IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 3 March 2024, accepted 13 March 2024, date of publication 20 March 2024, date of current version 27 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3379273

RESEARCH ARTICLE

FedDBO: A Novel Federated Learning Approach for Communication Cost and Data Heterogeneity Using Dung Beetle Optimizer

DONGYAN WANG[1,2], LIMIN CHEN[3], XIAOTONG LU[1,2], YIDI WANG[1,2], YUE SHEN[1,2], AND JINGJING XU[1,2]

[1]School of Mathematical Science, Mudanjiang Normal University, Mudanjiang 157011, China
[2]Institute of Applied Mathematics, Mudanjiang Normal University, Mudanjiang 157011, China
[3]School of Computer and Information Technology, Mudanjiang Normal University, Mudanjiang 157011, China

Corresponding author: Limin Chen (chenlimin_clm@126.com)

This work was supported in part by the Natural Science Foundation of Heilongjiang Province under Grant LH2019F051; and in part by the Science and Technology Innovation Projects of Mudanjiang Normal University under Grant kjcx2023-123mdjnu, Grant kjcx2023-124mdjnu, and Grant kjcx2023-126mdjnu.

ABSTRACT As an emerging distributed machine learning technology, federated learning has gained widespread attention due to its critical privacy protection mechanism. However, it also faces challenges such as high communication costs and heterogeneous client data.In order to address the above issues. This paper proposes a federated learning approach based on the dung beetle optimizer, named FedDBO. In this method, the model parameters uploaded from clients to the server are transformed into model scores. In each round of training, only some of the clients with high model scores need to be selected to upload their parameters to the server, thus reducing communication costs; Simultaneously, a model retraining strategy is introduced. After aggregating the model parameters sent by clients, the server performs a second iterative training on the aggregated model using its own metadata, thereby reducing data heterogeneity and improving model performance. In addition, a proof of convergence is provided, demonstrating that the model aggregated by FedDBO converges to the aggregated model of FedAvg after each training round. Finally, experiments indicate that when simulating various data heterogeneous environments on datasets, FedDBO exhibits higher accuracy and better stability compared to three other algorithms: FedAvg, FedShare, and FedPSO.

INDEX TERMS Federated learning, dung beetle optimizer, model scores, data heterogeneous, communication cost.

I. INTRODUCTION

In classical machine learning (ML), data is typically collected and stored on a centralized server or a single node, which is then used for training and testing [1], [2]. However, the majority of data is distributed across various mobile devices [3], posing significant challenges for data aggregation and model training [4]. Numerous scholars have proposed improvements [5], e.g., in the case of face recognition technology, there are no publicly available datasets that provide comprehensive face images and corresponding geometric interpretations of 3D faces, and thus Khan et al. [6] generate data by rendering a large number of realistic and uniquely attributed face images by using learning-based inverse face rendering. On the other hand, the privacy of data is not well-protected when models directly transmit data during the training process, leading to common examples of introducing privacy protection in ML algorithms [7], [8]. Nevertheless, inherent patterns in centralized machine learning data storage still pose privacy issues. In this context, attention has shifted from data aggregation to model aggregation.

The associate editor coordinating the review of this manuscript and approving it for publication was Yudong Zhang.

Federated learning (FL) has emerged as a new technological solution to address the aforementioned challenges [9]. It is a form of distributed ML, with an overall architecture that includes a central server and numerous local devices or nodes. The goal is to allow models to be trained on local devices, coordinated and aggregated through the central server to improve the global model. One of the key mechanisms of FL is ''data staying local''. Clients initially download the global model from the server, train the global model locally based on their own data, and then upload the trained model to the server. The server performs model aggregation updates, iteratively improving the global model. Throughout this training process, only the models trained with data are transferred between the server and clients, eliminating the direct transmission of data.

However, most FL models use the FedAvg and still face some challenges. On one hand, in scenarios with large volumes of data, frequent model transmissions can increase communication costs. On the other hand, in real-world applications, due to the unknown nature of local devices, data heterogeneity among different clients often leads to a decrease in model accuracy during the training process, and even convergence issues.

Current research aims to enhance model update speed by applying the Dung Beetle Optimizer (DBO), an algorithm that obtains optimal solutions in a distributed environment [10]. DBO requires multiple repetitions as it acquires optimal solutions through a random method, aligning well with the iterative nature of ML. It is well suited to the dynamic and heterogeneous environment of FL.

In this paper, for the shortcomings of the high communication cost in FL and the advantages of DBO which is very suitable for FL, a new FL method, FedDBO, is proposed, which transforms the form of the data transmitted from the client to the server from the model parameters to the model scores (the accuracy of the client's local model versus the global model), and uploads to the server the model parameters of the part of the client with higher model scores only in each round of training, so as to reduce the communication cost and improve the communication efficiency, and the experiments also show that the method can effectively alleviate the situation of the client's data isomorphism.

The remaining structure of this paper is as follows. Sec. II discusses the development and applications of FL and federated optimization, as well as related work combining them with swarm intelligence technologies. Sec. III describes FL and the underlying theory of DBO. Sec. IV introduces the main methods and provides key proofs. The experimental part will be shown in Sec. V. Finally, conclusions and future work are proposed in Sec. VI.

II. RELATED WORK

A. THE DEVELOPMENT AND APPLICATION OF FEDERATED LEARNING

With the proposal and development of FL, its scope has expanded significantly. Nilsson et al. [11] initially evaluated federated averaging (FedAvg), federated stochastic gradient descent (FedSGD), and CO-OP on the MNIST dataset, considering both independent and identically distributed (IID) and non-independent and identically distributed (Non-IID) data. In 2019, Yang et al. [12] categorized FL into horizontal FL, vertical FL, and federated transfer learning based on data features and samples. In 2022, Wu et al. [13] introduced the application of graph neural networks (GNN) in FL, proposing a federated GNN framework called FedPerGNN. This framework incorporates personalized advantages from GNN into FL, allowing for privacy-preserving exploration of decentralized graph data.

Traditional machine learning methods such as linear models (LM), decision trees (DT), and support vector machine models (SVM) have been applied to FL. Various approaches, including federated linear algorithms [14], federated tree models [15], [16], and federated support vector machines [17], have been proposed. As FL gained attention in research, scholars integrated expertise from the Internet of Things (IoT) and foundational mathematics to enhance and refine the basic framework. Examples include the introduction of the FL algorithm based on global momentum (FedCNM) for IoT [18], and the development of differential privacy FL algorithms based on function mechanisms [19]. Numerous other methods for improving FL have been proposed [20], [21], [22], [23], [24].

In 2020, Li et al. summarized the characteristics of existing FL approaches [25], followed by a detailed discussion of the opportunities and challenges faced by FL in practical applications by Mammen [26]. The application areas of FL have expanded to fields with high requirements for data security and privacy, such as industrial drones [27], smart healthcare [28], [29], [30], and finance and insurance [31]. The unpredictable prospects of FL, particularly in healthcare applications, where the stakes are high, have been highlighted [32]. In healthcare [33], where the privacy of personal data is crucial, FL plays a vital role. For instance, in DNA sequencing, collaboration among multiple hospitals is required to inform patients about diseases. FL can facilitate joint learning on diverse datasets from multiple hospitals, creating a federated model encompassing various hospital data. Simultaneously, the DNA repositories of each hospital and patient DNA sequences remain mutually unknown, ensuring data and privacy security for all parties involved. In 2021, Zhang et al. [34] presented broader applications of FL, providing a systematic introduction to existing work in five aspects: data partitioning, privacy mechanisms, machine learning models, communication architectures, and system heterogeneity. Subsequently, Li et al. [35] reviewed the challenges and future research directions of FL, summarizing its characteristics and practical applications.

B. FEDERATED OPTIMIZATION FOR COMMUNICATION COST AND DATA HETEROGENEITY

In practical scenarios of FL, the multitude of devices storing data and the contradiction between data heterogeneity

and the generalization of global models, leading to non-independent and identically distributed (Non-IID) datasets, is common. Lizhi-Peng and Yong [36] proposed a class-balanced federated learning (CBFL) method based on data generation to address this issue. CBFL aims to balance class distribution in client data through data generation techniques. It incorporates a class-balanced sampler and data generator, where the sampler prioritizes sampling classes with insufficient client data, and the generator produces virtual data for sampled classes to balance class distribution and aid subsequent model training.

In each round of global model training in FL, each participant may send different model parameter updates to the server, resulting in significant communication overhead. Researchers have proposed methods building on FedAvg to improve communication efficiency. In 2017, Konecný [37] introduced two strategies for updating model parameters, presenting structured updates (SU), which directly learns updates from a limited parameter space. It utilizes combinations of quantization, random rotation, and sub-sampling to compress client data. In 2020, Nishio and Yonetani [38] addressed the issue of long communication times in FL and limited client computational resources. They proposed a new method called FedCS, solving the problem of resource-constrained client selection, allowing the server to aggregate updates from multiple clients and accelerate performance improvements in ML models.

Given the scenario of Non-IID data in FL [39], personalized FL methods have been proposed. In 2023, Long et al. [40] introduced personalized FL, presenting a novel multi-center aggregation mechanism grouping client model parameters. It learns multiple global models as cluster centers from data and determines the optimal match between users and centers to send more appropriate model parameters. The same year, Mu et al. [41] proposed FedProc to address imbalances in data distribution. FedProc utilizes a global model to correct local training for each client, designing a local network structure and a global model contrast loss to regulate local model training, ensuring local objectives align with global objectives. Yanhua and Yahui [42] introduced FL based on meta-distillation, combining knowledge distillation and meta-learning with FL. In each global iteration, the local model of each client distills the global model while providing feedback on its own situation to the central server, enabling continuous updates and obtaining an improved global model for personalized learning. Addressing heterogeneous client data, Khojir et al. [43] proposed the FedShare, considering a secure multiparty computation setting where clients use additive secret sharing to model multiple servers. They demonstrated that as long as there are at least two non-colluding servers, the solution can provide secure aggregation. Additionally, mathematical proofs indicated that the secure aggregated model at the end of each training round is identical to the model provided by FedAvg, with efficient communication and computation.

