CLUSTERED FEDERATED LEARNING FOR HETEROGENEOUS FEATURE SPACES USING SIAMESE GRAPH CONVOLUTIONAL NEURAL NETWORK DISTANCE PREDICTION

Anonymous Authors[1]

ABSTRACT

Federated learning (FL) has been proposed to enhance performance of local machine learning models across multiple devices while maintaining data privacy. One of the main challenges in FL is data heterogeneity, which limits the benefits of knowledge exchange among federated models. Data heterogeneity is a two-fold problem: Non-IID (not independent and identically distributed) data, and heterogeneous feature space. While personalized federated learning (PFL) solutions address challenges with non-IID data, most of the solutions require the same feature space. The few PFL solutions that are applicable in feature heterogeneity scenarios map all local domains into a new common feature space, which may result in degraded performance because of the negative transfer effects from unrelated local models based on a very different feature space. To address this limitation, we propose a novel clustered federated learning based on a Siamese graph convolutional neural network (FedSGCNN). We predict positive transfer between clients using an SGCNN in order to create a distance matrix for clustering. This network-based prediction is more accurate as compared to other distance measures which fail to capture the structure of models. When there is feature space heterogeneity, we show that FedSGCNN outperforms the latest work by 1.2% in the Boston Housing dataset and 2.8% in the Obesity Level dataset.

1 INTRODUCTION

Federated Learning (FL) (McMahan et al., 2017) is a relatively new machine learning paradigm where multiple clients (e.g., smartphones or hospitals) train machine learning models collaboratively while keeping their raw data private. In an FL system, each client only sends its model parameters to a central server, the central server creates and updates a global model such as by averaging the client models' parameters, and finally, the central server sends the global model back to each client. One of the main challenges in FL is data heterogeneity. FL could suffer from significant model performance degradation when the clients' data are not independent and identically distributed (Non-IID) (Zhao et al., 2018). This is because in such cases optimal model parameters for clients will be very different across clients, and therefore, the average global model parameters could be far from optimal parameters for local client models.

Recently researchers have proposed various personalization-based solutions such as local fine-tuning(Finn et al., 2017), clustering(Sattler et al., 2020), and personalized layers(Liang et al., 2020) to address the data heterogeneity problem. However, most of such solutions are designed to only address the data distribution heterogeneity (Non-IID) and are inapplicable when feature spaces are heterogeneous. We show Non-IID and feature space heterogeneity in Figure 1. In case such feature heterogeneity exists, a server cannot simply average all local models because features may be in a different order or features might be very different, which leads to different model architectures. For instance, each hospital collects different features of patients' data and they cannot join an FL system due to this feature space heterogeneity. Addressing the feature heterogeneity problems could have a dramatic influence on FL applications because it accommodates data with any feature space as long as the output or task is the same.

To the best of our knowledge, there are only two approaches that are directly applicable to address feature heterogeneity in FL, namely, LG-FedAvg(Liang et al., 2020) and HH-HFL(Gao et al., 2019). These methods map all local feature spaces into one space and create a global network from the common space. However, with these approaches, each client could suffer from negative knowledge transfer if there are unrelated clients in the same FL system. So, there is a need for selecting clients who could provide positive knowledge transfer. HHHFL introduces maximum mean discrepancy (MMD) loss to reduce the discrepancy in a common embed-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the Machine Learning and Systems (MLSys) Conference. Do not distribute.

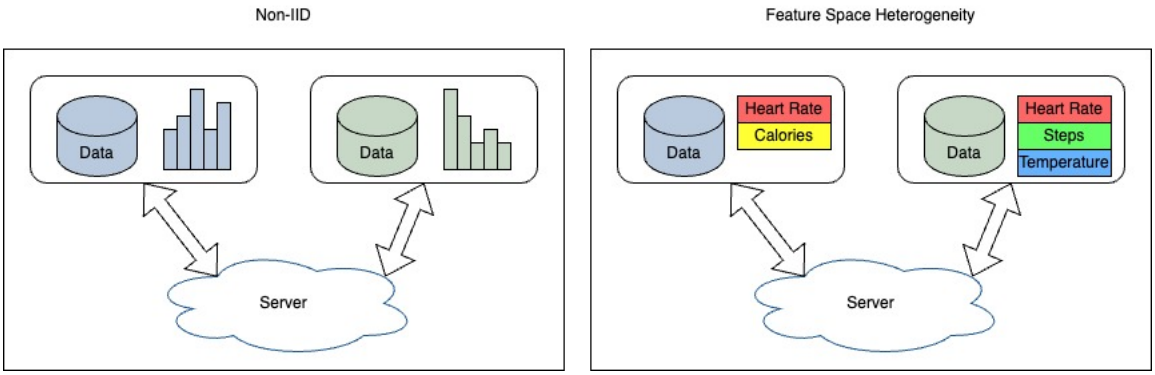

Figure 1. Non-IID and feature space heterogeneity in FL

ded space, but it could still suffer from negative knowledge transfer when the original feature space is very different. In addition to this issue, MMD calculation in HHHFL is required for each pair of domains, which is computationally expensive.

