A Federated Model Personalisation Method Based on Sparsity Representation and Clustering

Hailin Yang, Yanhong Huang*, Jianqi Shi and Fangda Cai*

National Trusted Embedded Software Engineering Technology Research Center,
East China Normal University, Shanghai, China
hlyang@stu.ecnu.edu.cn
{yhhuang, jqshi, fdcai}@sei.ecnu.edu.cn

***Abstract*—As federated learning (FL) becomes more extensively employed, it attracts an increasing number of scholars and practitioners. In contrast to traditional decentralized machine learning approaches that acquire users' raw data, FL gathers locally updated gradients, protecting their privacy. However, different users may have disparate data distributions, resulting in underperformance of the federated model. It is beneficial to adapt the federated model to various data distributions. Numerous personalisation approaches have been examined, but most of them are limited to a single device with minimal data, making them susceptible to bias and overfitting. In fact, the data distributions of certain users are similar, and these similarities can be leveraged to increase the efficacy of personalisation. In this research, we describe a sparsity-based clustering method, as well as a federated personalisation strategy based on it. Our method mitigates the impact of non-IID data and generates more accurate local models. The trials reveal that it outperforms several of its counterparts.**

***Index Terms*—Federated Learning, Privacy, Personalisation, Clustering**

I. Introduction

Increasingly, internet-connected devices are becoming an integral part of people's daily life. During their operation, they create a significant amount of data. Alough this data is of immense financial worth, most users don't want their privacy to be abused. It's meaningful to exploit local resources to collaboratively train machine learning (ML) models while keeping data on the devices. Federated learning [1] satisfies this requirement for it only requires participants to upload their locally determined gardiends to a sever and then aggregate them. The most well-known aggregation algorithm is the Federated Average (FedAvg), which simply averages each user's gradients. And vanilla FL refers to FL using FedAvg. However, the global model generated by vanilla FL is usually not satisfying for a specific user, because data is usually Non-independently and identically distributed(non-IDD) between clients [2].

A wealth of ways have been presented to solve the difficulties that data heterogeneity poses for FL. The global model can be improved by, for instance, retraining its parameters using input from a single user to produce a unique local model [3]. Yu et al. [4] introduced two further ways of personalization, namely multi-task learning and knowledge distillation. In addition, Arivazhagan et al. [5] suggested a federated learning strategy with personalization layers.

While all of these tactics contribute to personalisation, they all suffer from the same disadvantage: insufficient data. As a result, scholars investigated the feasibility of using federated approaches to indirectly augment the data used for personalisation. Liu et al. [7] describes a method that groups clients based on their data distribution, and use FedAvg within clusters to personalize the global model. Nonetheless, it was assumed that clients belonging to the same cluster would have the same data distribution, which occurs infrequently.

Clients' data distributions are similar but not identical, i.e., clients' preferences are consistent in general but varied in specifics. According to Liu et al. [7] the sparsity of an input image's feature map is often unique and may be utilized as a representation of the image. Simultaneously, this form of data representation is privacy-protective, as it conceals the underlying data. We have taken this idea and created some beneficial changes.

In this research, we offer a privacy-preserving federated personalisation technique to mitigates non-IID impact of FL. To begin, we transform the data distribution of each client into a sparsity vector. Then, we cluster these sparsity vectors to achieve client clustering. Following that, federated personalisation occurs within each cluster.

Our technique implemented federated learning at a finer grain, allowing clients in the same cluster to exchange critical knowledge without sacrificing their identity. Each client can obtain a model developed specifically for them. This work's contribution can be summarized as follows:

- We propose a privacy-preserving clustering method based on vector representations of clients' data distributions.
- Our strategy allows for some variation in the data distribution of clients within the same cluster, which is more practical.
- To maximize the effect of personalisation, we innovatively employ a FL approach with personalisation layers within clusters.
- We ran multiple relevant experiments and found that our technique outperforms its competitors.

