Grey Wolf Optimizer for Reducing Communication Cost of Federated Learning

Ammar Kamal Abasi, Moayad Aloqaily, Mohsen Guizani
Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, United Arab Emirates
E-mails: {ammar.abasi; moayad.aloqaily}@mbzuai.ac.ae, mguizani@ieee.org

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001681

***Abstract*—Federated Learning (FL) is a type of Machine Learning (ML) technique in which only learned models are stored on a server to sustain data security. The approach does not gather server-side data but rather directly shares only the models from scattered clients. Due to the fact that clients of FL frequently have restricted connection bandwidth, it is necessary to optimize the communication between servers and clients. FL clients frequently interact through Wi-Fi and must operate in uncertain network situations. Nevertheless, the enormous number of weights transmitted and received by existing FL aggregation techniques dramatically degrade the accuracy in unstable network situations. We propose a federated GWO (FedGWO) algorithm to reduce data communications. The proposed approach improves the performance under unstable network conditions by transferring score principles rather than all client models' weights. We achieve a 13.55% average improvement in the global model's accuracy while decreasing the data capacity required for network communication. Moreover, we show that FedGWO achieves a 5% reduction in accuracy loss compared to FedAvg and Federated Particle Swarm Optimization (FedPSO) methods when tested on unstable networks.**

***Index Terms*—Federated Learning, Grey Wolf Optimizer (GWO), Convolutional Neural Network (CNN), Deep Learning, Aggregation.**

I. Introduction

In recent years, the use of IoT devices such as tablets, mobile phones, and smartphones has increased exponentially. Mobile devices collect and gather a variety of different types of data, such as image, voice, and text. The more apps used on these mobile devices, the more data they generate. The collected data may be utilized in a variety of ways for Machine Learning (ML) [1]. For instance, Google's Gboard employs ML to learn which phrases users regularly write and, in return, predict the next words to be typed by the user [2]. There are typically four considerations when leveraging mobile device data for ML.

- Security risks: The exchange of private user information over the network significantly raises the possibility of data leaks.
- Inadequate computational capability: Mobile device processors lack the computational ability required for ML.
- Unstable networks: Mobile devices have to connect to wireless networks - typically over Wi-Fi. No matter how strong the wireless signal is, it will lose speed as more devices share the network bandwidth, making it harder to secure a stable network environment.
- There is a considerationable cost to mass data collection: storage expenses and network connectivity are expensive when gathering and handling massive volumes of users' data on a server.

To construct a successful ML model, especially one based on edge devices, it is required to decrease the quantity of the gathered data, reinforce its security, decrease the number of training parameters (weights), and handle operating within an Unstable Network Environment (UNE) [3].

Federated Learning (FL) research has made significant strides in overcoming the aforementioned challenges [4]. FL is a machine learning (ML) approach that uses decentralized data to train ML models. FL ensures data privacy by preventing the transfer of data from local machines such as personal devices to a central server [5]. It also cuts down on communication costs because it sends only recently trained model parameters to the server instead of large amounts of source data.

Because the computing time dominates the communication time in standard Artificial Neural Network (ANN) models, several strategies are utilized to minimize the computing time, including employing graphics processing unit (GPU) accelerators and parallelizing computations across multiple GPUs. Thus, the network communication time should be minimized to increase FL's efficiency. Due to the instability of the network, FL entails certain environmental circumstances [6]. As a result, in order to cut down on the communication cost with FL, it is necessary to speed up the network and deal with problems of the UNE.

Most FL models employ the global Federated Averaging (FedAvg) [7]. In FedAvg, the client is responsible for computing the gradient, updating the model, and sending the results back to the server. Current research proposes using Grey Wolf Optimizer (GWO), a distributed optimization technique, to accelerate global model updates. GWO takes a large number of trials since it achieves the best solution using a stochastic technique, which is consistent with how ML models are trained over many iterations. The GWO is well-suited for contexts that are dynamic and diverse, such as FL. Accordingly, we suggest a novel ANN model based on FL and GWO.

