International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 15, No.1 (Mar-24)

http://dx.doi.org/10.12785/ijcds/150198

Alpha-FedAvg: Safeguarding Privacy and Enhancing Forensic Analysis in Federated Learning on Edge Devices

Karam Muhammed Mahdi Salih [1] **and Najla Badie Ibraheem**[2]

[1]*Department of Computer Network, Ninevah University, Mosul, Iraq*
[2] *Department of Computer Science, University of Mosul, Mosul, Iraq*

Received 20 Jan. 2024, Revised 6 Mar. 2024, Accepted 8 Mar. 2024, Published 15 Mar. 2024

Abstract: In this paper, a novel federated learning algorithm for decentralized settings on edge devices—Alpha-FedAvg—is introduced. Using an adaptive learning rate approach based on Lipschitz and Smoothness parameters, Alpha-FedAvg dynamically modifies the learning rate for every node. Through federated averaging, the approach accomplishes model aggregation, exhibiting enhanced convergence and performance. An extensive test configuration includes using Kali Linux to simulate network assaults, an ESP32 microcontroller connected to a laptop equipped with a sound sensor, and Wireshark and Scapy for traffic analysis. The Alpha-FedAvg algorithm offers a privacy-preserving solution by effectively identifying and thwarting network attacks without gaining access to user data. The algorithm's performance is demonstrated in a comprehensive report generated. Evaluation against IID and non-IID datasets, such as Edge-IIoTset, and comparison with other models validate Alpha-FedAvg's efficacy in federated learning applications.

Keywords: Federated learning, Privacy-Preserving, Anomaly detection, Edge devices, Alpha-FedAvg, Forensic analysis

1. INTRODUCTION

In the field of federated learning, research on shared model training without access to user data has gained importance [1]. With an emphasis on enhanced anomaly detection precision, in order to address real-world challenges, this work introduces Alpha-FedAvg, a carefully designed model aggregation strategy. Federated learning is becoming more and more popular as a viable solution in privacy-sensitive domains because of its ability to train shared models without requiring access to user data. By investigating real-world problems in federated learning scenarios, the work seeks to increase the accuracy of anomaly detection, thereby advancing the field of machine learning. The strategy is divided into several important stages: Through a thorough literature review, the state of the art in federated learning was thoroughly examined, with particular attention paid to the various methods and protocols put forth to address issues with model aggregation and anomaly detection. Building on the knowledge gained from the literature review, a conceptual framework for federated learning was created, highlighting factors that are essential to learning process optimization. This framework was then used to develop and assess the Alpha-FedAvg algorithm. The Alpha-FedAvg algorithm was carefully designed, taking into account dynamic learning rate adaptation to improve efficiency and adaptability across a range of system configurations. It did this by utilizing mathematical concepts like Lipschitz and Smoothness parameters to guarantee stability and convergence in the federated learning process. Numerous tests were carried out to analyze the Alpha-FedAvg algorithm's performance in a range of machine learning models and system configurations. Both simulations and real-world deployments were used to gauge the algorithm's effectiveness in various contexts. After a thorough analysis of the experimental validation results, conclusions were reached regarding the algorithm's performance as well as its efficacy in resolving real-world federated learning challenges. An in-depth comprehension of the proposed was intended. Alpha-FedAvg algorithm and its implications for advancing federated learning methodologies by scheduling the entire research process, from literature review to experimental validation. Alpha-FedAvg, motivated by decentralized, synchronous, diverse, and privacy-preserving objectives, transforms federated learning for Edge IoT contexts. In digital forensics, it guarantees the validity and dependability of the evidence, improving the accuracy of IoT sensor data analysis, especially in attack detection. Its processing efficiency is optimized by its real-time adaptive mechanism, highlighting its critical role in scalable and successful edge IoT-based forensic investigations.

E-mail address: Karam.mahdi@uoninevah.edu.iq, Najlabadie@uomosul.edu.iq

Federated learning research highlights the difficulty of training shared models without user data access, requiring creative solutions for anomaly detection accuracy and model aggregation. Through research and analysis, the aim was to improve federated learning methods and significantly impact the domains of forensic analysis and deep learning.

2. Related work

The investigation of shared model training without gaining access to user data has gained prominence in the federated learning scene. This work presents a well-designed model aggregation approach that tackles practical issues, with a focus on improved anomaly detection precision. Federated learning has drawn notice recently for its ability to train shared models without requiring access to user data. The capacity of federated learning to train shared models without requiring access to user data has drawn attention in recent years.