*Corresponding Author
DOI reference number: 10.18293/SEKE2022-100

The remainder is structured as follows: Section II provides background information. Section III elaborates on the proposed method. Section IV details the experiments and their outcomes. Additionally, Section V discusses related work. Section VI concludes and suggests some potential directions worth pursuing.

II. Preliminaries

As a starting point, this section gives background information on the concepts of sparsity, clustering, and personalisation in order to aid in the comprehension of the technique we are proposing.

A. Sparsity

Sparsity is a numerical value that indicates the percentage of zeros in a matrix. It is frequently used to accelerate the inference process of Convolutional Neural Networks (CNNs) [10]. The first instance of channel-wise sparsity being used for data representation has appeared in [7]. A sparsity vector is constructed as follows, v_k represents the sparsity of the k_{th} channel.

$$V = \{v_1, v_2, v_3, v_4, ..., v_{k-1}, v_k\}$$

While feature maps are frequently used to portray raw data, they reveal too much fundamental information and are vulnerable to privacy leaks. In comparison, it is nearly impossible to deduce sufficient knowledge from a sparsity vector, offering a higher level of privacy protection. On the other hand, this representation format also contributes to the reduction of communication and processing overhead, as a sparsity vector is a number that can be transferred and calculated rapidly.

B. Clustering

Clustering is a technique for grouping similar objects together, with the core issue being determining how to measure similarity. There are a variety of distances that can be used to estimate similarity. Among these, Euclidean Distance [10] is a straightforward yet widely used one. The Euclidean distance between two vectors is calculated as follows:

$$Dist(V_a, V_b) = \sqrt{\sum_{j=1}^{j=k} (V_a^{(j)} - V_b^{(j)})^2} \tag{1}$$

V_a and V_b denote vectors, while $V_a^{(j)}$ and $V_b^{(j)}$ denote the j^{th} component of V_a and V_b, respectively.

Kmeans is a well-known technique for clustering data based on Euclidean distance. It initially selects k points as centroids and then groups the points closest to the centroid with them. The cluster centroid is iteratively updated until the total distance between it and the points inside the cluster no longer decreases. At this point, the clustering results are obtained.

C. Personalisation

FL collects and aggregates local gradients from participants during each communication round in order to update the global model. When users' data distributions are comparable, the global model easily converges and demonstrates superior performance versus the local models. However, this is not always the case. Once data distribution is highly variable, the global model typically underperforms.

A bulk of academics have examined the difficulties inherent in applying federated learning to heterogeneous data sources. A frequently used method is to further personalize the global model. Personalisation, in general, refers to optimization techniques that make the federated global model better suited to local clients.

III. Approach

This section details our strategy. For starters, we demonstrate how to generate sparsity vectors from the user's raw data which consist of images with different classes. Following that, we describe the clustering procedure using sparsity vectors. Finally, we describe the federated cluster-wise personalisation technique.

A. Generation of sparsity vectors

Prior to constructing a sparsity vector for a client, we must first determine the sparsity of each image on that client. The sparsity of an image is computed as shown in Figure 1.

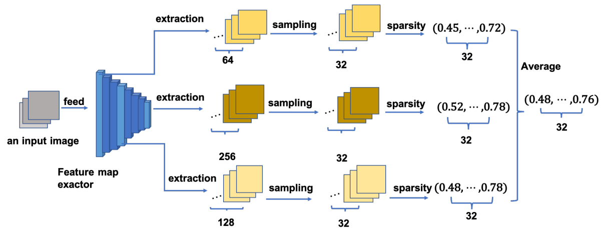

Fig. 1. The process of transforming a single image into a sparsity vector.