This paper focuses on minimizing network communication expenses through the usage of GWO in the communication operations of FL. We present a novel model, Federated GWO (FedGWO), that updates the optimization method of the global server model by using notches such as local model loss and accuracy rather than weights. Experimentally, we analyze

FedGWO's cost and accuracy for the network communication. FedGWO demonstrates an average accuracy improvement of 13.55% and a reduction in network communication costs in comparison to prior work. Additionally, we examine FedGWO in an UNE and compare the accuracy loss with existing algorithms.

The remainder of the paper is organized as follows. Section II summarizes past research utilizing GWO and FL. Section III details the proposed algorithm's mechanism for communicating operation from the client to the server. Section IV evaluates the proposed approach, and Section V concludes the paper.

II. LITERATURE REVIEW

A. Grey Wolf Optimizer (GWO)

GWO is a metaheuristic algorithm that belongs to population-based optimization methods. The grey wolves' hunting behavior and social leadership are the inspiration for this algorithm. The GWO was introduced by [8] in 2014 to tackle optimization problems. GWO employs social knowledge, which means that the pack members are encouraged to explore and find better areas in the search space. The wolf with more experience is selected to lead the pack and be in charge. The concept behind GWO is as follows: the set of individuals (i.e., the population) is classified into four primary levels $alpha$, $beta$, $delta$, and $omega$. Wolves' leaders indicate the best, second-best, and third-best individuals, which are the first three levels in the GWO population [9]. Then, the rest of the individuals are in the last level of the hierarchical classification system. These levels are taken into consideration by the GWO when searching for a global solution. The levels aim toward selecting the best solution in the search space. Encircling, hunting, and attacking the prey are the three main processes of hunting behavior [10].

1) Encircling prey: As previously mentioned, prey is circled by grey wolves when they are on the hunt. Mathematical models of encircling behavior are developed using the following equations:

$$\overrightarrow{ED} = |\overrightarrow{EC}.\overrightarrow{EX}_p(itr) - \overrightarrow{EX}(itr)|, \tag{1}$$

$$\overrightarrow{EX}(itr+1) = \overrightarrow{EX}_p(itr) - \overrightarrow{EA}.\overrightarrow{ED}, \tag{2}$$

Where $\overrightarrow{EA}$ and $\overrightarrow{EC}$ are coefficient vectors, itr is the current iteration, $\overrightarrow{EX}$ is a vector of the grey wolf position, $\overrightarrow{EX}_p$ is a vector of the prey position. The $\overrightarrow{EA}$ and $\overrightarrow{EC}$ are calculated as follows:

$$\overrightarrow{EA} = 2 \times \overrightarrow{ea} \times \overrightarrow{Er}_1 - \overrightarrow{ea}, \tag{3}$$

$$\overrightarrow{EC} = 2 \times \overrightarrow{Er}_2 \tag{4}$$

Where $\overrightarrow{Er}_1$ and $\overrightarrow{Er}_2$ are vectors of random numbers between [0,1], and $\overrightarrow{ea}$ is a vector of linear values. Over the number of iterations, these values are decreased linearly from 2 to 0 using Eq(5).

$$ea = 2 - current_{itr} \times \frac{2}{max_{itr}}, \tag{5}$$

Where $current_{itr}$ represents the current iteration and max_{itr} is the maximum number of iterations.

