Received 20 July 2023, accepted 5 August 2023, date of publication 11 August 2023, date of current version 16 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3304368

RESEARCH ARTICLE

GA Approach to Optimize Training Client Set in Federated Learning

DONGSEOK KANG AND CHANG WOOK AHN, (Member, IEEE)
AI Graduate School, Gwangju Institute of Science and Technology, Gwangju 61005, South Korea

Corresponding author: Chang Wook Ahn (cwan@gist.ac.kr)

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) funded by the Korean Government (MSIT) through the Artificial Intelligence Graduate School Program [Gwangju Institute of Science and Technology (GIST)] under Grant 2019-0-01842, and in part by the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant NRF-2021R1A2C3013687.

ABSTRACT Federated learning, where the distribution of distributed data is unknown, is more difficult and costly to train a central model with than traditional machine learning. In this study, we propose Federated Learning with Genetic Algorithm, which enables faster central model training at lower cost by providing an appropriate client selection method. A client can have its own communication cost depending on its data sharing preference, and based on this cost and the result of the client's local update, we can select the appropriate combination of clients each round with a genetic algorithm. In each round, the client's combinations are evaluated anew, which are continually explored. To evaluate the algorithm, we distributed the image dataset and communication costs in two ways and conducted federated learning for the image classification model. Experiments showed that the proposed algorithm can find a more efficient client combination and accelerate the training of federated learning.

INDEX TERMS Machine learning, federated learning, evolutionary computation.

I. INTRODUCTION

Federated learning [1], which is attracting attention as interest in data privacy increases, is a method of training a central model without collecting distributed raw data. Each entity with its own data, which is not shared, contributes to central model training with preserved data privacy. Thanks to these characteristics, federated learning is used as an algorithm that can easily access a lot of distributed data and is used in various fields such as computer vision and IOT [2], [3]. Given increasingly stringent privacy regulations, the range of applications of federated learning is expanding and impacting machine learning's real-world problem solving [4], [5].

However, as a price for according with the privacy, each participating entity requires frequent communication with either the server or clients [6]. Due to the need for frequent communication and the distribution of data unknown to each other, issues such as significant network cost, latency issues, and statistical heterogeneity arise in federated learning. In an attempt to reduce the cost of federated learning, model compression, faster training methods, and new communication protocols have been proposed [7], [8]. With such active research, federated learning has been able to steadily alleviate the problem of data security and communication cost between a server and a client.

In federated learning so far, it has been assumed that communication between a client and a server has the same cost for all clients. However, it is more realistic to assume that the characteristics of clients' communications are often different. Clients may have different latencies depending on the network, and sometimes the client may be busy and unable to respond to the server [9]. In addition to the physical characteristics of the communication, it is desirable for the server to coordinate the communication in consideration of the individual user's data sharing preference. This is because, in fact, each individual has different concerns about privacy, and even when using the same application, users have different preferences for sharing data according to factors such as age and gender [10].

If the server decides on the training contribution of the client based solely on the user's data sharing preference, it could lead to an undesirable performance of the central

The associate editor coordinating the review of this manuscript and approving it for publication was Wai-Keung Fung.

model. From the server's perspective, for the highest possible final performance of the central model, the server wants to optimize the central model as often as possible and for as much data as possible. At the moment, since the server does not know the distribution and size of the data held by the client at all, if only some clients who want to share data are selected, the training of the central model slows down and requires more time and cost. In the worst case, since the user's data sharing preference and user data size can be inversely proportional, clients participating in training should be selected considering various factors such as user data preference, local training results, and communication costs.

In order to solve the aforementioned client combination problem, the client combination to participate in training must be evaluated and selected in a complex manner each time. However, since the server cannot communicate with every client for every round, it is difficult to evaluate every client's information afresh. It is also computationally expensive to evaluate only possible combinations of clients rather than evaluating all clients at once. The number of possible client combinations is very large, and the number of possible cases is close to 3E+25 when selecting 30 out of 100 clients in each round of federated learning. Since a huge number of cases can occur with only 100 clients, it is necessary to apply an appropriate search algorithm and federated learning to solve this problem.

In this study, we assume that even if clients exchange the same central model in a single communication, each individual incurs different communication costs, in order to implement federated learning that addresses various issues such as users' different data sharing preferences and diverse communication environment problems. When a client transmits a central model of the same size, a low communication cost can be defined if the client prefers data sharing, and a high communication cost can be defined if it doesn't. In the real world, different communication costs can be imposed depending on the various communication environments. The algorithm we propose uses a genetic algorithm in such an environment with different communication costs and biased data distribution to efficiently carry out central model training.

Federated Learning for Optimization with Genetic Algorithm (FLOwGA), proposed in this study where it specifically targets the choice of client training participation, applies genetic algorithm (GA), a meta-heuristic method, to federated learning to optimize both performance and preference at the same time. We encoded the combinations of clients participating in federation learning training by chromosome in our genetic algorithm. Each chromosome is a sequence of bits representing an individual client's participation in training, and a new chromosome is created through the genetic algorithm reproduction process between chromosomes. The reproduction process selects the chromosome to be the base with respect to the fitness value, which is an evaluation measure of the chromosome, and the fitness value is calculated based on the training results of the clients.

The best way to know which client can give better central model training results is to test the model against evaluation data, but since the central server does not own any data, changes in the model's parameter values are used to calculate fitness. The fitness value was also formulated to be inversely proportional to the client's communication cost. In other words, chromosomes have higher fitness when they have larger updates of the central model with lower communication cost. This results in fitness that simultaneously considers client-specific communication costs and fast convergence of the central model. As federated learning rounds progress, the combinations of clients going through GA improve to higher fitness.

With the proposed FLOwGA as above, we can summarize our contribution as follows.

1) In federated learning, we proposed a more realistic environment in which each user's data sharing preference might be different, and this was embodied in the communication cost. In particular, we addressed a dramatic problem that can arise in this environment, where the size of each client's data and user preferences are inversely proportional. Our proposed method was able to train a central model with minimal communication based on the user's local training results and data preferences.
2) Federated learning is performed with the participation of a large number of distributed clients, and calculating the entire combination of them one by one requires a lot of time and computation cost. We solve this problem by deriving a new client combination in a short time every round through GA, and providing a way to gradually improve this combination based on previous knowledge.

We designed image dataset distributions under two Scenarios to verify the performance of FLOwGA. One is assuming the aforementioned worst case Scenario, and the other comes with random data sizes and costs. In all cases, each client's data has a non-IID distribution and disjoint data. We conduct image classification training using the CNN model on these image data sets and compare and evaluate them with the original federated learning.