In 2023, Sai and Tianrui [44] addressed the high communication costs and client heterogeneity issues in FL by proposing an algorithm optimized for communication costs. The server receives generated models from clients, generates simulated data, trains a global model with the simulated data, and communicates with clients only once in this process. Clients fine-tune their models with the global model to resolve client heterogeneity issues. Gong and Gao [45] introduced an adaptive FL algorithm based on evolutionary strategy. Each client is treated as an individual in the evolutionary strategy, adapting to generate different personalized sub-models through global optimization. They also introduced network pruning, where clients receive model parameters from the central server, perform chromosome-guided local network pruning and subnetwork generation, and send evolved chromosomes and corresponding sub-networks to the server for iterative updates. Consequently, each client establishes a subnet adapted to its private data distribution, optimizing it using evolutionary strategy.

C. APPLICATION OF SWARM INTELLIGENCE TECHNIQUES IN FEDERATED LEARNING

With the widespread adoption of swarm intelligence techniques in recent years, there is an increasing interest in applying such algorithms to FL. Particle swarm optimizer (PSO) [46] has demonstrated advantages such as fast convergence and good results for complex optimization problems or non-convex problems. In recent years, there has been a surge in research combining machine learning with PSO, primarily for optimizing hyperparameters of neural networks [47], [48], [49], aiming to improve model accuracy in classification tasks.

In the context of distributed learning environments, only a few studies have explored the combination of PSO and FL. Qolomany et al. [50] proposed a PSO based technique to optimize hyperparameters of client machine learning models in a FL environment. However, PSO technology is not employed in the training process of FL. Park et al. [51] introduced the FedPSO, which combines PSO with the training process of FL. Nevertheless, FedPSO simplifies the combination of PSO with FL, assuming an idealized IID client data distribution. The algorithm proves to be extremely unstable in the case of Non-IID data. Current applications of swarm intelligence techniques in FL are limited to dealing with IID data, overlooking the critical issues of communication cost and data heterogeneity mentioned earlier.

DBO primarily simulates the behaviors of dung beetles, such as rolling balls, dancing, foraging, stealing, and reproducing. It utilizes the rolling behavior of dung beetles for iterative position updates, introduces a boundary selection strategy to simulate the regions for female dung beetles to reproduce and for small dung beetles to forage. The algorithm then iterates the position information guided by thief dung beetles to seek the optimal solution. It is characterized

by strong optimization capabilities and fast convergence, making it widely applied in the model optimization process of ML after its proposal [52], [53].

In the field of air quality prediction models, Duan et al. [54] combined DBO, convolutional neural network (CNN), and long short-term memory (LSTM) to avoid the model's dilemma. They determined optimal hyperparameters to achieve higher predictive accuracy. Additionally, DBO has found extensive applications in various domains such as function optimization, image processing, ML [55], energy optimization [56], [57], [58], and path planning. Numerous researchers have made improvements to DBO parameters and applied them in practical scenarios. Zhang and Zhu [59] used fragmental linear chaotic mapping to generate the initial dung beetle population. They employed an adaptive nonlinear producer rate decay model to control the number of producers and applied dimension learning-enhanced foraging search strategies. They proposed an improved dung beetle optimization (IDBO), which was applied to optimize a backpropagation neural network for predicting five mechanical performance parameters of heat-treated larch sawn timber.

In this paper, FedDBO is proposed to solve the problem of high cost of federal learning communication as well as data heterogeneity, and the main work done can be described by the following four points:

- Combining DBO with the FL training process, FedDBO transforms the communication data format in FL from model parameters to model scores (accuracy). It selects a subset of client models based on their model scores and uploads their model parameters to the server, significantly reducing the issues of high communication costs and low communication efficiency in FL.
- Proposing model retraining strategies (MRS): Constructing server data and conducting secondary training optimization on the server. MRS not only alleviates the impact of data heterogeneity on FL but also improves model accuracy. Furthermore, this strategy guides the optimization of federated models in a controllable manner.
- Elaborating on the convergence of FedDBO's model and providing mathematical proofs. The proofs demonstrate that the aggregated model obtained by FedDBO converges to the aggregated model obtained by FedAvg.
- Simulating different data heterogeneity scenarios on the MNIST, FashionMNIST, and CIFAR-10 datasets, comparing the accuracy of FedDBO, FedAvg, FedShare, and FedPSO. Experimental results show that the model obtained by FedDBO has higher accuracy. The performance losses of each algorithm under different data heterogeneity scenarios are compared, demonstrating that FedDBO not only achieves higher accuracy compared to the other three algorithms but also exhibits higher robustness.

III. BASIC THEORIES

A. FEDERATED LEARNING

Federated learning (FL) is a decentralized ML approach designed to enable multiple devices or clients to locally train models without the need to centralize the original dataset on a single server as shown in Fig. 1.

FedAvg is the most typical FL algorithm. In this algorithm, each local device or client trains a local model. Subsequently, the parameters of these local models are transmitted to the central server. The central server aggregates these parameters through weighted averaging to obtain the global model. The global model is then transmitted back to the local devices, and this process is iterated further. The detailed procedure of FedAvg is illustrated in Alg. 1.

Algorithm 1 FedAvg; K = Number of Clients; E = Client Total Epochs; Select Client by the C Ratio

```
function SERVEREXECUTES
Initialize W_0
for each round t = 1, 2, ... do
    S_t ← random set of max (C · K, 1) clients
end for
for each client k ∈ S_t in parallel do
    W^k_{t+1} ← ClientUpdate(k, W_t)
    W_{t+1} ← averaging of the collected weights
    W_{t+1} of S_t clients
end for
function ClientUpdate(k, W_0)
Perform learning process on client k with weight W until the client reaches E epoch
W_{t+1} ← updated weight after learning
return W_{t+1} to server
```

B. DUNG BEETLE OPTIMIZER

DBO consists of a group of dung beetles, categorized into four types: rolling ball dung beetle, breeding dung beetle, small dung beetle, and stealing dung beetle. They perform rolling behavior, breeding behavior, foraging behavior, and stealing behavior, respectively. The specific processes are as follows:

Rolling Ball: The rolling ball sub-population is further divided into obstacle-free mode and obstacle mode. Obstacle-free dung beetles iteratively update their positions based on the intensity of the light source. The position update of a dung beetle is expressed as follows:

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \tau \cdot k \cdot \boldsymbol{x}_i^{t-1} + z \cdot \boldsymbol{\Delta x}. \tag{1}$$

$$\boldsymbol{\Delta x} = \mid \boldsymbol{x}_i^t - \boldsymbol{x}_i^{worst} \mid . \tag{2}$$

Here, t denotes the current iteration number, $\boldsymbol{x}_i^t$ represents the position information of the i-th dung beetle at the k-th iteration, $k \in (0,0.2]$ indicates a constant value that serves as a deviation coefficient, z represents a constant value belonging to the interval$(0,1)$,τ is a natural coefficient assigned as either

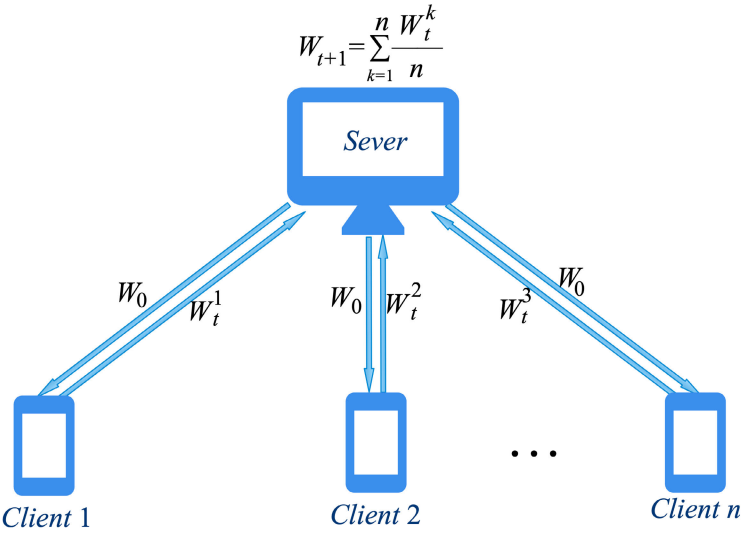

FIGURE 1. Federated learning training process: (1) The server initializes the global model W_0 and sends it down to each client. (2) The i-th client trains W_0 on its own local data to obtain the local model W_t^i and uploads it to the server. (3) The server uses FedAvg to aggregate the received local model parameters to update the global model to get W_{t+1}.

-1 or 1, and $\boldsymbol{x}_i^{worst}$ denotes the globally worst position, $\boldsymbol{\Delta x}$ used to simulate changes in light intensity.

In the presence of obstacles, the beetle will use a tangent function to simulate the dancing behavior and iteratively update its position. The position update in this case is given by Eq.(3):

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \tan\theta \mid \boldsymbol{x}_i^t - \boldsymbol{x}_i^{t-1} \mid . \tag{3}$$

where θ is the deflection angle belonging to $[0,\pi]$.