For feature extraction, we employ the CNN structure. CNNs are composed of multiple layers of neurons, with each layer producing a feature map. We calculate the sparsity of feature maps produced by Rectified Linear Unit (ReLU) layers because ReLU layers set a section of neurons' outputs to zero, resulting in sparse feature maps. The VGG [8] network

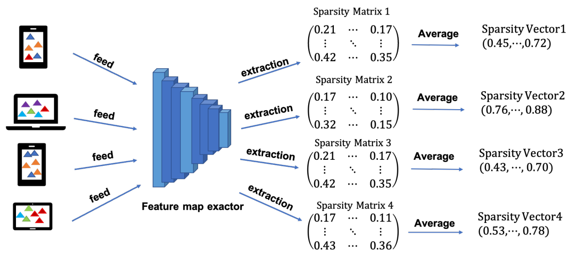
feed
feed
feed
feed
Feature map exactor
extraction
extraction
extraction
extraction
Sparsity Matrix 1
$\begin{pmatrix} 0.21 & \cdots & 0.17 \\ \vdots & \ddots & \vdots \\ 0.42 & \cdots & 0.35 \end{pmatrix}$
Average
Sparsity Vector1
(0.45,···,0.72)
Sparsity Matrix 2
$\begin{pmatrix} 0.17 & \cdots & 0.10 \\ \vdots & \ddots & \vdots \\ 0.32 & \cdots & 0.15 \end{pmatrix}$
Average
Sparsity Vector2
(0.76,···, 0.88)
Sparsity Matrix 3
$\begin{pmatrix} 0.21 & \cdots & 0.17 \\ \vdots & \ddots & \vdots \\ 0.42 & \cdots & 0.35 \end{pmatrix}$
Average
Sparsity Vector3
(0.43, ···, 0.70)
Sparsity Matrix 4
$\begin{pmatrix} 0.17 & \cdots & 0.11 \\ \vdots & \ddots & \vdots \\ 0.43 & \cdots & 0.36 \end{pmatrix}$
Average
Sparsity Vector4
(0.53,···, 0.78)

Fig. 2. All data are arranged in a matrix by their sparsity vectors, and the first dimension of this matrix is averaged to generate a vector representing the data distribution..

stacked small convolutional kernels instead of large ones repeatedly, increasing the network's depth while maintaining the same field of perception, and thereby improving the network's capacity to extract features. VGG-11 is the simpleset

VGG network. We replace the original fully connected layers of VGG-11 with a global average pooling layer, and then use it as the feature extractor. Feature maps become more representive with the modified VGG-11 for it establishes a direct connection between feature maps and classes.

Additionally, we take the first three ReLU layers as the extraction layers rather than a single one to enhance the extraction. Feature maps from multiple ReLU layers do not necessarily have the same number of channels, so we picked k common channels to calculate a sparsity vector.

As a result, the sparsity vectors from separate layers become identical in size and can be averaged into one. We will obtain a n$*$k dimensional sparsity matrix for a user with n data. A one-dimensional sparsity vector with k elements is created by averaging the sparsity matrix over the first dimension. Figure 2 depicts the process of expressing clients' data as sparsity vectors.

B. Clustering Based on Sparsity Vectors

The concept of sparsity-based clustering lies at the heart of our approach. With respect to this idea, it was first put forward by Liu et al. [7]. According to them, when the distance between two sparsity vectors is smaller than a certain threshold, they are considered similar. However, it's a challenge to determine this crucial hyperparameter in practice. It can be counterproductive if the clustering result is inaccurate. Considering the clustering categories are less extensive than the threshold, we employ k-means clustering to obtain better results.

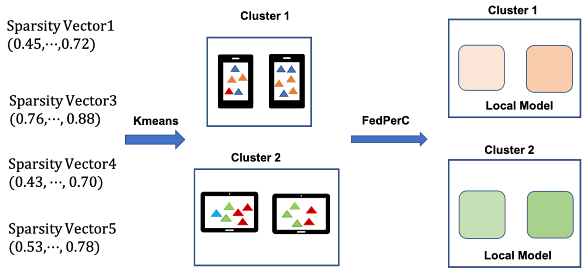

Fig. 3. Kmeans is performed based on sparsity vectors, and then FedPerC is conducted based on clusters.

