FedECCR: Federated Learning Method with Encoding Comparison and Classification Rectification

Yan Zeng[1,2,3], Hui Zheng[1], Xin Wang[1], Beibei Zhang[4], Mingyao Zhou[5(✉)], Jilin Zhang[1,2,3], and YongJian Ren[1,2,3]

[1] School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China
{yz,222050267,wangxin,jilin.zhang,yongjian.ren}@hdu.edu.cn
[2] Key Laboratory for Modeling and Simulation of Complex Systems, Ministry of Education, Hangzhou 310018, China
[3] Data Security Governance Zhejiang Engineering Research Center, Hangzhou 310018, China
[4] Zhejiang Lab, Hangzhou 311100, China
[5] Hangzhou Huawei Communication Technology Co., Ltd., Hangzhou 310052, China
zhoumingyao@huaewei.com

Abstract. Federated learning is a distributed training method that integrates multi-party data information using privacy-preserving technologies through dispersed client data sets to jointly construct a global model under the coordination of a central server. However, in practical applications, there is a high degree of data distribution skewness among clients, which causes the optimization direction of the client models to diverge, resulting in model bias and reducing the accuracy of the global model. Existing methods require the calculation and transmission of much information to correct the optimization direction of the client models, or only roughly limit the deviation of the client models end-to-end, ignoring targeted processing of the internal structure of the model, resulting in unclear improvement effects. To address these problems, we propose a federated optimization algorithm FedECCR based on encoding contrast and classification correction. This algorithm divides the model into an encoder and a classifier. It utilizes prototype contrastive training of the model encoder and unbiased classification correction of the classifier. This approach notably improves the accuracy of the global model while maintaining low communication costs. We conducted experiments on multiple data sets to evaluate the validity of our method, and the quantified results showed that FedECCR can improve the global model classification accuracy by approximately 1% to 6% compared to FedAvg, FedProx, and MOON.

Keywords: Federated Learning · Data Heterogeneity · Prototypical Learning · Contrastive Learning

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2024
Published by Springer Nature Switzerland AG 2024. All Rights Reserved
H. Gao et al. (Eds.): CollaborateCom 2023, LNICST 563, pp. 59–78, 2024.
https://doi.org/10.1007/978-3-031-54531-3_4

1 Introduction

In recent years, 5G technology has experienced rapid development, digital trends have accelerated, and the significant increase in internet data transmission speed and massive growth of terminal devices have led to the generation and storage of vast amounts of data, providing tremendous opportunities for the development and utilization of big data. Intelligent collaborative computing technology has begun to combine with artificial intelligence, enabling it to have more extensive and in-depth application prospects in cross-domain and cross-organizational collaboration and innovation in areas such as healthcare, intelligent manufacturing, and intelligent logistics. This leads to more efficient and accurate decision-making and improved production efficiency. Intelligent collaborative computing [1] is a method achieving collaboration and interaction between multiple computer systems with artificial intelligence technology, to process data efficiently.

Federated Learning [2,3] is a kind of intelligent collaborative technology that enables training a global model with the cooperation of multiple computer systems, which can process data efficiently and in privacy. It is a client-server architecture machine learning technology. Where the server aggregates the local model updated by clients to generate a global model or update the global model, and sends the updated global model to clients, and each client trains a local model with the local dataset and the received global model. Benefiting from its excellent privacy protection capabilities, federated learning has been widely applied, such as Google's use of federated learning technology in the GB Board mobile keyboard [4]. Apple also used federated learning technology in the QuickType keyboard in iOS13 [5]. Even though federated learning has been widely used for large-scale user data analysis, it still faces many problems in practical scenarios, such as the decrease in the accuracy of global models caused by data heterogeneity.

In federated learning, the production and storage of user data occur on the client side, and their data distribution is largely influenced by factors such as client device type, user preference, and organization. As the client group in practical scenarios is usually large in size and structurally complex, these factors generally exhibit significant differences, causing the data distribution between each local data set to be uneven and deviate from the global data distribution, exhibiting non-independent and identical distribution (Non-IID) characteristics. This phenomenon is known as data heterogeneity. The heterogeneity of client data results in significant differences between local models. After aggregating these different local models, the obtained global model has a considerable discrepancy from the ideal model, and this discrepancy accumulates with an increase in aggregation rounds, resulting in a reduction in the accuracy of the global model.

Currently, research on solving data heterogeneity mainly focuses on optimizing local training on the client side, roughly divided into two directions: (1) data-based optimization methods, such as solving the data heterogeneity problem of different clients through data enhancement, so as to improve accuracy; (2) algorithm-based optimization methods, reconstructing the client's loss function by adding a regularization term to reduce the difference between the local model

and global model. However, the former usually leads to training overfitting and poor prediction of new data samples; the regularization term in the latter is often an end-to-end limitation on model differences, providing limited precision improvement effects.

To address the above issues, we have proposed a federated optimization algorithm, FedECCR, based on encoding contrast and classification correction. FedECCR targets the optimization process through two stages: prototype contrastive training of the encoder and unbiased simulation correction of the classifier, which effectively suppresses the impact of heterogeneous data on the encoder and classifier. The main contributions of this paper are as follows.

- We classify all hidden layers of the model as encoders and the output layer as the classifier. By comparing the distribution differences between the global features extracted by the encoder in a data-heterogeneous environment and the client features, we found that there are significant differences in the distributions of client features and global features, and their discriminative power is poor.
- In response to differences in feature distribution, we designed a prototype comparison loss based on the prototype learning [6] and the feature similarity comparison of contrastive learning as a regularization term for local training. Then, we conducted an encoder prototype comparison training to align the feature mappings of the client encoder and improve the discriminability of the global encoder feature mappings.
- Based on high discriminability feature mappings, we estimated the statistical information of global features unbiasedly and used the corresponding Gaussian mixture model [7] to generate class-balanced simulated features for retraining the global classifier. This corrected the model parameters of the classifier and improved the classification accuracy of the global model on heterogeneous data.
- We conducted extensive experiments on datasets such as CIFAR-10 to evaluate the performance of TS-FedPC. The results indicate that compared to FedAvg, FedProx, and MOON, TS-FedPC can significantly improve the classification accuracy of the global model in heterogeneous data environments. Furthermore, it exhibits stability across different scenarios and possesses good scalability.

The arrangement of this article is as follows: first, we conducted related work in Sect. 2. Then, in Sect. 3, we introduced the FedECCR method. Experiments are presented in Sect. 4. Finally, in Sect. 5, we summarize the contents of this paper.