Breeding: To simulate the breeding process of female dung beetles and determine the boundaries of the breeding subpopulation, as shown in Eq.(4):

$$\begin{cases} \boldsymbol{Lb}^* & = \max\{\boldsymbol{x}_{lbest}^t \cdot (1-R), \boldsymbol{Lb}\} \\ \boldsymbol{Ub}^* & = \min\{\boldsymbol{x}_{lbest}^t \cdot (1+R), \boldsymbol{Ub}\} \end{cases} \tag{4}$$

where $\boldsymbol{x}_{lbest}^t$ represents the current local optimal position, $\boldsymbol{Lb}$ and $\boldsymbol{Ub}$ respectively represent the lower and upper bounds of the optimization problem, $\boldsymbol{Lb}^*$ and $\boldsymbol{Ub}^*$ respectively represent the lower and upper limits of the oviposition area, $R = 1 - t/T_{\max}$, $T_{\max}$ represents the maximum number of iterations.

The oviposition area dynamically adjusts with the number of iterations. Therefore, the position of the larva ball is also dynamic, denoted as $\boldsymbol{B}_i^t$ for the position of the i-th larva ball at the t-th iteration. The position of the larva ball must be strictly limited to the oviposition area, defined as follows:

$$\boldsymbol{B}_i^{t+1} = \boldsymbol{x}_{gbest}^t + \boldsymbol{A_1} \times (\boldsymbol{B}_i^t - \boldsymbol{Lb}^*) + \boldsymbol{A_2} \times (\boldsymbol{B}_i^t - \boldsymbol{Ub}^*). \tag{5}$$

Here, $\boldsymbol{B}_i^t$ represents the position information of the i-th nurturing ball at the t-th iteration, and $\boldsymbol{A_1}$,$\boldsymbol{A_2}$ denote two independent random vectors of size $1 \times D$, where D representing the dimensions of the optimization problem.

Foraging: The optimal foraging region for foraging beetles is also dynamically updated, where $\boldsymbol{x}_{lbest}^t$ is the current population's local optimal position, Lb^l and Ub^l are the lower and upper bounds of the foraging region, defined as Eq.(6):

$$\begin{cases} \boldsymbol{Lb}^l & = \max\{\boldsymbol{x}_{gbest}^t \cdot (1-R), \boldsymbol{Lb}\} \\ \boldsymbol{Ub}^l & = \min\{\boldsymbol{x}_{gbest}^t \cdot (1+R), \boldsymbol{Ub}\} \end{cases} \tag{6}$$

$\boldsymbol{x}_{gbest}^t$ represents the global optimal position, and the position of the small beetles is updated with the defined boundary positions, specifically defined as Eq.(7):

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + \zeta \cdot (\boldsymbol{x}_{gbest}^t - \boldsymbol{Lb}^l) + \boldsymbol{\Gamma} \times (\boldsymbol{x}_i^t - \boldsymbol{Ub}^l). \tag{7}$$

where ζ represents a random number following a normal distribution, and $\boldsymbol{\Gamma}$ represents a random vector belonging to the range (0,1).

Stealing: In the population of dung beetles, there are some stealing beetles. Their position is updated as follows:

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_{lbest}^t + S \cdot \boldsymbol{g} \times (\mid \boldsymbol{x}_i^t - \boldsymbol{x}_i^{gbest} \mid + \mid \boldsymbol{x}_i^t - \boldsymbol{x}_i^{lbest} \mid). \tag{8}$$

Here, $\boldsymbol{g}$ is a random vector of size $1 \times D$ following a normal distribution, and S represents a constant value.

IV. METHODOLOGY

A. FEDDBO

The DBO has the advantages of high parallelism, powerful global search capabilities, and strong adaptability, making it highly suitable for scenarios with heterogeneous clients in FL. In order to reduce communication costs and enhance the robustness of the FL model training process, this paper proposes FedDBO. FedDBO applies the DBO to the training process of Federated Learning, addressing the shortcomings of traditional FL training, where the server transfers

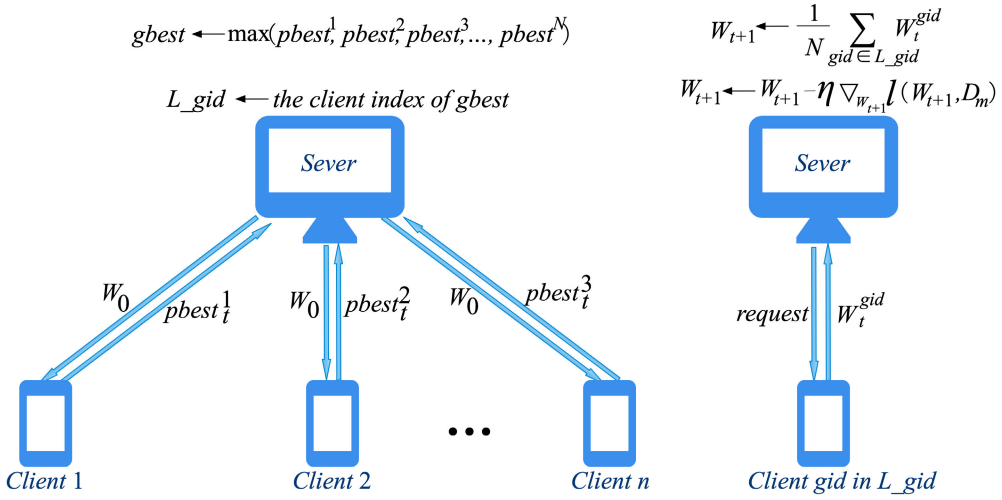

FIGURE 2. FedDBO training process:In the client update stage(left). The i-th client receives the D_s from the server, calculates the accuracy of its own local model with respect to the D_s as the model score denoted as $pbest_t^i$ (one round of dung beetle update with E round of local iteration is included). (3)Wait for the server to request the model parameters of the top N clients. In the server aggregation phase(right), after receiving the model parameters, the top N highest model scores are uniformly denoted as $gbest$, and their corresponding client indexes are L_gid. (2)Request the model parameter W_t^{gid} for the top N clients and update the model to W_{t+1} using the FedAvg aggregation parameters. (3)Train W_{t+1} using the server's own metadata to obtain a new model W_{t+1} trained on the global data.

model parameters with a focus on reducing communication costs. Instead, FedDBO introduces a novel approach by exchanging model scores rather than model parameters, aiming to improve the efficiency of communication and enhance the robustness of model training in FL, especially in heterogeneous client environments.

As shown in Fig. 2, FedDBO employs an alternative approach to the traditional FL method. At the beginning of the training process, the server sends the scoring dataset D_s (the selection of D_s is discussed specifically in Sec. IV-B). Then, the clients make updates and upload the accuracy as model scores to the server. The server selects the top N clients, asks them for model parameters, and updates the global model for the round by averaging the FedAvg parameters.

In the traditional FL framework, the transmission involves all client models' parameters. However, in the FedDBO framework, it only requires uploading the model scores on the evaluation model and the model parameters of N clients with higher scores, which significantly reduces communication costs and improves training efficiency.

During the training process, to leverage the guidance of both the global optimal model and the local optimal model on the client model training, analogous to the DBO algorithm, the update of the client neural network model in FedDBO involves one round of roach updates and E rounds of local iteration. Specifically, in the obstacle-free mode, the update process for the rolling ball model is as follows:

$$\boldsymbol{W}_i^{t+1} = \boldsymbol{W}_i^t + \tau \cdot k \cdot \boldsymbol{W}_i^{t-1} + z \cdot \boldsymbol{\Delta W}. \tag{9}$$

$$\boldsymbol{\Delta W} = \mid \boldsymbol{W}_i^t - \boldsymbol{W}_i^{worst} \mid . \tag{10}$$

where $\boldsymbol{W}_i^t$ riepresents the local optimal model of the rolling ball model in the tth iteration, and $\boldsymbol{W}_i^{worst}$ represents the global worst model, and $\boldsymbol{\Delta W}$ is used to simulate the variation of light intensity.

In the obstacle mode, the model is updated using a tangent function to simulate dancing behavior and iterate the model's position. The model update in this case is shown in Eq.(11):

$$\boldsymbol{W}_i^{t+1} = \boldsymbol{W}_i^t + \tan\theta \mid \boldsymbol{W}_i^t - \boldsymbol{W}_i^{t-1} \mid . \tag{11}$$

To simulate the process of roach reproduction and determine the boundaries of the breeding model's position, $\boldsymbol{Lb}^*$ and $\boldsymbol{Ub}^*$ represent the lower and upper limits of the spawning area, respectively:

$$\begin{cases} \boldsymbol{Lb}^* = \max\{\boldsymbol{W}_{\boldsymbol{lbest}}^t \cdot (1-R), \boldsymbol{Lb}\} \\ \boldsymbol{Ub}^* = \min\{\boldsymbol{W}_{\boldsymbol{lbest}}^t \cdot (1+R), \boldsymbol{Ub}\} \end{cases} \tag{12}$$

where $\boldsymbol{W}_{\boldsymbol{lbest}}^t$ represents the current local optimum model of the breeding model.