The paper [2] presents a selected model aggregation strategy that tackles real-world federated learning challenges and increases the accuracy of anomaly detection. The proposed Privacy-Preserving Asynchronous Federated Learning Mechanism for Edge Network Computing (PAFLM) allows multiple edge nodes to achieve more efficient federated learning without sharing their private data. PAFLM compresses communications between nodes and the parameter server during training without affecting accuracy, and it supports node join or quit at any learning process, making it suitable for highly mobile edge devices [3]. In a real-world cellular network, the research article suggests a novel FL protocol called FedCS that effectively carries out federated learning with heterogeneous clients. FedCS dramatically cuts training time as compared to the original FL technique, according to the experimental evaluation [4]. The study [5] suggests two methods for federated deep learning-based traffic flow prediction: FedGRU and FedGRU algorithms based on clustering. With a Joint-Announcement protocol, it presents an enhanced Federated Averaging (FedAVG) technique to lower communication cost in the aggregation mechanism. The study [6] suggests a novel federated learning algorithm that permits training over an enormous number of edge devices. The suggested algorithm's behavior is dependent on various parameters and demonstrates a convergence rate between FedAvg and single-thread SGD. To protect user privacy and accurately forecast demand for popular applications on mobile edge computing networks, this study [7] presents a federated learning architecture.

The system achieves high levels of accuracy in anticipating application demand by combining feedback from users' local training to develop a global and weighted model. In this work, a novel class of gradient algorithms for distributed machine learning called Lazily Aggregated Gradient (LAG) is presented. By adaptively omitting gradient calculations, LAG decreases computation and communication [8]. While [9] suggests a server-side method for federated learning systems' anomalous client detection. By creating low-dimensional surrogates of model weight vectors and applying them to anomaly detection, the authors surpass traditional defense-based techniques. Article [10] suggests using the FedPer method to train deep feed forward neural networks federatedly. It tackles the problem of statistical heterogeneity of data between user devices, which might deteriorate the efficacy of conventional federated averaging in machine learning applications such as deep learning personalization.Paper [11] suggests an edge computing-based federated learning strategy that is optimal. It attempts to solve the computational and privacy-sensitive data restrictions of mobile devices. Federated machine learning research that suggests an adaptive boosting technique for intensive care unit (ICU) data. The approach [12], known as LoAdaBoost, attempts to solve the problem of disparate data distributions across several sources. In order to handle heterogeneity in federated networks [13]-a distributed learning paradigm characterized by diversity in system parameters and non-identically distributed data across the network—the study presents the FedProx framework. In resource-constrained edge computing systems, where a lot of data is created at the network edge, this research [14] suggests an adaptive federated learning strategy. A general class of gradient descent-based machine learning models is the subject of the study. In order to improve forecasting accuracy in dynamic circumstances, the study [15] presents the MulticloudFL system, which focuses on deep learning local loss functions in multi-cloud contexts. In Table I an overview has been made for AlphaFedAvg related Models. The study that is being presented highlights the increasing importance of federated learning in the current machine learning environments, especially because of its ability to train shared models while maintaining user data privacy. The investigation of diverse approaches and procedures underscores the ongoing projects in the domain to optimize model aggregation strategies and augment anomaly detection accuracy.

3. Non-IID federated learning

A machine learning technique called federated learning enables many people to jointly train a model without disclosing any of their personal information. Federated learning is particularly helpful in the IoT environment because it enables us to make use of the data produced by various IoT devices while protecting data privacy [16]. However, in federated learning, the assumption of independently and identically distributed (IID) data is frequently broken in scenarios involving the Internet of Things (IoT), where devices may have various data distributions due to variances in their sensors, locations, or surroundings [17]. Recent studies have proposed alternative non-IID federated learning strategies, including as data partitioning, client weighting, and model personalization, to overcome this issue. In order to incentivize clients to exchange more diverse data samples during model updates, one frequent strategy is to incorporate a non-IIDness penalty term into the objective function of federated learning. It has been demonstrated that FedAvg

