FL using weight variation in clients. The algorithm assigns different weights to each participant based on their data quality and reliability, and these weights are

updated dynamically during the training process. The weight variation encourages more reliable participants to contribute more to the global model while penalizing unreliable or malicious participants.

We compared the accuracy of FedGCD with the methods in this section. Compared toFedProx, which had the highest average among the four methods, FedGCD showed a performance increase effect of at least 1.2% and at most 2.1%.

2.2 Graph-Based Methods for Federated Learning

FedGraphNN [10] is a FL algorithm designed for graph-structured data. The algorithm leverages GNNs to model the complex relationships and dependencies between data points in distributed settings. FedGraphNN introduces a novel federated optimization framework that allows each participant to update its local model using only its own data and a subset of the global model. The algorithm employs a graph aggregation mechanism to combine the local models and update the global model.

FedGCN [11] is an FL algorithm designed for graph-structured data. The algorithm employs Graph Convolutional Networks (GCNs) to model the graph structure and learn a shared feature representation that is independent of the specific attributes of each participant. FedGCN introduces a federated learning framework that allows each participant to update its local model using only its own data and a subset of the global model. The algorithm employs a weighted averaging scheme to combine the local models and update the global model.

FedGN [12] is an FL algorithm designed for graph-structured data.The algorithm employs GNNs to model the complex relationships and dependencies between data points in distributed settings. FedGN introduces a novel communication efficient approach that allows each participant to communicate only the updated gradients of its local model to the aggregator. The algorithm employs a message passing scheme to update the global model and ensure that the graph structure is preserved across participants.

2.3 Clustering-Based Techniques for Federated Learning

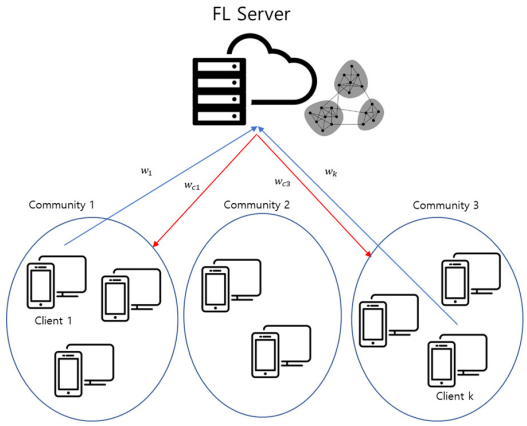
“Clustered Federated Learning” [13] leverages clustering techniques to improve the performance of FL in the presence of heterogeneity and non-IID data across participants. The algorithm first clusters the participants based on their data characteristics and trains a separate model for each cluster. The algorithm then combines the cluster models to obtain the global model. ″Clustered Distributed Federated Clustering with Federated Learning″ [14] combines federated learning and clustering techniques to improve the performance of FL in the presence of heterogeneity and non-IID data across participants. The algorithm first clusters the participants based on their data characteristics and trains a separate model for each cluster. The algorithm then combines the cluster models to obtain the global model. The algorithm also employs a distributed clustering approach that allows each participant to cluster its own data without sharing raw data.

FedDivide [15] is an FL algorithm that employs a divide-and-conquer approach to improve the performance of FL in the presence of heterogeneity and non-IID data across participants. The algorithm divides the participants into smaller groups based on their data characteristics and trains a separate model for each group. The algorithm then combines the group models to obtain the global model.

2.4 Community Detection in Graphs

“Community Detection in Networks: A User Guide” [16] is a comprehensive review of the field of community detection in networks. The paper provides an overview of the main approaches to community detection, including modularity-based methods, hierarchical clustering, and spectral methods. The paper also discusses the challenges and limitations of community detection and provides recommendations for future research.

“Community Detection in Graphs” [17] is a survey paper that provides an overview of the main methods for community detection in graphs. The paper covers traditional methods, such as modularity optimization, as well as more recent methods based on spectral clustering, probabilistic



(Fig. 3) Schematic Diagram of FedGCD

modeling, and deep learning. The paper also discusses the challenges and limitations of community detection and provides directions for future research.

3. Proposed Method

Fig. 3 is a schematic diagram of the method we propose. Our proposed method consists of the following steps to determine the weight assigned to each client in a federated learning environment, considering their participation in different communities:

3.1 Graph Constructions

We begin byconstructing graphs $G(V,E)$ to represent relationships between clients in a federated learning environment. A set of vertices V represents a client, and a set of edges E represents a relationship between clients based on data, communication patterns, or other related functions. The graph configuration consists of the distance between the end points of the tensors based on the result of the first round, and is configured as 0 if it deviates from a certain value.