II. RELATED WORKS

A. FEDERATED LEARNING

Federated learning has been researched since around 2015, and FedAvg [11] is one of the first studies that introduced it. This study showed how clients can train a central model with their own data, and show that models can be trained only with parameter collection, without exchanging raw data. When the data distribution of clients follows IID and Non-IID, the central model is able to converge through the parameter mean and an appropriate batch size.

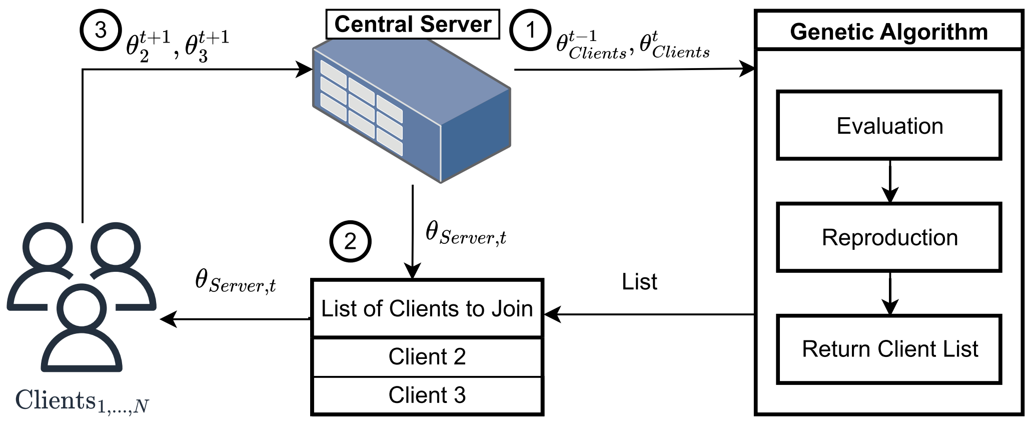


FIGURE 1. This figure shows how FLOwGA works in round t. First, the central server passes the parameters of the central model submitted by each client after local update to the GA. The GA then selects clients to participate in the next round through evaluation and reproduction using these parameters and the client's communication cost. The selected clients locally update the distributed central model parameters and return the results to the central server.

The server and clients exchange models repeatedly to train the central model, which can incur significant network costs. There are studies that have approached from various perspectives to reduce these costs. Li et al. [12] proposed a way to mitigate the problem of global model divergence when a client's heterogeneous data is overfitted locally. The proximal term composed of the parameter difference of the central model mitigated the statistical instability caused by heterogeneous data, achieving the target central model performance with fewer rounds of communication. Reddi et al. [13], on the other hand, presented an adaptive optimizer that was used by the central server rather than by the client. These studies achieved a more efficient gradient descent of the central model, allowing the central model to be trained with less communication cost.

The authors in [14], [15], and [16] have presented reasons and methods for selecting clients in federated learning. Nishio and Yonetani [14] proposed a method of selecting clients to solve the problem of the limitations of computing resources that clients have in a wireless network environment, which slows down federated learning. In [14], before the central server distributes the central model to the clients, it requests information on available computing resources from the clients. Based on the collected information, clients that correspond to the constraints are selected, and local training is conducted from these clients. This research is similar to our research in that it selects clients, but there are a few differences. Nishio and Yonetani assume devices that use wireless connections and have very limited computing resources, such as mobile devices. In such an environment, some clients often can only upload local training results much later than other clients. While the total physical time consumed in federated learning is reduced by setting a deadline as described above, our research proposes a method to reduce the total communication cost by selecting clients optimized to the unique communication costs of each client. Additionally, [14] exchanges additional communications with all clients in every round of federated learning to obtain computing resource information, but we do not exchange additional communications.

Kang et al. [15] proposed a method of selecting reliable clients, assuming that some clients may intentionally damage the central model. Each client is assigned a value *Reputation* from the central server, which is calculated using the communication history with the central server. A client has multiple reputation for various tasks, and these are combined to calculate the final reputation value. Central servers responsible for each task store and exchange reputation values for clients to select reliable clients. Kang et al. [15] has a common point with our work in that it proposes a new metric for client selection. In our case, we designed a fitness metric to reduce the total communication cost, and [15] proposed a reputation metric to exclude unreliable clients. There are also differences in how these metrics are managed. In [15], blockchain technology is used to manage the metric by storing it in a distributed manner, while in our research, a single central server that manages all clients stores and manages the metric.

Li et al. [16] selects clients to participate in training by evaluating whether the data held by the clients is relevant to the task being performed and whether the data diversity is high. In this method, each client generates an embedding vector representing their data, and after going through an encryption process to protect privacy, this vector is sent to the central server. The central server collects these vectors and selects clients with high-quality data. This method has a similarity with our work in that it evaluates clients based on the data they hold. However, while [16] requires additional encryption and communication for client evaluation, our method evaluates clients based on the history of local model updates from the clients, so no additional encryption and communication is needed. Additionally, while [16] bases client evaluation on how relevant each data point is to the task, our proposed method is based on how much more the local model has been trained relative to the communication cost.

Wang et al. [17] proposed a method where clients decide whether to upload to the central model by evaluating their local parameter updates. They compare the direction of the local update vector and the global update vector, and if the direction difference between the two vectors is above a threshold, the client does not upload the update to the central model. This is based on the assumption that when the directions of the two vectors are opposite, the local update does not contribute to the global update. However, since individual clients cannot know in advance the parameter changes of other clients, they compare their local parameter update with the global model update from the previous round. Their research shares similarities with ours in that it selects clients for participation in training and uses local model updates for client evaluation. However, in our research, client selection is conducted at the central server, while in [17], the selection process takes place at the local client. Therefore, in [17], there is a need to distribute the central model to all clients in every round for client selection, while in our research, it is sufficient to distribute the central model only to selected clients at the central server. Furthermore, while [17] communicates with all clients to compare the directionality of parameter vectors for higher stability and performance of the central model, we use the size of local updates for faster convergence to achieve lower communication costs.

B. EVOLUTIONARY COMPUTATION

Evolutionary computation(EC), inspired by natural evolutionary processes, is one of the global optimization methods [18], [19]. This method has been shown to be useful in solving optimization problems as a meta-heuristic method or in combination with other machine learning methods [20], [21]. Evolutionary computation starts with a number of candidate solutions and iteratively evaluates and reproduces the candidates, gradually converging to the optimal solution. These processes are based on the fitness value, the quality of the candidate solutions, which selects a few candidate solutions to participate in the reproduction process, and the newly created candidates are evaluated again.

Among evolutionary computation algorithms, Genetic Algorithm(GA) [22] plays a key role in our proposed federated learning algorithm. This population-based algorithm is an appropriate technique for finding a global optimal solution in a complex fitness space, avoiding local optima [23]. Since the process of finding the optimal client combination is non-differentiable, methods such as gradient descent cannot be used, so metaheuristic algorithms such as GA are appropriate. GA has been used in conjunction with other machine learning techniques as a metaheuristic search method.