The spawning region adjusts dynamically with the number of iterations, and thus, the position of the pupa is also dynamic. $\boldsymbol{W}_i^t$ represents the updated position of the i-th pupa in the t-th iteration, and the position of the pupa must be strictly confined to the spawning region, defined as in Eq.(13):

$$\boldsymbol{W}_i^{t+1} = \boldsymbol{W}_{\boldsymbol{gbest}}^t + \boldsymbol{A_1} \times (\boldsymbol{W}_i^t - \boldsymbol{Lb}^*) + \boldsymbol{A_2} \times (\boldsymbol{W}_i^t - \boldsymbol{Ub}^*). \tag{13}$$

The foraging model is similar to the boundary region of the reproduction model in that its optimal foraging region is also dynamically updated, with $\boldsymbol{W}_{\boldsymbol{lbest}}^t$ being the local optimal position of the current model, and $\boldsymbol{Lb}^{\boldsymbol{l}}$ and $\boldsymbol{Ub}^{\boldsymbol{l}}$ being the lower and upper bounds, respectively, of the position update region of the foraging model, which are defined in

equation (14) below:

$$\begin{cases} \boldsymbol{Lb}^{\boldsymbol{l}} = \max\{\boldsymbol{W}_{\boldsymbol{gbest}}^{\boldsymbol{t}} \cdot (1-R), \boldsymbol{Lb}\} \\ \boldsymbol{Ub}^{\boldsymbol{l}} = \min\{\boldsymbol{W}_{\boldsymbol{gbest}}^{\boldsymbol{t}} \cdot (1+R), \boldsymbol{Ub}\} \end{cases} \tag{14}$$

Here, $\boldsymbol{W}_{\boldsymbol{gbest}}^{\boldsymbol{t}}$ represents the global optimal model. The model position is updated according to the defined boundary position, specifically defined as follows in Eq.(15):

$$\boldsymbol{W}_{\boldsymbol{i}}^{\boldsymbol{t+1}} = \boldsymbol{W}_{\boldsymbol{i}}^{\boldsymbol{t}} + \zeta \cdot (\boldsymbol{W}_{\boldsymbol{gbest}}^{\boldsymbol{t}} - \boldsymbol{Lb}^{\boldsymbol{l}}) + \boldsymbol{\Gamma} \times (\boldsymbol{W}_{\boldsymbol{i}}^{\boldsymbol{t}} - \boldsymbol{Ub}^{\boldsymbol{l}}). \tag{15}$$

where $\boldsymbol{W}_{\boldsymbol{i}}^{\boldsymbol{t}}$ represents the local optimal model i of the foraging model at the t-th iteration.

In the overall client model population, there will be the presence of some stealing models, and their position updates are given by Eq.(16):

$$\boldsymbol{W}_{\boldsymbol{i}}^{\boldsymbol{t+1}} = \boldsymbol{W}_{\boldsymbol{lbest}}^{\boldsymbol{t}} + S \cdot \boldsymbol{g} \times (\mid \boldsymbol{W}_{\boldsymbol{i}}^{\boldsymbol{t}} - \boldsymbol{W}_{\boldsymbol{i}}^{\boldsymbol{gbest}} \mid + \mid \boldsymbol{W}_{\boldsymbol{i}}^{\boldsymbol{t}} - \boldsymbol{W}_{\boldsymbol{i}}^{\boldsymbol{lbest}} \mid). \tag{16}$$

where $\boldsymbol{W}_{\boldsymbol{lbest}}^{\boldsymbol{t}}$ represents the local optimal position of the current stealing model.

B. MODEL RETRAINING STRATEGY

In the FedDBO framework, the choice of the number N of clients uploading model parameters affects the training results of FL. A smaller N indicates that the server receives fewer client models in each training round, resulting in lower communication costs. However, selecting only a subset of client models will lead to the exclusion of other clients from training the global model, allowing a few clients with good performance on the scoring model to dominate the global model's training. This can result in unfairness in FL.

When N is small, such as N equals 1, the server only accepts the model from one client. However, a single client's data cannot represent all clients' data, especially in a Non-IID data environment. Simply setting a single client's model as the global model may lead to inconsistency between the data distribution used to train the global model and the actual data distribution, resulting in a decrease in model accuracy. For example, if the MNIST dataset is used for training but under extremely non-independent and identically distributed (Non-IID) conditions, where each client only has one class of data, the local model trained by a single client will have poor generalization. If only the model from a single client is sent to all clients as the global model, the global model will only improve the recognition capability for a specific class of label data. The final global model obtained through this training process will also lack generalization ability [60].

As N increases, the server receives more client models in one training round, which helps fully utilize client data and train a more generalizable global model. However, as N increases, the required communication costs also increase. When N equals the number of clients, the communication cost of FedDBO will be the same as FedAvg. To reduce communication costs, decrease the number of client models the server needs to receive, and alleviate the impact of Non-IID problems on federated learning training, FedDBO incorporates a model retraining strategy at the server side.

Specifically, in this paper, a server dataset D_m and a scoring dataset D_s are constructed. D_m is an independently and identically distributed (IID) subset of the overall training data D,with the ratio of the size of D_m to D denoted as α .D_s is an IID subset of D_m. In general, the server has better information resources and can construct D_m based on historical data and its own situation. Because only clients with the highest accuracy on D_m can send models to the server, D_s effectively represents the direction of client model optimization in FedDBO.

During each communication round, after receiving the client models, the server utilizes dataset D_m to retrain the models as follows:

$$W_t = W_{t-1} - \eta \nabla l(W_t, D_m). \tag{17}$$

Here,W_t represents the aggregated model at the server during the communication round, η is the learning rate, and D_m should reflect the data distribution of the overall client training data. The detailed process of FedDBO is shown in Alg. 2.

C. CONVERGENCE ANALYSIS

This section aims to demonstrate that the aggregated model obtained by FedDBO closely approximates FedAvg at the end of each training round. The strategy involves proving the convergence for an arbitrary client's training round and then extrapolating this evidence to all rounds, ultimately establishing that the FedDBO aggregated model converges to the FedAvg aggregated model.

Assume $C = \{C_1, C_2, C_3, \ldots, C_n\}$ to be a group of clients participating in the current training round, and $b = \{b_1, b_2, b_3, \ldots, b_n\}$ as the dataset of clients involved in the current training. Simultaneously, consider any client's training round denoted as t. Each client's model, obtained through training with local data over rounds of iterations, is denoted as $W_1^t, W_2^t, W_3^t, \ldots W_n^t$, where $l(W_0, b_k)$ represents the local loss function of the k-th client. The server initializes the global model as W_0 and aggregates the uploaded model parameters from clients after receiving them. The resulting aggregated global model is denoted as W^{t+1}.

Definition 1 (Locally Bounded Gradient): In this paper, considering the use of accuracy instead of model scores, in the context of non-convex optimization problems with the objective of minimizing the loss function $l(W_0, b_k)$, there exists a constant M, for all $W_0$and k, such that $\|(W_0, b_k)\| \leq M$.

Definition 2 (Bounded Learning Rate): There exists a constant M such that the learning rate η satisfies $\eta \leqslant \frac{1}{LM}$, where L is the Lipschitz constant.

Theorem 1: Let P be the aggregated result model by the server in t rounds of FedAvg, and R be the aggregated result model by the server in the same round of FedDBO, $\exists \sigma > 0$ such that $\mid P - R \mid < \sigma$.

Algorithm 2 FedDBO

function SERVEREXECUTES
Initialize W_0, $pbest$, $gbest$, gid;
for each round $t = 1, 2, \ldots$ **do**
  for each client k **in parallel do**
    $pbest \leftarrow$ ClientUpdate(k, W_t,gid)
  end for
end for
if $gbest > pbest$ **then**
  $pbest \leftarrow gbest$ W_{k+1}^t of S_t clients
  $gid \leftarrow k$
  $W_{t+1} \leftarrow$GetBestModel(gid)
  $W_{t+1} \leftarrow W_t - \eta \nabla l(W_t, D_m)$
end if
function CLIENTUPATE(k,W_t,gid)
Initialize W,W_{worst}, W_{gbest}, τ, A_1,A_2,ζ,Γ,S,g
$\beta \leftarrow$(split ρ_k into batches of size B)
l=1:$t+1$
if $l \in$PbroodRollingclient **then**
  $W_i^{t+1} = W_i^t + \tau \cdot k \cdot W_i^{t-1} + z \cdot \mid W_i^t - W_{worst}^t \mid$
  else
  $W_i^{t+1} = W_i^t + \tan\theta \mid W_i^t - W_i^{t-1} \mid$
end if
if $l \in$PbroodBallclient **then**
  $W_i^{t+1} = W_{lbest}^t + A_1 \cdot (W_i^t - Lb^*) + A_2 \cdot (W_i^t - Ub^*)$
end if
if $l \in$PSmallBallclient **then**
  $W_i^{t+1} = W_i^t + \zeta \cdot (W_i^t - Lb^l) + \Gamma \cdot (W_i^t - Ub^l)$
end if
if $l \in$Thiefclient **then**
  $W_i^{t+1} = W_{lbest}^t + S \cdot g \cdot (\mid W_i^t - W_i^{gbest} \mid + \mid W_i^t - W_{lbest}^t \mid)$
end if
for each client epoch i from 1 to E **do**
  for batch $b \in B$ **do**
    $W_{t+1} = W_t - \eta \nabla l(W_t, b)$
  end for
end for
return $pbest$ to sever
function GETBESTMODEL (gid)
request to Client(gid)
receive W_{t+1} from Client
return W_{t+1} to server

Proof: Let the Denclue algorithm estimate the density function using the improved Gaussian density function be f.

$$W_k = W_0 - \eta \nabla l(W_0, b_k). \tag{18}$$

In this context, η represents the learning rate, and $\nabla l(W_0, P_k)$ corresponds to the gradient of the local loss function for the k-th client.

$$\eta \nabla l(W_0, b_k) \longleftarrow W_0 - W_k. \tag{19}$$

According to Def. 1 and Def. 2, we can derive that,

$$\mid W_0 - W_k \mid \leq \frac{1}{LN} \cdot N = \frac{1}{L}. \tag{20}$$

Let $\delta = \frac{1}{L}$ be such that W_k converges to W_0,

Therefore, in the FedAvg training process, the globally aggregated model at the server is

$$P = \sum_{k=1}^{n} \frac{W_k}{n} \rightarrow W_0. \tag{21}$$

As a result,

$$\mid P - W_0 \mid \leq \delta. \tag{22}$$

For FedDBO, the server first initializes the global model W_0 using a subset D_s of its own data D_m and distributes it to various clients. The clients then calculate their model scores using W_0 and upload them to the server. The server selects the top N clients with the highest scores and requests their model parameters. Therefore, the globally aggregated model in FedDBO, as aggregated by the server, is:

$$W_g^{t+1} = \sum_{k=1}^{N} \frac{W_k}{N}. \tag{23}$$

From Eq.(21), $W_g^{t+1} \rightarrow W_0$, then the server utilizes its own data D_m to perform a second round of iterative training on the global model W_g^{t+1}, yielding:

$$R \longleftarrow W_g^{t+1} - \eta \nabla l(W_g^{t+1}, D_m). \tag{24}$$

Then, from Def. 1 and Def. 2, we can obtain:

$$R \longrightarrow W_g^{t+1} \longrightarrow W_0. \tag{25}$$

$$\mid R - W_0 \mid \leqslant \delta. \tag{26}$$

From Eq.(25) and Eq.(26), it can be observed that:

$$\mid P - R \mid \leqslant \delta. \tag{27}$$

□

Hence, P converges to R, and both converge to the initially initialized global model W_0.