In order to address these challenges, we propose a novel clustered federated learning solution based on Siamese graph convolutional neural network. In order to do clustering for maximizing positive knowledge transfer among clients within each cluster of models, we first need some sort of distance measure that can estimate the positive knowledge transfer between each pair of clients. The recent work in clustering in FL uses cosine similarity of model parameters because in FL one does not have access to raw data. However, cosine similarity cannot capture structural information of the models such as network topology. In order to capture the model architecture of neural networks and estimate positive transfer more accurately, we use Siamese graph convolutional neural network (SGCNN) to learn the similarity between client models. This could help us create a more accurate distance matrix, which in turn could lead to better clustering for maximizing positive transfer among clients. To the best of our knowledge, we are the first to apply SGCNN to predict positive knowledge transfer between two clients in an FL system. In addition to clustering, we introduce MMD loss from a centroid. Instead of calculating MMD between all possible client pairs, we create a centroid by averaging features in the same cluster, then we use MMD from this centroid in a loss function. This will reduce computational burden significantly while maintaining the original purpose of reducing discrepancy in an embedded space.

We summarize our contributions as follows:

(1) We introduce a new clustered federated learning algorithm utilizing a Siamese graph convolutional neural network (SGCNN) to address the feature heterogeneity problem in FL. As compared to cosine similarity, SGCNN can create a more accurate distance matrix among clients because it captures structural information of model parameters.

(2) We introduce MMD loss from a centroid, which increases computational efficiency by avoiding calculation between all possible pairs.

(3) We conduct extensive experiments with tabular datasets and show that our solution is more effective than the latest related work.

The remainder of this paper is structured as follows. Section 2 provides a summary of most related work. Section 3 defines our problem and notations. Section 4 describes our proposed solutions. Section 5 presents our experimental design, results, and our new findings. Section 6 concludes this paper by providing a summary and our future work.

2 Related Work

2.1 Personalized Federated Learning

The most common FL approach is FedAvg (McMahan et al., 2017) where each local client sends model parameters to a central server and the server sends back average model parameters to each local client. Since FedAvg could suffer from model performance degradation when local data are statistically heterogeneous (Zhao et al., 2018), personalized federated learning(PFL) has been proposed. There are four types of PFL (Tan et al., 2022), namely, data-based (Jeong et al., 2018; Wang et al., 2020), model-based (Li & Wang, 2019; Chen et al., 2020; Ji et al., 2021; Fallah et al., 2020), architecture-based (Arivazhagan et al., 2019) and similarity-based (Ghosh et al., 2020; Briggs et al., 2020; Duan et al., 2020; Sattler et al., 2020; Zhang et al., 2020). PFL approaches suffer from a number of shortcomings. PFL shows promising results when the local data distribution is statistically heterogeneous, but most PFL approaches are not applicable when feature space is heterogeneous because most PFL methods usually apply calculations (such as pa-

Algorithm	Non-IID	Feature heterogeneity
LG-FedAvg, HHHFL	×	○
CFL	○	×
FedSGCNN (ours)	○	○

Table 1. Comparison of related work

rameter averaging) to each parameter across clients in the same position in the same model architecture. When feature space is heterogeneous, parameters in the same position across clients come from totally different features. Alternatively, those parameters in the same position may not even exist for some clients due to the difference in the number of features.

The only PFL solutions that address feature space heterogeneity are LG-FedAvg (Liang et al., 2020) and HHHFL(Gao et al., 2019). LG-FedAvg allows each local client to have local layers which train local representations. Hence, as long as local representations have the same feature space, LG-FedAvg is applicable to clients with heterogeneous feature spaces. However, LG-FedAvg could suffer from negative knowledge transfer when local representations are heterogeneous. On the other hand, HHHFL introduces MMD between each manifold as a domain loss to reduce the discrepancy in a common embedded space. Yet HHHFL could suffer from negative transfer when featuring spaces of the clients are sufficiently different and embedded features are heterogeneous. HHHFL is specifically designed for EEG (Electroencephalography) data and a more generic framework is needed to relax the assumption that all clients are related and their feature spaces are similar. Moreover, HHHFL requires MMD calculation between pairs of manifolds, which will be computationally expensive. Therefore, a novel framework is needed which can reduce negative transfer from unrelated clients and reduce the discrepancy in a common embedded space across multiple clients efficiently.

2.2 Clustered Federated Learning

One of the common solutions for data heterogeneity in the similarity-based PFL is clustered federated learning (CFL) (Sattler et al., 2020). CFL groups local clients into a number of clusters which contain "similar" clients. With CFL, knowledge is shared only among models within each cluster to avoid negative knowledge transfer from unrelated clients. The original CFL method (Sattler et al., 2020) applies a recursive bi-partitioning algorithm to find the optimal clusters. However, this is computationally expensive, so CFL methods that compute client models based on some sort of distance measure have been recently proposed. In particular, FeSEM (Xie et al., 2020) uses L_2 distance between parameters as a measure of similarity between two models; however, L_2 does not capture similarity properly with high dimensional data because it suffers from distance concentration phenomenon in high dimension (Sarkar & Ghosh, 2019). FedGroup (Duan et al., 2020) uses cosine similarity as the distance measure to address this limitation. However, since we need to convert model parameters into one-dimensional data to calculate cosine similarity, this measure cannot capture the structural information and topology of the model. In order to address this limitation, we use a graph convolutional neural network in order to capture the structural similarity between models and learn similarity using a Siamese neural network to create a more accurate distance matrix among clients. It is important to note that CFL methods are not directly applicable as a solution for feature heterogeneity problems because they require the same local model architectures when aggregating model parameters. In order to circumvent this issue, we first map local data into a common embedded space so that we can apply CFL.