The sparsity vector representing a client is computed using all its data. Then we compute the Euclidean Distance between each pair of vectors and then cluster them using the k-means algorithm. Based on the clustering results, we ran FedPerC to personalise the federated global model that was previously trained. This procedure is depicted in Figure 3. FedPerC is a cluster-based algorithm for federated personalization that will be talked about in the next section.

For clustering, k is critical. If k is set too high, clustering will be less successful, and if set too low, the benefits of federated learning will be missed. We can choose k according to our prior understanding of the dataset. If no prior knowledge exists, we can do several trials with randomly chosen clients to determine the best value of k and then apply it to all clients.

C. Federated Personalisation within Clusters

We expand personalisation from a single device to all devices inside a cluster, and we refer to this approach as FedPerC. To take use of the similarities in data distributions of users inside a cluster without losing sight of their differences, we split the model in two. The concept is derived from [5].

$W_{B,i}^{(k)}$ and $W_{P,i}^{(k)}$ signify the base layer and personalisation layer parameters of client i at the k_{th} training round, respectively. $W_{B,i}^{(k)}(W_{P,i}^{(k)})$ denote the entire model consist of this two part. LocalUpdate function refers to the training process on a single device. $W_{L,i}^{(k)}$ donate the local model of client i at the k_{th} training round. A formal description of FedPerC is given in Algorithm 1.

Algorithm 1 FedPerC

```
Input: $D_i$-the data of the $i_{th}$ client
C-clusters of clients
$W_g$-federated global model
$N$-training rounds
Output: $W_l$-personalised local models
 1: Initialisation: $W_l \leftarrow W_g$
 2: for k = 0,1,2,...,N-1 do
 3:     for c in C do
 4:         for i in c do
 5:             $W_{l,i}^{(k)} \leftarrow$ LocalUpdate($W_{l,i}^{(k-1)}, D_i$)
 6:             $W_{B,i}^{(k)}(W_{P,i}^{(k)}) \leftarrow W_{l,i}^{(k)}$
 7:         Aggregate: $W_{B,c}^{(k)} \leftarrow \sum_{i=1}^{c} W_{B,i}^{(k)}$
 8:         for i in c do
 9:             $W_{l,i}^{(k)} \leftarrow W_{B,c}^{(k)}(W_{P,i}^{(k)})$
10: return $W_l$ to local clients
```

Only the weights of base layers must be uploaded to the parameter server throughout the personalization process, the weights of personalization layers remain local. In Figure 4, we illustrate the process of personalizing within clusters.

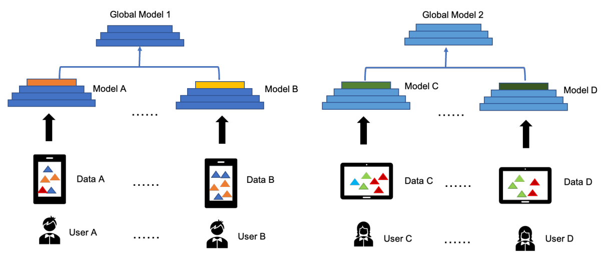

Fig. 4. Users within a cluster share the common base layers, and their personalisation layers can be adapted to individual data.

A formal description of the personalised goals is as follows: Suppose M_g donates the global model trained federally on data from all clients and M_l^i denotes the local model of the i^{th} client after personalisation. The number of clients is n, and C_i represents the cluster to which the i^{th} client belongs. The goal of FedPerC is to update M_g with the assistance of C_i

TABLE I
ACHIEVED ACCURACY (%) ON CIFAR-100 OF THE BASELINE AND DIFFERENT PERSONALIZATION METHODS USING RESNET34.