2) Hunting: Grey wolves can figure out where their prey is and then encircle them. The alpha is typically responsible for the hunt. Occasionally, the beta and delta can engage in hunting as well. The search space in optimization is usually unknown. Thus, the optimum (prey) location is also unknown. Based on this assumption, the first three levels of the population (i.e., $alpha$, $beta$, and $delta$) should have a deeper insight into the possible location of the prey. Accordingly, the locations of the first three levels are used to update the rest of the population's (including the omega's) positions [11]. In this regard, the following formulas have been presented.

$$\overrightarrow{HD}_\alpha = |\overrightarrow{EC}_1 \times \overrightarrow{HX}_\alpha - \overrightarrow{HX}|, \tag{6}$$

$$\overrightarrow{HD}_\beta = |\overrightarrow{EC}_2 \times \overrightarrow{HX}_\beta - \overrightarrow{HX}|, \tag{7}$$

$$\overrightarrow{HD}_\delta = |\overrightarrow{EC}_3 \times \overrightarrow{HX}_\delta - \overrightarrow{HX}|, \tag{8}$$

$\overrightarrow{HC}_1$, $\overrightarrow{HC}_2$, and $\overrightarrow{HC}_3$ are calculated using Eq(4).

$$\overrightarrow{HX}_1 = \overrightarrow{HX}_\alpha - \overrightarrow{EA}_1 \times \overrightarrow{HD}_\alpha, \tag{9}$$

$$\overrightarrow{HX}_2 = \overrightarrow{HX}_\beta - \overrightarrow{EA}_2 \times \overrightarrow{HD}_\beta, \tag{10}$$

$$\overrightarrow{HX}_3 = \overrightarrow{HX}_\delta - \overrightarrow{EA}_3 \times \overrightarrow{HD}_\delta, \tag{11}$$

Where $\overrightarrow{EA}_1$, $\overrightarrow{EA}_2$, and $\overrightarrow{EA}_3$ are calculated using Eq (3). Eq(12) is used to describe $\overrightarrow{HD}_\alpha$, $\overrightarrow{HD}_\beta$, and $\overrightarrow{HD}_\delta$

$$\overrightarrow{HX}(itr+1) = \frac{\overrightarrow{HX}_1 + \overrightarrow{HX}_2 + \overrightarrow{HX}_3}{3}, \tag{12}$$

3) Attacking: As soon as the prey has stopped moving, the wolves begin attacking it. In each iteration, the ea value is decreased from 2 to 0, as seen in Eq(5). According to [12], in a smooth manner, half of the iterations will be dedicated to exploration and the other half to exploitation. This process involves wolves moving to a random location between their current location and their prey's location.

The GWO pseudocode is provided in algorithm 1. The algorithm starts by randomly creating a set of wolves (or population) within the search space. The fitness function evaluates the wolves' locations. Following that, the GWO steps are repeated until the stop criteria are reached, such as reaching the maximum number of iterations or a specific fitness function value. In each iteration, the three initial wolves with the highest finesses are referred to as $alpha$, $beta$, and $delta$. The location of each wolf is then updated based on the initial processes of encircling, hunting, and attacking the prey. Finally, the optimal location of the $alpha$ can be obtained by repeating these processes.

Algorithm 1: Pseudocode of the GWO algorithm

Set the GWO parameter settings ($\overrightarrow{EC}$,$\overrightarrow{EA}$, Number of dimensions, Number of wolves, Number of iterations).
Create the population of wolves (solutions).
while The current iteration less than the maximum number of iterations **do**
    calculate the objective function for each solution.
    Select $alpha$, $beta$, and $delta$ as best, second-best, and third-best solution, respectively.
    for each solution y **do**
        Using Eqs(6 $\rightarrow$ 12), determine the current location of the current grey wolf.
    end for
    Update $\overrightarrow{EC}$ ea, and $\overrightarrow{EA}$,.
end while
Return $alpha$ which is the best solution.

There are many advantages to using the GWO algorithm over other population-based intelligence techniques. It does not require any derivative information, and it has a smaller set of parameters compared to other algorithms. Due to these advantages, the GWO algorithm has proven to be helpful in solving a variety of optimization problems, including feature selection for classification problems, planning, economic dispatching, scheduling, robotics, and engineering, as explained in [13].