In summary, as Table 1 shows, LG-FedAvg and HHHFL suffer from negative knowledge transfer due to Non-IID in an embedded space, while CFL is not applicable to feature space heterogeneity problems because it is under an assumption that feature space is the same. The naive solution could be to combine HHHFL and CFL into a hybrid approach. However, the efficacy of clustering (distance measure between clients) in such a hybrid approach is questionable, and also combining the two algorithms will significantly increase computational overhead. Therefore, we propose a novel distance measure between clients based on SGCNN, which works better than the recent work, and we also introduce MMD from a centroid for computational efficiency.

3 Problem Definition

3.1 Notations

We assume that there are n local clients $C = \{C_1, C_2, ...C_n\}$ in an FL system, where C_i refers to the ith client. C_i has (X_i, Y_i) as local data, where X_i represents feature data and Y_i represents label data. The feature space and/or dimensionality of X_i can vary across clients.

3.2 Problem Formalization

The objective of standard FL where we only consider one single global model is described as follows:

$$\min_{\theta_g} F(\theta_g) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f_g(\theta_g; X_i))$$

where θ_g represents global model parameters, ℓ refers to a loss function, and $f_g(x)$ denotes a global neural network.

With the feature heterogeneity problem, we cannot use a

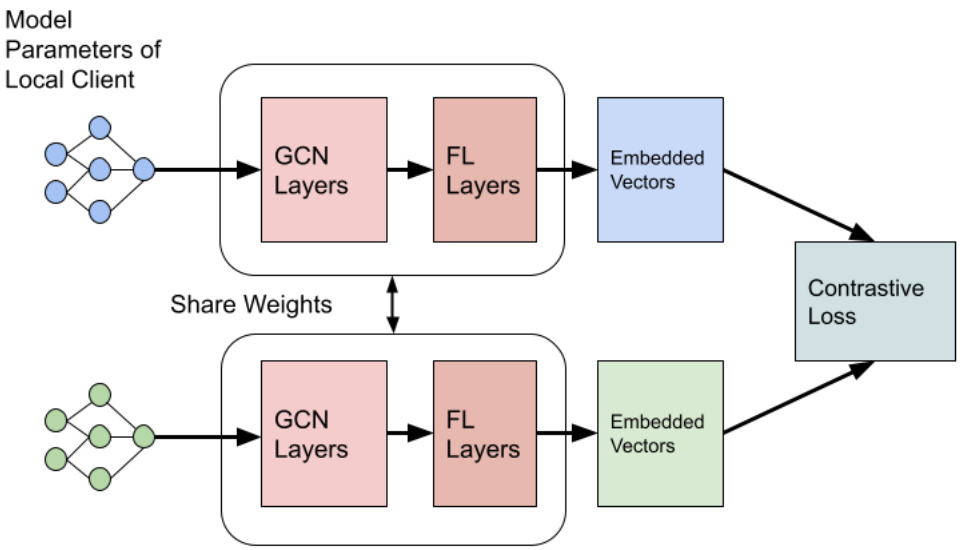

Figure 2. Architecture of the Siamese graph convolutional neural network

single global model because the feature space X_i varies across clients. Suppose Q_i denotes the ith embedded feature data in a common space. The objective of the feature heterogeneity problem is described as follows:

$$\min_{\theta_g} F(\theta_g) = \frac{1}{n}\sum_{i=1}^{n} \ell(Y_i, f_g(\theta_g; Q_i))$$

$$Q_i = f_{l,i}(X_i)$$

where $f_{l,i}(x)$ refers to a local neural network that projects feature data X_i to a common space.

When we assume that there are m clusters, this objective function could be extended as follows:

$$\min_{\theta_{g,1},\theta_{g,2},..\theta_{g,m}} F(\theta) = \sum_{j=1}^{m} w_j \sum_{i=1}^{k_j} \ell(Y_i, f_{g,j}(\theta_{g,j}; Q_i))$$

$$Q_i = f_{l,i}(X_i)$$

$$w_j = \frac{k_j}{n}$$

where $\theta_{g,i}$ represents the ith cluster global model parameters, k_j denotes the number of local clients in jth cluster, k_j refers to the number of local clients in jth cluster, and $f_{g,j}$ is a global neural network for jth cluster.

4 METHODOLOGY

In this section, before we provide an overview and discuss details of our proposal method, FedSGCNN, we will explain the benefit of the distance measure based on the Siamese graph neural network approach versus other distance measures.