TABLE I. Overview of AlphaFedAvg related models

Paper	Research Problem	Proposed Solution
(Nishio & Yonetani, 2018)	Reduce training time in federated learning with heterogeneous clients	FedCS
(Huang et al., 2018)	Adaptive boosting for federated learning in ICU data	LoAdaBoost
(T. Li et al., 2018)	Handle heterogeneity in federated networks	FedProx
(Wang et al., 2018)	Adaptive federated learning for resource-constrained edge computing	Adaptive federated learning strategy
(Chen et al., 2018)	Reduce computation and communication in federated learning	Lazily Aggregated Gradient (LAG)
(S. Li et al., 2019)	Detect anomalous clients in federated learning systems	Server-side method
(Arivazhagan et al., 2019)	Train deep feed forward neural networks federatedly	FedPer
(Qin et al., 2020)	Selected model aggregation strategy for anomaly detection	Privacy-Preserving Asynchronous Federated Learning Mechanism for Edge Network Computing
(Lu et al., 2020)	Efficient federated learning without sharing private data	PAFLM
(Liu, Zhang, et al., 2020)	Federated deep learning-based traffic flow prediction	FedGRU, FedGRU*, Joint-Announcement
(Fantacci & Picano, 2020)	Protect user privacy and forecast demand for applications	Federated learning architecture
(Ye et al., 2020)	Federated learning for edge computing	Optimal federated learning strategy
(Xie et al., 2020)	Train over a large number of edge devices	Novel federated learning algorithm
(Stefanidis et al., 2023)	Improve forecasting accuracy in dynamic federated learning	MulticloudFL

frequently produces greater model accuracy. FedAvg takes advantage of collaborative learning by aggregating models rather than just gradients, which gives the central server a more complete picture of the data. This might result in better model performance [18]. It can be expressed mathematically as follows 1 and 2:

$$8k; w_{t+1}^{k} \quad w_t \quad g_k \tag{1}$$

$$w_{t+1} \quad \sum_{k=1} \frac{n_k}{n} w_t^k + 1 \tag{2}$$

Were the k is the client, nk is the number of data points on client k, w_t^k is the model weights on communication round t on client k, and is the learning rate. The application of Lipschitz parameter and smoothness parameter in the context of privacy guarantees and federated learning emphasizes its significance in shaping theoretical foundations and model performance considerations. The Lipschitz constant, or Lipschitz parameter, is a fundamental concept in mathematical analysis, particularly in the realms of functions and optimization. Lipschitz continuity is a key attribute associated with functions and plays a vital role in optimization algorithms, especially those leveraging gradient-based methods. Lipschitz continuity is defined for a real-valued function $f : Rn \; ! \; R$. A function is considered Lipschitz continuous with a Lipschitz constant L if, for all pairs of points x_1 and x_2 in the domain, the absolute difference of the function values is bounded by the Lipschitz constant times the distance between the points. This condition is expressed as [19]:

$$\mathrm{j}f(x_1) \quad f(x_2)\mathrm{j} \quad \mathrm{k}x_1 \quad x_2\mathrm{k} \tag{3}$$

where $\mathrm{k}x_1 \quad x_2\mathrm{k}$ is a norm (e.g., Euclidean norm) measuring the distance between x1 and x2. The Lipschitz constant is crucial in optimization, it indicates that the gradient (or derivative) of the function changes in a controlled manner. This property aids in optimizing algorithms, making onvergence more predictable and stable. In optimization contexts, the Lipschitz constant is often denoted as L and is associated with the gradient of the function. For a differentiable function, if the gradient is Lipschitz continuous with constant L, it implies that the magnitude of the gradient at any point doesn't exceed L times the distance between points. This information is valuable for controlling the step size in gradient-based optimization methods, preventing excessively large steps that could lead to divergence. The Lipschitz constant serves as a metric for quantifying how much a function can change concerning changes in its input.

It contributes to ensuring stability and convergence in the optimization process. The Euclidean norm, utilized in the Lipschitz continuity definition, is a measure of the magnitude of a vector in Euclidean space, providing a distance

measure in n-dimensional space. The smoothness parameter is a fundamental concept also in mathematical analysis, particularly in the study of functions' behavior concerning changes in their input. In the context of federated learning, the smoothness parameter (L) is associated with the Lipschitz continuity of the gradient of the global loss function $f(\ )$, ensuring stability during the optimization process. The smoothness condition can be expressed mathematically as [20]:

$$\| \nabla f(\ ) \quad \nabla f(\ )_0 \|_2 \quad \| \quad _0 \|_2 \qquad (4)$$

Here: $\nabla f(\ )$ represents the gradient of the global loss function with respect to the model parameters ( ). and $_0$ are different parameter values. $\| \|_2$ denotes the Euclidean norm, measuring the distance in parameter space. This equation indicates that the squared difference between the gradients at different parameter points is bounded by L times the squared distance between those points. In federated learning, enforcing smoothness is crucial for controlling the step size in optimization algorithms, ensuring convergence across distributed devices, and adapting to the dynamic and heterogeneous nature of the data sources.