3.2 GNN-Based Community Detection

We use a GNN model to detect the community of graphs $G(V,E)$. The GNN model enables effective community detection by learning node embeddings that capture the structural properties of graphs. Then, community detection is achieved using the M-NMF algorithm, which performs best in the most diverse of the overlapped community detection algorithms. This step identifies N communities and yields corresponding membership scores $s_{i,j}$ for each client a_i of the community c_j. Then the membership scores $s_{i,j}$ may be expressed as Eq. 1.

$$s_{i,j}=\frac{\sum_{x\in D_i} 1(x\in c_j)}{|\ D_i\ |} \qquad (1)$$

The sum of the dataset size included in each community to the total dataset size is the member score, which plays an important role in calculating the participation in communities with different weights for each client. The important thing is that only the size of dataset is sent to server, not the whole dataset.

3.3 Find Optimal Number of Communities

To determine the optimal number of communities, we employ the Akaike Information Criterion (AIC)[18]. The AIC is a widely used and well-established criterion for model selection. It is designed to find the balance between the goodness of fit of amodel and its complexity, avoiding overfitting or underfitting. In the context of community detection, it helps us find the number of communities that best represent the underlying structure of the data.

The reason forusing AIC to determine the optimal number of communities is that it allows us to identify a model that is neither too simple nor too complex. A model with too few communities may not capture the true structure of the data, leading to poor performance in federated learning. On the other hand, a model with too many communities may overfit the data, leading to poor generalization to newclients or data. AIC is computed as Eq. 2.

$$AIC(k)=2k-2\log L(k) \qquad (2)$$

Where k is the number of communities, and $L(k)$ is the maximum likelihood of the model for k communities. The

term $2k$ represents the penalty for model complexity, while the term $-2\log L(k)$ represents the goodness of fit. The objective is to find the value of k that minimizes the AIC value.

To find the optimal number of communities, we perform the following steps: For a range of possible values for k (e.g., $k=100$ to a predetermined maximum value), train the GNN-based community detection model and compute the maximum likelihood $L(k)$ for each k. And then Compute the AIC values for each k using the formula above. Finally, Choose the value of k that results in the lowest AIC value as the optimal number of communities.

To calculate AIC and the likelihood term ($L(k)$) with the FEMNIST dataset, we need to first fit a model to the data and then calculate the relevant components for AIC computation.

As the FEMNIST dataset consists of images of hand-written characters, we will use a simple logistic regression model for demonstration purposes.

Suppose we have a FEMNIST dataset with the following variables:

X: Features representing the images of handwritten characters.

Y: Labels representing the corresponding classes of the characters (e.g., digits 0-9).

We will start with a logistic regression model. We will fit a logistic regression model to predict the class labels Y based on the image features X.

Calculate the likelihood of the model using the maximum likelihood method like Eq. 3.

$$L(k) = \prod [P(Y_i \mid X_i)^{Y_i} \times (1 - P(Y_i \mid X_i)^{1-Y_i} \quad (3)$$

where $\prod$ denotes the product over all data points (i =1 to n), P(Yi | Xi) is the predicted probability of the correct class label Yi given the features Xi, and Yi is the actual class label (0 or 1).

Determine the number of parameters in the model, which is k = number of features + 1 (including the intercept term). Then, calculate the value of -2ln(L). We will calculate the log-likelihood and then multiply by -2 to get the -2ln(L) value. Next, compute the AIC value. Let's assume we have fitted the logistic regression model to the FEMNIST dataset and obtained the log-likelihood value as -1500. Further, suppose the model has 100 features (including the intercept term). We calculate the likelihood term (L) using the formula mentioned above based on the logistic regression model predictions and the actual class labels.

While repeating this process from 100 to 1, the k value with the smallest AIC value is determined as the number of communities. In this example, we have obtained an AIC value of -2798 for the logistic regression model fitted to the FEMNIST dataset. The lower the AIC value, the better the model's trade-off between goodness of fit and complexity. Therefore, the model with the lowest AIC value would be preferred for further analysis and inter-pretation.