4.1 Similarity Measure to Predict Positive Transfer

In this section, we will empirically observe whether a similarity measure based on a Siamese graph neural network could outperform the typical distance measures used for calculating the distance between client models in CFL, namely, cosine similarity and L_2 distance. Since our goal is to avoid negative knowledge transfer and maximize positive knowledge transfer among clients, it is essential to define similarity/distance between client models such that it can accurately predict the amount of positive knowledge transfer between models. Our hypothesis here is that the Siamese graph convolutional neural network could outperform the baselines in terms of predicting positive transfer between two client models because graph neural networks can capture structural information such as the connection between neurons, which are essential to determine the functionality of neural networks. Siamese graph convolutional neural network (SGCNN) has two identical graph convolutional neural networks to learn whether two inputs are similar or not. Figure 2 shows the architecture of SGCNN. Graph convolutional neural network takes a graph as input and embeds a graph into a vector. This way, a graph neural

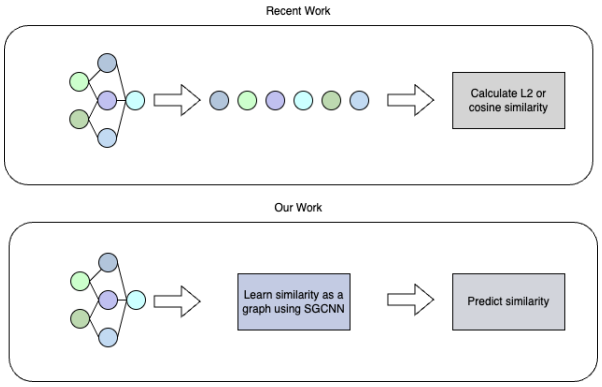

Figure 3. Comparison of similarity measures of neural network models

network can capture structural information. In particular, SGCNN performs well when the structured data involves a large number of classes, but only a few examples for each label are available in predicting similarity (Krivosheev et al., 2020; Lv et al., 2022; Gu et al., 2022). In case feature space heterogeneity exists and each feature space is considerably different from others, the neural networks corresponding to different models also vary considerably. Therefore, we might not be able to collect a sufficient number of labels (whether one neural network is similar to another network). It is under this constraint that we can make the most benefit of the SGCNN strength.

As Figure 3 describes, cosine similarity and L_2 distance flatten neuron connections and calculate distance, which means they cannot utilize structural information of model architecture in calculating similarity. On the other hand, SGCNN takes a network as a graph and learns similarity with the structural information, which could lead to better similarity prediction.

Here, we first define the positive transfer in an FL system, then we make observations on the performance of similarity measures as discussed above. In particular, we can define positive knowledge transfer between two clients as follows:

$PT(\theta_1^{local}, D_1, \theta_2^{local}, D_2) = \frac{1}{2}\sum_i^2 \ell(\theta_i^{fed}; D_i^{test}) - \frac{1}{2}\sum_i^2 \ell(\theta_i^{local}; D_i^{test})$

where θ_i^{fed} refers to the parameters of ith client after federated learning, θ_i^{local} refers to parameters of ith client after local training, D_i is local data of ith client and D_i^{test} is test data of i-th client.

To evaluate the performance of various similarity measures in predicting positive knowledge transfer according to the aforementioned definition, we performed an experiment as follows. With this experiment, using the Boston Housing dataset from the UCI dataset, we assign 5 out of 13 features randomly to each client to simulate feature heterogeneity. We use a multi-perceptron consisting of 64, 32, 16 neurons in each layer, respectively. We create 25 clients and apply LG-FedAvg to each possible pair 100 rounds and calculate positive transfer between all possible pairs. We use 5 clients as test cases. As a baseline, we consider L_2 distance and cosine similarity because these are the common distance measures for clustering in FL. The particular architecture we use for SGCNN contains two GCN layers with one fully connected layer and we use Adam with a learning rate 0.01 for the optimizer and train it 100 rounds with 10 local epochs. Since the main purpose of CFL is to distinguish and separate pairs of clients that can incur negative transfer of knowledge, we use classification accuracy as an evaluation metric to compare the performance of the aforementioned distance measures. Here, if a value of the positive transfer is greater than 0, we consider it as "positive", and otherwise "negative".

similarity measure	test accuracy
$l2$ distance	0.48 ±0.143
cosine similarity	0.4 ±0.158

Table 2. Comparison of baseline similarity measure

number of clients for training	test accuracy
5	0.460±0.148
10	0.690±0.155
15	0.74±0.309
20	0.699±0.027

Table 3. Comparison of baseline similarity measure

Tables 2 and 3 show the comparison of classification accuracy of positive transfer. Table 2 shows that L_2 distance and cosine similarity of model parameters (gradients updates) perform similarly to a random predictor, which means that these measures fail to accurately predict positive transfer in the feature heterogeneity scenario. Table 3 shows that the classification accuracy of SGCNN with a different number of clients exceeds the baselines when the number of clients is equal to or greater than 10. We attribute this to the capability of SGCNN in capturing structural information of neural models, which is essential for the accurate prediction of positive knowledge transfer.

4.2 Overview

Now that we have empirically shown SGCNN can provide better similarity measures of two neural networks in FL, we apply this to an FL training process. Figure 4 provides an overview of the proposed FedSGCNN algorithm, including our two novel contributions to address the feature

