Entrofuse: Clustered Federated Learning Through Entropy Approach

Kaifei Tu, Wenhao Yuan, and Xuehe Wang(✉)

School of Artificial Intelligence, Sun Yat-sen University, Zhuhai, China
{tukf,yuanwh7}@mail2.sysu.edu.cn, wangxuehe@mail.sysu.edu.cn

Abstract. Conventional machine learning method typically relies on collecting vast quantities of data, which usually results in serious private information leakage and a huge communication burden. To tackle this severe challenge, Federated Learning (FL), which served as a novel paradigm of distributed machine learning, is recently proposed. Under the framework of FL, clients cooperatively train a shared global model with their own data and transmit the model parameter to the central server while keeping their private data localized. However, FL still encounters some limitations, particularly in confronting non-independent and non-identically distributed (Non-IID) data which results in poor model performance. In light of the above concerns, we propose an entropy-based clustering federated learning model named **Entrofuse**, which aims to partition the clients into different clusters characterized by data distribution and subsequently, the model training process performs within each cluster. As it is hard to acquire the distribution of data samples, we adopt Kernel Density Estimation (KDE) method to estimate the data distribution of heterogeneous clients. Our approach takes into account both entropy and vector angle of the model parameter, and proves the rationality of our method through rigorous theoretical analysis. Experimental results show that our proposed method is superior to the non-clustered case on the EMNIST dataset and significantly improves the accuracy by 10% to 12%.

Keywords: federated learning · cluster · entropy

1 Introduction

Traditional machine learning models typically require data to be centralized-stored for training purposes, which raises concerns about the potential risk of privacy leakage [11,13]. In contrast, federated learning adopts a decentralized approach by conducting model training on mobile devices, ensuring that data remains localized and safeguarding data privacy [10,17]. Moreover, federated learning facilitates cross-institutional data sharing, thereby enhancing the efficiency of data utilization [7,14,15,18]. Therefore, FL holds significant potential for various applications in the realms of data privacy protection and data sharing, rendering it a promising and fertile area for further research.

V. C. M. Leung et al. (Eds.): Qshine 2023, LNICST 574, pp. 79–94, 2024.
https://doi.org/10.1007/978-3-031-65123-6_6

Nevertheless, it is important to acknowledge that federated learning is not without its limitations [13]. One significant challenge arises from the fact that the data sources of federated learning users are commonly Non-IID. Heterogeneous datasets may inject varying influences on the FL model, and certain data samples may even have a detrimental effect on the performance. Consequently, training a robust model under such circumstances can prove to be a formidable task.

In order to address aforementioned challenge, numerous researchers have proposed clustering methods aimed to classify clients with similar data distribution for better model performance [2]. Presently, clustering methods in the context of federated learning typically involve evaluating the similarity between data from different clients through various methods, such as geographical location, device type, or data distribution [5,20]. In order to achieve better training results, classification based on the data distribution usually serves as a better entry point [6,11,13], based on which, data instances exhibiting high similarity are integrated into clusters and then, each cluster is trained as a cohesive unit in the federated learning process.

In recent literature, some scholars have used cosine similarity to classify users [20], while others may use Euclidean distance. However, these methods have certain limitations. The cosine-based clustering method only considers the direction of the data, but it is obvious that the length of the data is also very important. The Euclidean distance-based clustering method only considers the position of the data in space, and ignores other attributes of the data itself, e.g., the direction and the length. In this article, we use an improved entropy-based clustering method to partition users under the federated learning framework [1,22], which effectively addresses the shortcomings of the previous works. The main contributions of this paper are summarized as follows:

- *Generalized entropy approach:* To our best knowledge, this paper is the first work studying clustering federated learning via the generalized entropy method. We define a generalized entropy term, and initialize each cluster according to the K-means algorithm, based on which, each client is allocated to the corresponding cluster with the highest entropy.
- *Rationality guarantee for clustering:* For our clustering method, we verify its rationality theoretically: First, we show that the value of entropy is maximized when each probability is equal, and then prove that distribution of the gradient of the loss function for the IID dataset is also IID, which indicates that the generalized entropy can be maximized in the case of IID dataset, thus our algorithm can cluster the data of IID together, which shows the rationality of our algorithm.
- *Experiment verification:* Compared with the case without clustering, our clustering method has a significant improvement in accuracy. Furthermore, we conduct experiments to show the impact of client numbers on the model accuracy under different hyperparameters.

1.1 Related Work

In this section, we discuss the related work in regard to clustered federated learning.