Zhu and Jin [24] proposed a method in federated learning to optimize the artificial neural network structure of the central server through evolutionary computation, achieving multi-objective optimization of lower communication costs and higher performance. They iteratively simplified the structure of the central model during training rounds, indirectly reducing communication costs with the server. They demonstrated that using EC can achieve the multi-objective problem of federated learning, which aims to optimize both performance and communication costs. Liu [25] utilized federated learning and EC for training a central model on medical data. They presented a study that respects patient privacy in medical data through federated learning while simultaneously exploring appropriate neural network structures for medical data.

Both [24] and [25] proposed methods to improve the communication cost of federated learning or the performance of the central model using EC as an optimization tool. Their research and our work commonly apply the heuristic optimization algorithm EC to federated learning, but there are differences in the targets of optimization. They focused on the fact that the structure of artificial neural networks is composed of encodable components such as layers and connections. What [24] and [25] proposed is to gradually create a more optimized central model structure through the addition and subtraction of components, not the parameters of the artificial neural network model. The method we propose is not the optimization of the central model structure, but the optimization of the selection of clients participating in training in each round. We assume that each client has its own communication cost, data distribution, and size, and we noted that different communication costs and training results can be produced when the combination of clients participating in training is different. Our research proposes a method to evaluate the fitness of client combinations and a method to optimize it using EC. Although all these studies with different optimization targets use EC as an optimization tool, there are differences in their targets, methods, and assumptions.

III. METHOD

A. PROBLEM FORMULATION

Before explaining the proposed FLOwGA, we first describe the problem to be solved and the general method of federated learning.

$$\arg\min_{\theta} loss(\theta) = \sum_{i} f(x_i; \theta) \tag{1}$$

We want to obtain parameters of the central model θ that satisfy the above equation by training the central model with each client's data x_i through federated learning. The central model's *loss* function is the sum of the loss functions for each client's data x_i. Here, the server has no information about each client's data, and the distribution of data for each client has nothing to do with other clients.

$$\min \sum_{k}^{K} cost(k) = \min \sum_{k}^{K} \sum_{c_i}^{C_k} g(i) \tag{2}$$

Federated learning typically operates with the goal of optimizing only Equation 1. However, in this research, we aim to minimize the sum of communication costs incurred by each client in every round, in addition to Equation 1, as expressed in Equation 2. The communication cost associated with each ith client is denoted as $g(i)$. The communication cost of each client can represent the size of the model communicated between the client and the central server, or it can be a client-specific value indicating user data sharing preferences, network conditions, and other factors. We denote the set of clients that participate in training in federated learning round k as C_k. To achieve Equation 2, we need to select the clients to participate in the next round, round k, for training after the central model update in round $k-1$ is completed.

If the central server knows the data distribution and data size of all clients, it can make the optimal client selection in each round. However, since the server cannot access the data or gradients of individual clients, we plan to utilize fitness values in a genetic algorithm to evaluate client combinations. To measure fitness, we use the change in the model's parameter θ after local training on client c_i, denoted as $\Delta\theta_i$, and the communication cost $g(i)$ of each client. In each round of federated learning, to indicate the clients participating in training, we use chromosomes in the genetic algorithm. These chromosomes are composed of a bit sequence of length N, which can represent all N clients. Each chromosome consists of N bits, where the ith bit being 1 indicates that ith client participates in training, and the ith bit being 0 indicates that ith client does not participate.

$$fitness(c_1, \dots, c_N) = \frac{1}{\max \Delta\theta_i} \sum_{i=1}^{N} \frac{\Delta\theta_i \cdot c_i}{g(i)} \quad , c_i \in \{0, 1\} \tag{3}$$

Equation 3 represents the calculation of the fitness value for a chromosome of length N. Each c_i represents a bit indicating whether client i participated in the training, and $\Delta\theta_i$ represents the difference between the parameter of the local model obtained from the ith client's last local training and the current parameter of the central model. The fitness evaluation of the chromosome involves finding the maximum $\Delta\theta_i$ among all clients and assigning a higher value to the chromosome combination with participating clients having larger $\Delta\theta_i$ values and lower communication costs.

It should be noted that not all clients participate in a round of federated learning at once. Some clients who do not participate in the current round will have θ_i from a past round, not the updated θ_i from the current round. This outdated θ_i could provide inaccurate information for the fitness evaluation of the chromosome in the current round. To address this issue in our Algorithm 1, each client will have a chance to participate in the training with a low probability, regardless of their fitness value.

B. ALGORITHM

Algorithm 1 and Figure 1 demonstrate the overall flow of the proposed method. This algorithm follows the training approach of federated learning, where the genetic algorithm optimizes which clients will participate in the training. The determination of participating clients goes through several steps. Firstly, the algorithm selects the chromosome with the highest $fitness(chromosome)$ value from randomly generated chromosomes. From this chromosome, $\alpha\beta|C|$ clients are randomly extracted, while the remaining $(1-\alpha)\beta|C|$ clients are selected from C. Consequently, we use $\beta|C|$ clients in a single round. β is a variable predetermined in the experiment, and a higher value allows more clients to participate in the training at once. The selected clients receive the parameters of the central model and perform local training, and the central model is replaced by the average of the trained models. Through a reproduction process, H new chromosomes are generated.

Algorithm 1 Federated Learning With Genetic Algorithm

```
Data: $C$ = Client Set, $\theta$ = Initial Parameters,
      $R$ = Maximum round, $H$ = #Chromosomes,
      $\alpha$ = Evaluation factor, $\beta$ = Coefficient to limit
      maximum # clients in a single round
Result: $\theta_E$
1  $X_1, \dots, X_H \leftarrow$ Initialize each chromosomes with
     $\{x_1, \dots, x_{|C|} \mid x_i \in \{0, 1\}\}$
2  $t \leftarrow 1$
3  while $t \leq R$ do
4      $X_{max} \leftarrow \arg\max_{X_i} fitness(X_i)$
5      $A \leftarrow$ Randomly pick $\alpha\beta|C|$ elements from
         $\{x_i \in X_{max} \mid x_i \neq 0\}$
6      $j \leftarrow \arg\max_i \Delta\theta_i$
7      In $A$, set $x_j$ to 1
8      $B \leftarrow$ Randomly pick $\beta|C| - |A|$ elements from
         $C - A$
9      $C_t \leftarrow A \cup B$
10     Distribute $\theta$ to clients in $C_t$ and starts local train
11     $\theta_1, \dots, \theta_{|Ct|} \leftarrow$ Collect $\theta_i$ from $C_t$
12     $\theta \leftarrow \frac{1}{|C_t|} \sum_{i=1}^{|C_t|} \theta_i$
13     $X_1, \dots, X_H \leftarrow Reproduce(X_1, \dots, X_H)$
14     $t \leftarrow t + 1$
15 end
```

The reason we do not select all clients from X where $fitness(chromosome)$ is the highest is due to the outdated $\Delta\theta_i$ that some clients possess. Fitness is deeply tied to $\max \Delta\theta_i$, and since $\max \Delta\theta_i$ is often significantly influenced by these outdated $\Delta\theta_i$s, there is a need to update a client's $\Delta\theta_i$ even if it does not fall into the combination with high $fitness$. Especially in the initial rounds of federated learning, $\max \Delta\theta_i$ tends to have a very large value, which results in very low fitness values for all chromosomes. For instance, if some clients obtained $\Delta\theta_i$ in the initial rounds of federated learning and never updated it again, these clients continue to have very high values of $\Delta\theta_i$. This will lower the value in the fitness calculation of other chromosomes, and consequently, all fitness values will appear to decrease gradually.