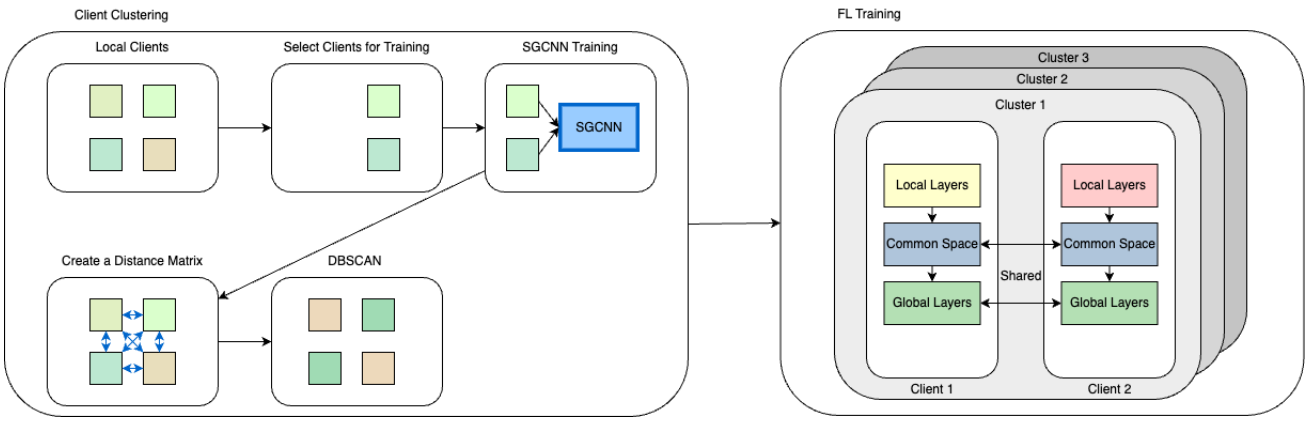

Figure 4. Overview of FedSGCNN

space heterogeneity problem. As mentioned before, the first contribution is to introduce a Siamese graph convolutional network to predict positive transfer between clients, which helps with increasing positive transfer among clusters. The second contribution is to introduce an MMD loss from a centroid, which will help FedSGCNN reduce the discrepancy in an embedded space while increasing computational efficiency compared to standard MMD loss that requires MMD calculation among all possible pairs. Algorithm 1 shows the details of FedDNPR. We further elaborate on the implementation steps as follows:

Step 1: We select some clients for SGCNN training. As we have seen in Section 4.1, we need at least 10 clients to make an SGCNN model useful.

Step 2: Each selected client performs a local training for E epochs. Each client uses the same initial parameters. These networks will be the inputs for SGCNN.

Step 3: For all pairs of selected clients, we calculate positive transfer using the following formula (explained in Section 4.1). We first train two models individually to obtain local model performance. Thereafter, we perform LG-FedAvg between those two models to obtain federated learning-based model performance. Since the feature space in each model could be different, each model maps features into the same feature space, then they share global layers. Lastly, we calculate those differences for the positive transfer. If the value is negative, it means those models suffer from negative transfer when they are in the same FL training.

$$PT(\theta_1^{local}, D_1, \theta_2^{local}, D_2) = \frac{1}{2}\sum_i^2 \ell(\theta_i^{fed}; D_i^{test}) - \frac{1}{2}\sum_i^2 \ell(\theta_i^{local}; D_i^{test})$$

Step 4: We train SGCNN taking two networks as graph input and positive transfer as an output (we consider positive transfer as 0 and negative transfer as 1). Since the feature space in each model could be different, only global layers are used as input.

Step 5: We create a distance matrix between all clients by using SGCNN. We first train local models for E epochs. Thereafter, we predict the similarity between two clients using SGCNN, which we will use for a distance matrix.

Step 6: We apply DBSCAN (Ester et al., 1996) to create client clustering using the distance matrix we create. This clustering groups clients which can incur positive transfer on each other.

Step 7: The FL training starts at this step. Each client performs local training for E epochs. The loss function is a combination of the standard loss function (i.e., mean squared error) and MMD loss from a centroid in the same cluster.

Step 8: Each client sends the global layers of the updated models and embedded vectors in a common feature space of local data to a central server. Since embedded vectors need each client's local layers to replicate the original data, there is no significant privacy issue here.

Step 9: The central server updates the global layers of each cluster by averaging global layers in local clients from the same cluster. The central server randomly selects some embedded vectors to create a distribution of a centroid for the MMD loss calculation of each client.

Step 10: Iterate Steps 7-9 until a predefined communication round or until models converge.

4.3 Clustering based on Siamese Graph Convolutional Neural Network Distance Prediction

When feature spaces of the various local clients in an FL system are different, the FL system can suffer from the challenge of negative knowledge transfer–even with domain adaptation–because the embedded features in a shared space

Algorithm 1 FedSGCNN

Input: Clients C = $\{C_1, C_2, ...C_n\}$
Local datasets $D = \{D_1, D_2, ...D_n\}$
Initialized global network θ_g^0
Initialized local network $\theta_1^0, \theta_2^0, ...\theta_n^0$

```
Select s clients randomly and save it in S
for all possible pairs C_i and C_j in S do
    local training for E epochs and save them for input of SGCNN
    local training and calculate validation accuracy
    for each round t = 0, 1.... do
        Update local models by updating global layers
        Local training for C_i and C_j
        Average shared parameters to update global layers for the next round
    end for
    Calculate positive transfer
end for
Train SGCNN θ_SGCNN
Create distance matrix based on predictions of SGCNN θ_SGCNN
Apply DBSCAN clustering based on the distance matrix
for each cluster j = 0, 1...., k do
    for each round t = 0, 1.... do
        θ^t_{g,j} = θ^{t-1}_g
        Average embedded distribution Q_{j,t} is randomly sampled from embedded vectors of all clients in the same cluster
        Randomly select r clients and save it in S_{j,t}
        for each client C_i in S_{j,t} do
            Update global layers of θ^t_i by θ^t_{g,j}
            Update θ^t_i by gradient descent based on D_i with the loss function of normal loss function with λMMD(Q_{j,t}, θ^t_i(D_i))
        end for
        Update θ^t_{g,j} by averaging only shared parameters
    end for
end for
return θ^t_{g,1}, θ^t_{g,2}..., θ^t_{g,k}, θ^t_1, θ^t_2, ...θ^t_n
```

could be statistically heterogeneous.