To tackle the negative influence brought by the Non-IID data distribution, e.g., low accuracy and convergence rate, clustered federated learning has been proposed recently. Jia *et al.* [6] propose differential privacy-based algorithms for clustered federated learning, which protects participants' privacy by adding artificial noise. Zhao *et al.* [24] introduce a framework that shares a small subset of public data with heterogeneous clients to reduce the weight divergence between trained local models, thus increasing robustness and stability during training. A semi-cyclic method [3] is proposed to train pluralistic models that perform model averaging over blocks of clients, in which, clients are clustered based on where they are located in a shared timezone. However, this method requires extra knowledge (i.e., shared timezone) about the clients. Ghosh *et al.* [5] adopt the one-time clustering and non-shared clustering method. The central server runs the K-means algorithm to cluster the participants, which heavily relies on the selection of initial points. [21] investigates the application of clustered federated learning to byzantine settings, where a subset of clients behaves unpredictably or tries to disturb the joint training effort in an unpredictable way. [20] employs iterative clustering and non-shared clusters, which conduct cluster division based on the parameter similarity of nodes in each iteration. Ghosh *et al.* [4] propose iterative clustering and inter-cluster sharing. In each iteration, the nodes are re-estimated for their cluster membership by leveraging the average parameters of their respective clusters. Whereas, the convergence rate of algorithms [4,20] is relatively slow, and lacks strict theoretical proof for the rationality of clustering. [9] proposes a three-phased data clustering algorithm, named generative adversarial network-based clustering, cluster calibration, and cluster division to overcome these three limitations of general federated learning: the risk of data privacy breach, the fixed shape of clusters, and the non-adaptive number of clusters. However, the convergence rate of this algorithm is relatively slow. To capture the complex nature of real-world data, soft clustering methods [12] with overlapping clusters have been proposed that attain superior performance over hard ones.

Noted that most of the above methods face certain limitations, such as slow convergence rate [4,9,20], low accuracy [24], and lack of analysis for the rationality of clustering methods [9]. In our article, we introduce a new algorithm and clustering method, which is proven to be rational for clustering and has good performance on the dataset EMNIST. The remainder of this paper is organized as follows: Sect. 2 introduces our model Entrofuse, then describes the initialization method of the clusters and the entire clustering process is also stated in this section. Section 3 proves the rationality of our clustering method, that is, our clustering method can gather clients with similar data distribution. Section 4 shows the simulation results. Finally, Sect. 5 summarizes this paper.

2 System Model and Problem Formulation

Federated learning has certain requirements for data quality and is more like to achieve good results on independent and identically distributed (IID) data. Integrating heterogeneous clients with similar data distributions will highly improve the performance of FL model. For the past few years, plenty of researchers have explored clustered federated learning in academia. In this paper, we will introduce a clustered method that measures the degree of proximity of data distributions based on entropy.

2.1 Federated Learning

Federated learning is a decentralized machine learning mechanism where a group of distributed local clients collaboratively train a globally shared model. Each client performs one step or multiple steps of mini-batch stochastic gradient descent (SGD) in each global iteration to update the model parameters. After aggregating all local parameter from each client, the central server updates the global model and send it back to all participants for the next round of local training.

Without loss of generality, we assume that there are n clients who are willing to participate in FL model training process and client $i \in \{1, 2, \ldots, n\}$ utilizes its local dataset D_i with datasize $|D_i|$.

For a batch data $\{x_i^j, y_i^j\}_{j=1}^{|D_i|}$ of client i, the FL model aims to find the optimal model parameter w which is able to predict the label output $\hat{y}_i^j$ based on the input x_i^j with the loss function $f_i^j(w, x)$. Then the loss function of client i on his dataset D_i is:

$$F_i(\omega(t)) = \frac{1}{|D_i|} \sum_{j \in D_i} f_i^j(\omega(t), x_i^j). \tag{1}$$

At global iteration $t + 1 \in \{1, 2, ..., T\}$, client i updates the local parameter according to their own dataset D_i based on last global parameter $\omega(t)$ sent by the central server:

$$\omega_i(t + 1) = \omega(t) - \eta \nabla F_i(\omega(t)), \tag{2}$$

where η is the learning rate and $F_i(\cdot)$ is the local loss function. The central server updates the global model by weighting and summing the local parameters from all clients:

$$\omega(t + 1) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} \omega_i(t + 1), \tag{3}$$

where $D = \sum_{i=1}^{n} D_i$. Updated global model $\omega(t + 1)$ will be transmitted back to each client for a new iteration of global training.

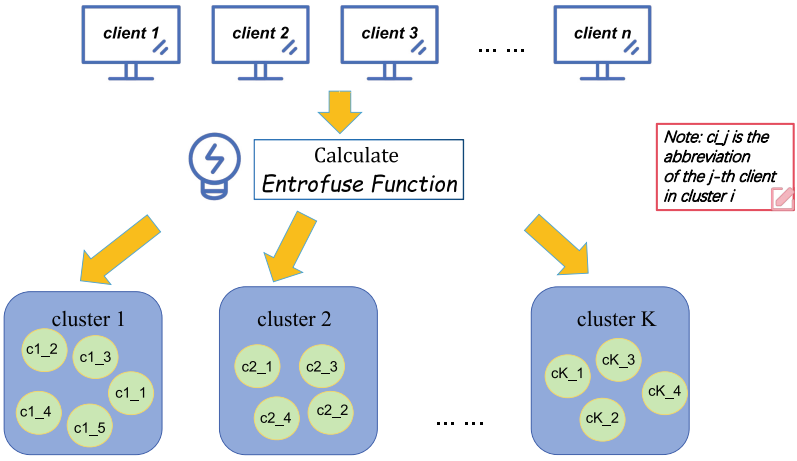

Fig. 1. Entrofuse

2.2 Entropy

Entropy is a statistical metric for quantifying the chaos of a system, which is first proposed by Shannon [23]. Due to the negative impact of chaotic data on training results, it is desirable to integrate data that causes a small increase in system entropy. In general, for a set of points $x = \{x_1, x_2, \ldots, x_n\}$ that follow a probability distribution $p(\cdot)$, the entropy of the system x is given by (Fig. 1):