B. Federated Learning (FL)

FL is a technique for distributed datasets planned and presented by [14]. In order to train a model, it uses datasets distributed across several devices while also avoiding data leakage. FL has the advantage of increasing privacy and lowering communication costs. Artificial Neural Network (ANN) models may be trained on centralized servers without exposing sensitive data or personal information using FL. Additionally, migrating data from several local machines to a single server results in storage expenses and network traffic. FL dramatically lowers communication costs by transmitting just the weights acquired during local model training [4]. The FL process is depicted in Fig.1.

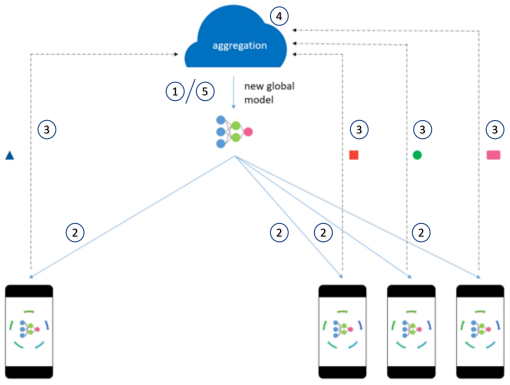

Fig. 1. The Federated Learning protocol.

1) The global trained model is distributed to all clients by the server.
2) The models are trained locally using each client's local data.
3) The trained models' weights are sent back to the server from all clients.
4) The collected models are aggregated into one learning model by the sever.
5) The updated global model is sent to all clients from the server, and steps 1 to 5 are frequently repeated to continue updating the global model representation using local clients' updates.

Federated Averaging (FedAvg) and Federated Stochastic Gradient Descent (FedSGD) are two algorithms commonly used in FL studies to perform the fourth step in Fig.1 and optimize an objective function. McMahan [15] pioneered the FedAvg approach that has been used to update models acquired on servers in a variety of FL initiatives. Based on averaging the results collected from each client, both approaches produce the global model parameters. In FedSGD, the gradient is transmitted to the server, which then calculates the average weights. After that, the global weights are modified in order to generate a global model that is then transmitted to the clients. FedSGD and mini-batches are used in FedAvg to update models directly on the clients, with the weights being averaged on the server to build a new global model. FedAvg is then used to directly adjust models on the client end, as seen in Fig.2.

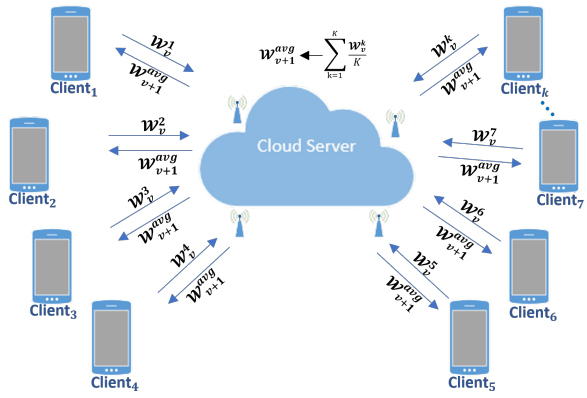

Fig. 2. Weighted aggregation processes like FedAvg receive an average of the W_t values that K clients transmit to the server and send an average of the updated $W_t + 1$ weights back to clients.

FL is reliant on the existence of a mobile device environment that is distributed. Instead of learning in a wired network environment, mobile devices must learn in a wireless network environment [16]. If the network is insecure, the client is unable to send the trained local model to the server.

C. Related Work

Numerous studies on client-server communication have been undertaken in order to enhance the efficacy of FL. FL faces a number of obstacles as a consequence of the mobile device's UNE, including frequent node failures, frequent group switching, increased latency as the number of nodes rises, communications overhead, and high central server overhead. In order to enhance the learning accuracy of ANNs, multilayer models have been used, although the number of node

weights increases as the layers go deeper. FL is limited by the data size due to increased network traffic between the client and the server.

In recent years, numerous studies have been done to address these problems. In order to increase the performance of FL's network, temporal weights [17] as well as low rank, and random masks [18] have been studied. These studies, on the other hand, may suffer from decreased accuracy under UNE.