V. EXPERIMENTATION AND ANALYSIS

A. EXPERIMENTAL ENVIRONMENT

The experimental environment was set up with an Apple M2 8-core central processor running MacOS 14.0. The experiments were implemented using the Python language and the PyTorch neural network framework. Python version 3.11 was used, and the PyTorch version employed was 2.0.1. The specific details of the experimental environment are presented in the following table.

TABLE 1. Experimental environment parameter settings.

Equipment	Parameter
CPU	Apple M2
Hard Disk	512.0GB
Internal Storage	8.0GB
IDE	Jupyter Notebook 6.5.4
Computing Environmentk	Python3.11

B. EXPERIMENTAL DATA

This paper conducts experiments to evaluate FedDBO using the MNIST dataset, FashionMNIST dataset, and CIFAR-10 dataset. The evaluation involves comparing FedDBO with FedAvg, FedShare, and FedPSO algorithms under different data heterogeneity scenarios. Additionally, experiments are conducted to analyze the impact of server data on FedDBO. The accuracy of FedDBO is compared across different scenarios on the MNIST dataset, FashionMNIST dataset, and CIFAR-10 dataset.

The MNIST dataset consists of 60,000 training samples and 10,000 test samples, each representing a grayscale hand-written digit image of size 28 pixels $\times$ 28 pixels. The FashionMNIST dataset comprises 70,000 images of 10 different categories of fashion items, divided into 60,000 training samples and 10,000 test samples. Each sample is a grayscale image of size 28 pixels $\times$ 28 pixels.

The CIFAR-10 dataset is composed of 50,000 training samples and 10,000 test samples, with each sample being a color image of size 32 pixels $\times$ 32 pixels. In contrast to the samples in the MNIST and FashionMNIST datasets, CIFAR-10 samples represent real-world objects with varying features, sizes, and substantial noise, making object recognition more challenging.

To simulate the data distribution environment in FL, the paper utilizes 100 clients as training nodes. Following the dataset partitioning approach outlined in literature [57], the training data is divided into IID, NonIID_one and NonIID_two distributions.

(1) IID: The training data is uniformly and randomly distributed among each client. Taking the partitioning of the MNIST dataset as an example, each class in the dataset has 6,000 training samples. The 10 class datasets are distributed randomly and uniformly among the 100 clients, ensuring that each client receives 600 samples with all 10 class labels. This partitioning method ensures that each client's training dataset has diversity and representativeness, contributing to improved model generalization capabilities.

(2) NonIID_one: Each client possesses data with only one class label. Using the partitioning of the MNIST dataset as an example, the dataset is initially divided into 10 groups based on the labels. Each group of data is evenly split into 100 data slices. Subsequently, each client randomly selects 10 data slices from a randomly chosen group, forming the client's training data.

(3) NonIID_two: Each client possesses data with only two class labels. Using the partitioning of the MNIST dataset as an example, the dataset is initially divided into 10 groups based on the labels. Each group of data is evenly split into 200 data slices. Subsequently, each client randomly selects 10 data slices from two randomly chosen groups, forming the client's training data.

TABLE 2. FedAvg and FedShare experimental parameters (data sample size) settings.

Dataset	Data sample size
MNIST(IID)	100
MNIST(NonIID_one)	100
MNIST(NonIID_two	100
FashionMNIST(IID)	100
FashionMNIST(NonIID_one)	1000
FashionMNIST(NonIID_two	1000
CIAFR(IID)	120
CIAFR(NonIID_one)	1200
CIAFR(NonIID_two	1200

C. PARAMETER SETTING

The experimental model adopts a CNN with two 5 $\times$ 5 convolutional layers, each followed by a 2 $\times$ 2 max-pooling layer. Subsequently, three fully connected layers are appended, leading to a final output vector with 10 dimensions.

For the MNIST, FashionMNIST, and CIFAR-10 dataset, batch_size is set to 20, the number of local iterations E is set to 10, and the learning rate $\eta = 0.001$. the ratio of selecting clients for the FedAvg is $B = 1.0$. FedShare is an improved algorithm of FedAvg, which is mainly investigated in the FL NonIID problem in the algorithm, the parameter settings of this algorithm refer to the literature [61], take a number of samples from MNIST, FashionMNIST, and CIFAR-10 datasets to form a global shared dataset G, and then assign each client proportion of the global shared dataset of β and $\beta = 0.05$. To ensure that the experimental data sample size is the same, the FedAvg, FedShare experimental parameters will be set against FedDBO, and the data sample sizes under different distributions of different datasets are shown in the following table.

FedPSO and FedDBO are improved algorithms that combine FedAvg with swarm intelligence techniques. They primarily focus on addressing the challenges of high communication costs and data heterogeneity in FL. The parameters for these algorithms are based on references [60], where a subset of samples from the MNIST, FashionMNIST, and CIFAR-10 datasets is used to form the server data D_m. The ratio of the size of data D_m to data D is set to $\alpha = 0.2$, and the optimal strategy parameter N is set to 10. The experimental parameters for FedPSO and FedDBO are detailed in the Tab3 (further explanations will be provided in Sec. V-D.4)).

D. EXPERIMENTAL PERFORMANCE ANALYSIS

1) ANALYSIS OF NUMERICAL RESULTS

To validate the experimental results of FedAvg, FedShare, FedPSO, and FedDBO on the MNIST dataset, FashionMNIST dataset, and CIFAR-10 dataset, the experiments

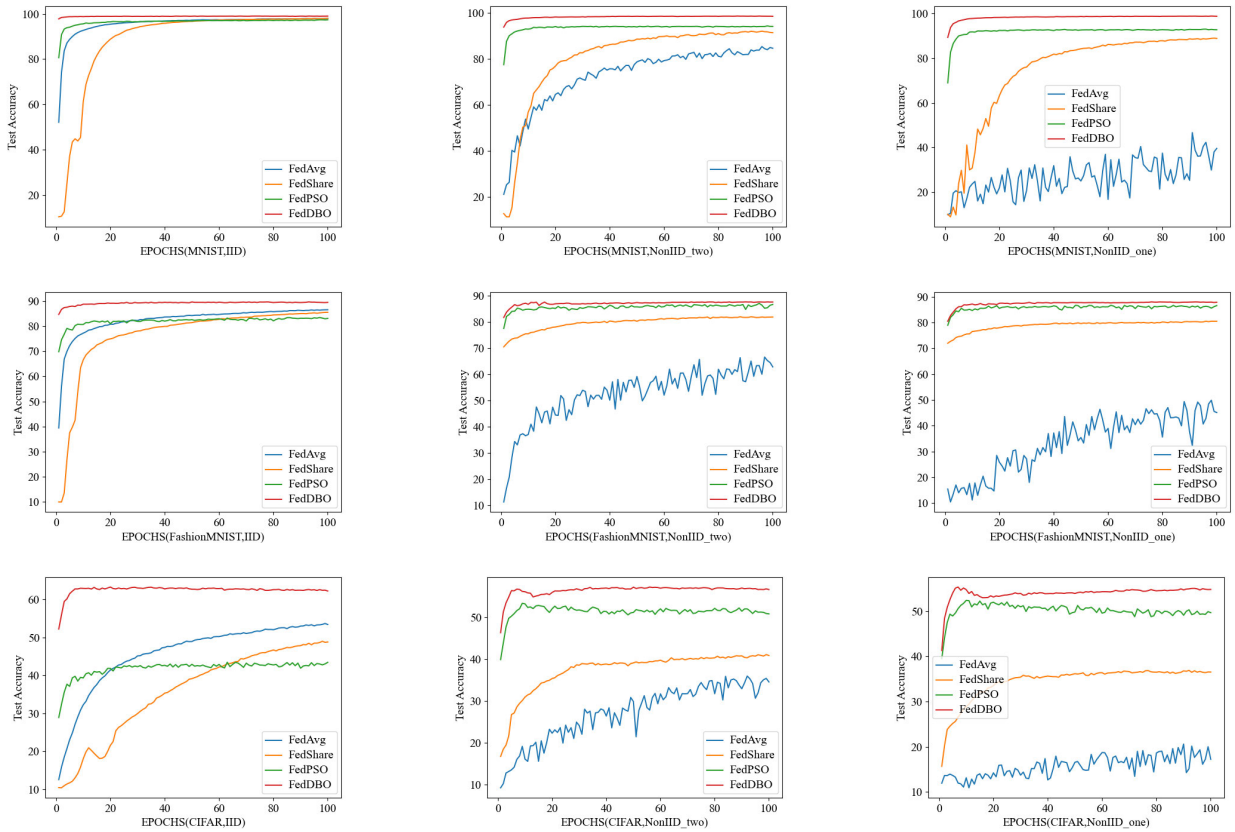

FIGURE 3. Accuracy of FedAvg, FedShare, FedPSO and FedDBO on three datasets under three client data heterogeneity conditions.

TABLE 3. FedPSO and FedDBO experimental parameter (data sample size) settings.