$$Entropy(x) = -\sum_{i=1}^{n} p(x_i) \log(p(x_i)). \tag{4}$$

In t-th global training iteration, based on the local parameter transmitted to the central server, we can evaluate the similarity S_{ij} between any clients i and j by comparing the gradient of the loss function with respect to model parameters as follows:

$$S_{i,j} = S(\nabla F_i(\omega(t)), \nabla F_j(\omega(t))). \tag{5}$$

As the model parameters may take the form with various dimensions, we first transform the gradient $\nabla F_i(\omega(t))$ of client i in t-th iteration into vectors of size $1 \times n_0$ for the purpose of convenience in processing, where n_0 is the total number of parameters of machine learning model.

Since *Entropy* reaches its maximum value when $p(x_i)$ is approximate to each other according to Eq. (4), we leverage the entropy function to estimate the similarity of the data distribution of clients:

$$E = -\sum_{i=1}^{n} p(\nabla F_i(\omega(t)) \log(\nabla F_i(\omega(t))), \tag{6}$$

where function p is defined as:

$$p(\nabla f_i(\omega(t))) = \frac{\nabla F_i(\omega(t))^T \nabla F_i(\omega(t))}{\sum_j \nabla F_j(\omega(t))^T \nabla F_j(\omega(t))}. \tag{7}$$

When the data distribution of clients is similar, the gradient of model parameters is also close. Consequently, the value of p in Eq. (7) will be relatively close, leading to a relatively large value of entropy.

2.3 Generalized Entropy Function

In the previous calculations, p in Eq. (7) only consider the magnitude of the vectors, which means that when a series of points have similar magnitude, the entropy Eq. (6) will reach a relatively high value. However, the magnitude of the vectors cannot deliver all the information of the vectors, and the direction is also crucial. Therefore, we introduce the angle deviation between vectors as another factor needed to be considered:

$$\cos(\theta_{ij}) = \frac{\langle \nabla F_i(\omega(t)), \nabla F_j(\omega(t)) \rangle}{|\nabla F_i(\omega(t))| \, |\nabla F_j(\omega(t))|}. \tag{8}$$

Cluster i has a series of clients $C_i = \{i_1, i_2, \cdots i_{i0}\}$ as well as the corresponding parameter gradients $\{\nabla F_{i_1}(\omega(t)), \nabla F_{i_2}(\omega(t)), \ldots, \nabla F_{i0}(\omega(t))\}$. The similarity between a client q to be clustered and a cluster class C_i is defined as follows:

$$El = -\alpha \sum_{j \in C_i \cup q} p(\nabla F_j(\omega(t)) \log(\nabla F_j(\omega(t))) + (1-\alpha) \frac{\langle \nabla F_q(\omega(t)), x_c(t) \rangle}{|\nabla F_q(\omega(t))| \cdot |x_c(t)|}, \tag{9}$$

where $\alpha \in [0, 1]$, x_c is defined as the average of the gradient of the loss function in cluster C_i:

$$x_{C_i}(t) = \frac{1}{|C_i|} \sum_{j \in \text{class} C_i} \nabla F_j(\omega(t)). \tag{10}$$

For the convenience of the description, we refer to Eq. (9) as the Entrofuse Function, based on which, each client is assigned to the cluster corresponding to the maximum value in this equation.

2.4 Cluster Centroid Initialization

We initialize and update the centroids of the clients based on the gradient of loss function of each client $C_c = \{\nabla F_1(\omega(t)), \nabla F_2(\omega(t)), \ldots, \nabla F_n(\omega(t))\}$.