The selected clients receive the parameters of the central model and train the model using their local data. The trained model parameters are gathered at the central server, and their average parameters become the central model parameters for the next round. To start the next round, Algorithm 2 generates new chromosomes. The generation of new chromosomes is based on the existing chromosomes, using the updated parameters of the central model and the results of the previous local training.

Algorithm 2 Reproduce

```
Data: $P$ = Existing Chromosomes
Result: $P_{new}$ = New Chromosomes
1  $P_{new} \leftarrow \emptyset$
2  while $P_{new} \leq P$ do
3      $P_1 \leftarrow$ Randomly pick 2 elements from $P$ and
         return one with bigger $fitness(X)$
4      $P_2 \leftarrow$ Randomly pick 2 elements from $P$ and
         return one with bigger $fitness(X)$
5      $K \leftarrow$ Random integer $1 < K < |P_1|$
6      $P_{1,new} \leftarrow \{x_i \in P_1 \mid x_1, \ldots, x_K\} \cup \{x_i \in P_2 \mid$
         $x_K, \ldots, x_{|P_2|}\}$
7      $P_{2,new} \leftarrow \{x_i \in P_2 \mid x_1, \ldots, x_K\} \cup \{x_i \in P_1 \mid$
         $x_K, \ldots, x_{|P_1|}\}$
8      foreach set $P \in \{P_{new,1}, P_{new,2}\}$ do
9          foreach element $x_i \in P$ do
10             Flip $x_i$ with probability $\frac{1}{|P|}$
11         end
12     end
13     $P_{new} \leftarrow P_{new} \cup \{P_{new,1}, P_{new,2}\}$
14 end
```

Algorithm 2 probabilistically selects two chromosomes to create a new chromosome based on the fitness values of the existing chromosomes. The method of selecting two existing chromosomes is to randomly select two from all chromosomes, choosing the one with the greater fitness value. The method used at this time is called the tournament selection method in GA. This tournament selection method is described in lines 3-4. By selecting the chromosome with a higher fitness, we expect the newly combined chromosome to have a higher fitness value than the existing one.

After selecting two chromosomes through tournament selection, they are combined to create a new client combination. This is a process of connecting different parts of the two chromosomes, called the Crossover process. We randomly select a number between the length P_1 of one chromosome and 1 to use as the index for the combination process. Based on this index, the new chromosome is a new combination where the front and back of the existing chromosome have crossed over. This new combination is stored in a new chromosome set after going through the mutation process with a low probability of $\frac{1}{|P|}$. Algorithm 2 repeats this process until the new chromosomes are generated as many as the existing chromosomes.

The client selection method we proposed requires the calculation of fitness for all clients and the *Reproduce* that creates all chromosomes anew each time for the operation of GA. The time complexity required for each process is determined by the number of chromosomes H and the total number of clients C. In Algorithm 1, in one round of federated learning, it is necessary to refer to all C clients to calculate the fitness of one chromosome. We need to repeat this calculation H times for all chromosomes, and this process occurs in every round of federated learning. The *Reproduce* process creates H new chromosomes, and it takes time equivalent to the length of a chromosome C to create one chromosome. Assuming that the central server conducts federated learning for R rounds, the time complexity that GA uses is $O(RCH)$.

In federated learning, distributing the central model to all clients and each client training the central model locally require relatively large computational resources and time. Compared to the entire process of federated learning, the time consumed by GA is negligible. The part where GA generally takes the most time is the calculation of fitness, but in our case, the calculation of fitness for a single chromosome is simple, denoted as $O(C)$, so we can say that the computational resources and time required for GA are very small. In other words, our proposed FLOwGA does not demand additional computational resources on edge devices. While there is some additional computational resource consumption on the server, it is only marginally more than that of the central model training process.

IV. EXPERIMENTS

To verify whether FLOwGA can train the central model faster with lower communication costs through the optimization of client combinations, we conducted image classification experiments using two Scenarios. The two Scenarios use the same original dataset, but the correlation between the number of data each client has and the communication cost is set differently. The algorithms that were tested alongside FLOwGA are FedAvg [11] and CMFL [17].

In Scenario 1, the communication cost assigned to the client is inversely proportional to the size of the data held by the client. We assign communication costs based on the size of the data each client has, assigning lower costs to clients with larger data sizes and higher costs to those with smaller data sizes. In this Scenario, the central server needs to more frequently select clients with larger data sizes for faster convergence of the central model at a lower communication cost. There are three reasons why we designed such data size-communication cost relationship.

First, Scenario 1 more clearly demonstrates that FLOwGA works well. When generating arbitrary client combinations in each of Scenario 1 and 2, when the sum of the data sizes of

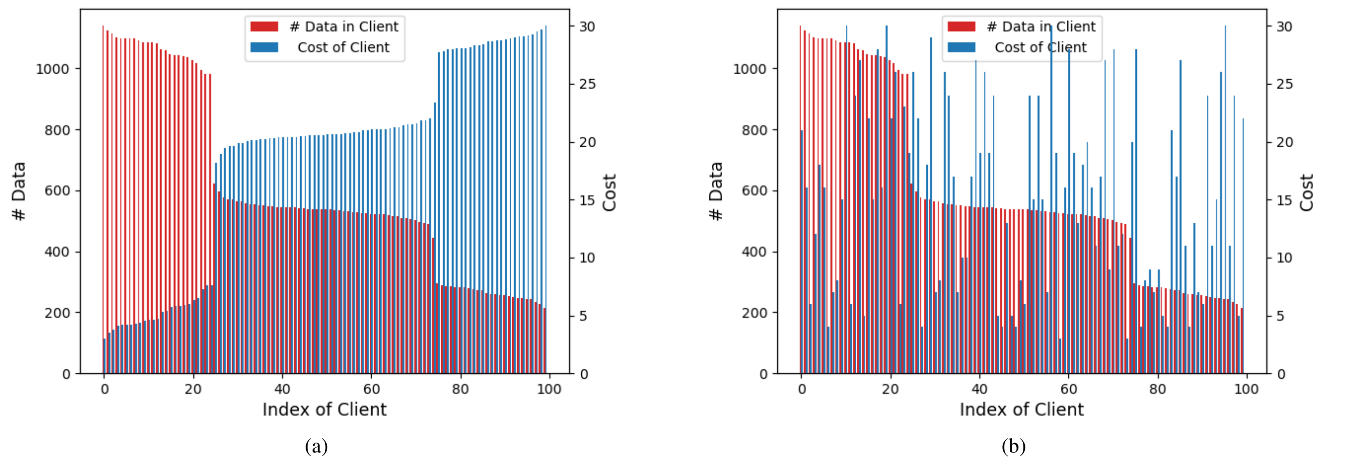


FIGURE 2. **The amount of data each client has and its cost in Scenarios 1 and 2.**