4. Overview of propsed ALPHA-FedAVG Algoritm

As shown in Figure 1, the system consists of a network of edge nodes, which are Internet of Things (IoT) sensors that connect to the central server as local clients and engage in synchronous federated learning. The server gathers models from these dispersed nodes in this collaborative setup and returns updated models to the clients. One of the system's primary contributions is a novel convergence bound that offers a framework for comprehending the dynamics of learning in federated learning. Furthermore, the aggregation can be adjusted in real-time thanks to the Alpha-FedAvg algorithm. frequency, enhancing the efficiency of federated learning by dynamically adjusting the learning process. Through extensive experiments, the system has demonstrated near-optimal performance in a range of machine learning models and configurations. Alpha-FedAvg algorithm offers a dynamic and adaptive approach to federated learning, allowing the system to optimize the trade-off between parallelism and accuracy by adjusting the step size based on real-time information. The algorithm emerges as a crucial component, facilitating efficient and effective federated learning in a wide range of scenarios.

The algorithm described involves a federated learning framework where a central server (Alpha server) collaborates with multiple edge nodes (Alpha clients) to optimize a global model. A crucial aspect of this optimization process lies in the dynamic adjustment of the learning rate ( ) based on Lipschitz and Smoothness parameters, along with the average delta gradient divergence across all participating nodes. The Lipschitz and Smoothness parameters are computed to quantify the local variations in model updates, ensuring a balance between rapid convergence and stability. The delta gradient divergence, representing the disparity between local and global model weights, contributes to the adaptability of the learning rate. Of $_{new}$ encapsulates this adaptive learning rate mechanism. This equation reflects integrating Lipschitz and Smoothness considerations, and is designed to promote convergence in the federated learning process while accounting for diverse local data characteris-

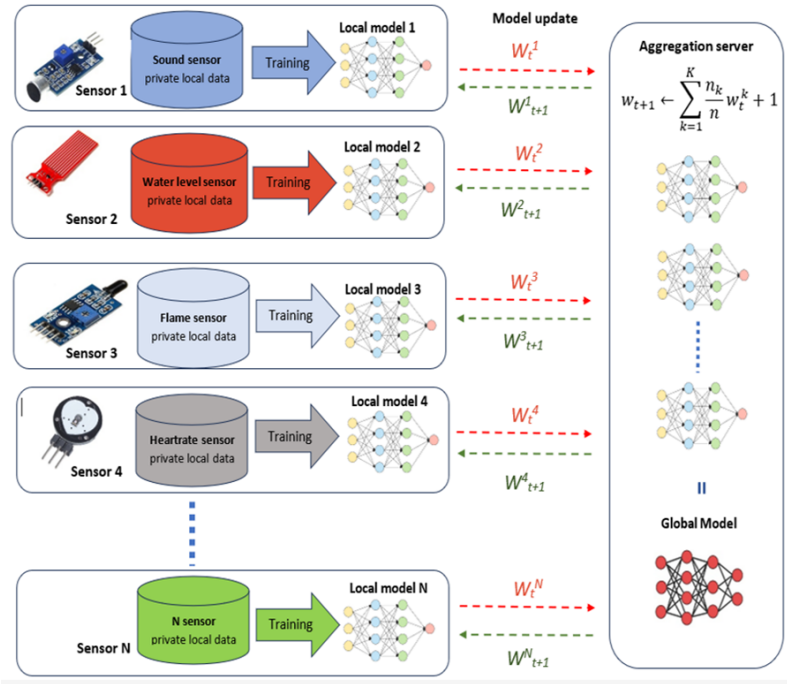

Figure 1. The proposed synchronous federated learning system

tics.

In federated learning scenarios, achieving effective model aggregation requires careful consideration of the tuning parameters for Alpha-FedAvg. By optimizing the trade-off between parallelism and accuracy, the algorithm is guaranteed to dynamically adapt to the distinct features of the data distribution of each edge node thanks to specially designed parameters. Through the adjustment of parameters like the learning rate according to Lipschitz and Smoothness considerations, Alpha-FedAvg is able to effectively handle the diverse complexities associated with local data updates, facilitating the convergence of the model towards global optimization. Furthermore, during the aggregation process, stability and convergence checks are crucial defenses against divergence and inconsistency. These checks prevent disruptions and promote a coherent learning trajectory by ensuring that model updates from various nodes meaningfully contribute to the overall improvement of the model. Fundamentally, careful parameter tuning selection and stringent stability checks are the cornerstones of Alpha-FedAvg's pursuit of efficient model aggregation, guaranteeing that federated learning systems can realize their full potential in decentralized settings.