Dataset	Data sample size	The ratio of D_m to D is α
MNIST(IID)	100	0.02
MNIST(NonIID_one)	100	0.02
MNIST(NonIID_two)	100	0.02
FashionMNIST(IID)	100	0.02
FashionMNIST(NonIID_one)	100	0.2
FashionMNIST(NonIID_two)	100	0.2
FashionMNIST(NonIID_one)	100	0.2
CIAFR(IID)	100	0.02
CIAFR(NonIID_one)	100	0.2
CIAFR(NonIID_one)	100	0.2

consider three different data distribution scenarios: IID, NonIID_one, and NonIID_two. The parameters for the experiments are set according to Tab. 2 and Tab. 3. IID data follows an independent and identically distributed distribution, representing a scenario with no client data heterogeneity. NonIID_one and NonIID_two data have non-independent and non-identically distributed characteristics, with NonIID_one having a higher degree of heterogeneity. The experimental results are shown in Fig. 3.

It can be seen that, for the comparison between FedAvg, FedPSO, and FedShare, in the case of IID client data, FedAvg has a lower accuracy in the first few rounds but gradually increases with more iterations. By the 100th iteration, FedAvg performs roughly on par with FedPSO, while FedShare has lower accuracy.

In the case of Non_IID client data, FedAvg's accuracy significantly decreases, and the accuracies of FedShare and FedPSO increase with the number of iterations, reaching a stable state. Particularly, as the data volume increases, and data heterogeneity becomes more severe, FedShare and FedPSO achieve much higher accuracy than FedAvg.

For the proposed FedDBO in this paper, its accuracy is higher than the three aforementioned algorithms in the MNIST, FashionMNIST, and CIFAR-10 datasets, under IID, NonIID_two, and NonIID_one scenarios. Moreover, with an increase in the number of iterations, FedDBO maintains stable accuracy.

2) COMPARISON OF ALGORITHM PERFORMANCE LOSS IN CASE OF DATA HETEROGENEITY

The accuracies of FedAvg, FedShare, FedPSO and FedDBO after 100 rounds of iterations of the experiment for the three datasets as well as for the three data distributions are shown in the Tab. 4 below.

It can be observed that, in the MNIST dataset, when the data distribution changes from IID to NonIID_one, the accuracies of FedAvg and FedShare decrease by 63.62% and 1.88%, respectively. In contrast, the proposed FedDBO experiences only a slight decrease of 0.12%. In the FashionMNIST and CIFAR-10 datasets, the accuracy drop of FedDBO is within 2%, significantly less than the performance loss observed in FedAvg and FedShare.

TABLE 4. Accuracy of FedAvg, FedShare, FedPSO and FedDBO in three datasets after 100 rounds of training.

	MNIST			FashionMNIST			CIFAR-10		
	IID	NonIID_two	NonIID_one	IID	NonIID_two	NonIID_one	IID	NonIID_two	NonIID_one
FedAvg	97.93	85.72	34.31	86.47	62.81	45.08	53.40	34.56	17.21
FedShare	97.93	96.39	96.05	85.46	81.86	80.52	48.79	40.89	36.53
FedPSO	97.32	98.18	98.15	83.04	86.67	86.62	43.43	50.85	49.67
FedDBO	**98.96**	**98.78**	**98.84**	**89.40**	**87.58**	**87.89**	**62.21**	**56.67**	**54.81**

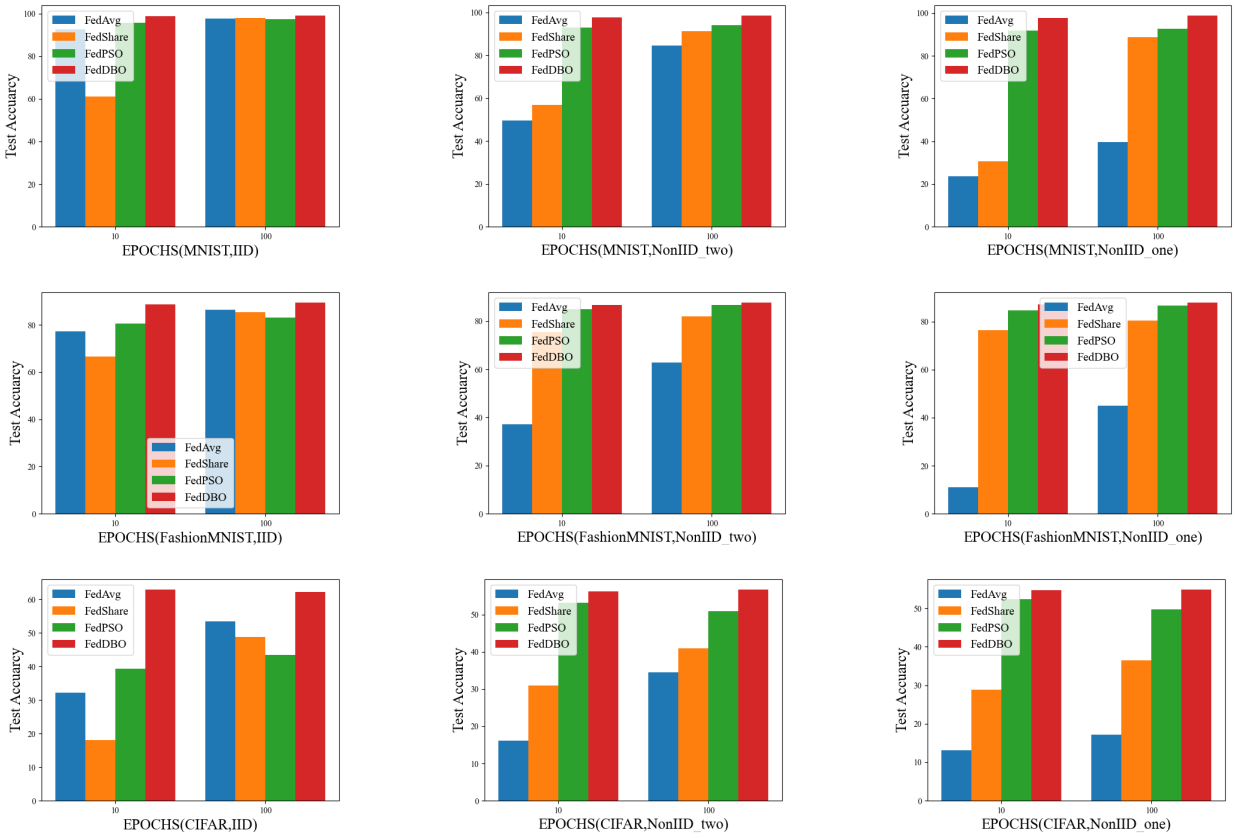

FIGURE 4. Accuracy of FedAvg, FedShare, FedPSO and FedDBO on three datasets under three client-side data heterogeneity conditions with 10 and 100 iterations respectively).

Compared to FedPSO, in the MNIST, FashionMNIST, and CIFAR-10 datasets, as the degree of data heterogeneity increases, the performance loss of FedPSO is lower than that of FedDBO. However, FedDBO consistently achieves significantly higher accuracy than FedPSO.

In summary, the proposed FedDBO exhibits higher performance loss compared to FedPSO as the degree of data heterogeneity increases, but its accuracy remains significantly higher than the other three algorithms, regardless of the dataset or the level of data heterogeneity.

3) ANALYZING THE INFLUENCE OF THE NUMBER OF ITERATIONS ON THE ALGORITHM'S IMPACTABILITY

The number of iterations m has a certain impact on the accuracy of FedDBO. In order to compare the influence of m on accuracy, this study sets the number of iterations to 10 and 100 for each of the four algorithms. The experimental results are shown in the Fig. 4.

It can be seen that compared to FedAvg, FedShare, and FedPSO, FedDBO has higher accuracy for both iteration numbers m at 10 and 100. Moreover, the accuracy of FedDBO remains relatively stable as m changes from 10 to 100, without significant fluctuations. This indicates that FedDBO not only outperforms the other three algorithms in terms of accuracy but also exhibits strong robustness with less performance loss.

4) IMPACT ANALYSIS OF α ON FEDDBO

α represents the size of the required data samples in the experiment, and the size of the data samples has a significant impact on the accuracy of FedDBO. To analyze the specific impact of data sample size on its accuracy, this paper conducted experiments with four different settings of $\alpha = 0.002$, 0.02, 0.1, and 0.2 under different data distributions on various datasets.The experimental results are shown in the Fig. 5.