the clients is above average, Scenario 1 shows a lower sum of communication costs, and when the sum of the data sizes of the clients is below average, Scenario 1 shows a higher sum of communication costs. Therefore, if FLOwGA operates negatively, it will show worse figures than in a random data size-communication cost relationship, and if it operates well, it will show better figures.

Second, we test through Scenario 1 whether FLOwGA is too greedy for communication costs. In Scenario 1, it is not the optimal choice for the GA to always select clients with lower communication costs and more data. If the central server blindly selects the same clients all the time, it could lead to overfitting of the central model, falling into local optima, and inaccurate calculation of fitness values. Therefore, through this Scenario, we can more clearly confirm than in Scenario 2 whether FLOwGA can find cost-optimal clients while not falling into local optima and training the central model well.

Third, we believe that a data size-communication cost relationship like Scenario 1 can exist in the real world. Studies like [26] show that user groups unfamiliar with electronic devices have lower willingness to share data than user groups familiar with electronic devices. It also shows that user groups familiar with electronic devices own more wearable devices. Based on these findings, we represent the fact that user groups producing less data have less willingness to share data through the data size-communication cost relationship in Scenario 1.

In Scenario 2, there is no correlation between the size of the data held by the client and the communication cost. This assumption better represents real-world situations compared to Scenario 1. Because there is no correlation between cost and size, if the central server simply selects low-cost clients, it will result in lower performance at a higher cost. What we want from the central server in Scenario 2 is a more dynamic client selection strategy. For example, in the early stages of federated learning, the central model is barely trained, so it would be advantageous for the central server to select clients with larger data sizes, even if it means paying a slightly higher communication cost.

The image dataset used in the two Scenarios is Fashion MNIST [27]. This grayscale image dataset has 10 labels, each image is 28×28 in size, and it has a total of 60,000 training images and 10,000 test images. We divided 100 clients into three groups and distributed the training images with different probabilities. When distributing an image, clients in group A have a 1.78% chance of receiving the image, group B has a 0.89% chance, and group C has a 0.45% chance. The size of group A is 25, group B is 50, and group C is 25. At this time, each client was allowed to have images belonging to a maximum of 4 labels. The training data distributed in this way is disjoint between each client and forms Non-IID training data.

After distributing the training images, we set the communication cost for each client according to the Scenarios. In Scenario 1, we assigned a communication cost inversely proportional to the size of the data held by the client, with a minimum of 3 and a maximum of 33. For the data formation process for Scenario 2, we set an arbitrary cost between 3 and 33 for the client, regardless of the size of the data held by the client. The results of distributing images and communication costs in this way are shown in Fig. 2.

To train the central model in all image classification experiments, we use SGD with a learning rate of 1E-3. The central model was implemented using a convolution layer and ReLU activation. β was fixed at 0.3, allowing a total of 30 clients to train in one round. α is set to 0.9, so out of 30 clients, 27 clients will be selected from the chromosome with the greatest fitness, and 3 clients will be selected randomly. To drive FLOwGA to use 30 clients in a round, we made the lowest fitness value if the chromosome represents more than 30 clients. We ran a total of 10 times with these settings and averaged the values to evaluate the experiment.

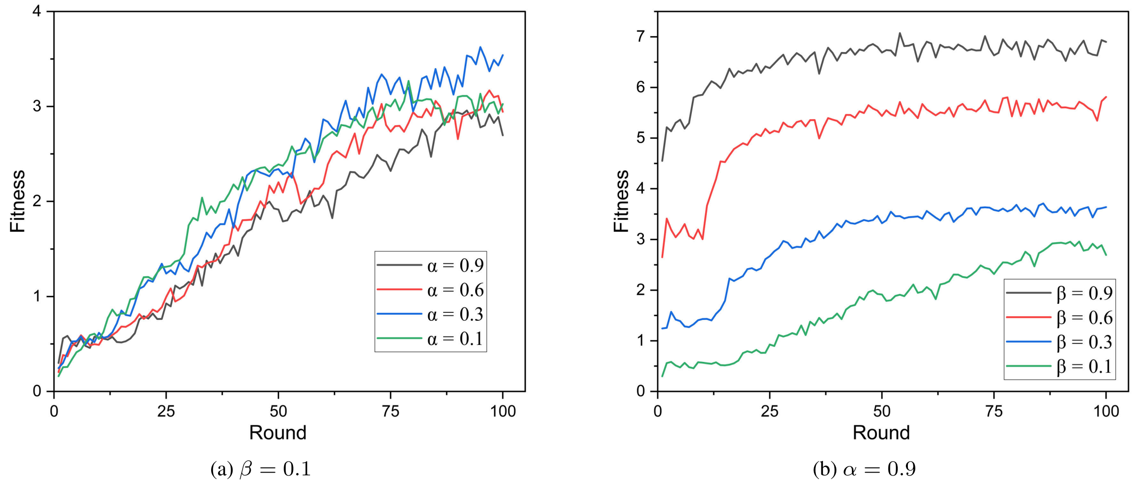


FIGURE 3. Maximum fitness value found in each round according to α and β. (a) β is 0.1, and (b) α is 0.9.