As shown in Algorithm 1, the first client in the first cluster $cent_1$ is chosen randomly from the gradient of all clients. For $i \in \{2, 3, \ldots, K\}$, we calculate the entropy between all the clients unclustered in C_c and the clients already clustered $C_a = \{\sigma_1, \sigma_2, \ldots, \sigma_{i-1}\}$. For any $x \in C_c$ the distance between x and the cluster C_α is defined as:

Algorithm 1. Centroid Selection for Each Cluster

1: Acquire the gradients of the loss function: $C_c = \{\nabla F_1(\omega(t)), \nabla F_2(\omega(t)), \nabla F_3(\omega(t)), \ldots, \nabla F_n(\omega(t))\}$
2: Choose the first client $\nabla F_1(\omega(t))$ as the first point σ of the first cluster C_1,
3: **for** $i \in \{2, 3, \ldots, K\}$ **do**
4: **for** $x \in \{\nabla F_1(\omega(t)), \nabla F_2(\omega(t)), \nabla F_3(\omega(t)), \ldots, \nabla F_n(\omega(t))\}$ **do**
5: Compute

$$E(x) = \sum_{x_j \in \text{cluster} i \cup x} p(x_j) \log(p(x_j))$$

6: **end for**
7: Choose the point which has the minimum entropy $E(x)$ with the current centroids:

$$\sigma_i = \arg\min_{x \in C_c} E(x)$$

8: Delete σ_i from C_c
9: **end for**
10: Output centroids $\{\sigma_1, \sigma_2, \ldots, \sigma_K\}$

$$E(x) = -\sum_{y \in C_a \cup x} p(y) \log(p(y)). \tag{11}$$

The unclustered client which has the minimum entropy with the centroids would be selected as the first client in a new cluster:

$$C_i = \arg\max_{x \in C_c} E(x) \tag{12}$$

Repeating the above procedure until K centroids are selected. After obtaining a series of centroids $\{\sigma_1, \sigma_2, \ldots, \sigma_K\}$, we treat them as the first client in each cluster, and utilize Eq. (4) to classify the clients who have not yet been clustered.

2.5 Threshold-Based Strategy for Clustering

Traditional federated learning models perform well on independently and identically distributed (IID) data, while training on Non-IID data, catastrophic results may occur. Therefore, we propose a clustered federated learning approach, which clusters clients with similar data distribution and trains each cluster separately to achieve better results. Note that for clients with similar data distribution, the clustering procedure may result in unnecessary and additional resource waste. Thus, we propose a threshold-based strategy that judges the data distribution and only applies clustering operation to clients with sufficiently heterogeneous data distribution. For any $x \in G(t) = \{\nabla F_1(\omega(t)), \nabla F_2(\omega(t)), \ldots, \nabla F_N(\omega(t))\}$, we have:

$$p(x) = \frac{\nabla x^T \nabla x}{\sum_j \nabla x^T \nabla x}. \tag{13}$$

Algorithm 2. Client Clustering

Note unclustered clients as $C_u = \{\text{client}_1, \text{client}_2, ..., \text{client}_n\}$
Initialize the clusters $C_1, C_2, ..., C_K$ by Algorithm 1
Delete the clients chosen as centroids from set C_u
for iteration $t \in \{1, 2, ..., T\}$ **do**
  compute $\delta = -\sum_{i=1}^{n} p(\nabla F_i(\omega(t)) \log(p(\nabla F_i(\omega(t)))$
  if $\delta \leq \varepsilon$ **then**
    Break and start to cluster
  end if
end for
for client$i \in C_u$ **do**
  Note the gradient of loss function $g_i = \nabla(F_i(\omega(t)))$
  for cluster j **do**
    The number of clients in cluster j is m_j
    Note the gradients of loss functions in cluster j as G_j
    Compute $x_c(t) = \frac{1}{m_j} \sum_{x \in G_j} x$
    Compute $El_i = -\alpha \sum_{x \in G_j \cup g_i} p(x) \log(p(x)) + (1 - \alpha) \frac{\langle g_i, x_c \rangle}{|g_i||x_c|}$
  end for
  Put client$i$ into the cluster where El_i reaches its maximum value
  Delete client$i$ from set C_u
end for

Then we can calculate the entropy of the whole system:

$$\delta = -\sum_{i=1}^{n} p(\nabla F_i(\omega(t))) \log(p(\nabla F_i(\omega(t)))). \tag{14}$$

Smaller entropy means greater difference in the data distributions between the clients, highlighting the necessity of clustering clients with similar data distributions to expedite the convergent rate of the model. The threshold-based clustering is triggered when the following criterion is met:

$$\delta \leq \varepsilon, \tag{15}$$

where ε is a preset hyperparameter. With threshold-based strategy, our clustering algorithm becomes integral and is described in detail in Algorithm 2.

3 Theoretical Analysis

3.1 Preliminary

Lemma 1. *Entropy* $E = -\sum_{i=1}^{n} p_i \log(p_i)$ *reaches its maximum when* $p_i, i \in \{1, ..., n\}$ *are equal, i.e.,* $p_i = \frac{1}{n}$ *for all* $i \in \{1, 2, \ldots, n\}$.

Proof. The maximize problem of entropy $H(p_1, p_2, \ldots, p_n) = -\sum(p_i \log(p_i))$ is subject to the constraint $\sum p_i = 1$, whose optimal solution can be obtained by

applying the Lagrange multiplier method. We define the Lagrangian function [19] L as follows:

$$L = -\sum_{i=1}^{n}(p_i \log(p_i)) + \lambda\left(\sum_{i=1}^{n} p_i - 1\right), \tag{16}$$

where λ is the Lagrange multiplier.