Method	client1	client2	client3	client4	client5	client6	client7	client8	client9	client10
BaseLine	45.17	41.39	43.18	41.11	41.67	35.83	38.33	39.72	38.06	41.39
FineTune	64.49	61.39	64.49	59.44	64.17	62.50	61.67	63.89	64.44	62.50
PFA	63.92	57.50	65.63	58.33	67.22	63.89	60.83	63.06	61.94	64.17
Ours	**65.34**	**65.28**	**65.06**	**63.06**	**68.89**	**65.56**	**63.89**	**64.72**	**65.28**	**65.56**
Method	client11	client12	client13	client14	client15	client16	client17	client18	client19	client20
BaseLine	37.78	41.94	40.56	38.06	34.44	43.06	39.17	44.47	41.76	41.76
FineTune	56.94	62.22	60.2	58.33	57.22	65.00	59.17	64.24	63.92	68.47
PFA	58.33	64.17	59.44	60.56	60.56	67.50	58.61	66.86	63.07	63.64
Ours	**61.39**	**65.83**	**61.39**	**62.50**	**61.67**	**67.50**	**63.61**	**70.35**	**66.19**	**70.17**

TABLE II
ACHIEVED ACCURACY (%) ON CIFAR-100 OF THE BASELINE AND DIFFERENT PERSONALIZATION METHODS USING MOBILENETV1.

Method	client1	client2	client3	client4	client5	client6	client7	client8	client9	client10
BaseLine	37.22	33.33	34.38	31.11	38.33	35.83	35.56	40.29	37.79	41.68
FineTune	66.48	61.39	62.22	61.11	66.39	65.57	66.11	60.56	63.33	63.89
PFA	65.63	62.78	63.35	61.11	65.56	66.39	64.72	63.89	66.39	60.00
Ours	**68.75**	**65.56**	**65.34**	**65.56**	**70.56**	**68.06**	**68.33**	**65.56**	**67.50**	**69.44**
Method	client11	client12	client13	client14	client15	client16	client17	client18	client19	client20
BaseLine	37.50	41.11	35.00	29.44	33.33	41.11	34.44	40.70	40.63	31.82
FineTune	58.33	63.33	60.28	56.11	58.61	62.78	57.50	64.54	60.51	63.64
PFA	60.56	57.22	59.17	53.61	57.50	63.33	60.56	67.73	60.23	64.21
Ours	**62.50**	**65.56**	**63.06**	**58.61**	**63.33**	**66.94**	**63.61**	**69.19**	**63.92**	**67.33**

and generate a personalised model M_l^i. The following is the objective function:

$$min \sum_{i=0}^{n} \mathcal{L}(D_i, C_i, M_g) \tag{2}$$

where $\mathcal{L}$ denotes the loss function, and in this work we use the cross-entropy loss.

IV. EXPERIMENT

Detailed descriptions of the experimental setup, including the datasets and models used, as well as the implementation details and methodologies for comparison, are provided in this part.

A. Settings

On two widely used network architectures, ResNet [11] and MobileNet [12], we conducted experiments to validate our technique. ResNet addresses the issue of gradient vanishing through a residual structure, improving performance. MobileNet pioneered the notion of depthwise separable convolution, drastically reducing model parameters and making itself suitable for low-resource mobile devices. We employed ResNet34 and MobileNetV1 on the CIFAR-10 and CIFAR-100 datasets [6], respectively. They are both image classification datasets, with 10 and 100 categories, respectively. Additionally, we implement our strategy in Python using PyTorch [13].

At first, we simulate a federated environment. To be more precise, we simulate 20 clients and 5 clients roughly belonging to a type of distribution for each dataset. For clients from CIFAR-10 that are expected to be drawn from the same distribution, 80% of their data came from two common categories and 20% from a random category. CIFAR-100 is processed similarly, with similar clients sharing twenty classes of data and holding an additional five classes of data. Thus, four distinct data distributions correspond to four clusters. Furthermore, to mimic the finite nature of client data, we limit the quantity of data available to each client to no more than 1000.