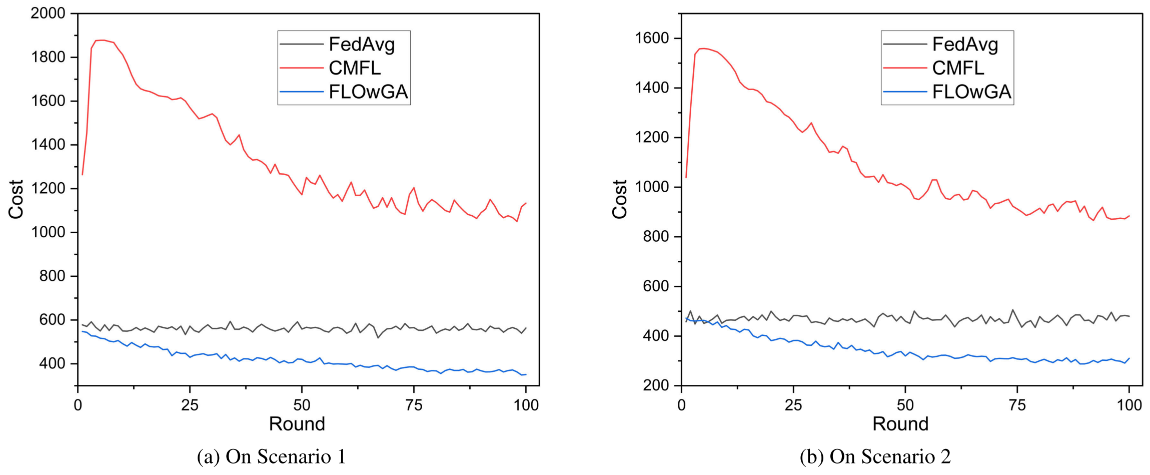


FIGURE 4. Illustration showing the communication cost incurred by the central server in each round.

In CMFL, all clients exceeding the threshold must participate in training each round, so there was no way to limit the training participation to a maximum of 30 clients per round. We used 0.65 as the threshold for CMFL. We tried values of {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.65, 0.7, 0.8} as [17] did, and also tried to decrease the threshold over time. As a result, a fixed threshold of 0.65 was the most efficient CMFL we could find. During the experiment, if CMFL could not find any client exceeding the threshold, we randomly selected 30 clients from all clients to use for the global update. We use the average of the results obtained through 10 repeated experiments of the CMFL algorithm for result comparison.

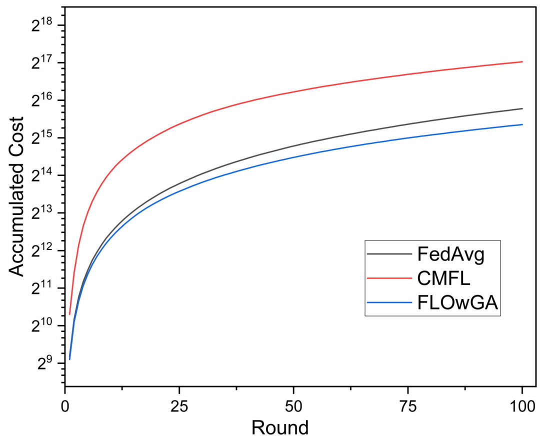
In addition, to evaluate the impact of α and β on the algorithm's operation, we conducted experiments to observe the fitness evaluation results when they are set to 0.1, 0.3, 0.6, 0.9 respectively. We conducted image classification experiments in the aforementioned Scenario 1 environment, setting β to 0.1 when investigating α, and setting α to 0.9 when investigating β.

TABLE 1. Results of the Experiment on Scenario 1.

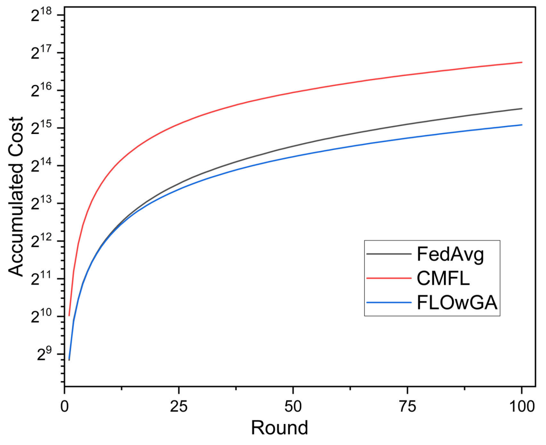
Accumulated Cost	FLOwGA		CMFL		FedAvg	
	Accuracy	Round	Accuracy	Round	Accuracy	Round
21173	0.497	46	0.429	12	0.412	37
48301	0.589	118	0.475	28	0.460	86
82463	0.614	217	0.533	54	0.501	147
122501	0.649	332	0.579	90	0.519	217

V. RESULTS

The results of our experiments are shown in Figs. 3-8 and Tables 1-2. Figure 3 presents the experimental results in Scenario 1, demonstrating the influence of α and β on the client combination optimization of FLOwGA. Figure 4 shows the incurred communication costs for each round. Figure 5 illustrates the cumulative sum of communication costs used by each algorithm. The sum of communication costs in each round represents the sum of costs carried by the participating clients in that round. Figure 6 displays the accuracy of the

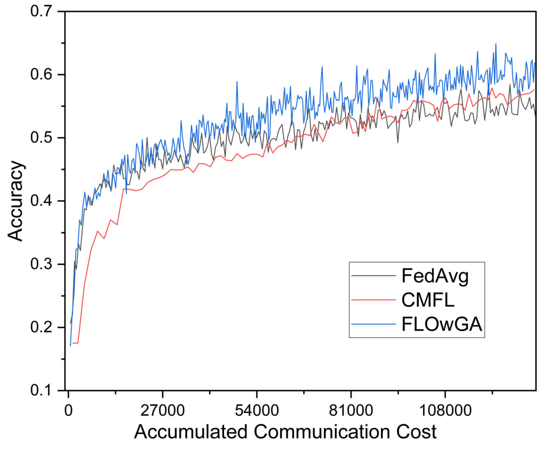


(a) On Scenario 1

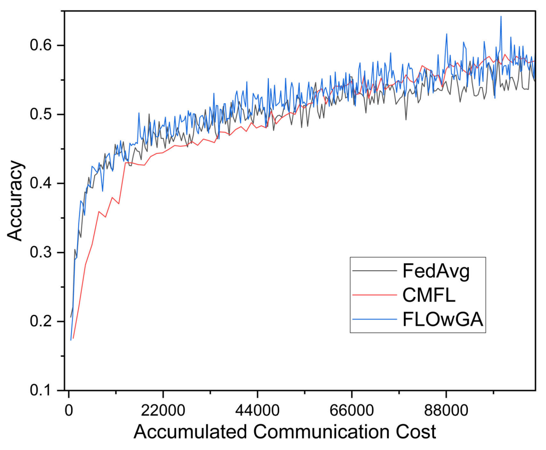


(b) On Scenario 2

FIGURE 5. Shows the total accumulated cost that the central server has consumed up to each round. The y-axis is on a log scale.



(a) On Scenario 1



(b) On Scenario 2

FIGURE 6. Illustration showing the performance of the central model achieved through federated learning. (a) shows the performance in Scenario 1, and (b) shows the performance in Scenario 2.