2 Related Work

There are two mainstream optimization methods for the problem of decreased overall model accuracy caused by data heterogeneity, including the optimization

methods based on data and the optimization methods based on the algorithm of federated learning.

The Optimization Methods Based on Data. This method improves the model accuracy by resolving the data heterogeneity between different clients, including data sharing and data augmentation [8].

For the optimization method based on data sharing, Zhao et al. [9] proposed maintaining a globally shared data set on a central server. Clients would then randomly sample a portion of the data to mix with their local data set, reducing Non-IID. Similarly, [10,11] have shared local data with the server to alleviate non-IID. While this method can improve global model performance, generating the shared data set is challenging because the server can't sense the local data set status. Additionally, downloading a portion of the shared data set violates privacy protection requirements.

Data augmentation is a technique to increase the diversity of training data by applying random transformations or knowledge transfer. It has been applied to federated learning in [12]. This method sends label distribution information of each client to the server, which calculates the sample number N_c and its mean value $\bar{N}$ for each category. If $N_c < \bar{N}$, the client needs to enhance the class data and use the locally augmented data set to train the model. Studies [13–15] have shown that data augmentation can improve the learning performance of the global model on non-IID data sets. However, most of these techniques require sharing label distribution information or some samples, which increases the risk of data privacy leakage.

The Optimization Methods Based on the Algorithm. The research on the algorithm level can be divided into three directions: gradient correction, regularization methods, and personalized federated learning.

The representative method of the optimization method based on the gradient correction direction is SCAFFOLD [16]. This method uses a control variable with the same size as the model gradient to predict the update direction of the global model and then uses this control variable in the local training to correct the client's gradient, adjusting the client's update direction to the update direction of the global model to alleviate the local model differences caused by Non-IID. [17,18] also added control variables similar to SCAFFOLD to correct client gradients during the local training process. Li et al. proposed a method [19] to estimate the average update direction of the client and server-level classifiers (i.e., the last few fully connected layers), and use their differences as control variables to reduce the divergence of classifier update direction. It should be noted that the gradient correction method will lead to a significant increase in communication volume because an additional control variable containing multiple high-dimensional matrices needs to be transmitted between the server and clients in each aggregation round.

Regularization methods are commonly used in federated learning to suppress the divergence of local models caused by heterogeneous data. FedProx uses the squared Euclidean distance between the local model and the global model as the regularization term [20], while FedCurv and FedCL use Elastic Weight Consoli-

dation (EWC) to prevent catastrophic forgetting [21,22] based on the idea of lifelong learning [23] and using EWC [24] regularization term to penalize significant updates to important parameters by local training, where important parameters refer to model parameters that have a greater impact on federated tasks. MOON applies contrastive learning to federated learning, adding the model contrastive loss between the global model and the local model as the regularization term [25] based on the contrastive learning technique [26], which significantly improves the accuracy of the global model under data heterogeneous environment. However, regularization methods may bring additional communication overhead or neglect the targeted handling of the internal structure of the model, resulting in an insignificant accuracy improvement.

Unlike the above two approaches, personalized federated learning aims to provide personalized solutions by adjusting the model according to the client's local task. Meta-learning and multi-task learning are the main types of personalized methods [27,28]. G-FML [29] is a group-based federated meta-learning framework that adaptively divides clients into groups based on the similarity of data distribution. SpreadGNN [30] is a federated multi-task learning framework that uses a novel optimization algorithm with a convergence guarantee. Although personalized federated learning can solve the problem of data heterogeneity by building multiple personalized models, it does not consider the performance optimization of the global model, making it unsuitable for scenarios that require a powerful single global model.

Based on the research mentioned above, data augmentation and sharing methods pose privacy risks, gradient correction methods increase communication volume, and personalized federated learning isn't suitable for scenarios requiring a single global model. To train a powerful global model for practical applications, we propose a federated optimization method that uses encoding contrast and classification correction for non-IID optimization. This method aims to improve global model accuracy in a data heterogeneous environment by alleviating the impact of heterogeneous data on model encoders and classifiers without increasing communication costs.

3 Method

In this section, we will introduce the federated optimization algorithm FedECCR, which is based on encoding contrast and classification correction. First, we introduce the overall process and ideas of FedECCR, and then focus on introducing the two core steps of the algorithm, including encoder prototype contrastive training and classifier unbiased simulation correction.

3.1 Method Overview

FedECCR divides a model ω into two parts: the encoder ωe and the classifier ωc. The encoder is the collection of all hidden layers in the neural network, which is designed to encode the input data into features. The classifier is the output layer

of the neural network, which is responsible for making classification decisions based on the features extracted by the encoder. The working principle of both is as follows: the encoder encodes data that is originally low-discriminatory to high-discriminatory features, and the classifier makes accurate classification decisions based on these high-discriminatory features.

Due to the heterogeneity of the data, both the encoder and classifier of the model have model biases. The model bias of the encoder can cause different clients to have different feature encoding methods (also known as feature mapping), which reduces the discriminatory power of the global model feature mapping and increases the difficulty of the classifier's decision-making. The model bias of the classifier can cause significant differences in the classification decisions of different clients for the same input data and decrease the overall classification accuracy. To address this issue, FedECCR first adopts encoder prototype comparison training to align the feature mapping of different client encoders, while improving the discriminatory power of the global encoder feature mapping. Then, through unbiased simulation correction of the classifier, the decision accuracy of the global classifier is improved. The overall architecture is shown in Fig. 1.