The derivation of the provided equation of $_{new}$ in Alpha server algorithm involves Lipschitz and Smoothness parameters, which are crucial in optimizing machine learning models. Lipschitz (LP) and Smoothness (SP) parameters quantify how sensitive a model's output is to changes in its input and parameters, respectively. These parameters play a pivotal role in controlling the learning rate during optimization to ensure convergence. The Delta Gradient Divergence ($_i$) signifies the difference between local and global model weights for each node, reflecting the divergence in model updates across the network. The equation for the New Learning Rate ($_{new}$) dynamically adjusts the learning rate based on these parameters and the average delta gradient divergence. It incorporates the Lipschitz and Smoothness parameters in a product, representing the overall control factor. The adaptive adjustment involves rounding to the nearest integer for practical implementation.

A heuristic or empirical adjustment equation commonly employed in machine learning for hyperparameter tuning. The equation's components demonstrate a thoughtful consideration of Lipschitz and Smoothness parameters, reflecting an effort to adapt the learning rate to the varying characteristics of local data in a federated learning setting, ultimately aiding convergence with stability in the optimization process. Although empirical in nature, such heuristics are valuable tools in fine-tuning models for optimal performance. Table II provides an overview of the main notations of the algorithm.

TABLE II. Summary of main notations

w_f	Final global model parameters
g_t	Global iteration
l_t	Local iteration
	The quantity of local update stages that separate two global aggregations.
$w(g_t)$	Global model parameter in global iteration g_t
$w_i(g_t)$	Local model parameter at node i in global iteration g_t
F(w)	Global function
F_i(w)	Local function at node i
L_P	Lipschitz parameter
S_P	Smoothness parameter
$_i$	Gradient divergence
D_i	Local data of node i

A. Alpha Server Algorithm

Through iterative communication, the server sends and receives local model updates as well as global model weights from edge nodes. These updates are aggregated using federated averaging, which allows for cooperatively improving the global model. In order to assess convergence, the act of the current global model is linked to that of the best-performing model found thus far. If improvements are noted, the global model is updated. This information sharing process enhances convergence and refines the adaptive learning rate. Every edge device on the client side participates in local training, which helps the group refine the global model. The process keeps going until the predefined number of rounds is reached or the convergence criteria are satisfied, producing a globally optimized model that is well-suited to the various features of the edge devices that are involved in the federated learning environment. In order to compute $_{new}$ dynamically on the server, clients submit control parameters, such as weight vectors and losses, to the server-side algorithm, which initializes with pertinent parameters. In the federated learning process, the algorithm seeks to balance accuracy and parallelism. Furthermore, it improves adaptive control by using parameters such as LP, SP, and delta in the computation of the new optimal . Here is the algorithm of Alpha-server:

- Input Model_name, search range parameter (Max_case), n_node, $_{max}$, _range.
- Initialize = 1, $g_t = 0$.
- Initialize Model weights w(0) to random variable.
- Initialize $w_f = w(0)$

Repeat

Send $w(g_t)$, , (is last round) flag to all edge nodes

$g_{t0} = g_t$

$g_t = g_t +$

Receive $w_i(g_t)$ from each node i

Call federated_average for global aggregation

$$w(g_t) = \frac{\sum_{i=1}^{N} w_i(g_t)}{N}$$

If $g_{t0} > 0$ then

Receive L_P; S_P, Weights after each local training.

$$\text{Compute } F(w) = \frac{\sum_{i=1}^{N} D_i w_i(t)}{D}$$

If $F(w(g_{t0})) < F(wf)$ then

$w_f = w(g_{t0})$

$$\text{Calculate total } L_P = \frac{\sum_{i=1}^{N} D_i L_{Pi}}{D}$$

$$\text{Calculate total } S_P = \frac{\sum_{i=1}^{N} D_i sp_i}{D}$$

Calculate delta gradient divergence

$$_i = \frac{\sum_{i=1}^{N} D_i delta_i}{D}$$

for each node i

where

$$delta_i = \| wi(gt) \quad w(gt) \|$$

Compute new value of :

$$_{new} = round\left(\frac{}{\max(1; L_P : S_P : Avg(\ _i) : \max(1; L_P : S_P))}\right)$$

$$\min\{ _{new};\ _{max}\}$$

Send $w(g_t)$ to all edge nodes

Receive $w_i(g_t)$ from each node i

$$\text{Compute } F(w) = \frac{\sum_{i=1}^{N} D_i w_i(t)}{D}$$

If $F(w(g_t)) < F(w_f)$ then $w_f = w(g_t)$

B. Alpha Client's Algorithm

Client side plays a crucial role in coordinating federated learning across Internet of Things (IoT) devices. It changes global model depending on local computations, simplifies round preparation, and initializes important variables while working in rounds. Notably, it analyzes variations in local and global weights and losses to compute adaptive control parameters (LP and SP). In handling loss computation exceptions, the class gets ready to talk to the central server. It provides ways to communicate with the server by sending messages that specify whether control parameters were computed locally. In the federated learning process, this adaptive control method makes sure that local IoT devices and the central server coordinate and communicate effectively, improving the system's overall efficiency. Here is the algorithm of Alpha-client:

- Initialize l_t =0
- While not (last_round)

Receive model_name, $w(g_t)$, and $_{new}$ from the Server.
Get model according to model_name
Set control algorithm to class ControlAlgAlphaClient()
If lt > 0 then

$$\text{Calculate } LP = \frac{\| Fi(wi(lt)) \quad Fi(w(gt)) \|}{\| wi(lt) \quad w(gt) \|}$$

$$\text{Calculate } SP = \frac{\| 4Fi(wi(lt)) \quad 4\ Fi(w(gt)) \|}{\| wi(lt) \quad w(gt) \|}$$

For = 0; 1; : : : ;

- $l_t = l_t + 1$
- Read D_i
- Train model on D_i
- Set $w(l_t)$ = model weights If =

  Set $w_i(g_t) = w(l_t)$

Send $w_i(g_t)$ to server

If $l_t > 0$ then

Send $L_P; S_P$, $w_i(g_t)$ to server.

-Until (last_round) flag set and received from server.

- Set $w_i(g_t) = w(g_t)$

With federated learning, the Alpha-FedAvg algorithm uses an adaptive learning rate technique with the goal of improving convergence and performance in a decentralized environment. By initializing the alpha value and configuring the global model parameters, the central server starts the procedure. This adaptive learning rate, represented by the letter alpha ( ), is dynamically modified according on the Smoothness and Lipschitz parameters. By capturing the unique local characteristics of each edge device, these parameters make sure that the learning rate is adjusted adaptively for every node. Through iterative communication, the server sends and receives local model updates as well as global model weights from edge nodes. These updates are aggregated using federated averaging, which allows for cooperatively improving the global model. In order to assess convergence, the act of the current global model is linked to that of the best-performing model found thus far. If improvements are noted, the global model is updated. This information sharing process enhances convergence and refines the adaptive learning rate. Every edge device on the client side participates in local training, which helps the group refine the global model. The process keeps going until the predefined number of rounds is reached or the convergence criteria are satisfied, producing a globally optimized model that is well-suited to the various features of the edge devices that are involved in the federated learning environment.

5. Experement results

Using datasets from a variety of sensors, such as sound, flame, water level, and heart rate sensors, federated learning on edge devices was used in the study to conduct several tests. The Edge-IIoTset provided the raw data that was used to create these datasets. Federated learning was started after one of these datasets was allocated to each client in the system. Furthermore, a real-case study involving the ESP32 microcontrollers and sensor

connection was carried out. The hardware and software configurations for this case study are detailed in Table III. For instance, the ESP32 microcontroller communicated wirelessly with a computer to transmit sound sensor data, as depicted in Figure 2 which displays the serial monitor of sound sensor dataOne of the most important parts of the experiment was the simulation of fourteen distinct attack scenarios, including Backdoor, DDoS_ICMP, DDoS_HTTP, DDoS_TCP, DDoS_UDP, Fingerprinting, MITM, Password, Ransomware, Port_Scanning, Uploading, SQL_injection, Vulnerability_scanner, and XSS.

TABLE III. Hardware and Software configuration

Hardware	Type	Version
Computer	Lenovo 20VG	ThinkBook 15 G2 ARE
ESP32	WROOM	ESP32 WROOM-32
Sound Sensor	Microphone	MEMS Microphone
Wire GNG	Connector	GPIO Connector
Wire 3v3	Connector	Voltage Regulator
Wire g34	Connector	GPIO Connector
Power wire	Cable	USB-C Power Cable
Router	Wireless Router	TP-Link Archer C7
Port com7	Port	COM7
Software Tool	**Type**	**Version**
Arduino IDE	IDE	2.0.0
Spyder	IDE	5.0.5
Wireshark	Network Analyzer	3.6.2
Tshark	Command Line	3.6.2
CMD	Command Prompt	Windows Command Prompt
Excel	Spreadsheet	Microsoft Excel 2023
Keras	Deep Learning Library	2.8.0
Scapy	Packet Manipulation	2.4.5