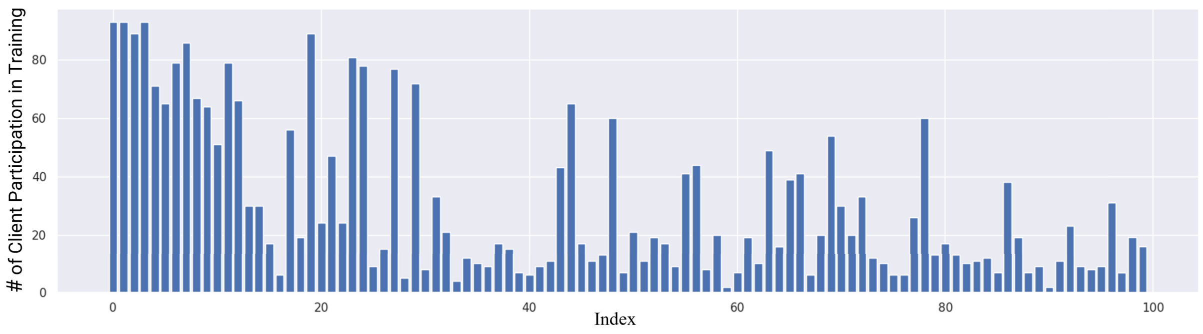


FIGURE 7. The number of times each client participated in training with Scenario 1. Clients with smaller indices (larger data size) got more opportunities to participate in training.

central model at each round, which represents the classification results on a separate test dataset of 10,000 images that were not used for training. Figure 7 reveals how many times FLOwGA involved each client in federated learning in Scenario 1. The fitness values of the combinations found by FLOwGA in each round are depicted in Fig. 8.

TABLE 2. Results of the Experiment on Scenario 2.

Accumulated Cost	FLOwGA		FedAvg		CMFL	
	Accuracy	Round	Accuracy	Round	Accuracy	Round
16346	0.502	41	0.430	11	0.409	34
41513	0.547	123	0.488	30	0.460	88
67885	0.575	216	0.557	55	0.501	144
100770	0.642	333	0.584	89	0.519	215

Table 1 and 2 show the performance of each algorithm based on the cumulative sum of communication costs.

α and β play an important role in the operation of GA in the algorithm, and the effects they have on fitness evaluation are shown in Fig. 3. In Figure 3a, we can see that when α is 0.3, GA can find a chromosome with higher fitness. α determines how greedily the client combination found by GA will be adopted, and we can see that when the algorithm operates at a moderate value that is neither too high nor too low, it can find a combination with a higher fitness value. This is because a high α increases the probability that a client will have an outdated θ_i, and a low α slows down the exploration of GA.

Figure 3b shows the impact of four values of β on fitness evaluation. β determines how much of the total clients will participate in one round of federated learning, and the more clients participated, the more combinations with higher fitness values could be found. The more clients participate at once, the more often each client can update θ_i, and GA has the opportunity to more accurately evaluate various combinations. Therefore, as shown in the experimental results, the higher the β, the higher the fitness value that can be found.

The communication costs consumed by the three algorithms in each round of federated learning are shown in Fig. 4. FedAvg always shows similar costs because clients participate randomly in each round, while FLOwGA and CMFL decrease their round-by-round communication cost consumption as federated learning progresses. However, CMFL started with a very high communication cost consumption and eventually had a reduced communication cost that was higher than the other two algorithms. In FLOwGA, the consumed communication cost steadily decreased at a constant rate until the end of federated learning, which allowed it to ultimately train the central model with much less communication cost than the other two algorithms. Comparing Scenarios 1 and 2 in Fig. 4, both FLOwGA and CMFL generally required higher communication costs in Scenario 1 and optimization of communication costs was more difficult. From this, we can see that communication cost-efficient algorithms like FLOwGA and CMFL face greater difficulties in training when clients with less data require more communication costs.

In Scenarios 1 and 2, when comparing the cumulative sum of communication costs consumed by federated learning, FLOwGA consumes less cost in all rounds as seen in Fig. 5. In Scenario 1, when federated learning finished the 100th round, FLOwGA consumed 41843 (100%) of communication costs, FedAvg consumed 56112 (134%), and CMFL consumed 133433 (318%). Looking at the communication costs consumed, it can be seen that FLOwGA requires less communication costs throughout the federated learning process. In particular, as the rounds progress, the cumulative sum of communication costs consumed by FLOwGA becomes even less than the other two algorithms, which shows that the EC used by FLOwGA is gradually converging to the optimal client combination. In the case of CMFL, it can be seen from Fig. 5 that it consumed much more cost than the other two algorithms, which indicates that CMFL can have more stable training performance at the expense of higher communication costs. CMFL accepts all client updates as long as they exceed the threshold for the stability of the global update, which required more communication costs.

As can be seen in Fig. 5, even though the three algorithms have gone through the same federated learning rounds, the sum of communication costs they consumed is very different from each other. For a fair evaluation of the central model performance, we decided to show the performance achieved by the central model when the same communication cost was consumed. To this end, FLOwGA and FedAvg continued federated learning until they consumed the same communication cost as CMFL. Figure 6 shows the performance of the central model in terms of communication costs. As we can see from Fig. 6, in both Scenarios 1 and 2, the learning curves of the three algorithms fluctuate due to the statistical heterogeneity inherent in federated learning, but FedAvg and FLOwGA show similar patterns. CMFL showed lower performance than the other algorithms when the communication cost consumed was low, but recorded higher performance than FedAvg as the training progressed. As shown in Tables 1-2, in both Scenarios 1 and 2, FLOwGA was able to achieve highest performance for the same communication cost. These results indicate that FLOwGA achieved more cost-effective federated learning when selecting clients using GA. FLOwGA showed rapid performance improvement especially at the beginning of learning, which we believe is due to the fitness evaluation method we use. Fitness increases when one client has a larger central model update compared to other clients, and the method of preferring larger central model updates at the beginning of learning allowed for faster achievement of target performance.

In both Scenarios 1 and 2, FLOwGA worked as we expected, but there is a slight difference in the size of the fitness value and communication costs. The main factor is that the average communication cost of the clients is different in the two Scenarios. In Scenario 1, the average cost is 18.7, and in Scenario 2, it is 15.6, which is about 83% of Scenario 1. The communication cost incurred by FLOwGA in each round is also about 80% less in Scenario 2.

Looking at Figs. 4 and 8, we can see that the increase in fitness value is due to finding a combination of lower communication costs. This is evident in both Scenarios 1 and 2,

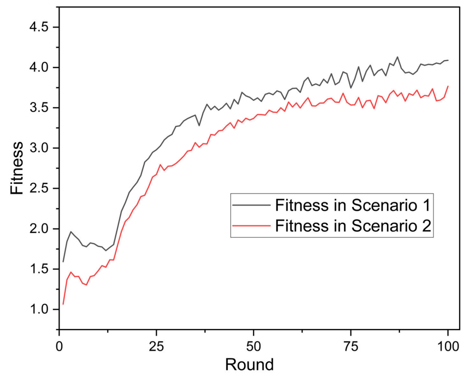


FIGURE 8. **Illustration showing the highest fitness value found by FLOwGA in each round in Scenarios 1 and 2.**

where the communication cost consumed per round is halved while the fitness value increases by about twice. Despite the fact that the size of the central model's parameter update generally decreases as federated learning progresses, FLOwGA was able to adapt to this and find a client combination with a higher fitness value through communication cost optimization.