Furthermore, the conventional FL design introduces security vulnerabilities. FL models broadcast the whole set of model weights to the server on a regular basis. Sending all of the weights via the network, as demonstrated by [7], is potentially damaging due to the possibility of sensitive data being gathered from the reverse computation of model weights.

The majority of the previous research has focused on the clients interaction and global optimization in order to improve FL's coordination. There is no body of work that has studied how to maintain robustness in FL's UNE. Additionally, although efforts have been made to employ GWO to facilitate FL in a number of ways, GWO has never been utilized to enhance network communication performance by improving the global model's performance. This paper aims to improve the efficiency of the global model in FL by modifying the data format used in communication between clients and servers based on GWO.

III. Federated Grey Wolf Optimizer

Artificial neural networks (ANNs) are computer programs based on biological principles and are intended to imitate how the human brain processes information. ANNs work by passing information through layers of nodes, or neurons, that process data in a manner similar to how our brains process information. The ANN is composed of hundreds of single units, known as processing elements (PE) or artificial neurons, connected through weights or coefficients that comprise a neural structure organized in layers.

In neural computations, the strength of a network is obtained from the connections of neurons. The network's ability to process information and make decisions improves with the number of neurons it contains. For each PE, there is a single output, as well as a weighted input and a transfer function. The neural network is influenced by its architecture, the learning rule, and the transfer functions of its neurons. In this respect, a neural network is a parametric system, with the weights serving as the trainable parameters. The neuron's activation is determined by the weighted sum of the inputs being fed into the neuron. In order to create a single output, the activation signal must be transferred through the transfer function. Nonlinearity is introduced into the network through the transfer function [19].

An overall strategy for increasing the performance of ANN models is to increase the model's layer depth. This is what is referred to as a 'deep neural network". The more layers, the more weight characteristics are present to train. Network communication costs increase dramatically when the model trained on the device is sent to the server in traditional FL. As a result, we present the FedGWO method, which delivers the server the best result (in terms of loss or accuracy) by exploiting GWO features to transfer any sized trained model.

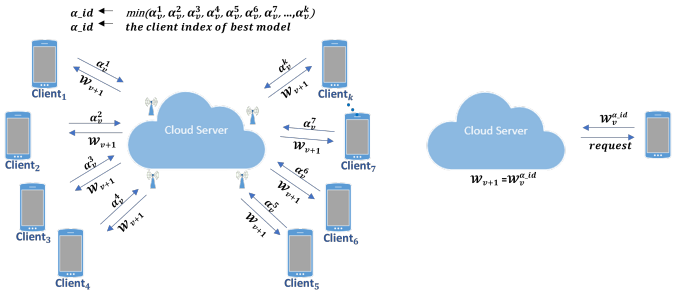

Fig. 3. The process of updating the weights of the FedGWO algorithm. The client with the best score value receives a request from the server to be used as the global model after the server receives the scores from all clients

The FedGWO process is provided in Fig.3. First, the model and GWO parameters are initialized by the server. Second, the model is sent to the clients who will participate in the round. Third, the model is trained using FedGWO weights by each selected client. Following that, the score is calculated based on the lowest loss value or highest accuracy. The best score value is sent to the server from each client. The loss or accuracy values are only four bytes. Finally, the best model's weighted collection is updated (on the server) by the best client with the best score (i.e., $alpha$ solution). In order to reduce communication costs, we assume the second and third best solutions would be in the same region in the search space as the best solution. The algorithm of FedGWO is presented in Algorithm 2. The FedGWO code is available in [20]

Algorithm 2: FedGWO algorithm

1: Initialize $\overrightarrow{EC}$,$\overrightarrow{EA}$, Number of dimensions, Number of wolves, Number of iterations w_0 , $alpha$).
2: **while** The current iteration less than the maximum number of iterations **do**
3: **for** each client k **do**
4: **for** each weight layer l **do**
4: Update the weight using Eq(12)
5: **end for**
6: **end for**
7: Best Score $\leftarrow \alpha_{id}$.
8: **end while**
9: Request from server to client(α_{id}).
10: Receive w from client to server.
11: Return $w + 1$, which is the best model.