By taking the first derivative of Eq. (16) with respect to each p_i, $i \in \{1, 2, \ldots, n\}$, we have:

$$\frac{\partial L}{\partial p_i} = -\log(p_i) - 1 + \lambda = 0. \tag{17}$$

Then, we get the expression for p_i as follows:

$$p_i = e^{\lambda - 1}. \tag{18}$$

Since the sum of all probabilities is equal to 1, we have:

$$\sum_{i=1}^{n} p_i = n e^{\lambda - 1} = 1. \tag{19}$$

Thus, we easily get the expression of λ with respect to the number of probabilities as follows:

$$\lambda = 1 - \log(n). \tag{20}$$

By inserting the above expression of λ in Eq. (20) into the expression of p_i in Eq. (18), we have:

$$p_i = e^{-\log(n)} = \frac{1}{n}. \tag{21}$$

By checking the second-order derivative of the Lagrangian function:

$$\frac{\partial^2 L}{\partial p_i^2} = -\frac{1}{p_i} < 0. \tag{22}$$

According to Eq. (17–22), the maximization problem achieves the optimal solution when $p_i, i \in \{1, 2, \cdots, n\}$ are equal, i.e., $p_i = \frac{1}{n}$ for all i. □

3.2 Rationality Analysis

Karimireddy et al. [8] have proved that training on the heterogeneous dataset (i.e., Non-IID dataset) will slow down the convergent rate even making the model fail to converge. In the following, we provide a theoretical and rigorous analysis to demonstrate that the Entrofuse Function in Eq. (9) reaches its maximum value with the IID dataset. Therefore, we can effectively partition clients into clusters and improve the model performance significantly.

Lemma 2. *If the datasets of n clients satisfy IID, the gradients of the loss functions of client i, denoted as $\nabla F_i(\omega(t))$, can also be approximated as IID.*

Proof. Client i participates in FL and utilizes its local dataset $D_i = \{x_i^j, y_i^j\}_{j=1}^{|D_i|}$, where $x_1, x_2, \cdots, x_{|D_i|}$ is the input sample and $y_1, y_2, \cdots, y_{|D_i|}$ is the corresponding label. Then, the loss function can be defined as follows based on Eq. (1):

$$F_i(\omega(t), D_i) = \frac{1}{|D_i|} \sum_{j=1}^{|D_i|} f(w(t), x_i^j), \tag{23}$$

Then, $\nabla F_i(\omega(t))$ is utilized to update the parameters $\omega(t)$, and the update rule can be expressed as:

$$\omega_i(t+1) = \omega(t) - \eta \nabla F_i(\omega(t)), \tag{24}$$

where η is the learning rate.

According to the Central Limit Theorem and Eq. (23), for a sequence of IID random variables, the distribution of their means $F_i(\omega(t))$ converges to a normal distribution:

$$F_i(\omega(t)) \sim N(\mu_i, Var_i^2). \tag{25}$$

Since the dataset of client i and j is IID, and $\xi_i \in D_i$ and $\xi_j \in D_j$ share the same data distribution, we have:

$$E(\xi_i) = E(\xi_j), Var(\xi_i) = Var(\xi_j). \tag{26}$$

Then, based on Eq. (23), we also have:

$$\mu_i = \mu_j, Var_i = Var_j, \text{for any } i, j \in \{1, 2, \cdots, n\}. \tag{27}$$

Thus, it can be derived that $F_i(\omega(t))$ and $F_j(\omega(t))$ are IID respectively, as well as $\nabla F_i(\omega(t))$ and $\nabla F_j(\omega(t))$. Therefore, if the dataset satisfies the assumption of IID samples, the gradient of loss function of each client $\nabla F_i(\omega(t))$ can also be approximated as IID. □

In the following, we show that the Entrofuse Function Eq. (9) reaches its maximum value for the IID dataset. Given IID datasets $\{D_1, D_2, \ldots, D_\zeta\}$, the gradients of the loss function is denoted as $\{\nabla F_1(\omega), \nabla F_2(\omega), \ldots, \nabla F_\zeta(\omega)\}$. According to Lemma 2, the gradients of the loss functions of the clients can be approximated as IID. Denote p_i as:

$$p_i = \frac{\nabla F_i(\omega(t))^T \nabla F_i(\omega(t))}{\sum_{j=1}^{\zeta} \nabla F_j(\omega(t))^T \nabla F_j(\omega(t))}, i \in \{1, 2, \cdots, \zeta\} \tag{28}$$

Regarding $p_i, i \in \{1, 2, \ldots, \zeta - 1\}$ as the points that are already clustered and p_ζ as the point to be clustered. Then, the average of the points in the cluster can be denoted by $p_c = \frac{1}{\zeta - 1} \sum_{i=1}^{\zeta - 1} p_i$. The generalized entropy is defined as:

$$El(p_\zeta) = -\alpha \sum_{i=1}^{\zeta} p_i \log(p_i) + (1 - \alpha) \frac{\langle p_\zeta, p_c \rangle}{|p_\zeta| \cdot |p_c|}, \tag{29}$$