It can be seen that, under the IID setting of the MNIST dataset, as well as the IID settings of the FashionMNIST

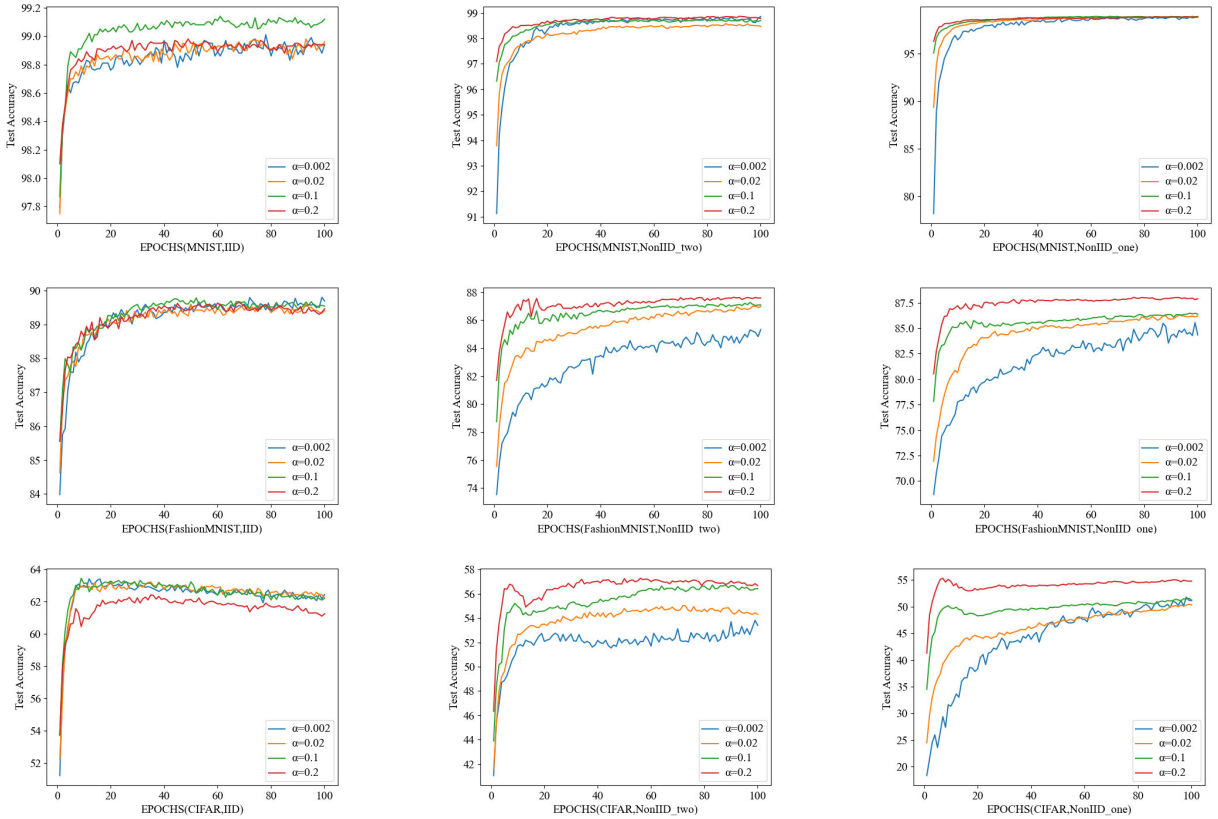

FIGURE 5. Accuracy of FedDBO under three client data heterogeneity conditions $\alpha = 0.002, 0.02, 0.1, 0.2$ on three datasets.

and CIFAR-10 datasets, the impact of $\alpha = 0.002, 0.02, 0.1, 0.2$ on experimental accuracy is negligible. However, under the NonIID settings of the FashionMNIST and CIFAR-10 datasets, the experimental accuracy increases with the increase of the dataset ratio. As the data sample size increases, the communication cost increases, and the data privacy and security decrease. Therefore, for the five data heterogeneous scenarios with small impact on experimental accuracy due to dataset ratio, and for the sake of experimental stability, this paper chooses to set the experimental parameter $\alpha = 0.2$ under these scenarios. For the other four data heterogeneous scenarios, this paper chooses to set $\alpha = 0.2$.

5) PERFORMANCE COMPARISON OF FEDDBO AND SIMILAR ALGORITHMS

To further validate the robustness of FedDBO, we conducted a performance loss comparison with FedPSO, specifically examining the impact of performance loss on FedPSO and FedDBO. In this paper, we set up two sets of experiments $\alpha = 0.02$ and 0.2 to compare the performance loss of the two algorithms.The experimental results are shown in the Fig. 6.

It can be observed that as α changes from 0.2 to 0.02, regardless of the dataset and data heterogeneity, FedPSO exhibits increasing performance loss with the deepening of data heterogeneity. However, the performance loss of FedDBO is consistently lower than that of FedPSO, indicating that FedDBO has stronger stability compared to FedPSO.

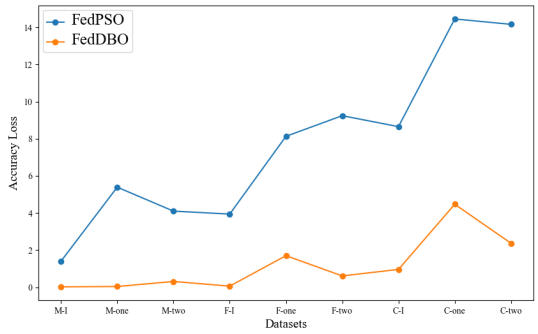

FIGURE 6. Accuracy loss values of FedPSO and FedDBO on three datasets when α changes from 0.2 to 0.02 under three client data heterogeneity conditions.

VI. CONCLUSION AND FUTURE WORK

This paper addresses challenges in federated learning, such as high communication costs, low communication efficiency, and heterogeneous client data. It introduces a federated learning algorithm based on the cockroach optimization algorithm, named FedDBO. Unlike traditional federated learning, FedDBO replaces model parameters with model scores during communication between the server and clients. This innovation enables a communication process where only a subset of clients needs to transmit model parameters, significantly reducing communication costs. Additionally, FedDBO incorporates a model retraining strategy at the server, effectively improving the accuracy of the global

model. It demonstrates robust performance even in scenarios with heterogeneous client data. Finally, FedDBO provides mathematical proofs supporting the convergence of its aggregated model to the aggregated model provided by FedAvg, ensuring accuracy without compromising efficiency and robustness.

To achieve the goal of reducing communication costs, FedDBO selects a limited number of clients for uploading model parameters. Consequently, the model accuracy of FedDBO is highly dependent on the quality of server data. Future research in the federated learning field should focus on finding ways to reduce communication costs while maintaining accuracy and constructing high-quality server data while preserving data privacy. These areas are likely to be hot topics in future studies in FL.

REFERENCES

[1] C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, "Communication-efficient federated learning via knowledge distillation," *Nature Commun.*, vol. 13, no. 1, p. 2032, Apr. 2022.
[2] Y. Li, R.-G. Zhou, R. Xu, J. Luo, W. Hu, and P. Fan, "Implementing graph-theoretic feature selection by quantum approximate optimization algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 2, pp. 2364–2377, Feb. 2024.
[3] A. Khan, S. Hayat, M. Ahmad, J. Wen, M. U. Farooq, M. Fang, and W. Jiang, "Cross-modal retrieval based on deep regularized hashing constraints," *Int. J. Intell. Syst.*, vol. 37, no. 9, pp. 6508–6530, Sep. 2022.
[4] K. Bonawitz, F. Salehi, J. Konecný, B. McMahan, and M. Gruteser, "Federated learning with autotuned communication-efficient secure aggregation," in *Proc. 53rd Asilomar Conf. Signals, Syst., Comput.*, Nov. 2019, pp. 1222–1226.
[5] A. Khan, S. Hayat, Y. Zhong, A. Arif, L. Zada, and M. Fang, "Computational and topological properties of neural networks by means of graph-theoretic parameters," *Alexandria Eng. J.*, vol. 66, pp. 957–977, Mar. 2023.
[6] A. Khan, S. Hayat, M. Ahmad, J. Cao, M. F. Tahir, A. Ullah, and M. S. Javed, "Learning-detailed 3D face reconstruction based on convolutional neural networks from a single image," *Neural Comput. Appl.*, vol. 33, no. 11, pp. 5951–5964, Jun. 2021.
[7] H.-Z. Xia, L.-M. Chen, F. Qi, X.-D. Mao, L.-Q. Sun, and F.-Y. Xue, "DP-denclue: An outlier detection algorithm with differential privacy preservation," in *Proc. IEEE 24th Int. Conf. High Perform. Comput. Commun.; 8th Int. Conf. Data Sci. Syst., 20th Int. Conf. Smart City; 8th Int. Conf. Dependability Sensor, Cloud Big Data Syst. Appl. (HPCC/DSS/SmartCity/DependSys)*, Dec. 2022, pp. 2264–2269.
[8] H. Xia, L. Chen, D. Wang, and X. Lu, "Improved denclue outlier detection algorithm with differential privacy and attribute fuzzy priority relation ordering," *IEEE Access*, vol. 11, pp. 90283–90297, 2023.
[9] B. McMahan, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html
[10] J. Xue and B. Shen, "Dung beetle optimizer: A new meta-heuristic algorithm for global optimization," *J. Supercomput.*, vol. 79, no. 7, pp. 7305–7336, May 2023.
[11] A. Nilsson, S. Smith, G. Ulm, E. Gustavsson, and M. Jirstrand, "A performance evaluation of federated learning algorithms," in *Proc. 2nd Workshop Distrib. Infrastructures Deep Learn.*, Dec. 2018, pp. 1–8.
[12] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.
[13] C. Wu, F. Wu, L. Lyu, T. Qi, Y. Huang, and X. Xie, "A federated graph neural network framework for privacy-preserving personalization," *Nature Commun.*, vol. 13, no. 1, p. 3091, Jun. 2022.
[14] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022–2035, Mar. 2020.
[15] Y. Liu, Y. Liu, Z. Liu, Y. Liang, C. Meng, J. Zhang, and Y. Zheng, "Federated forest," *IEEE Trans. Big Data*, vol. 8, no. 3, pp. 843–854, Jun. 2022.
[16] Z. Xue and L. Xiaohui, "Federated gradient boosting decision tree for non-IID dataset," *Appl. Res. Comput.*, vol. 40, pp. 2184–2191, 2023, doi: 10.19734/j.issn.1001-3695.2022.12.0764.
[17] V. Hartmann, K. Modi, J. M. Pujol, and R. West, "Privacy-preserving classification with secret vector machines," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2020, pp. 475–484.
[18] J. Ye, T. Wei, L. Hu, S. Luo, and X. Li, "An efficient federated learning algorithm for artificial intelligence of things," *Comput. Eng.*, vol. 49, pp. 243–251 and 261, 2023, doi: 10.19678/j.issn.1000-3428.0066803.
[19] S.-X. Cao, C.-M. Chen, P. Tang, and S. Su, "Differentially private federated learning with functional mechanism," *Chin. J. Comput.*, vol. 46, pp. 2178–2195, 2023.
[20] J. Chen, R. Li, G. Huang, T. Liu, H. Zheng, and Y. Cheng, "Survey on vertical federated learning: Algorithm, privacy and security," *Chin. J. Netw. Inf. Secur.*, vol. 9, pp. 1–20, 2023.
[21] X. Xu, G. Yang, and Y. Huang, "Differential privacy clustering algorithm in horizontal federated learning," *J. Comput. Appl.*, vol. 44, no. 1, p. 217, 2024.
[22] C. Y. Xu and L. N. Ge, "Federated learning method based on differential privacy protection knowledge transfer," *Appl. Res. Comput.*, vol. 40, no. 2473, pp. 1–9, 2023.
[23] Y. Xia and W.-Q. Cui, "Personalized federated learning algorithm based on reptile," *Comput. Syst. Appl.*, vol. 31, pp. 294–300, 2022, doi: 10.15888/j.cnki.csa.008875.
[24] F. Chen, H. Zhou, and Y. Zhang, "FCAT-FL: An efficient federated learning algorithm based on non-IID data," *J. Nanjing Univ. Posts Telecommun., Natural Sci. Ed.*, vol. 42, pp. 90–99, 2022, doi: 10.14132/j.cnki.1673-5439.2022.03.011.
[25] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Comput. Ind. Eng.*, vol. 149, Jan. 2020, Art. no. 106854.
[26] P. M. Mammen, "Federated learning: Opportunities and challenges," 2021, *arXiv:2101.05428*.
[27] Y. Xiancai and Z. Jianchao, "Joint optimization of UAV trajectory and resource allocation for federal learning," *Comput. Eng. Appl.*
[28] Z.-B. Ma, Y. Mi, B. Zhang, Z. Zhang, J.-Y. Wu, H.-W. Huang, and W.-D. Wang, "Review on deep learning algorithms for heterogeneous medical image processing," *J. Softw.*, vol. 34, pp. 4870–4915, 2023, doi: 10.13328/j.cnki.jos.006680.
[29] J. Ogier du Terrail, A. Leopold, C. Joly, C. Béguier, M. Andreux, C. Maussion, B. Schmauch, E. W. Tramel, E. Bendjebbar, and M. Zaslavskiy, "Federated learning for predicting histological response to neoadjuvant chemotherapy in triple-negative breast cancer," *Nature Med.*, vol. 29, no. 1, pp. 135–146, 2023.
[30] N. Rieke, J. Hancox, W. Li, F. Milletarì, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, S. Ourselin, M. Sheller, R. M. Summers, A. Trask, D. Xu, M. Baust, and M. J. Cardoso, "The future of digital health with federated learning," *NPJ Digit. Med.*, vol. 3, no. 1, p. 119, Sep. 2020.
[31] F. Geng, Z. Li, and X. Chen, "Incentive mechanism design for hierarchical federated learning based on multi-leader Stackelberg game," *J. Comput. Appl.*, vol. 43, pp. 3551–3558, 2023.
[32] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Comput. Ind. Eng.*, vol. 149, Jan. 2020, Art. no. 106854.
[33] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *J. Healthcare Informat. Res.*, vol. 5, no. 1, pp. 1–19, Mar. 2021.
[34] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowl.-Based Syst.*, vol. 216, Jan. 2021, Art. no. 106775.
[35] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.
[36] Z.-P. Li, Y. Guo, Y.-F. Chen, Y.-W. Wang, W. Zeng, and M.-K. Tan, "Class-balanced federated learning based on data generation," *Chin. J. Comput.*, vol. 46, pp. 609–625, 2023.
[37] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. Theertha Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*.
[38] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–7.
[39] X. Ma, J. Zhu, Z. Lin, S. Chen, and Y. Qin, "A state-of-the-art survey on solving non-IID data in federated learning," *Future Gener. Comput. Syst.*, vol. 135, pp. 244–258, Oct. 2022.
[40] G. Long, M. Xie, T. Shen, T. Zhou, X. Wang, and J. Jiang, "Multi-center federated learning: Clients clustering for better personalization," *World Wide Web*, vol. 26, no. 1, pp. 481–500, Jan. 2023.