Experiments was conducted to determine the values of the hyperparameters. The final experimental parameters are as follows: 50 and 30 rounds were conducted respectively for training the federated model and personalizing. The batch size and local epochs are set to 64 and 4, while the learning rate and momentum are set to 0.01 and 0.5. The number of channels used for feature extraction is 32, and the indexes of the ReLU layers involved are 0, 1, 2. In federated personalization, the final fully connected layer serves as the personalization layer for both networks.

B. Results

Principal component analysis (PCA), a dimension reduction technique, was used to visualize the clustering. In Figure 5, we plot these reduced two-dimensional points to visualize the clustering, with different colors signifying different clusters. Figure 6 depicts the process of federally training a global

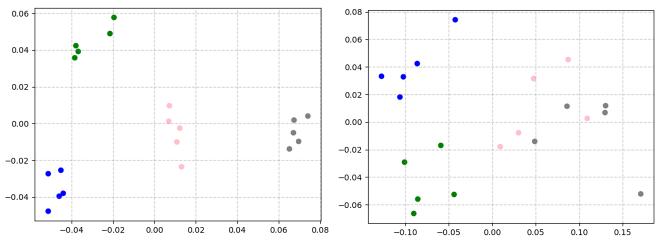

Fig. 5. Visualization of clustering on CIFAR-10(left) and CIFAR-100(right).

model on the CIFAR-100 dataset using ResNet34 and MobileNetV1. As can be observed, the heterogeneous distribution of the data causes the global model to over-fit, necessitating personalisation. It can be observed that each cluster contains 5

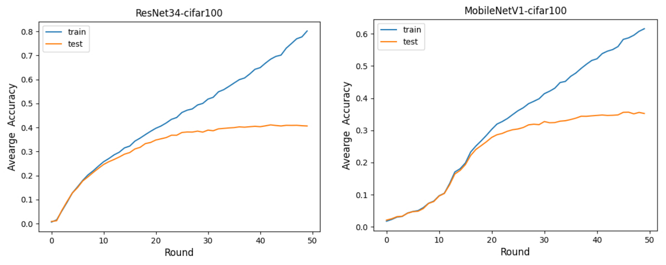

Fig. 6. The ResNet34 and MobileNetV1 training processes on CIFAR-100.

clients, and different clusters are spread in different positions in space. The clustering findings demonstrate that our approach matches our expectations.

we use the global model's test accuracy to establish a baseline for each client. The global model is then optimized using a variety of personalization methods, including finetune, PFA, and FedPerC. Fine-tuning is a popular migrating learning strategy that retrains all or part of the parameters. In this case, we refer to it as "retraining all parameters." And it represents the personalisation effect on a single device. PFA is a framework introduced in paper [7] that organizes clients first and then personalizes within groups with the FedAvg algorithm.

TABLE III
THE CLUSTER-LEVEL AVERAGE ACCURACY AND OVERALL AVERAGE ACCURACY OF RESNET34 ON CIFAR-100.

Method	cluster1	cluster2	cluster3	cluster4	Avg
BaseLine	42.50	38.67	38.56	42.04	40.44
FineTune	62.28	61.56	58.61	62.58	61.26
PFA	61.66	62.50	61.65	61.39	61.80
Ours	**65.52**	**65.00**	**62.56**	**67.56**	**65.16**

TABLE IV
THE CLUSTER-LEVEL AVERAGE ACCURACY AND OVERALL AVERAGE ACCURACY OF MOBILENETV1 ON CIFAR-100.