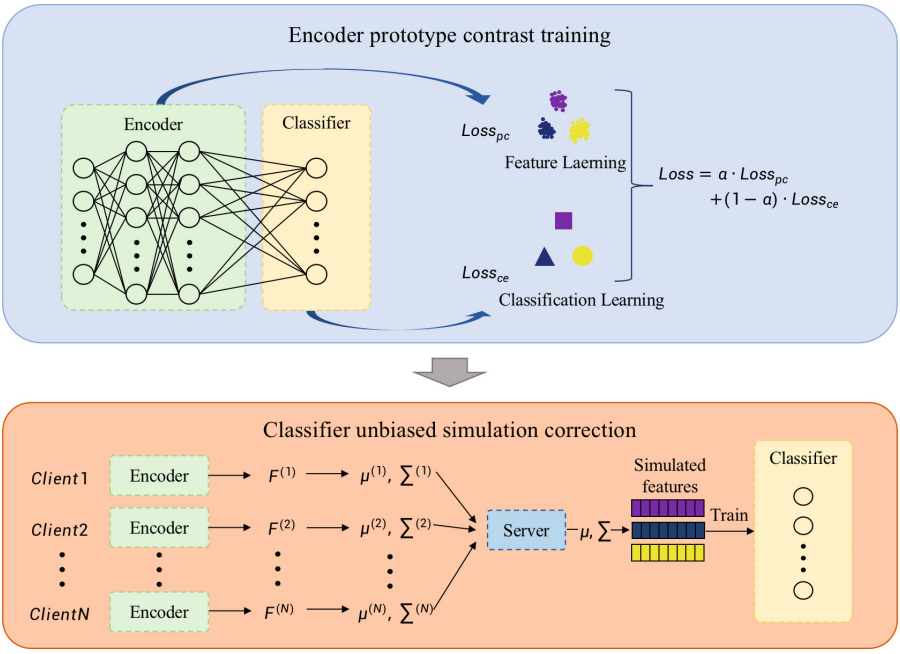


Fig. 1. FedECCR algorithm structure

As shown in Fig. 1, FedECCR mainly consists of the following two core stages:

(1) **Encoder prototype contrast training**: In order to align the feature mappings of different client encoders while improving their discriminability, FedECCR reconstructed the client loss function by adding the prototype contrast loss $Loss_{pc}$ on the basis of the original classification loss $Loss_{ce}$. The prototype refers to the center of each feature cluster. $Loss_{pc}$ can make the encoder extract feature vectors that aggregate towards the prototype of the corresponding category and away from the prototypes of other categories, thus supervising each client encoder to learn consistent and highly discriminative feature mappings. In the local training process, the client first calculates the prototype contrast loss $Loss_{pc}$ using the prototype and the features extracted by the encoder, then calculates the classification loss $Loss_{ce}$ based on the output of the classifier, and finally combines $Loss_{pc}$ and $Loss_{ce}$ as the final loss value for local model parameter optimization.

(2) **Classifier unbiased simulation correction**: In order to eliminate the model bias of the classifier and improve classification accuracy, FedECCR generates class-balanced simulation features based on a Gaussian mixture model and retrains the global classifier model parameters using the simulation features to correct classification decisions. Specifically, the client first extracts features on the local data set and calculates the mean $\mu^{(i)} = \{\mu_1^{(i)}, \mu_2^{(i)}, ..., \mu_C^{(i)}\}$ and covariance $\sum^{(i)} = \{\sum_1^{(i)}, \sum_2^{(i)}, ..., \sum_C^{(i)}\}$ of feature $F^{(i)}$. The server then aggregates the statistical information of the global features, i.e., the mean μ and covariance $\sum$. Based on the statistical information of the global features, the server uses a Gaussian distribution generator to generate an equal amount of simulation features for each data category and retrains the global classifier with these simulation features. As the classifier is the output layer of the model, the optimized output layer parameters obtained from the above steps can make more accurate classification decisions.

3.2 Encoder Prototype Comparison Training

The powerful data-fitting ability of deep learning models relies on the multi-level model structure and the discriminatory power of the encoder feature mapping directly affects the classification accuracy of the classifier. However, in federated learning, model bias caused by data heterogeneity causes different client encoders to learn inconsistent feature mappings. After model aggregation, the globally extracted features of the global encoder often do not have high discriminatory power, which in turn affects the classification accuracy of the model.

To address the above problems, we construct prototype contrast loss based on the feature similarity comparison of prototype learning and contrastive learning techniques, in order to supervise the learning of feature mapping for each client-side feature and enable the encoder of the client-side to learn consistent and highly separable feature mapping, thus reducing the model bias of the encoder.

According to reference [6], we refer to the prototype calculated based on client-side features as a local prototype, while the global prototype is obtained

by averaging the local prototypes of each client side. The design idea of prototype contrast loss $Loss_{pc}$ is to move the features closer to the global prototype of the corresponding category while moving away from the global prototypes of other categories. The specific calculation method can be seen in Eq. (1).

$$\begin{aligned} Loss_{pc} &= \log \frac{exp(sim(f_j^{(i)}, z_k)/\tau)}{\sum_{\tilde{k} \neq k} exp(sim(f_j^{(i)}, z_{\tilde{k}})/\tau)}, \\ where\ sim(f_j^{(i)}, z_k) &= (\mathbf{f_j^{(i)}}^{\top} z_k)/(\parallel f_j^{(i)} \parallel_2 \cdot \parallel z_k \parallel_2 \end{aligned} \tag{1}$$

Here, $f_j^{(i)}$ represents the features corresponding to the j input data of client i with category k, and z_k represents the global prototype of the k-th category features. sim represents the cosine similarity, and τ is the temperature parameter used to adjust the sensitivity of the $Loss_{pc}$ function to difficult-to-distinguish features. z_k can be calculated according to Eq. (2):

$$\begin{aligned} z_k &= \frac{1}{N} \sum_{i=1}^{N} z_k^{(i)}, \\ where\ z_k^{(i)} &= \frac{1}{n_{i,k}} \sum_{(x_j, y_j) \in D_{i,k}} f_j^{(i)} \end{aligned} \tag{2}$$

Here, $z_k^{(i)}$ represents the local prototype of the k-th category features of client i, $D_{i,k}$ is the collection of the k-th category data in the local data set of client i, and $n_{i,k}$ represents the amount of data in $D_{i,k}$.

In order to coordinate the feature mapping learning of the client's encoder and the classification learning, based on the research results of Mu et al. [31], we use a decay factor α to adjust the ratio between the client's prototype contrast $Loss_{pc}$ and classification $Loss_{ce}$. The final client loss function is represented by Eq. (3).

$$Loss = \alpha \cdot Loss_{pc} + (1 - \alpha) \cdot Loss_{ce} \tag{3}$$

Here, $\alpha = 1 - r/R$, r represents the current aggregation round of federated learning, and R represents the total aggregation rounds.

Based on the loss function, conducting prototype contrast training of the encoder can obtain a globally high feature-mapping discriminator encoder. The specific process is shown in Algorithm 1. In each round of aggregation, the server sends the global model and global prototype to the clients. After completing the local training, the clients upload the local models and local prototypes to the server, which aggregates them into new global models and global prototypes. Iterate this process until the round of aggregation is complete.

After the comparison training of the encoder prototype, the feature maps of each client encoder are supervised by the global prototype. The feature distribution shows the characteristics of intra-class cohesion and inter-class separation, which promotes the significant improvement of the feature mapping distinction of the global encoder aggregated by the client encoder.