[41] X. Mu, Y. Shen, K. Cheng, X. Geng, J. Fu, T. Zhang, and Z. Zhang, "FedProc: Prototypical contrastive federated learning on non-IID data," *Future Gener. Comput. Syst.*, vol. 143, pp. 93–104, Jun. 2023.
[42] Y. Sun, Y. Shi, Z. Wang, M. Li, and P. Si, "A personalized federated learning algorithm based on meta-learning and knowledge distillation," *J. Beijing Univ. Posts Telecommun.*, vol. 46, pp. 12–18, 2023, doi: 10.13190/j.jbupt.2021-327.
[43] H. Fazli Khojir, D. Alhadidi, S. Rouhani, and N. Mohammed, "FedShare: Secure aggregation based on additive secret sharing in federated learning," in *Proc. Int. Database Engineered Appl. Symp. Conf.*, May 2023, pp. 25–33.
[44] S. Zheng, T. Li, and W. Huang, "Federated learning algorithm for communication cost optimization," *J. Comput. Appl.*, vol. 43, pp. 1–7, 2023.
[45] M. Gong, Y. Gao, J. Wang, Y. Zhang, S. Wang, and F. Xie, "Adaptive federated learning algorithm based on evolution strategies," *Scientia Sinica, Informationis*, vol. 53, pp. 437–453, 2023.
[46] R. E. J. Kennedy, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw.*, Perth, WA, Australia, 1995, pp. 1942–1948.
[47] T. Serizawa and H. Fujita, "Optimization of convolutional neural network using the linearly decreasing weight particle swarm optimization," 2020, *arXiv:2001.05670*.
[48] B. Wang, B. Xue, and M. Zhang, "Particle swarm optimisation for evolving deep neural networks for image classification by evolving and stacking transferable blocks," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2020, pp. 1–8.
[49] A. R. Syulistyo, D. M. Jati Purnomo, M. F. Rachmadi, and A. Wibowo, "Particle swarm optimization (PSO) for training optimization on convolutional neural network (CNN)," *Jurnal Ilmu Komputer dan Informasi*, vol. 9, no. 1, p. 52, Feb. 2016.
[50] B. Qolomany, K. Ahmad, A. Al-Fuqaha, and J. Qadir, "Particle swarm optimized federated learning for industrial IoT and smart city services," in *Proc. GLOBECOM IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–6.
[51] S. Park, Y. Suh, and J. Lee, "FedPSO: Federated learning using particle swarm optimization to reduce communication costs," *Sensors*, vol. 21, no. 2, p. 600, Jan. 2021.
[52] Z. Chang, J. Luo, Y. Zhang, and Z. Teng, "A mixed strategy improved dung beetle optimization algorithm and its application," Tech. Rep., 2023.
[53] M. Zhao, H. Jiang, and Q. Chen, "Identification of procymidone in rapeseed oils based on olfactory visualization technology," *Microchemical J.*, vol. 193, Oct. 2023, Art. no. 109055.
[54] J. Duan, Y. Gong, J. Luo, and Z. Zhao, "Air-quality prediction based on the ARIMA-CNN-LSTM combination model optimized by dung beetle optimizer," *Sci. Rep.*, vol. 13, no. 1, Jul. 2023, Art. no. 12127.
[55] W. Wang, Y. Wang, R. Jiang, and K. Cui, "Machine learning-enabled early prediction of dimensional accuracy for complex products of investment casting," Tech. Rep., 2023.
[56] C. Wu, J. Fu, X. Huang, X. Xu, and J. Meng, "Lithium-ion battery health state prediction based on VMD and DBO-SVR," *Energies*, vol. 16, no. 10, p. 3993, May 2023.
[57] T. Hu, H. Zhang, and J. Zhou, "Prediction of the debonding failure of beams strengthened with FRP through machine learning models," *Buildings*, vol. 13, no. 3, p. 608, Feb. 2023.
[58] Z. Mao and Y. Xu, "Gas emergence prediction based on a combination of improved dung beetle optimizer algorithm and temporal convolutional network," in *Proc. 4th Int. Conf. Comput. Eng. Appl. (ICCEA)*, Apr. 2023, pp. 930–935.
[59] R. Zhang and Y. Zhu, "Predicting the mechanical properties of heat-treated woods using optimization-algorithm-based BPNN," *Forests*, vol. 14, no. 5, p. 935, May 2023.
[60] D. Chao-Fan and L. Chen-Yang, "Particle swarm optimization-based federated learning method for heterogeneous data," *Comput. Sci.*
[61] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*.

DONGYAN WANG was born in Nanyang, China, in 2000. He is currently pursuing the M.S. degree in applied mathematics with Mudanjiang Normal University, China. His research interests include machine learning and federated learning.

LIMIN CHEN received the Ph.D. degree from Harbin Engineering University, China. She is currently a Professor with Mudanjiang Normal University. Her research interests include machine learning and data mining.

XIAOTONG LU was born in Suihua, China, in 2000. She is currently pursuing the M.S. degree in applied mathematics with Mudanjiang Normal University, China. Her research interests include machine learning and data mining.

YIDI WANG was born in Jiamusi, China. She is currently pursuing the M.S. degree in applied mathematics with Mudanjiang Normal University, China. Her research interests include machine learning and data mining.

YUE SHEN was born in Shenyang, China. She is currently pursuing the M.S. degree in applied mathematics with Mudanjiang Normal University, China. Her research interests include machine learning and data mining.

JINGJING XU was born in Zhoukou, China. She is currently pursuing the M.S. degree in applied mathematics with Mudanjiang Normal University, China. Her research interests include uncertainty theory and its applications.