Method	cluster1	cluster2	cluster3	cluster4	Avg
BaseLine	34.87	38.23	35.27	37.74	36.53
FineTune	63.52	63.89	59.33	61.79	62.13
PFA	64.12	64.28	57.61	63.21	62.31
Ours	**67.15**	**67.78**	**62.61**	**66.20**	**65.94**

During the experiment, each approach is run three times, and the results of the best performing rounds are averaged as a reflection of the method's performance. Table I and Table II demonstrate experimental findings. When compared to alternative localization approaches, we can see that our approach provides a variable degree of accuracy enhancement to each client. In Table III and Table IV, we display the average accuracy within and across clusters. The findings indicate the superiority of our strategy.

TABLE V
THE IMPACT OF CLUSTERING ON RESNET34'S PERFORMANCE ON CIFAR-100.

Client Number	Random Clustering	No Clustering	Sparsity-based Clustering
client1	63.07	61.93	**65.34**
client2	60.83	63.06	**65.28**
client3	61.65	61.08	**65.06**
client4	61.11	60.28	**63.06**
client5	63.33	66.39	**68.89**
client6	60.83	59.72	**65.56**
client7	59.17	58.06	**63.89**
client8	58.61	61.39	**64.72**
client9	61.11	62.50	**65.28**
client10	62.22	61.94	**65.56**
client11	56.67	56.11	**61.39**
client12	61.11	61.67	**65.83**
client13	57.78	58.61	**61.39**
client14	54.44	60.00	**62.50**
client15	55.56	54.72	**61.67**
client16	64.17	65.28	**67.50**
client17	60.00	58.06	**63.61**
client18	61.92	63.66	**70.35**
client19	59.38	60.23	**66.19**
client20	63.07	63.35	**70.17**
Avg	60.30	60.90	**65.16**

Furthermore, to see whether clustering affects the impact of personalisation, we examined the experimental findings of ResNet34 on CIFAR-100 under three conditions: clustering, no clustering, random clustering. Table V displays detailed experimental data.

The experimental investigations indicate that random clustering has a comparable or perhaps slightly lower personalising impact than no clustering. When employing our clustering approach, both models increase their accuracy on CIFAR-100 by 5% roughly.

TABLE VI
CIFAR-10 RESULTS FOR THE BASELINE AND DIFFERENT PERSONALIZATION METHODS.

Model	Method	Avg
ResNet34	BaseLine	53.25
	FineTune	82.55
	PFA	79.45
	FedPerC	**84.48**
	FedPerC(Random Clustering)	82.55
	FedPerC(No Clustering)	81.88
MobileNetV1	BaseLine	51.28
	FineTune	83.53
	PFA	78.95
	FedPerC	**85.08**
	FedPerC(Random Clustering)	82.85
	FedPerC(No Clustering)	83.33

When the clustering is suitable, our technique is supposed to have a favorable impact since it takes into account the users' similarities and differences. However, when the clustering is inaccurate, it might be detrimental. When all clients are treated as belonging to a cluster, the result is identical to not clustering, and our technique degrades to the FedPer algorithm given in article [5].

Table VI presents the average accuracy of all previously mentioned personalisation techniques on CIFAR-10. The accuracy boost on CIFAR-10 is not as significant as on CIFAR-100, most likely because CIFAR-100 has a higher number of classes and the variances across clients within a cluster are greater. Additionally, a user often owns various classes of images in reality, so our approach has its practicalities.

V. Related Work

Federated learning has extended the range of applications for artificial intelligence (AI). In recent years, it has become a popular research topic. However, It also confronts obstacles on a variety of fronts [2], including privacy breaches and heterogeneous data distribution.

Privacy violations may cause a great deal of grief in people's lives. As a result, individuals are becoming more conscious of their right to privacy. Furthermore, the implementation of applicable rules, like the General Data Protection Regulation (GDPR) [14], not only officially protects users' privacy but also drives adjustments in machine learning algorithms that need access to users' raw data.