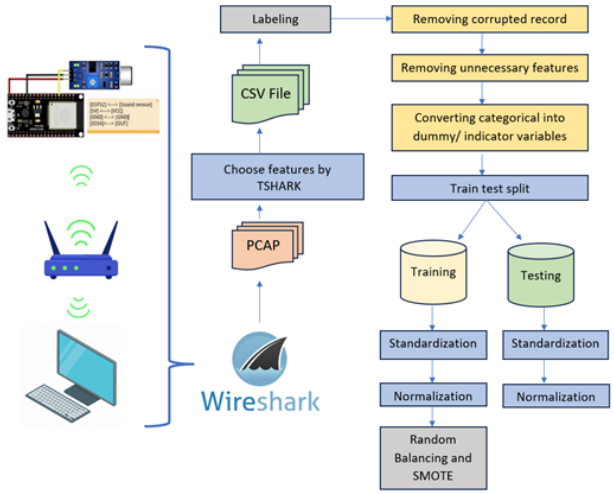

Figure 2. The steps of collect real time dataset

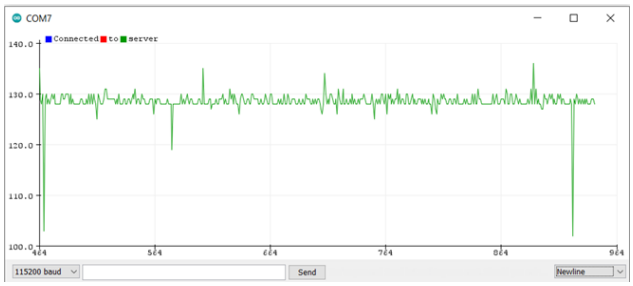

Figure 3. The collection of the sensor values in serial monitor

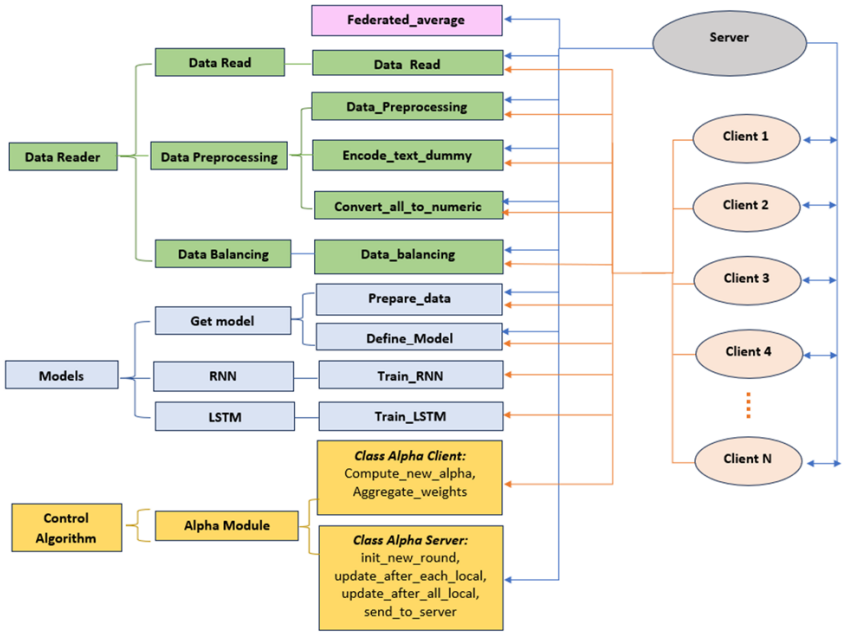

Figure 4. The main components of the system

TABLE IV. Comparison of Experimental Results: Alpha-FedAvg vs. Related Approaches in Federated Learning Models