Observing the algorithm process of the comparison training of the encoder prototype, it can be found that in each round of communication during the training process, in addition to transmitting the model, it is also necessary to transmit the prototype. However, the prototype is usually a collection of low-dimensional vectors, and its size can be ignored compared to the model parameters. Therefore, it will not increase too much communication cost.

3.3 Unbiased Simulation Correction of Classifier

The classifier takes the features extracted by the encoder as input, and the distribution of the features will directly affect the classification accuracy. In a data-heterogeneous environment, the heterogeneity of the input data is further amplified after passing through the layers inside the encoder and is ultimately manifested as a strong non-IID feature distribution. Therefore, heterogeneous data has a more serious model bias on the classifier than on the encoder.

We propose unbiased simulation correction of classifiers, which uses the generated simulation features with balanced categories to retrain the global classifier, eliminates the model bias of the classifier, and improves the global model classification accuracy.

Ideally, aggregating the data features of each client for retraining the global classifier is the best way to correct the model parameters of the classifier but it will cause a leakage of user privacy. Therefore, we assume that the values of each feature follow a Gaussian distribution [32], and simulated features are generated based on this distribution. Where each cluster of features follows a Gaussian distribution with corresponding mean and covariance, by aggregating the means and covariances of each cluster of features from all local data sets, we obtain an unbiased estimate of the mean and covariance of the global features and thus can generate simulated features that follow the Gaussian distribution of the global features.

Specifically, the server sends the globally trained encoder, which has been compared with the prototype encoder, to each client. Client i uses the global encoder to extract feature $F^{(i)} = \{f_1^{(i)}, f_2^{(i)}, ..., f_{n_i}^{(i)}\}$ from its local data set, where n_i is the local data set size of client i. After feature extraction, client i calculates the mean $\mu_k^{(i)}$ and covariance $\sum_k^{(i)}$ of each feature according to Eqs. (4) and (5).

$$\mu_k^{(i)} = \frac{1}{n_{i,k}} \sum_{(x_j, y_j) \in D_{i,k}} f_j^{(i)} \tag{4}$$

$$\sum_k^{(i)} = \frac{1}{n_{i,k} - 1} \sum_{(x_j, y_j) \in Di,k} (f_j^{(i)} - \mu_k^{(i)})(\mathbf{f_j^{(i)}} - \mu_{\mathbf{k}}^{(\mathbf{i})})^{\top} \tag{5}$$

Then, clients upload statistical information of their own features, which includes the means $\mu^{(i)} = \{\mu_1^{(i)}, \mu_2^{(i)}, ..., \mu_C^{(i)}\}$ and covariances $\sum^{(i)} = \{\sum_1^{(i)}, \sum_2^{(i)}, ..., \sum_C^{(i)}\}$ of various features, to the server.

Next, the server aggregates the statistical information of all clients' features and estimates the mean and covariance of global features without bias. The

aggregation method for feature mean is shown as Eq. (6), where N represents the total number of the k-th data class in the global data set.

$$\mu_k = \sum_{i=1}^{N} \frac{n_{i,k}}{n_k} \cdot \mu_k^{(i)} \tag{6}$$

Where n_k represents the total number of the k-th type of data in the global data set. The aggregation method of feature covariance $\sum_k$ is complex and can be calculated using the Eq. (7).

$$\sum_k = \sum_{i=1}^{N} \frac{n_{i,k}-1}{n_k-1} \sum_k^{(i)} + \sum_{i=1}^{N} \frac{n_{i,k}}{n_k-1} \mu_k^{(i)} \mu_{\mathbf{k}}^{(\mathbf{i})\top} - \frac{n_k}{n_k-1} \mu_k \mu_{\mathbf{k}}{}^{\top} \tag{7}$$

After aggregation, the server obtains the means $\mu = \{\mu_1, \mu_2, ..., \mu_C\}$ and covariances $\sum = \{\sum_1, \sum_2, ..., \sum_C\}$ of various features. Then, according to the Gaussian distribution $\mathcal{N}(\mu, \sum)$, it generates M simulated features for each category.

Finally, after freezing the parameters of the global encoder, the server uses simulated features to retrain the global classifier, completes the correction of the classifier model parameters, and improves the classification accuracy of the global model. And the detail of this method is described as Algorithm 2.

It is worth noting that during the stage of unbiased simulation correction of classifiers, the client only exposes the statistical information of its own features to the server instead of the feature source, and, the server generates simulated features based solely on this statistical information instead of real features, which ensures the privacy and security of the client in the whole process. In addition, thanks to the high discriminative feature mapping that aggregates classes and separates them within the encoder, the simulation features generated by the server are closer to the real features. Therefore, a small number of simulation features can achieve a significant effect in correcting the classifier.

3.4 Algorithm Implementation

In this section, we provide detailed descriptions of the overall process of FedECCR, and the details are as follows:

(1) The server initializes the global model and global prototype, which are sent to the client during each round of aggregation (lines 1 to 5 of Algorithm 1).
(2) The client optimizes the local model during local training using classification loss and prototype comparison loss (lines 13 to 22 of Algorithm 1).
(3) After local training, the client uploads the local model and local prototypes to the server, which aggregates them to generate a new global prototype and global model (lines 23 to 28 of Algorithm 1).
(4) The client uses the global model to extract features from its local data set and uploads the mean and covariance of each feature to the server(lines 21 to 24 of Algorithm 2).

(5) The server aggregates the unbiased estimated values of the global feature mean and covariance (line 6 of Algorithm 2) and generates class-balanced simulated features to retrain the classifier of the global model (lines 7 to 17 of Algorithm 2).