IV. Experiments

In order to evaluate the proposed algorithm (i.e., FedGWO), we carried out experiments to analyze the convergence speed and accuracy, as well as studied FedGWO under UNE in order to determine its efficacy. We intended to assess if the model had acceptable convergence speed and accuracy in the first trial, considering its lower network connection requirements than FedAvg. We compared the accuracy of three algorithms (i.e., FedAvg, FedPSO [21], and FedGWO) using the Canadian Institute for Advanced Research's (CIFAR-10) dataset and examined the cost of data transfer across clients and servers.

We then examined the accuracy of FedGWO and FedAvg in a variety of network conditions under the second set of experiments.

A. Experimental Setup

The studies were done on a Windows server equipped with a CPU 2.50 GHz (2 processors) with 128 gigabytes (128 gigabytes usable), a Xeon(R), an Intel (R), and 465 gigabytes of memory. TensorFlow version 2.3.0 and Keras version 2.4.3 were used to create our experimental code. The study was proposed to enhance Federated Learning's (FL) network communication performance. By utilizing GWO, we were able to update the distributed model's weights while also changing the data format sent from the client to the server. The Convolutional Neural Network (CNN) model provided great accuracy. As a result, a two-layer CNN model with 32 and 64 channels, each followed by 2 x 2 maximum pooling, was employed in our experiments. This model is identical to the one used by FedAvg [15]. Table I illustrates the layers of the corresponding model.

TABLE I
CNN PARAMETER SETTINGS

ID	Shape	Layer
1	5 x 5 x 32	Conv2D
2	32	Conv2D
3	5 x 5 x 64	Conv2D
4	64	Conv2D
5	1024 x 512	Dense
6	512	Dense
7	512 x 10	Dense
8	10	Dense

The experiment utilized the CIFAR-10 dataset. This dataset is an image dataset utilized regularly for image classification problems. It contains 32 x 32-pixel images from ten different categories, such as airplanes, vehicles, and cats, as well as 10,000 testing and 50,000 training images. To begin training, the CIFAR-10 dataset was shuffled, allocated to client numbers, and dispersed to each client. Except for the dropout layer, no independent tuning method was performed to increase accuracy throughout the training phase.

FedGWO and FedAvg employed SGD techniques to train the clients, with a learning rate of 0.0025. Table II also contains the hyperparameter values utilized in the conducted experiments.

TABLE II
PROPOSED MODEL CONSTANT

ID	Parameter	FedGWO	FedAvg
1	Batch	10	10
2	Client-epoch	5	5
3	Epoch	30	30
4	C	-	0.1, 0.2, 0.5, 1.0
5	Client	10	10

B. Experimental Results for Accuracy

Fig.4 and Table III show the accuracy of experimental findings using the CIFAR-10 dataset. The results presented were generated using test data. FedGWO showed enhanced performance at an accuracy of 76.1% in comparison to FedPSO under all circumstances and from early iterations. FedAvg had a maximum accuracy of 67.14% when C = 1.0.

It is worth mentioning here that the maximum number of clients that can be trained at one time is specified by the constant C, which can take a value between 0 and 1. The experiment was done in rounds of communication by randomly picking a client ranked as high as C from all the clients. When the value of C in Figs.4 and 5 is increased, the accuracy increases proportionately, as well as the amount of data exchanged between the client and server. At C = 0.5, the accuracy difference is higher.