The first step of FedSGCNN is to create model clusters, where each cluster consists of models that are expected to benefit from positive knowledge transfer when they share parameters. By grouping client models into clusters, the FL system can benefit from more positive transfer among intra-cluster pairs of models and avoid negative transfer among inter-cluster pairs of models. Toward this end, we train and use a Siamese graph convolutional neural network to accurately predict positive transfer between two clients based on their network structures. As we demonstrated in the previous section, SGCNN provides better positive transfer prediction as compared to other standard distance measures such as cosine similarity which are used to generate the model clusters in CFL approaches.

To train SGCNN, we first need to create training data. Toward this end, we randomly select a number of clients (e.g., 10 clients) and calculate a positive transfer between all possible pairs. Then, we can train SGCNN based on this data. Thereafter, we can use trained SGCNN to predict positive transfer among all clients. Once we have a distance matrix, we apply DBSCAN (Ester et al., 1996) to create the model clusters using the distance matrix derived by SGCNN.

4.4 Discrepancy Reduction from Centroid in a Common Space

HHHFL (Gao et al., 2019) includes MMD as a loss function to reduce discrepancies in the common space. MMD is calculated as follows:

$$MMD(P, Q) = ||E_{X\sim P}[\phi(X)] - E_{Y\sim Q}[\phi(Y)]||_{\mathcal{H}}$$

where P and Q are distributions, the function ϕ maps initial distributions to kernel Hilbert space $\mathcal{H}$.

Even though this is helpful to make embedded features non-IID, when we have multiple domains, the objective of MMD reduction is extended to the following:

$$\min \sum_{i\neq j} \sum_{i\neq j} MMD(Q_i, Q_j)$$

$$Q_i = \theta_i(D_i)$$

where θ_i is a local network for the ith client and D_i is a local dataset for the ith client. This is many-objective optimization problem, which is computationally expensive. Here, we propose

$$\min \sum_{i} MMD(Q_i, Q_{avg})$$

where Q_{avg} represents average distribution of $Q_1, Q_2, ...Q_k$, which is randomly sampled from total distribution of $Q_1, Q_2, ...Q_k$. This objective is much more computationally efficient because we can solve it independently for each local client.

For the overall loss of each local training, we use the following loss function:

$$\ell(\theta_i, D_i) + \lambda \sum_i MMD(Q_i, Q_{avg})$$

where ℓ represents a standard loss function, i.e., a classification loss (e.g., cross-entropy) for classification problems and mean square error for regression problems, and λ denotes a weight of the MMD loss.

5 Experimental Evaluation

5.1 Experimental Setup

5.1.1 Dataset

We use the Boston Housing dataset, the wine quality dataset, and the obesity level dataset from the UCI dataset. We use one-dimensional datasets for this experiment because feature heterogeneity problem often happens in one-dimensional data when different organizations collect different feature data. Table 4 shows the detailed specifications of the datasets in our experiment. For instance, The Boston Housing dataset contains 13 features, so we assign 5 features randomly to each client. We also simulate the clustered structure of feature heterogeneity. For instance, we split 13 features into 2 groups, then we assign 5 features from one of the groups to one of local clients in the Boston Housing dataset. We create 2 clusters for the Boston Housing dataset, 2 for the Wine Quality dataset, and 3 for the Obesity Level dataset.

5.1.2 Baselines

We use local training (termed *Standalone* here) and LG-FedAvg (Liang et al., 2020) as baselines. Local training is the standard solution without FL and LG-FedAvg is the most general and recent work that addresses the feature heterogeneity problem in FL. In order to isolate and demonstrate the effectiveness of our proposed solutions, we report results based on three models: (1) Model 1, which is LG-FedAvg with MMD loss from a centroid, (2) Model 2: which is similar to Model 1 but with random clustering, and (3) Model 3, which is similar to Model 1 but with clustering using SGCNN as distance/similarity measure.

We do not use CFL as baselines because it is not readily applicable in case of feature space heterogeneity. This is due to the assumption that CFL requires the same feature space. In addition to that, in Section 4.1, we empirically showed that our similarity measure is dominant, so we do not see necessity to show the efficacy of our clustering in comparison with other clustering methods.

5.1.3 Hyperparameters

We use a multi-perceptron consisting of 64, 32, and 16 neurons in each layer, respectively, for the Boston Housing dataset. We use a multi-perceptron consisting of 16 and 8 neurons in each layer, respectively, for both the Wine Quality and Obesity Level datasets because this simpler model architecture provides better performance based on our experiment. We train these models with the following hyperparameters: the local epoch is 10, the local batch size is 32, the optimizer is SGD with a learning rate 0.01 and momentum 0.5 for the Boston Housing dataset and Adam with the learning rate 0.01 for the Wine Quality and the Obesity Level dataset. We train 100 rounds with all clients joining all rounds for the Boston Housing dataset, and we train 1,000 rounds for both Wine Quality and Obesity Level datasets with 10% of the clients joining each round. For LG-FedAvg and our models, we use the first layer as local layers to map all features into the same embedding space. For FedSGCNN, the weight of MMD loss λ is 0.5 for all the datasets. We use 3 different seeds and report those 3 times average with 95% confidence interval. The architecture of SGCNN contains two GCN layers with one fully connected layer, and we use Adam with the learning rate 0.01 for the optimizer and train it 100 rounds with 10 local epochs.