Algorithm 1: Encoder prototype contrast training

Input: Total aggregation rounds R, local data set D_i, local training epochs E, number of data categories C, number of clients C, temperature parameter τ, and learning rate η

Output: Global model w_R

Server:
Initialize global modelw_0 and global prototypez_0
for $r = 0, 1, ..., R-1$ **do**
  for $i = 1, 2, ..., N$ **do**
    Send w_r and $z_r$to client i
    $W_{r+1}^{(i)}, z_{r+1}^{(i)} = LocalTraining(i, r, w_r, z_r)$
  end
  $z_{r+1} = \frac{1}{N}\sum_{i=1}^{N} z_{r+1}^{(i)}$, $w_{r+1} = \sum_{i=1}^{N} \frac{n_i}{n} w_{r+1}^{(i)}$
end
return w_R
Client:
$LocalTraining(i, r, w_r, z_r)$:
for *epoch* $= 0, 1, 2..., E_l$*-1* **do**
  foreach *batch*$\{x_j, y_j\}$ *in* D_i **do**
    Extracting Features from $w_{e,r}^{i}$ to f_j^i
    $Loss_{pc} = -log\frac{exp(sim(f_j^{(i)}, z_{r,k})/\tau)}{\sum_{k'} exp(sim(f_j^{(i)}, z_{r,k'})/\tau)}$
    $\alpha = 1 - r/R$
    $Loss = \alpha \cdot Loss_{pc} + (1-\alpha) \cdot Loss_{ce}$
    $w_r^{(i)} = w_r^{(i)} - \eta \nabla Loss$
  end
end
$w_{r+1}^{(i)} = w_r^{(i)}$
Encoder $w_{e,r+1}^{(i)}$ extracts feature $f_j^{(i)}$
for $k = 1, 2, ..., C$ **do**
  $z_{r+1,k}^{(i)} = \frac{1}{n_{i,k}} \sum_{(x_j, y_j) \in D_{i,k}} f_j^{(i)}$
end
$z_{r+1}^{(i)} = \{z_{r+1,1}^{(i)}, z_{r+1,2}^{(i)}, ..., z_{r+1,C}^{(i)}\}$
return $w_{r+1}^{(i)}, z_{r+1}^{(i)}$

Algorithm 2: Classifier unbiased simulation correction

Input: Global model w_R, number of simulation features for each category M, epoch number of global classifier training E, learning rateη.
Output: Global model w_{R+1}

Server:
for $i = 1, 2..., N$ **do**
  Send encoder $w_{e,R}$ to client i
  $\mu^{(i)}, \sum^{(i)} = LocalFeatureStatistics(i, w_e, R)$
end
$\mu = \{\mu_1, \mu_2, ..., \mu_C\}$, $\sum = \{\sum_1, \sum_2, ..., \sum_C\}$
for $k = 1, 2..., C$ **do**
  $F_k^{'} = \{F_{1,k}^{'}, F_{2,k}^{'}, ..., F_{M,k}^{'}\}$ based on Gaussian distribution $\mathcal{N}(\mu_k, \sum^k)$
end
for $epoch = 0, 1, 2..., E-1$ **do**
  foreach $batch\{f_j^{'}, y_j^{'}\}$ *in* $F^{'} = \{F_1^{'}, F_2^{'}, ..., F_C^{'}\}$ **do**
    $Loss_{ce} = CrossEntropyLoss(w_{c,R}; f_j^{'}, y_j^{'})$
    $w_{c,R} = w_{c,R} - \eta \nabla Loss_{ce}$
  end
end
$w_{c,R+1} = w_{c,R}, w_{R+1} = (w_{e,R}, w_{c,R+1})$
return w_{R+1}
Client:
LocalFeatureStatistics(Encoder $w_{e,R}$, Server i):
foreach $batch\{x_j, y_j\}$ *in* D_i **do**
  Extract feature $f_j^{(i)}$ from $w_{e,R}$
end
$\mu^{(i)} = \{\mu_1^{(i)}, \mu_2^{(i)}, ..., \mu_C^{(i)}\}$ for feature $\sum^{(i)} = \{\sum_1^{(i)}, \sum_2^{(i)}, ..., \sum_C^{(i)}\}$ by formulas (4) and (5).
return $\mu^{(i)}, \sum^{(i)}$

4 Experiment

In this section, we experimentally verify the federated optimization algorithm FedECCR based on encoder contrast and classifier correction. We conduct experiments to verify the accuracy and scalability of FedECCR with Non-IID data, compared with MOON, FedProx, and FedAvg algorithms. We also analyze the feature distribution of different algorithms to verify the high distinguishability of FedECCR features and conduct ablation experiments of classifier correction to demonstrate the accuracy improvement effect of FedECCR.

4.1 Experimental Setup

(1) Datasets and models: We used three popular datasets: EMNIST, CIFAR-10, and CIFAR-100. To introduce data heterogeneity, we partitioned non-IID datasets using the Dirichlet distribution [33]. We use a simple CNN model for EMNIST and CIFAR-10, which includes two convolutional layers, two pooling layers, and five fully connected layers, and a ResNet-50 for CIFAR-100.
(2) Parameter Setting: We used 10 clients, 100 aggregation rounds, 10 local epochs per client, a learning rate of 0.01, batch size of 64, and a Non-IID data set divided by 0.5. We adjusted the regularization coefficients for MOON and FedProx and set the temperature parameter to 0.5 in MOON and 0.5 or 1 in FedECCR, depending on the dataset. In the EMNIST dataset experiments, all algorithms converged after 30 rounds of aggregation, so the aggregation rounds were set to 30. For the unbiased classifier simulation correction phase, we generated 400 simulated features for each data class and set the number of epochs for service retraining classifiers to 20 based on multiple rounds of experimentation and testing.
(3) Environment: All experiments were conducted on Ubuntu using Python with the PyTorch framework. Data stream processing relied on the torchvision library, while Matplotlib was used for visualization. Other core libraries used include NumPy, Scikit-learn, and Wandb.
(4) Baseline: We used FedAvg, FedProx, and MOON as baselines. FedAvg is a classic and effective federated learning algorithm. FedProx introduces a regularization term to solve the heterogeneity problem of data, while MOON addresses the heterogeneity problem of data by introducing prototype features into federated learning.

4.2 Accuracy

We calculated the test accuracy of FedAvg, FedProx, MOON, and FedECCR algorithms on the EMNIST, CIFAR-10, and CIFAR-100 data sets respectively, and calculated the error range of the accuracy through multiple experiments, as shown in Table 1.