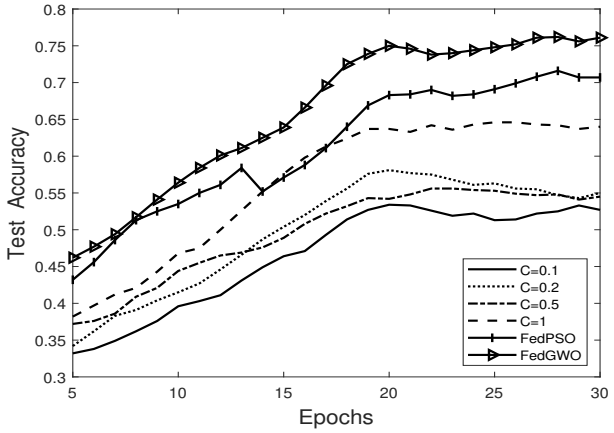

Fig. 4. A comparison of the test accuracy of different algorithms.

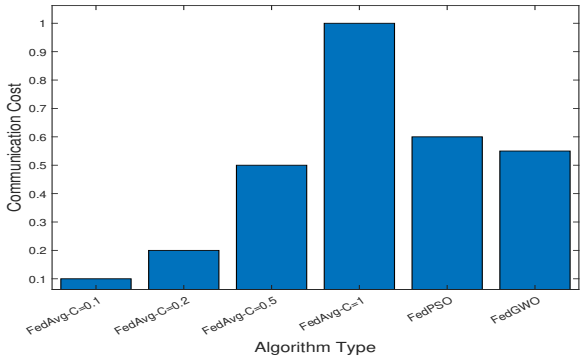

Fig. 5. A comparison of the communication cost of different algorithms.

C. Results for Unstable Network Environments

Under this experimental setup, we created a network environment that was unstable. During each communication cycle, data was randomly dropped from the client to the server. To highlight the accuracy difference between the three algorithms in this context, data were discarded in increments of 0%,

TABLE III
TEST ACCURACY COMPARISONS

Algorithm	Accuracy (Testing)
FedAvg,C = 0.1	51.39%
C = 0.2	59.07%
C = 0.5	65.00%
C = 1.0	67.14%
FedPSO	70.12%
FedGWO	**76.10%**

10%, 20% and 50%. Finally, to evaluate the experiment's validity, we calculated the results by averaging over ten trial runs. Table IV contains detailed accuracy findings. Our proposed FedGWO showed an improvement of 5% accuracy in comparison to FedAvg under an UNE, where data is not being fully delivered.

TABLE IV
ACCURACY AGAINST COMMUNICATION FAILURE PROBABILITY.

Algorithm	Failure Rate50%	20%	10%	0%
FedAvg,C = 1.0	59.55%	61.09%	61.48%	67.14%
FedPSO	65.47%	68.41%	69.18%	70.12%
FedGWO	**72.64%**	**74.78**	**75.16**	**76.10%**

V. CONCLUSION

This paper proposes a novel Federated Grey Wold Optimization (FedGWO) to enhance the performance of FL's network communication and lower the bulk of data delivered between the server and clients. By sharing the accuracy or loss values, the proposed technique aggregates the model learned on the server. The client with the highest score provides the server with the trained model. The proposed method was trained using a two-layer CNN and evaluated on the CIFAR-10 datasets. On average, FedGWO showed a 13.55%. While training the same number of devices, the accuracy increased by 4.54%, even while network connection costs were reduced by 59%. FedGWO is capable of successfully implementing FL even when the network connection is unreliable and large amounts of data are difficult to transmit to servers. Additionally, FedGWO is, on average 5% more resilient than FedAvg when communication data is randomly deleted. In the future, we plan to combine other metaheuristic algorithm components with the GWO technique to enhance the network communication performance. For instance, we will investigate the hybridization of GWO with other optimization algorithms. Additionally, We want to use various network protocols, such as the gossip protocol, to improve the network communication efficiency in the presence of frequent client dropouts and restricted network capacity.

ACKNOWLEDGEMENTS

This work was supported by the Machine Learning Department, Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI), UAE under grants No: 8481000021.