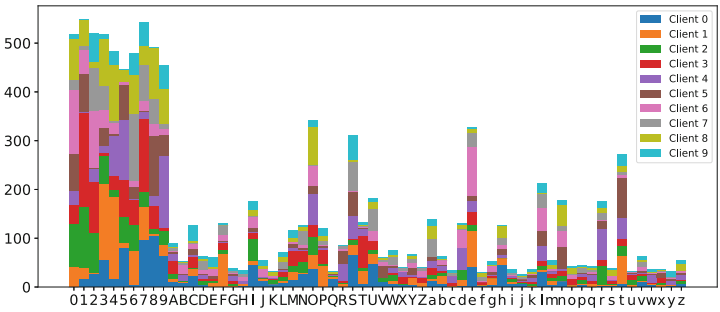

Fig. 2. The distribution of the dataset

To prove our statement, we need to demonstrate that when all probabilities $p_1, p_2, \ldots, p_\zeta$ are equal, the generalized entropy El reaches maximum. According to Lemma 1, El reaches its maximum when $p_1 = p_2 = \ldots = p_n = p$. By substituting these values into the entropy formula, we have:

$$\begin{aligned} El(p_\zeta) &= -\alpha \sum_{i=1}^{\zeta} p_i \log(p_i) + (1-\alpha) \frac{\langle p_\zeta, p_c \rangle}{|p_\zeta| \cdot |p_c|} \\ &= -\alpha \cdot \zeta \cdot p \log(p) + (1-\alpha) \frac{\langle p, p \rangle}{|p| \cdot |p|} \\ &= -\alpha \cdot \zeta \cdot \frac{1}{\zeta} \log(\frac{1}{\zeta}) + 1 - \alpha \\ &= \alpha \cdot \log(\zeta) + 1 - \alpha \\ &= El_0. \end{aligned} \tag{30}$$

El_0 is the maximum of Eq. (9) when there are $\zeta - 1$ clients in a cluster and a new client is to be clustered. So we have demonstrated our Entrofuse Function (9) takes on larger values when the data is closer to being independently and identically distributed (IID), and smaller values when the data is more dispersed. Therefore, our clustering approach effectively groups user data that satisfies the IID assumption into a cluster.

4 Experiments

In this section, we conduct experiments on the EMNIST dataset to evaluate the performance of our proposed framework. First of all, we introduce the model setup. Then we compare the performance of our clustered federated learning method with traditional FL algorithms (FedAVG) [16] in terms of model accuracy and convergence rate. Note that traditional federated learning without clustering is FedAVG by default. Finally, we verify the impact of the weight α on the performance of the FL model.

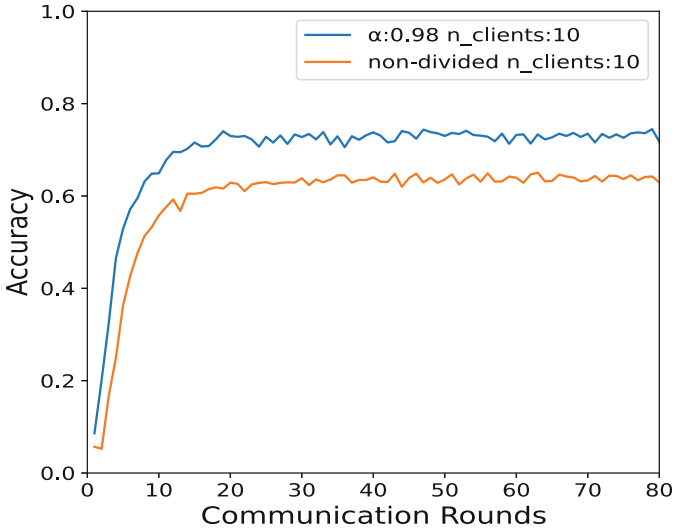

Fig. 3. The performance of our model compared with traditional FL method

4.1 Experiment Setup

The experimental setup is given as follows:

- **Dataset:** We use the EMNIST dataset, which is divided into training sets and test sets. We divide 10,000 images in the training set and 10,000 images in the test set. Defaultly, the images in the EMNIST dataset are 28×28 pixels, with a total of 62 categories.
- **Hyperparameters:** We set learning rate $\eta = 0.1$, the weight $\alpha = 0.98$ and communication round equals to 80 in total. During each global training iteration, the batch size is 128. We first consider 10 clients and divide the Non-IID dataset according to Dirichlet distribution, where $DIRICHLET\_ALPHA = 1.0$. The data distribution is shown in Fig. 2. In our experiments, we will also adjust client number n and allocate data based on the Dirichlet distribution.
- **Model:** For the image classification task, the model for EMNIST is a convolutional neural network (CNN) model that consisted of two 5×5 convolution layers (6 and 16 output channels, each of which is activated by batch normalization and ReLU, followed by 2×2 max pooling convolution layer), one fully-connected layer with 256 units and a ReLU output layer with 62 units.