Table 1. Test accuracy of FedAvg, FedProx, MOON, and FedECCR on EMNIST, CIFAR-10, and CIFAR-100 data sets

	EMNIST	CIFAR-10	CIFAR-100
FedAvg	$84.77\% \pm 0.2\%$	$67.13\% \pm 0.2\%$	$66.85\% \pm 0.4\%$
FedProx	$85.33\% \pm 0.1\%$	$68.12\% \pm 0.3\%$	$67.27\% \pm 0.2\%$
MOON	$84.93\% \pm 0.1\%$	$69.18\% \pm 0.5\%$	$68.83\% \pm 0.4\%$
FedECCR	$\mathbf{85.66\% \pm 0.1\%}$	$\mathbf{70.89\% \pm 0.3\%}$	$\mathbf{74.32\% \pm 0.3\%}$

From the test accuracy results in Table 1, it can be seen that FedECCR has the highest test accuracy in different data set experiments, with test accuracies of

85.66%, 70.89%, and 74.32% on the EMNIST, CIFAR-10, and CIFAR-100 data sets respectively. The accuracy of FedProx is not significantly improved, compared with FedAvg, as it only uses end-to-end regularization to restrict model bias, and the regularization coefficient remains unchanged during training. On the EMNIST data set, it increases by about 0.56%, on CIFAR-10 it increases by about 0.99%, and on CIFAR-100 the accuracy decreases by 0.55%. MOON obtains a more significant improvement in accuracy by restricting the targeted bias of the encoder. Compared to FedAvg, it improves the model accuracy by about 2.05% and 0.96% on CIFAR-10 and CIFAR-100 respectively. However, in the experiments on EMNIST, the accuracy improvement of MOON is generally not significant, and the model accuracy is only about 0.16% higher than that of FedAvg. We believe this may be due to MOON's feature comparison technology being better suited for handling RGB image classification tasks.

The FedECCR algorithm benefits from the separate optimization of the model encoder and classifier, with the highest accuracy improvement. On EMNIST, it is approximately 0.89% higher than FedAvg, approximately 0.33% higher than FedProx, and approximately 0.73% higher than MOON. On CIFAR-10, it is approximately 3.76% higher than FedAvg, approximately 2.77% higher than FedProx, and approximately 1.71% higher than MOON. On CIFAR-100, it is approximately 5.13% higher than FedAvg, approximately 5.68% higher than FedProx, and approximately 4.17% higher than MOON.

In summary, compared to other algorithms, FedECCR can more effectively improve the accuracy of global models in federated learning in heterogeneous data environments.

4.3 Scalability

In order to verify the scalability of the FedECCR algorithm, we conducted experiments on different values of the Non-IID degree of the data set and the number of clients, and comprehensively compared the model accuracy of FedECCR with FedAvg, FedProx, and MOON in multiple scenarios.

(1) Degree of Non-IID of the data set:
We use prior distribution hyperparameters of the Dirichlet to control the degree of Non-IID of the data set. The smaller the value, the higher the degree of Non-IID of the data set. We set three different values of 0.1, 0.5, and 5, and compared

Table 2. Model accuracy of the algorithm under different degrees of Non-IID

	$\beta = 0.1$	$\beta = 0.5$	$\beta = 5$
FedAvg	$61.86\% \pm 0.6\%$	$67.13\% \pm 0.2\%$	$73.34\% \pm 0.1\%$
FedProx	$61.65\% \pm 0.4\%$	$68.12\% \pm 0.3\%$	$73.03\% \pm 0.1\%$
MOON	$62.55\% \pm 0.1\%$	$69.18\% \pm 0.5\%$	$73.47\% \pm 0.1\%$
FedECCR	$\mathbf{65.49\% \pm 0.2\%}$	$\mathbf{70.89\% \pm 0.3\%}$	$\mathbf{73.80\% \pm 0.1\%}$

the model accuracy under these three values on the CIFAR-10 data set, as shown in Table 2.

The results show that when $\beta = 0.1$, the model accuracy of FedECCR is approximately 3.63% higher than FedAvg, approximately 3.84% higher than FedProx, and approximately 2.94% higher than MOON. When $\beta = 0.5$, the model accuracy of FedECCR is approximately 3.76% higher than FedAvg, approximately 2.77% higher than FedProx, and approximately 1.71% higher than MOON. When $\beta = 5$, due to the low degree of Non-IID of the data set, we reduced the prototype contrastive loss of FedECCR by a factor of 10 to reduce the impact of feature mapping learning on classification learning, and the final model accuracy is approximately 0.46% higher than FedAvg, approximately 0.5% higher than FedProx, and approximately 0.33% higher than MOON.

From the above results, we can see that the FedECCR can maintain stable performance on data sets with different degrees of Non-IID, and the model accuracy is higher than FedAvg, FedProx, and MOON. In addition, it is worth noting that the model accuracy of FedProx decreases when the degree of Non-IID of the data set changes, which may be because its regularization term only considers end-to-end model differences and cannot control the penalty strength of internal structural differences in the model.

(2) Number of clients:

In order to test the performance of the algorithm under different client capacities, we expanded the number of clients from the default 10 to 50 and 100 respectively, and compared the model accuracy of all algorithms. The specific results are shown in Table 3. It should be noted that we used different client sampling rates for the setting of 50 clients and 100 clients. In the case of 50 clients, the client sampling rate is 1, and all clients participate in every round of model aggregation; in the case of 100 clients, we refer to MOON [27] and set the client sampling rate to 0.2, with only 20 clients participating in every round of model aggregation.

Table 3. Accuracy under different client numbers and aggregation rounds settings

	client = 50	client = 100
FedAvg	$66.42\% \pm 0.1\%$	$62.28\% \pm 0.4\%$
FedProx	$66.51\% \pm 0.2\%$	$62.22\% \pm 0.1\%$
MOON	$66.27\% \pm 0.3\%$	$62.99\% \pm 0.1\%$
FedECCR	$\mathbf{67.09\% \pm 0.2\% (\tau = 0.5)}$	$\mathbf{62.76\% \pm 0.2\% (\tau = 0.5)}$

The results showed that when Clients = 50, the model accuracy of FedECCR is higher than other algorithms, with an increase of approximately 0.67% over FedAvg, approximately 0.58% over FedProx, and approximately 0.82% over MOON. When Clients = 100, the model accuracy of FedECCR is better than FedAvg and FedProx, with approximately 0.48% higher than FedAvg and approximately 0.54% higher than FedProx.

It should be noted that when Clients = 100, the model accuracy of FedECCR is slightly lower than MOON, with a decrease of approximately 0.23%. At this time, we observed that the silhouette coefficient of the encoder feature distribution of FedECCR was highest only at 0.047, while it was highest at 0.092 when Clients = 50. This is because the sampling rate of 0.2 appears low when Clients = 100. A sampling rate that is too low will reduce the discrimination of the encoder feature map, resulting in a decrease in the effect of improving model accuracy.

4.4 Feature Discrimination

In order to verify that the prototype comparison training can promote the model encoder to learn high-discriminatory feature mapping, we compared the feature distributions of different algorithms on the EMNIST and CIFAR-10 data sets. From the result in Fig. 2, we can see that, compared with FedAvg, FedProx, and MOON, FedECCR has significantly higher cohesion and farther cluster distances for various feature clusters, and the feature distribution has higher discriminability.