5.1.4 Evaluation Metrics

As an evaluation metric, we use R-squared for the regression problems (the Boston Housing dataset) and accuracy for the classification problems (the Wine Quality and Obesity Level datasets). We use the maximum test R-squared or accuracy as a final result in the same way as (Duan et al., 2020) because we could assume early stopping (Yao et al., 2007) for each cluster.

5.2 Experimental Results

5.2.1 Model Performance Comparison

Table 5 shows the model performance comparison with the Boston Housing dataset, the Wine Quality dataset, and the Obesity Level dataset, in two scenarios with and without clustered feature heterogeneity structure. Noticeably, FedSGCNN (Model 3 in the table) performs the best in all the scenarios. When there is a clustered structure in feature sets, we observe that FedSGCNN outperforms the latest work by 1.2% in the Boston Housing dataset; 0.5% in the Wine Quality dataset; and 2.8% in the Obesity Level dataset.

LG-FedAvg performs worse than Standalone with the

Dataset Name	Number of Features	Features per Client	Samples per Client	Number of Clients
Boston Housing	13	5	20	20
Wine Quality	11	4	10	100
Obesity Level	17	4	20	100

Table 4. Dataset details

Method	Boston	Boston(c)	Wine	Wine(c)	Obesity	Obesity(c)
Standalone	0.0973±0.0357	0.1344±0.0433	46.92±1.03	46.27±3.30	29.52±8.57	28.13±4.51
LG-FedAvg	0.0764±0.0551	0.1285±0.0333	46.93±0.59	46.32±2.35	28.80±4.08	28.41±2.71
LG-FedAvg+MMD loss (1)	0.1130±0.0547	0.1297±0.0158	49.61±4.43	48.32±2.26	28.61±2.24	28.89±5.38
(1)+ random clustering (2)	0.1256±0.0320	0.1363±0.0240	52.86±2.39	51.73±2.75	32.26±4.59	31.80±6.18
(1)+ SGCNN clustering (3)	**0.1305±0.0434**	**0.1481±0.0217**	**53.04±0.70**	**52.27±6.18**	**34.30±5.76**	**34.64±4.96**

Table 5. The R square value (Boston) or classification accuracy (Wine and Obesity) with 95% confidence interval. C stands for Clustered structure feature heterogeneity.

Boston Housing dataset and non-clustered scenario in the Obesity Level dataset, and it performs as well as Standalone in the other cases. This indicates that simply averaging global layers could be detrimental to each model, meaning that there is significant heterogeneity in the common feature space and each model suffers from negative knowledge transfer. On the other hand, the observation that FedSGCNN outperforms Standalone and LG-FedAvg implies that FedSGCNN effectively groups similar clients and increases positive knowledge transfer among clients.

When we pay attention to the difference in performance among Model 1, Model 2, and Model 3, we make the observation that computing MMD loss from a centroid increases performance when feature heterogeneity is relatively insignificant, while client clustering based on SGCNN always increases model performance. First of all, the performance of LG-FedAvg is less than or equal to Standalone. This indicates that even though LG-FedAvg maps all local heterogeneous feature spaces into a common space and shares knowledge in the common space, it suffers from negative knowledge transfer because of data heterogeneity in the embedding space. Introducing an MMD loss from centroid results in increased performance as compared to the original LG-FedAvg with the Boston Housing dataset and Wine Quality dataset, while it decreases performance with the Obesity Level dataset. This could be explained by observing that the Obesity Level dataset has more features (17) than other datasets (11 and 13), so there are many more patterns in feature space in this dataset, making all clients' optima far from a centroid where in turn reducing discrepancy from a centroid provides a negative effect on model performance. On the other hand, the Boston Housing dataset and the Wine Quality dataset have relatively fewer features, which makes the feature space of all clients relatively less heterogeneous in those datasets. In this case, it seems that the advantage of reducing data heterogeneity in a common space outweighs the disadvantage of reducing discrepancy from unrelated clients.

Finally, introducing client clustering based on SGCNN increases model performance as compared to random client clustering. This could be explained based on the observation that client clustering with SGCNN groups clients into clusters with positive knowledge transfer and reduces negative transfer from unrelated clients. We also observe that the increase in model performance is more significant when there is a clustered structure in feature heterogeneity, which reaffirms the previous observation.

5.2.2 *Impact of Weight of MMD Loss*

We also illustrate the impact of the weight in MMD loss in Figure 5. We train the same model with the same hyperparameters with one seed using the Boston Housing dataset (20 samples per client). This figure shows that the R-square increases as weight increases up to 0.5, and the R-square decreases as weight increases beyond 0.5. The Pearson correlation coefficient 0.70 with a weight up to 0.5, and it is -0.43 when the weight is larger than 0.5. This means that introducing MMD loss increases model performance as long as the weight is not too large, and when the weight is too large, MMD loss could be dominant in the loss function and becomes detrimental to the other standard loss function.