By using AIC to determine the optimal number of communities, we ensure that our model strikes a balance between complexity and goodness offit. This approach helps to improve the performance of federated learning by capturing the true structure of client communities while avoiding overfitting or underfitting.

3.4 Weight Calculation

For each client a_i, we calculate the weight w_i by considering participation in different communities as Eq. 4.

$$w_i = \sum_{j=1}^{N} \frac{w_j \times s_{i,j}}{\sum_{k=1}^{N} s_{i,k}} \quad (4)$$

Here, w_i represents a weight allocated to the community c_j. Fractions within the sum normalize the membership scores of clients a_i in community c_j with respect to membership in all communities. The final weight w_i is the weighted sum of these normalized scores, where the weights of each community are considered. Such weighting calculations can explain the nature of overlapped communities because clients participating in multiple communities have a higher overall weight.

3.5 Federated Learning with Weighted Clients

We incorporate client weights by updating the federated

Algorithm 1 FedGCD : Federated Learning Algorithm with GNN based Community Detection

Input1 : Federated learning environment with M clients and N overlapping communities
Input2 : Local dataset D_i for each client a_i
Input3 : Community weights w_j for j=1,...,N

Output1 : Community assignment for each client
Output2 : Weight w_i for each client a_i

```
weights = [w_1, w_2, ..., w_N]
client_data = [D_1, D_2, ..., D_M]

for client ∈ clients do
    community = comm_detect(client_data[client])
    membership_scores=cal_scores(client_data[client], community, N)
    weight = cal_weights(membership_scores, weights)
    server.aggregate(weight)

procedure CAL_SCORES(data, community, N)
    membership_scores = zeros(data.size, N))
    for i = 1,2,...,len(data) do
        for j = 1,2,...,N do
            membership_scores[i, j] = sum(community[i] == j)
            membership_scores[i, j] = data.size
    return membership_scores

procedure CAL_WEIGHTS(membership_scores, weight)
    normalized_scores = []
    for i = 1,2,...,len(membership_scores) do
        score_sum = sum(membership_scores[i])
        normalized_score = [membership_scores[i]/score_sum]
        normalized_scores.append(normalized_score)
    weighted_scores = []
    for i = 1,2,...,len(normalized_scores) do
        weighted_score = sum([normalized_scores[i] * weights)
        weighted_scores.append(weighted_score)
    client_weight = sum(weighted_scores)
    return client_weight
```

(Fig. 4) Pseudo Code of FedGCD

learning process using calculated weights. During aggregation of client updates, the server uses the calculated weight w_i to combine local model updates for each client to update the global model. This weighted aggregation allows customers who participate more in multiple communities to contribute more to the global model and be influenced by the weights of different communities.

We define the above method as one round and repeat community redetection every 50 rounds. The pseudocode is as Fig. 4. In summary, our proposed method utilizes a GNN-based community detection algorithm to identify overlapping communities in a federated learning environment and compute client weights based on participation in these communities. The federated learning process is then updated to incorporate these weights, improving performance inthe presence of overlapping communities

4. Experimental Results

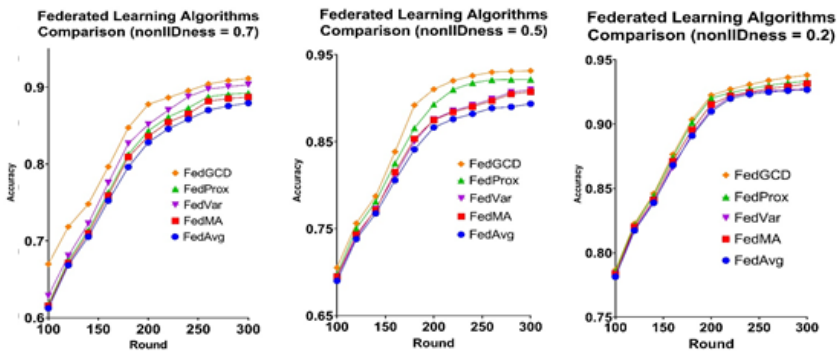
Prior to the experiment, we redefined values for data heterogeneity. The data heterogeneity values defined in Synthetic Dataset are too complex to approach. We graph the distribution of the weighted tensor norm values and redefined the heterogeneity value "nonIIDness" using the kurtosis of the corresponding graph. The order of calculation is as follows. First, the kurtosis of the weights is calculated. The equation of the kurtosis is the same as Eq. 5, and the range is -2 or more and positive infinity or less. w_i is the weight of client i, and w is the average of weights for all clients.