Secure multiparty computing (SMC) [15] is a privacy-preserving technique that based on mathematical theory. It safeguards all parties' input data while generating accurate results, and is especially beneficial in the absence of a trustworthy third party. However, SMC requires substantial computer power and network resources for encryption and decryption, which is not achievable for resource-constrained devices.

Differential privacy (DP) [16] preserves privacy by introducing noise into the data to diminish its sensitivity. Noise must be provided to make it more difficult to derive information about users without significantly affecting the distribution of data. On the other hand, DP makes the data less reliable, which has an effect on how well the training works.

As a result, when the idea of federated learning was initially proposed, it sparked a great deal of attention. Most importantly, FL enables resource-constrained devices to cooperate together to train a shared model that benefits each device without requiring access to any device's raw data. When the distribution of data for separate clients is roughly the same, it proves to be a realistic technique [1]. Unfortunately, this is not always the case. Therefore, FL has established itself as a research center dedicated to tackling the issues raised by data non-IDD.

Mansour et al. [17] introduces a personalisation method based on clustering. However, users are required to give out some raw data in return for precise clustering results, compromising data privacy. Wang et al. [3] applies transfer learning to personalisation, retraining all or part of the parameters of the federated model on local data. Another one, based on the concept of transfer learning, is provided in [18], in which meta-learning is utilized to establish the global model and then fine-tuning approaches are employed to achieve personalisation. Yu et al. [4] recommended that networks be divided into two components: base layers and personalisation layers, with the former being trained by all users collectively and the latter by each user individually.

VI. Conclusion And Future Work

In this work, we propose an algorithm that implements personalisation by clustering users based on a privacy-protected representation of their original data. This is an exploratory way to addressing the non-IID dilemma of FL, although it has certain downsides. For example, though the sparsity patterns of various inputs are often distinct, there may be exceptions that result in incorrect clustering conclusions. How to enhance the logic of clustering is a topic that deserves to be investigated more in the future.

Since it is almost impossible to derive the original data from the sparsity representation, we regard it as privacy-preserving. However, this representation's ability to protect privacy has not been formally demonstrated. Additionally, incorporating established privacy-protection mechanisms such as DP is a point of improvement. On the other hand, our method is limited to CNNs, leaving the possibility of extending it to other network architectures for future implementation.

References

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

[2] P. Kairouz et al., "Advances and open problems in federated learning," arXiv preprint arXiv:1912.04977, 2019.

[3] K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, and D. Ramage, "Federated evaluation of on-device personalization," arXiv preprint arXiv:1910.10252, 2019.

[4] T. Yu, E. Bagdasaryan, and V. Shmatikov, "Salvaging federated learning by local adaptation," arXiv preprint arXiv:2002.04758, 2020.

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

[6] A. Krizhevsky, G. Hinton, and others, "Learning multiple layers of features from tiny images," 2009.

[7] B. Liu, Y. Guo, and X. Chen, "PFA: Privacy-preserving Federated Adaptation for Effective Model Personalization," in Proceedings of the Web Conference 2021, 2021, pp. 923–934.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[9] B. Graham and L. van der Maaten, "Submanifold sparse convolutional networks," arXiv preprint arXiv:1706.01307, 2017.

[10] P. E. Danielsson, "Euclidean distance mapping," Computer Graphics and image processing, vol. 14, no. 3, pp. 227–248, 1980.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[12] A. G. Howard et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.

[13] A. Paszke et al., "Pytorch: An imperative style, high-performance deep learning library," Advances in neural information processing systems, vol. 32, pp. 8026–8037, 2019.

[14] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," A Practical Guide, 1st Ed., Cham: Springer International Publishing, vol. 10, p. 3152676, 2017.

[15] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in Annual Cryptology Conference, 2012, pp. 643–662.

[16] M. Abadi et al., "Deep learning with differential privacy," in Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, 2016, pp. 308–318.

[17] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," arXiv preprint arXiv:2002.10619, 2020.

[18] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in International Conference on Machine Learning, 2017, pp. 1126–1135.