4.2 Experimental Results

We conduct extensive experiments on the EMNIST dataset. Compared with the traditional federated learning method (FedAVG), our model achieves better performance as shown in Fig. 3. In particular, we obtain almost 10% improvement

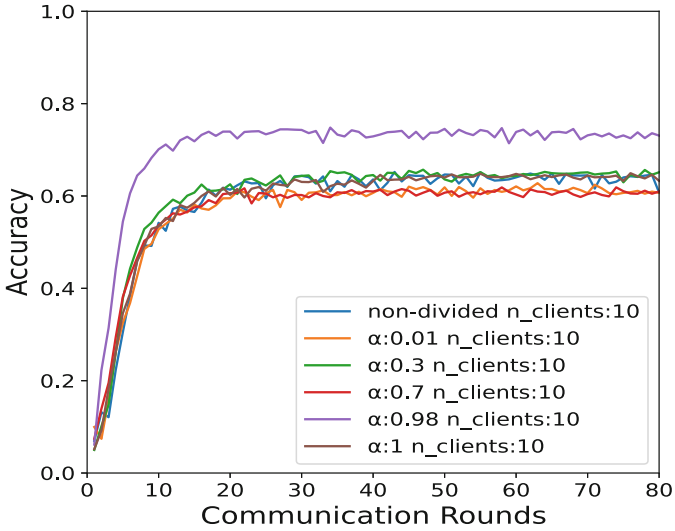

Fig. 4. The impact of the selection for α on accuracy

in accuracy compared with the non-divided situation (i.e., FedAVG) on the Non-IID data distribution. In the following, we further evaluate the effect of weight α and client number on the model accuracy.

Effect of Weight α: Given that different weight α implies the preference between the magnitude and angle deviation of the data, we randomly select the values of $\alpha \in [0, 1]$ and compare the results to determine the optimal value that yields the best performance for our method.

Our model reaches the highest accuracy with $\alpha = 0.98$, which makes significant improvement on accuracy for 10% to 12%. Besides, it is worth noting that the accuracy of the global model is relatively low on some value of α, even worse than that of a non-divided situation, as shown in Fig. 4. Specifically, when considering α as 1, the model accuracy is not optimal, suggesting the need for a combined evaluation of both magnitude and angle deviations in the data to attain improved performance.

Effect of Client Number: The previous experiments consider the scenario with a fixed client number, i.e., $n = 10$. In this section, we demonstrate the generalization ability of our proposed method by investigating the performance with different weights α and the number of clients on the EMNIST dataset.

For the case when the clients are not divided, it is shown in Fig. 5(a) that with the increase of the client number, the accuracy of FedAVG has a slight improvement, which is generally stable at around 60%. When $\alpha = 0.01$ and $\alpha = 0.3$ as shown in Fig. 5(b) and Fig. 5(c) respectively, the accuracy doesn't improve very well, or even worse than that of FedAVG in some cases. That is to say, the entropy term plays a crucial role in client clustering. When the value

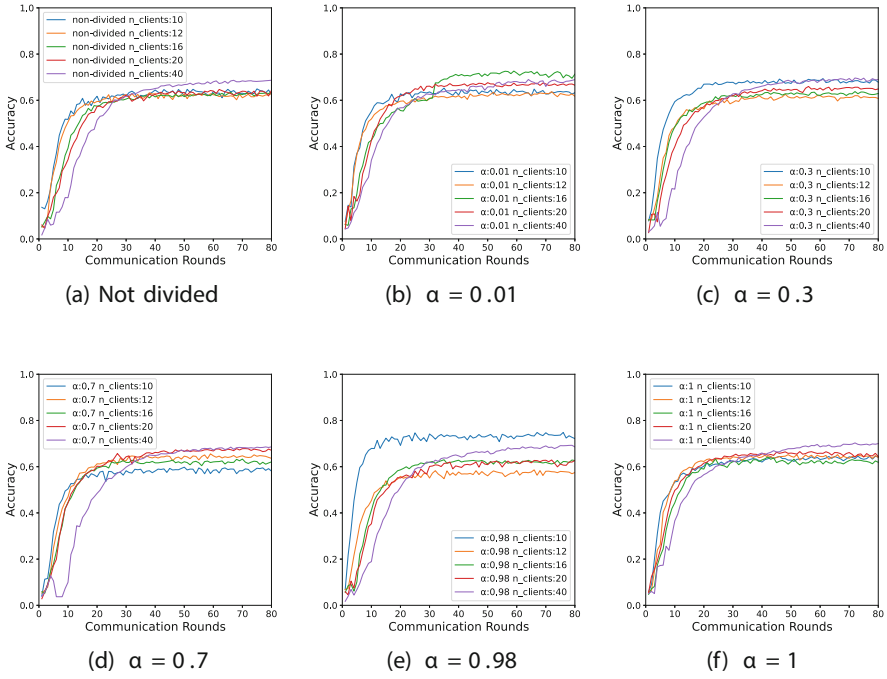

Fig. 5. The performance of our model with different numbers of clients

of weight α is 0.7 as shown in Fig. 5(d), we find that with the increment of client number, the accuracy improves. When $\alpha = 0.98$ in Fig. 5(e), the model achieves the best performance with client number $n = 10$. While in the last set of experiments as shown in Fig. 5(f), when the angle deviation is ignored, the model accuracy drops, which indicates that both entropy and angle deviations play important roles in client clustering for improving the model accuracy.