$$K = \frac{\sum (w_i - w)^4}{(\sum (w_i - w)^2)^2} - 3 \tag{5}$$

Eq. 5 is used to normalize these values to values between 0 and 1.

$$s = \frac{1}{e^K + 1} \tag{6}$$

According to Eq. 6 and the range of the kurtosis value, the value of nonIIDness is 0 to 0.88. In a typical federated learning environment, the number of clients is at least 100, so when the weights are most evenly distributed, the value of the kurtosis converges to -1.2, where the value of nonIIDness is about 0.77. Suppose this value is a non-IID environment. Therefore, when the value of nonIIDness is 0.77, it is non-IID, and as the value of nonIIDness decreases, it can be thought that it approaches IID Using Synthetic Dataset, we created a dataset by specifying various non-IID values and applied them to the nonIIDness we created to confirm the correlation, and the results are shown in the Fig. 5. As can be seen in the Fig. 5, the heterogeneity value of Synthetic Dataset and nonIIDness defined by us have a positive correlation. Then, we present the experimental results of our proposed method, FedGCD, and compare it with FedProx, a state-of-the-art federated learning algorithm and the other methods. We conducted experiments on various nonIIDness values to analyze the performance of our method with different levels of data heterogeneity.



(Fig. 5) Experimental Results

4.1 Experimental Setup

We evaluate our method on a FEMNIST dataset with varying degrees of nonIIDness. We set the nonIIDness to 0.7, 0.5, and 0.2 to represent non-IID, semi-IID, and IID data environments, respectively. For each setting, we compare the performance of FedGCD with that of FedProx and the other methods in terms of test accuracy and loss convergence.

The federated learning system consisted of 100 clients, and the experiments are conducted for 300 communication rounds. We use the same model architecture and training hyperparameters for all federated learning methods to ensure a fair comparison. The experimental results are shown in Fig. It is summarized in Fig. 5 and shows Accuracy, Precision, Recall, and F1-Score foreach scenario. Additionally, the best value for each result is written in bold.

(Table 1) Experimental Results

		FedAvg[1]	FedMA[7]	FedProx[8]	FedVar[9]	Ours
Non-IID	Accuracy	0.8793	0.8871	0.8924	0.9032	**0.9113**
	Precision	0.8749	0.8801	0.8902	0.9053	0.8798
	Recall	0.8797	0.8853	0.8949	0.9047	**0.9149**
	F1-Score	0.8773	0.8827	0.8926	**0.9050**	0.8972
Semi-IID	Accuracy	0.8936	0.9074	0.9216	0.9099	**0.9315**
	Precision	0.8901	0.9051	**0.9348**	0.9147	0.9321
	Recall	0.8946	0.9094	0.9180	**0.9403**	0.9300
	F1-Score	0.8923	0.9072	0.9260	0.9273	**0.9312**
IID	Accuracy	0.9268	0.9314	0.9335	0.9277	**0.9381**
	Precision	0.9199	0.9251	**0.9371**	0.9352	0.9303
	Recall	0.9247	0.9298	0.9302	**0.9397**	0.9379
	F1-Score	0.9223	0.9275	0.9335	**0.9374**	0.9341

4.2 Result and Analysis

As we can see from Fig. 5 and Table. 1, our experimental results demonstrate that FedGCD consistently outperforms FedProx and the others in terms of test accuracy and loss convergence across all levels of nonIIDness. Federated learning was conducted for 300 rounds for each experiment, and the experiment was repeated 20 times, and the average accuracy of each 20 rounds was recorded.

4.2.1. Non-IID setting (nonIIDness = 0.7)

In this high data heterogeneity scenario, FedGCD achieved significantly better test accuracy and faster loss convergence compared toFedProx or FedVar. This result highlights the effectiveness of our method in handling severe data heterogeneity by leveraging community detection and client weight calculation based on their participation in different communities.

4.2.2 Semi-IID setting (nonIIDness = 0.5)

In this intermediate data heterogeneity setting, FedGCD continued to outperform FedProx in terms of test accuracy and loss convergence. This indicates that our method is robust and adaptable to various data distribution scenarios, providing improved performance even when the data heterogeneity is less severe.