Figure 7 shows the number of times each client participated in training until FLOwGA terminated in Scenario 1. As shown in Fig. 2a, clients with smaller indices have smaller communication costs and larger local datasets, and FLOwGA found and selected these clients. The larger datasets these clients have contribute to the generalization of the central model, and the cost of communicating with them is lower than other clients. It can also be seen that the client selection is not limited to a subset of clients with lower communication costs, but that all clients are guaranteed a chance to participate in training. It demonstrates that FLOwGA can utilize all of the data of all clients while prioritizing a few.

VI. CONCLUSION

Clients participating in federated learning can have different data and may also have different costs for communication or preferences for data sharing. We proposed a method that uses Genetic Algorithm (GA) to utilize the characteristics of clients for more efficient federated learning in such an environment, rather than randomly selecting clients. Our proposed FLOwGA was able to accelerate federated learning through client combination search by GA. As federated learning progressed, the central server was able to find clients that contributed more to parameter updates with less communication cost.

In future research, we can use heuristic algorithms like GA for multiobjective optimization to obtain a Pareto set that can present various aspects such as communication cost, latency, and training time. In this study, we assumed that all clients have a constant communication cost over time, but in future research, we can consider an environment where the characteristics of clients change.

REFERENCES

[1] S. Banabilah, M. Aloqaily, E. Alsayed, N. Malik, and Y. Jararweh, "Federated learning review: Fundamentals, enabling technologies, and future applications," *Inf. Process. Manage.*, vol. 59, no. 6, Nov. 2022, Art. no. 103061.

[2] Y. Liu, A. Huang, Y. Luo, H. Huang, Y. Liu, Y. Chen, L. Feng, T. Chen, H. Yu, and Q. Yang, "FedVision: An online visual object detection platform powered by federated learning," in *Proc. 32nd Conf. Innov. Appl. Artif. Intell. (IAAI)*, Apr. 2020, vol. 34, no. 8, pp. 13172–13179.

[3] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4177–4186, Jun. 2020.

[4] J. P. Albrecht, "How the GDPR will change the world," *Eur. Data Protection Law Rev.*, vol. 2, no. 3, pp. 287–289, 2016.

[5] Y. Liu, J. J. Q. Yu, J. Kang, D. Niyato, and S. Zhang, "Privacy-preserving traffic flow prediction: A federated learning approach," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7751–7763, Aug. 2020.

[6] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 2031–2063, 3rd Quart., 2020.

[7] J. Mills, J. Hu, and G. Min, "Communication-efficient federated learning for wireless edge intelligence in IoT," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5986–5994, Jul. 2020.

[8] W. Wu, L. He, W. Lin, R. Mao, C. Maple, and S. Jarvis, "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead," *IEEE Trans. Comput.*, vol. 70, no. 5, pp. 655–668, May 2021.

[9] G. Zhu, Y. Wang, and K. Huang, "Broadband analog aggregation for low-latency federated edge learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 491–506, Jan. 2020.

[10] F. Beierle, V. T. Tran, M. Allemand, P. Neff, W. Schlee, T. Probst, J. Zimmermann, and R. Pryss, "What data are smartphone users willing to share with researchers?" *J. Ambient Intell. Humanized Comput.*, vol. 11, no. 6, pp. 2277–2289, Jun. 2020.

[11] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*.

[12] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*.

[13] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," 2020, *arXiv:2003.00295*.

[14] 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.

[15] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 72–80, Apr. 2020.

[16] A. Li, L. Zhang, J. Tan, Y. Qin, J. Wang, and X.-Y. Li, "Sample-level data selection for federated learning," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2021, pp. 1–10.

[17] L. Wang, W. Wang, and B. Li, "CMFL: Mitigating communication overhead for federated learning," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 954–964.

[18] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. Hoboken, NJ, USA: Wiley, 1995.

[19] Z.-H. Zhan, L. Shi, K. C. Tan, and J. Zhang, "A survey on evolutionary computation for complex continuous optimization," *Artif. Intell. Rev.*, vol. 55, no. 1, pp. 59–110, Jan. 2022.

[20] C. Wang, C. Xu, X. Yao, and D. Tao, "Evolutionary generative adversarial networks," *IEEE Trans. Evol. Comput.*, vol. 23, no. 6, pp. 921–934, Dec. 2019.

[21] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 19–35.

[22] J. H. Holland, "Genetic algorithms," *Sci. Amer.*, vol. 267, no. 1, pp. 66–73, 1992.

[23] Y.-T. Kao and E. Zahara, "A hybrid genetic algorithm and particle swarm optimization for multimodal functions," *Appl. Soft Comput.*, vol. 8, no. 2, pp. 849–857, Mar. 2008.

[24] H. Zhu and Y. Jin, "Multi-objective evolutionary federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1310–1322, Apr. 2020.
[25] X. Liu, J. Zhao, J. Li, B. Cao, and Z. Lv, "Federated neural architecture search for medical data security," *IEEE Trans. Ind. Informat.*, vol. 18, no. 8, pp. 5628–5636, Aug. 2022.
[26] S. Schneegass, R. Poguntke, and T. Machulla, "Understanding the impact of information representation on willingness to share information," in *Proc. CHI Conf. Human Factors Comput. Syst.* New York, NY, USA: Association for Computing Machinery, May 2019, pp. 1–6.
[27] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

DONGSEOK KANG received the B.S. degree in computer science and engineering from Inha University, South Korea, in 2014. He is currently pursuing the Ph.D. degree with the School of Artificial Intelligence, Gwangju Institute of Science and Technology (GIST). He is also working on heuristic search algorithm and federated learning. His research interests include evolutionary computation and deep learning.

CHANG WOOK AHN (Member, IEEE) received the Ph.D. degree from the Department of Information and Communications, Gwangju Institute of Science and Technology (GIST), Republic of Korea, in 2005. From 2005 to 2007, he was with the Samsung Advanced Institute of Technology, South Korea. From 2007 to 2008, he was a Research Professor with GIST. From 2008 to 2016, he was an Assistant/Associate Professor with the Department of Computer Engineering, Sungkyunkwan University (SKKU), Republic of Korea. He is currently a Professor with the School of Artificial Intelligence, GIST. His research interests include genetic algorithms/programming, multiobjective optimization, neural networks, and quantum machine learning.