5 Conclusion

In this paper, we have designed Entrofuse, a clustered federated learning framework based on entropy. We cluster each client using the Entrofuse Function, which takes into account both the modulus and direction of the model parameter, and groups clients with similar data distribution into the same cluster. According to rigorous mathematical analysis, it is proved that our clustering method can divide clients with IID data into a cluster and distinguish the clients with different data distributions. We have also conducted several groups of contrast experiments on the dataset EMNIST and the results demonstrate the efficiency of Entrofuse, which shows significant improvement in accuracy compared with the baseline algorithm FedAVG.

References

1. Bein, B.: Entropy. Best Pract. Res. Clin. Anaesthesiol. **20**(1), 101–109 (2006)
2. Briggs, C., Fan, Z., 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)
3. Eichner, H., Koren, T., McMahan, B., Srebro, N., Talwar, K.: Semi-cyclic stochastic gradient descent. In: International Conference on Machine Learning, pp. 1764–1773. PMLR (2019)
4. Ghosh, A., Chung, J., Yin, D., Ramchandran, K.: An efficient framework for clustered federated learning. Adv. Neural. Inf. Process. Syst. **33**, 19586–19597 (2020)
5. Ghosh, A., Hong, J., Yin, D., Ramchandran, K.: Robust federated learning in a heterogeneous environment. arXiv preprint arXiv:1906.06629 (2019)
6. Jia, B., Zhang, X., Liu, J., Zhang, Y., Huang, K., Liang, Y.: Blockchain-enabled federated learning data protection aggregation scheme with differential privacy and homomorphic encryption in iiot. IEEE Trans. Industr. Inf. **18**(6), 4049–4058 (2021)
7. Kairouz, P., et al.: Advances and open problems in federated learning. Found. Trends® Mach. Learn. **14**(1–2), 1–210 (2021)
8. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T.: SCAFFOLD: stochastic controlled averaging for federated learning. In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 5132–5143. PMLR (2020). https://proceedings.mlr.press/v119/karimireddy20a.html
9. Kim, Y., Hakim, E.A., Haraldson, J., Eriksson, H., da Silva, J.M.B., Fischione, C.: Dynamic clustering in federated learning. In: ICC 2021 - IEEE International Conference on Communications, pp. 1–6 (2021). https://doi.org/10.1109/ICC42927.2021.9500877
10. Konečnỳ, J., McMahan, H.B., Ramage, D., Richtárik, P.: Federated optimization: Distributed machine learning for on-device intelligence. arXiv preprint arXiv:1610.02527 (2016)
11. Konečnỳ, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D.: Federated learning: Strategies for improving communication efficiency. arXiv preprint arXiv:1610.05492 (2016)
12. Li, C., Li, G., Varshney, P.K.: Federated learning with soft clustering. IEEE Internet Things J. **9**(10), 7773–7782 (2022). https://doi.org/10.1109/JIOT.2021.3113927
13. Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated learning: challenges, methods, and future directions. IEEE Signal Process. Mag. **37**(3), 50–60 (2020)
14. Liu, S., Pan, S.J., Ho, Q.: Distributed multi-task relationship learning. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 937–946 (2017)
15. McDonald, R., Hall, K., Mann, G.: Distributed training strategies for the structured perceptron. In: Human Language Technologies: The 2010 Annual Conference of the North American chapter of the Association for Computational Linguistics, pp. 456–464 (2010)
16. McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)
17. Mohri, M., Sivek, G., Suresh, A.T.: Agnostic federated learning. In: International Conference on Machine Learning, pp. 4615–4625. PMLR (2019)

18. Povey, D., Zhang, X., Khudanpur, S.: Parallel training of DNNs with natural gradient and parameter averaging. arXiv preprint arXiv:1410.7455 (2014)
19. Powell, M.J.: Algorithms for nonlinear constraints that use lagrangian functions. Math. Program. **14**, 224–248 (1978)
20. Sattler, F., Müller, K.R., Samek, W.: Clustered federated learning: model-agnostic distributed multitask optimization under privacy constraints. IEEE Trans. Neural Netw. Learn. Syst. **32**(8), 3710–3722 (2020)
21. Sattler, F., Müller, K.R., Wiegand, T., Samek, W.: On the byzantine robustness of clustered federated learning. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8861–8865. IEEE (2020)
22. Srednicki, M.: Entropy and area. Phys. Rev. Lett. **71**(5), 666 (1993)
23. Wehrl, A.: General properties of entropy. Rev. Mod. Phys. **50**(2), 221 (1978)
24. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-IID data. arXiv preprint arXiv:1806.00582 (2018)