4.2.3 IID setting (nonIIDness = 0.2)

In this low data heterogeneity environment, FedGCD still

(Table 2) Number of Communities

	class 1	class 2	class 3	class 4	class 5
WSCC[19]	3	10	15	7	6
ClusterGAN[20]	3	9	13	8	5
Ours	4	9	12	8	7

achieved better test accuracy than FedProx. However, the difference in performance was not as significant asin the other settings. This is because FedGCD is specifically designed to address non-IID issues, and in an IID environment, the benefits of our method are less pronounced. Nonetheless, the results show that our method does not adversely affect the performance in an IID setting.

Overall, our experimental results confirm that FedGCD is an effective method for dealing with data heterogeneity in federated learning settings. It consistently outperforms the state-of-the-art FedProx algorithm, particularly in non-IID and semi-IID environments, where data heterogeneity poses significant challenges. Compared to FedProx, the method most commonly used, the average accuracy saw an increase ranging from 1.2% to 2.1% across all scenarios. The number of communities did not show a significant difference from the number of clusters found in the clustering-based method, which is the approach most similar to this study. By incorporating community detection and weighted client contributions, FedGCD offers a promising solution for improving the performance of federated learning systems in heterogeneous data scenarios.

4.3 Number of Communities

To verify ourmethod, we tested how the number of communities appeared when compared to other previous studies.

Unfortunately, there is no previous study that solved the non-IID data problem using the GNN-based community detection method in federated learning. However, there is a precedent for applying a clustering method similar to the community detection method, so for each test case in the FEMNIST dataset, the number of clusters and communities in the study and our study was compared.

WSCC [19] is the federated learning method with clustering that is based on weight similarity. ClusterGAN [20] is the federated learning method based on GAN. When compared with WSCC and ClusterGAN, the number of communities of our method for some FEMNIST class are shown in Table 2.

4.4 Discussion

Our proposed method incorporates a GNN-based community detection algorithm to identify overlapping communities in a federated learning environment, which is crucial for capturing the inherent complexities of the underlying data distribution structure among clients. The identification of overlapping communities allows FedGCD to better account for the shared information and similarities between clients that belong to multiple communities. The use of AIC for determining the optimal number of communities ensures that our model is not overly complex and preventsoverfitting. By calculating client weights based on their participation inthese communities, our method effectively accounts for the overlapping natureof communities and adjusts the contributions of clients in the federatedlearning process accordingly.

Experimentalresults on a FEMNIST dataset with varying degrees of nonIIDness demonstrate theeffectiveness of FedGCD in providing improved test accuracy and loss convergence in comparison to the state-of-the-art federated learning algorithm, FedProx. Across all levels of data heterogeneity, FedGCD consistently outperforms FedProx, with particularly noticeable performance improvements in non-IID and semi-IID settings. Even in IID settings, FedGCD maintains comparable performance to other algorithms, highlighting its versatility and adaptability.

4. Conclusions

In this paper, we presented FedGCD, a novel federated learning method designed to address the challenges of data heterogeneity and overlapping communities in federated learning settings. By leveraging community detection through graph neural networks (GNNs) and calculating client weights based on their participation indifferent communities, we have developed a more robust and adaptive federated learning algorithm.

By addressing data heterogeneity and overlapping communities, FedGCD represents a significant step forward in improving the performance of federated learning systems, making them more robust and adaptable to the diverse and complex data distributions encountered in real-world applications. This advancement in federated learning has the potential to greatly benefit a wide range of industries and applications, from healthcare to finance, by enabling more effective and efficient collaborative learning across distributed networks

In future work, we plan to extend our method to real-world datasets and further investigate other factors that might influence the performance of federated learning systems, such as communication efficiency, client dropout, and privacy preservation. Moreover, we aim to further optimize the community detection process by incorporating additional similarity metrics, adopting more advanced GNN architectures, or exploring other clustering techniques.

Reference

[1] McMahan, Brendan, et al., “Communication-efficient learning of deep networks from decentralized data,” Artificial intelligence and statistics. PMLR, 2017.

[2] Li, Tian, et al., “Federated learning: Challenges, methods, and future directions,” IEEE Signal Processing Magazine, 37(3), 50-60, 2020. https://doi.org/10.1109/MSP.2020.2975749

[3] Kipf, Thomas N., and Max Welling, “Semi-supervised classification with graph convolutional networks,” arXiv preprint arXiv: 1609.02907, 2016. https://doi.org/10.48550/arXiv.1609.02907

[4] Alet, Pierre, et al., “Gradient-based community detection in the weight space of deep models,” arXiv preprint arXiv: 2006.14033, 2020.