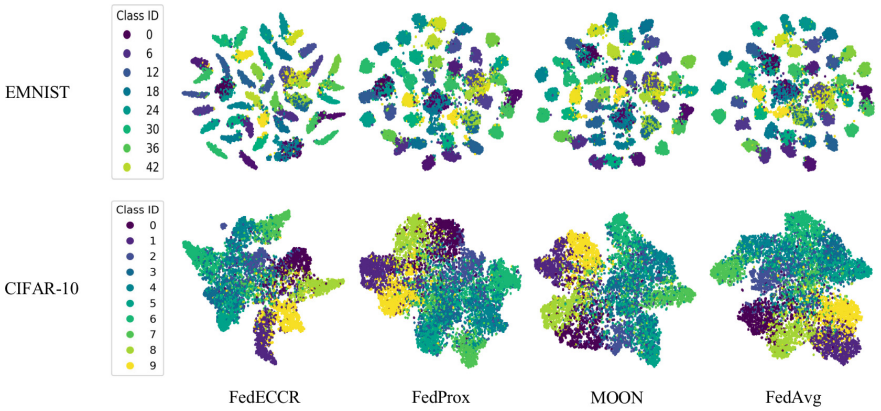
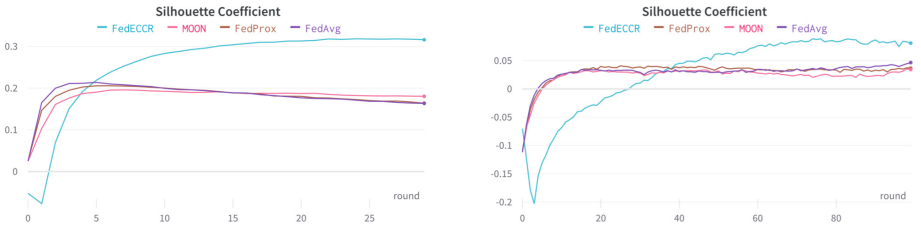


Fig. 2. Feature distribution of FedECCR, FedProx, MOON, and FedAvg

In order to observe the changes in feature mapping discriminability during the training process, we calculated the changes in silhouette coefficient of the feature distribution of each algorithm during the training process, and plotted the change curves, as shown in Fig. 3.

In Fig. 3, the final silhouette coefficients of FedECCR, FedProx, MOON, and FedAvg on the EMNIST data set are 0.3159, 0.1638, 0.1803, and 0.1633, respectively; on the CIFAR-10 data set, the final silhouette coefficients are 0.0809, 0.0366, 0.0343, and 0.0466, respectively. It can be seen that on these two data sets, the silhouette coefficients of FedECCR's feature distribution after training are the highest, and significantly higher than those of other algorithms.



EMNIST CIFAR-10

Fig. 3. The change curves of silhouette coefficients of the feature distribution of FedECCR, FedProx, MOON, and FedAvg.

This indicates that the feature mapping discriminability of the global encoder of FedECCR is higher than that of other algorithms.

The above results show that the prototype comparison training of FedECCR's encoder can indeed promote the characteristics of intra-cluster aggregation and inter-cluster separation of feature distribution, thus effectively improving the discrimination of the global encoder's feature mapping.

4.5 Classifier Correction Ablation Experiment

The higher the discriminability between features of different categories, the clearer the differences between the corresponding Gaussian distributions. At this time, the simulated features generated based on the mean and covariance of the features are closer to the real features, the correction effect on the classifier is better, and the model accuracy is significantly improved.

In this section, we performed unbiased simulation correction on FedAvg, FedProx, and MOON, and analyzed the effect of feature mapping discriminability on classifier correction from the results of accuracy improvement, as shown in Table 4. We use USCC (Unbiased Simulation Correction of Classifier, the second stage of the FedECCR algorithm) to identify the classifier unbiased simulation correction operation, such as FedAvg-USCC, which represents the FedAvg algorithm after being corrected by the classifier unbiased simulation.

On the EMNIST data set, unbiased simulation correction brought about approximately 0.16%, 0.14%, and 0.21% accuracy improvement to FedAvg, FedProx, and MOON, respectively; on the CIFAR-10 data set, it brought about approximately 1.05%, 1.58%, and 0.32% accuracy improvement, respectively; on the CIFAR-100 data set, it brought about approximately 0.23%, 0.09%, and 0.14% accuracy improvement, respectively.

From the results, even after adding unbiased simulation correction to other algorithms, FedECCR's accuracy improvement effect is still the best, which indicates that high-discriminatory feature mapping usually helps the classifier make more accurate classification decisions. And it mainly consists of two parts of calculation time for FedECCR. One is the encoder prototype comparison training phase, an additional 982.6 s of computation time was required to calculate

Table 4. Comparison of unbiased correction accuracy of classifiers of different algorithms

	EMNIST	CIFAR-10	CIFAR-100
FedAvg	$84.77\% \pm 0.6\%$	$67.13\% \pm 0.2\%$	$73.34\% \pm 0.1\%$
FedProx	$61.65\% \pm 0.4\%$	$68.12\% \pm 0.3\%$	$73.03\% \pm 0.1\%$
MOON	$62.55\% \pm 0.1\%$	$69.18\% \pm 0.5\%$	$73.47\% \pm 0.1\%$
FedAvg-USCC	$62.55\% \pm 0.1\%$	$69.18\% \pm 0.5\%$	$73.47\% \pm 0.1\%$
FedProx-USCC	$62.55\% \pm 0.1\%$	$69.18\% \pm 0.5\%$	$73.47\% \pm 0.1\%$
MOON-USCC	$62.55\% \pm 0.1\%$	$69.18\% \pm 0.5\%$	$73.47\% \pm 0.1\%$
FedECCR	$\mathbf{65.49\% \pm 0.2\%}$	$\mathbf{70.89\% \pm 0.3\%}$	$\mathbf{73.80\% \pm 0.1\%}$

prototype features for each epoch trained on local clients. The other is classifier stage, it was executed once at the end of training and only took 27 s. The total training time is about 25,000, so these two stages account for 4.03% of the overall computational cost.