6 Conclusions and Future Work

In this paper, we proposed a new solution to the feature heterogeneity problem in FL based on client clustering using SGCNN as well as discrepancy reduction from a centroid in an embedded space. We empirically showed that

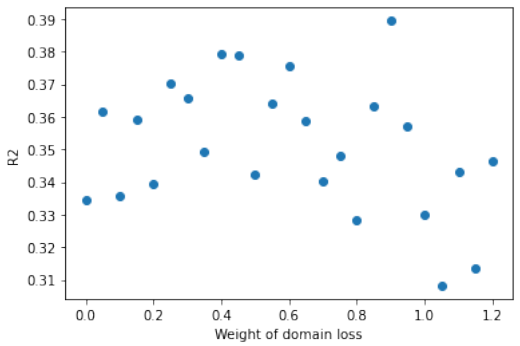

Figure 5. Impact of the weight for MMD Loss

SGCNN could outperform standard distance measures in terms of predicting positive transfer between two clients, and demonstrated that FedSGCNN outperforms the latest work especially when there is a clustered structure in feature heterogeneity. Our solution relaxes the assumption that each client in FL has to have a same feature space, and our solution is applicable as long as a task or output is the same across clients, which makes our solution much more widely adaptable.

Our future work is to extend our algorithm to two-dimensional datasets with convolutional neural networks and observe whether SGCNN can capture structural information of the convolutional neural networks. Another future direction is to extend SGCNN to other scenarios where we need to predict positive transfer among neural networks, such as multi-task learning and transfer learning.

REFERENCES

Arivazhagan, M. G., Aggarwal, V., Singh, A. K., and Choudhary, S. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019.

Briggs, C., Fan, Z., and Andras, P. Federated learning with hierarchical clustering of local updates to improve training on non-iid data. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9. IEEE, 2020.

Chen, Y., Qin, X., Wang, J., Yu, C., and Gao, W. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4):83–93, 2020.

Duan, M., Liu, D., Ji, X., Liu, R., Liang, L., Chen, X., and Tan, Y. Fedgroup: Efficient clustered federated learning via decomposed data-driven measure. *arXiv preprint arXiv:2010.06870*, 2020.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. Density-based spatial clustering of applications with noise. In *Int. Conf. Knowledge Discovery and Data Mining*, volume 240, 1996.

Fallah, A., Mokhtari, A., and Ozdaglar, A. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33:3557–3568, 2020.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Gao, D., Ju, C., Wei, X., Liu, Y., Chen, T., and Yang, Q. Hhhfl: Hierarchical heterogeneous horizontal federated learning for electroencephalography. *arXiv preprint arXiv:1909.05784*, 2019.

Ghosh, A., Chung, J., Yin, D., and Ramchandran, K. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33: 19586–19597, 2020.

Gu, Y., Yang, X., Tian, L., Yang, H., Lv, J., Yang, C., Wang, J., Xi, J., Kong, G., and Zhang, W. Structure-aware siamese graph neural networks for encounter-level patient similarity learning. *Journal of Biomedical Informatics*, 127:104027, 2022.

Jeong, E., Oh, S., Kim, H., Park, J., Bennis, M., and Kim, S.-L. Communication-efficient on-device machine learning: Federated distillation and augmentation under non-iid private data. *arXiv preprint arXiv:1811.11479*, 2018.

Ji, S., Saravirta, T., Pan, S., Long, G., and Walid, A. Emerging trends in federated learning: From model fusion to federated x learning. *arXiv preprint arXiv:2102.12920*, 2021.

Krivosheev, E., Atzeni, M., Mirylenka, K., Scotton, P., and Casati, F. Siamese graph neural networks for data integration. *arXiv preprint arXiv:2001.06543*, 2020.

Li, D. and Wang, J. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019.

Liang, P. P., Liu, T., Ziyin, L., Allen, N. B., Auerbach, R. P., Brent, D., Salakhutdinov, R., and Morency, L.-P. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020.

Lv, J., Li, Z., Chen, H., Qi, Y., and Wu, C. Path-aware siamese graph neural network for link prediction. *arXiv preprint arXiv:2208.05781*, 2022.

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

Sarkar, S. and Ghosh, A. K. On perfect clustering of high dimension, low sample size data. *IEEE transactions on pattern analysis and machine intelligence*, 42(9):2257–2272, 2019.

Sattler, F., Müller, K.-R., and Samek, W. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE transactions on neural networks and learning systems*, 32(8):3710–3722, 2020.

Tan, A. Z., Yu, H., Cui, L., and Yang, Q. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Wang, H., Kaplan, Z., Niu, D., and Li, B. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pp. 1698–1707. IEEE, 2020.

Xie, M., Long, G., Shen, T., Zhou, T., Wang, X., Jiang, J., and Zhang, C. Multi-center federated learning. *arXiv preprint arXiv:2005.01026*, 2020.

Yao, Y., Rosasco, L., and Caponnetto, A. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

Zhang, M., Sapra, K., Fidler, S., Yeung, S., and Alvarez, J. M. Personalized federated learning with first order model optimization. *arXiv preprint arXiv:2012.08565*, 2020.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.