[5] Caldas, Sebastian, et al., “Leaf: A benchmark for federated settings,” arXiv preprint arXiv: 1812.01097, 2018.

[6] Smith, Virginia, et al., “Federated multi-task learning,” Advances in Neural Information Processing Systems, 2017.

[7] Khodak, Mikhail, et al., “Adaptive Federated Optimization,” arXiv preprint arXiv: 2003.00295, 2020.

[8] Li, Tian, et al., “Federated optimization in heterogeneous networks,” arXiv preprint arXiv: 1812.06127, 2018.

[9] Shin, Wooseok, and Jitae Shin, “FedVar: Federated Learning Algorithm with Weight Variation in Clients,” 2022 37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC), IEEE, 2022.

[10] He, Chaoyang, et al., “Fedgraphnn: a federated learning benchmark system for graph neural networks,” ICLR 2021 Workshop on Distributed and Private Machine Learning (DPML), 2021.

[11] Hu, Kai, et al., “Fedgcn: Federated learning-based graph convolutional networks for non-euclidean spatial data,” Mathematics, 10(6), 1000, 2022. https://doi.org/10.3390/math10061000

[12] Kainan Zhang, Zhipeng Cai, and Daehee Seo, “Privacy-Preserving Federated Graph Neural Network Learning on Non-IID Graph Data,” Wireless Communications and Mobile Computing, 2023. https://doi.org/10.1155/2023/8545101

[13] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek, “Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints,” IEEE transactions on neural networks and learning systems, 32(8), 3710-3722, 2020.

[14] Don Kurian Dennis, Tian Li, and Virginia Smith, “Heterogeneity for the win: One-shot federated clustering,” International Conference on Machine Learning, PMLR, 2021.

[15] Chandran, Pravin, et al., “Divide-and-Conquer Federated Learning Under Data Heterogeneity,” CS & IT Conference Proceedings, Vol. 11. No. 13, 2021.

[16] Santo Fortunato, Darko Hric, “Community detection in networks: A user guide,” Physics reports, 659, 1-44, 2016. https://doi.org/10.1016/j.physrep.2016.09.002

[17] Santo ortunato, “Community detection in graphs,” Physics reports, 486(3-5), 75-174, 2010. https://doi.org/10.1016/j.physrep.2009.11.002

[18] Scott I Vrieze, “Model selection and psychological theory: a discussion of the differences between the Akaike information criterion (AIC) and the Bayesian information criterion (BIC),” Psychological methods, 17(2), 228-243, 2012. https://doi.org/10.1037/a0027127

[19] P. Tian, W. Liao, W. Yu and E. Blasch, “WSCC: A Weight-Similarity-Based Client Clustering Approach for Non-IID Federated Learning,” in IEEE Internet of Things Journal, vol. 9, no. 20, pp. 20243-20256, 2022, https://doi.org/10.1109/JIOT.2022.3175149

[20] Y. Kim, E. A. Hakim, J. Haraldson, H.Eriksson, J. M. B. da Silva and C. Fischione, “Dynamic Clustering in Federated Learning,” ICC 2021 - IEEE International Conference on Communications, Montreal, QC, Canada, pp. 1-6, 2021. https://doi.org/10.1109/ICC42927.2021.9500877

◐ 저 자 소 개 ◑

Wooseok Shin

2017.08 B.S Semiconductor System Engineering, Sungkyunkwan University

2017.09~ Combined M.S and Ph.D Degree, Dept. of Electrical and Computer Engineering, Sungkyunkwan University

Research Interests: Federated Learning, Graph Neural Networks, AI/ML on 5G Networking

E-mail : swsda95@skku.edu

Jitae Shin

1986 B.S. Electrical Engineering, Seoul National University

1988 M.S. Nuclear Engineering, Korea Advanced Institute of Science & Technology

1998 M.S. Electrical Engineering, University of Southern California, USA

2001 Ph.D. Electrical Engineering, University of Southern California, USA

Professor of Sungkyunkwan University Department of Electrical & Computer Engineering

Research Interests: Medical Image Processing and System, Image/Video Communication and System

E-mail : jtshin@skku.edu