REFERENCES

[1] G. Wainer, M. Aloqaily *et al.*, "Machine learning-based indoor localization and occupancy estimation using 5g ultra-dense networks," *Simulation Modelling Practice and Theory*, vol. 118, p. 102543, 2022.
[2] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *arXiv preprint arXiv:1811.03604*, 2018.
[3] A. K. Abasi, M. Aloqaily, and M. Guizani, "sine cosine algorithm for reducing communication costs of federated learning," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE, 2022.
[4] S. Banabilah, M. Aloqaily, E. Alsayed, N. Malik, and Y. Jararweh, "Federated learning review: Fundamentals, enabling technologies, and future applications," *Information Processing Management*, vol. 59, no. 6, p. 103061, 2022.
[5] L. Shi, J. Shu, W. Zhang, and Y. Liu, "Hfl-dp: Hierarchical federated learning with differential privacy," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–7.
[6] Z. Yang, M. Chen, K.-K. Wong, H. V. Poor, and S. Cui, "Federated learning for 6g: Applications, challenges, and opportunities," *Engineering*, 2021.
[7] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
[8] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in engineering software*, vol. 69, pp. 46–61, 2014.
[9] S. N. Makhadmeh, A. K. Abasi, M. A. Al-Betar, M. A. Awadallah, I. A. Doush, Z. A. A. Alyasseri, and O. A. Alomari, "A novel link-based multi-objective grey wolf optimizer for appliances energy scheduling problem," *Cluster Computing*, pp. 1–28, 2022.
[10] O. A. Alomari, S. N. Makhadmeh, M. A. Al-Betar, Z. A. A. Alyasseri, I. A. Doush, A. K. Abasi, M. A. Awadallah, and R. A. Zitar, "Gene selection for microarray data classification based on gray wolf optimizer enhanced with triz-inspired operators," *Knowledge-Based Systems*, vol. 223, p. 107034, 2021.
[11] Z. A. A. Alyasseri, O. A. Alomari, S. N. Makhadmeh, S. Mirjalili, M. A. Al-Betar, S. Abdullah, N. S. Ali, J. P. Papa, D. Rodrigues, and A. K. Abasi, "Eeg channel selection for person identification using binary grey wolf optimizer," *IEEE Access*, vol. 10, pp. 10 500–10 513, 2022.
[12] A. K. Abasi, A. T. Khader, M. A. Al-Betar, S. Naim, S. N. Makhadmeh, and Z. A. A. Alyasseri, "An improved text feature selection for clustering using binary grey wolf optimizer," in *Proceedings of the 11th National Technical Seminar on Unmanned System Technology 2019*. Springer, 2019, pp. 503–516.
[13] H. Faris, I. Aljarah, M. A. Al-Betar, and S. Mirjalili, "Grey wolf optimizer: a review of recent variants and applications," *Neural computing and applications*, vol. 30, no. 2, pp. 413–435, 2018.
[14] J. Konečnỳ, B. McMahan, and D. Ramage, "Federated optimization: Distributed optimization beyond the datacenter," *arXiv preprint arXiv:1511.03575*, 2015.
[15] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.
[16] O. Bouachir, M. Aloqaily, Ö. Özkasap, and F. Ali, "Federatedgrids: Federated learning and blockchain-assisted p2p energy sharing," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 1, pp. 424–436, 2022.
[17] Y. Chen, X. Sun, and Y. Jin, "Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 10, pp. 4229–4238, 2019.
[18] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.
[19] B. Wang, Y. Sun, B. Xue, and M. Zhang, "A hybrid ga-pso method for evolving architecture and short connections of deep convolutional neural networks," in *pacific rim international conference on artificial intelligence*. Springer, 2019, pp. 650–663.
[20] A. K. Abasi, M. Aloqaily, and M. Guizani. [Online]. Available: https://github.com/Artifitialleap-MBZUAI/Grey-Wolf-Optimizer-for-Reducing-Communication-Costs-of-Federated-Learning
[21] 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, 2021.