5 Conclusion

The federated learning method is a popular distributed machine learning method, which cooperates with the clients and server to mine the large-scale data value while keeping data locally, which can protect the privacy of data. However, in practical applications, there is significant heterogeneity in the data on the federated learning client, which can cause divergence in the model update direction during local training, leading to local model bias. So the global model generated by aggregating biased models has significant differences from the ideal model, resulting in performance loss. To address the problem of poor accuracy of the global model due to local model bias in data heterogeneous environments, we propose a federated optimization algorithm FedECCR based on encoding contrast and classification correction. It specifically resolves model encoder and classifier bias, introducing prototype contrast loss as a regularization term in local training of the encoder, promoting consistent and separable feature maps to be learned by different clients, and then using simulated features generated from the statistical information of each client's features to retrain the classifier and correct its model parameters. Through experiments, we verified that compared with FedAvg, FedProx, and MOON, FedECCR can improve the accuracy of the global model by approximately 1% to 6% on multiple data sets such as CIFAR-10. In addition, FedECCR has good scalability and can maintain the high accuracy of the global model in situations where the Non-IID degree and the number of clients change. Since the classifier stage generates simulation features, which may cause overfitting during training, we will research the optimization for this problem in the future.

Acknowledgement. This work is supported by the Key Research and Development Program of Zhejiang Province 2023C03194; the National Natural Science Foundation of China under Grant No. 62072146; and the Natural Science Foundation of Zhejiang Province under Grant No. LQ23F020015.

References

1. Wang, X., Gao, H., Huang, K.: Artificial intelligence in collaborative computing. Mobile Netw. Appl. **26**, 2389–2391 (2021). https://doi.org/10.1007/s11036-021-01829-y
2. Yang, J., Zheng, J., Zhang, Z., Chen, Q.I., Wong, D.S., Li, Y.: Security of federated learning for cloud-edge intelligence collaborative computing. Int. J. Intell. Syst., 9290–9308 (2022). https://doi.org/10.1002/int.22992
3. McMahan, H.B., Moore, E., Ramage, D., Hampson, S., Arcas, B.: Communication-efficient learning of deep networks from decentralized data. arXiv: Learning (2016)
4. Hard, A., et al.: Federated learning for mobile keyboard prediction. arXiv: Computation and Language (2018)
5. Geyer, R.C., Klein, T., Nabi, M.: Differentially private federated learning: a client level perspective. Cornell University - arXiv (2017)
6. Tan, Y., Long, G., Liu, L., Zhou, T., Jiang, J.: FedProto: federated prototype learning over heterogeneous devices. arXiv: Learning (2021)
7. Reynolds, D.A.: Gaussian Mixture Models (2009)
8. Yan, Y., Zhu, L.: A Simple Data Augmentation for Feature Distribution Skewed Federated Learning (2023)
9. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-IID data. Cornell University - arXiv (2018)
10. Tuor, T., Wang, S., Ko, B., Liu, C., Leung, K.K.: Overcoming noisy and irrelevant data in federated learning. arXiv: Learning (2020)
11. Yoshida, N., Nishio, T., Morikura, M., Yamamoto, K., Yonetani, R.: Hybrid-FL: Cooperative Learning Mechanism Using Non-IID Data in Wireless Networks (2019)
12. Wicaksana, J., et al.: FedMix: Mixed Supervised Federated Learning for Medical Image Segmentation (2022)
13. Seol, M., Kim, T.: Performance enhancement in federated learning by reducing class imbalance of non-IID data. Sensors, 1152 (2023)
14. Shin, M., Hwang, C., Kim, J., Park, J., Bennis, M., Kim, S.-L.: XOR mixup: privacy-preserving data augmentation for one-shot federated learning. Cornell University - arXiv (2020)
15. Jeong, E., Oh, S., Park, J., Kim, H., Bennis, M., Kim, S.-L.: Multi-hop federated private data augmentation with sample compression. arXiv: Learning (2019)
16. Karimireddy, S., Kale, S., Mohri, M., Reddi, S.J., Stich, S.U., Suresh, A.: SCAFFOLD: stochastic controlled averaging for federated learning. In: International Conference on Machine Learning (2020)
17. Gao, L., Fu, H., Li, L., Chen, Y., Xu, M., Xu, C.-Z.: FedDC: Federated Learning with Non-IID Data via Local Drift Decoupling and Correction
18. Liu, Y., Sun, Y., Ding, Z., Shen, L., Liu, B., Tao, D.: Enhance Local Consistency in Federated Learning: A Multi-Step Inertial Momentum Approach (2023)
19. Li, B., Schmidt, M.N., Alstrøm, T.S., Stich, S.U.: Partial Variance Reduction improves Non-Convex Federated learning on heterogeneous data (2022)
20. Li, T., Sahu, A., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. arXiv: Learning (2018)

21. Shoham, N., et al.: Overcoming forgetting in federated learning on non-IID data. Cornell University - arXiv (2019)
22. Yao, X., Sun, L.: Continual local training for better initialization of federated models. In: 2020 IEEE International Conference on Image Processing (ICIP) (2020). https://doi.org/10.1109/icip40778.2020.9190968
23. Li, H., Krishnan, A., Wu, J., Kolouri, S., Pilly, P.K., Braverman, V.: Lifelong learning with sketched structural regularization. Cornell University - arXiv (2021)
24. Kirkpatrick, J., et al.: Overcoming catastrophic forgetting in neural networks. In: Proceedings of the National Academy of Sciences, pp. 3521–3526 (2017). https://doi.org/10.1073/pnas.1611835114
25. Li, Q., He, B., Song, D.: Model-contrastive federated learning. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021). https://doi.org/10.1109/cvpr46437.2021.01057
26. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.E.: A simple framework for contrastive learning of visual representations. Cornell University - arXiv (2020)
27. Vanschoren, J.: Meta-learning: a survey. arXiv: Learning (2018)
28. Zhang, Y., Yang, Q.: An overview of multi-task learning. Natl. Sci. Rev., 30–43 (2018). https://doi.org/10.1093/nsr/nwx105
29. Yang, L., Huang, J., Lin, W., Cao, J.: Personalized federated learning on non-IID data via group-based meta-learning. ACM Trans. Knowl. Discov. Data., 1–20 (2023). https://doi.org/10.1145/3558005
30. He, C., Ceyani, E., Balasubramanian, K., Annavaram, M., Avestimehr, A.S.: SpreadGNN: serverless multi-task federated learning for graph neural networks. Cornell University - arXiv (2021)
31. Mu, X., et al.: FedProc: prototypical contrastive federated learning on non-IID data. arXiv: Learning (2021)
32. Miller, J.W., Harrison, M.T.: Mixture models with a prior on the number of components. arXiv: Methodology (2015)
33. Hsu, H., Qi, H., Brown, M.: Measuring the effects of non-identical data distribution for federated visual classification. arXiv: Learning (2019)