		Method	Dataset	Accuracy	AI local model	No. of class	No. of node	Layer	Tunning
IID	(Nishio & Yonetani, 2019) [4]	FedCS	CIFAR-10	0.76	CNN	NaN	NaN	3 × 3 convolution layers	NaN
	(Nishio & Yonetani, 2019) [4]	FedCS	Fashion-MNIST	0.91	Residual network	NaN	NaN	NaN	NaN
	(Lu et al., 2020) [3]	LAG	MNIST	0.8913	MLP	10	10	3	Fixed interval
	(Lu et al., 2020) [3]	PAFLM	MNIST	0.9235	MLP	10	10	3	Fixed interval
	Alpha-FedAvg		Edge-IIoTset	0.926	CNN-autocoder	15	5	22 layers	dynamic
			Edge-IIoTset	0.989	RNN	2	4	6	dynamic
			Edge-IIoTset	0.9921	LSTM	2	4	6	dynamic
Non-IID	(Nishio & Yonetani, 2019) [4]	FedCS	CIFAR-10	0.54	CNN	NaN	NaN	3 3 convolution layers	NaN
	(Nishio & Yonetani, 2019) [4]	FedCS	Fashion-MNIST	0.71	Residual network	NaN	NaN	NaN	NaN
	(Qin et al., 2020) [2]	FedAvg	Fashion MNIST	0.866	NaN	2	5	NaN	dynamic
	(Qin et al., 2020) [2]	Score-based	Fashion MNIST	0.928	NaN	2	5	NaN	dynamic
	(Qin et al., 2020) [2]	Score-based + threshold	Fashion MNIST	0.942	NaN	2	5	NaN	dynamic
	Alpha-FedAvg		Four datasets generated from Edge-IIoTset	0.9305	RNN	2	4	6	dynamic
			Four datasets generated from Edge-IIoTset	0.9984	LSTM	2	4	6	dynamic
			Real-time data	0.8151	RNN	2	4	6	dynamic
			Real-time data	0.9745	LSTM	2	4	6	dynamic

Malicious packets, generated in Kali Linux, were combined with regular network traffic to produce a comprehensive packet capture file. The command prompt was utilized to extract relevant features from the pcap file using Tshark commands, subsequently translated to CSV format for additional analysis. Using programs like Excel, any gaps in the dataset were filled up, and a target variable was created to indicate attack or typical occurrences. The Alpha algorithm received the preprocessed dataset and, without accessing user data, globally aggregated model updates across edge devices while dynamically adjusting its learning rate .

Subsequently, the finished model underwent a rigorous testing process using the dataset to evaluate its capacity to identify and thwart network assaults. Lastly, a thorough report was produced utilizing the Scapy library, offering a comprehensive examination of the Alpha algorithm's performance in the context of federated learning for network traffic anomaly detection. The primary client-side and server-side components of the system are shown in Figure 4.

A dynamic alpha experiment was conducted with a range of Alpha_setup_all values [-1, 1, 2, 3, 5, 7, 10, 20, 30, 50, 70, 100], and control parameters set to 0.000025 and 0.00005. The optimal performance was achieved with control parameter values of 0.00025 and 0.0005 for prediction accuracy metrics, as shown in the Table V. Using the global model, At the end of the testing process, the system is thoroughly assessed. Scapy is a Python package used for network analysis and digital forensics that is used in packet analysis.

A comprehensive text report detailing system performance, possible vulnerabilities, and anomalies is created from the gathered data. The report provides technical data and contextual information that can be used as legal proof in the event that an assault is detected as a one of digital forensics important steps as shown in the Figure 5.

This approach guarantees the effectiveness of the system in technical and legal circumstances. The approach's effectiveness is demonstrated in terms of communication efficiency, loss function optimization, and model accuracy through comparison with related work, as summarized in the tables. Notably, the model demonstrates its robustness and adaptability in various federated learning settings by outperforming existing methods in several scenarios. Lastly, a thorough comparison of the results with similar federated learning studies in both IID and Non-IID scenarios is shown in Table IV. In terms of accuracy, model architecture, and adaptability to non-IID settings, the Alpha-FedAvg model consistently outperforms existing methods and exhibits competitive performance across a variety of datasets.

A thorough summary of federated learning techniques for both IID (Independent and Identically Distributed) and Non-IID Non-Independent and Non-Identically Distributed) scenarios is provided in the table. Within the IID category, CNN and residual network local models are used in the "FedCS" approach on datasets such as CIFAR-10 and Fashion-MNIST. On the MNIST dataset, the "LAG" and "PAFLM" approaches are used with MLP local models and fixed interval tuning. Notably, using a variety of AI local models and dynamic tuning, the suggested "Alpha-FedAvg" model is demonstrated in the IID context, attaining good accuracy across a variety of datasets, including Edge-IIoTset.Using CNN, Residual network, and other models, the "FedCS," "FedAvg," and "Score-based" approaches are applied to the CIFAR-10 and Fashion-MNIST datasets in the Non-IID context. When applied to real-time data and datasets generated from Edge-IIoTset, the proposed "Alpha-FedAvg" model consistently beats existing approaches. By using dynamically tuned RNN and LSTM local models, this model exhibits adaptability and achieves impressive accuracy. The trend that has been seen suggests that the "Alpha-FedAvg" model that has been suggested performs exceptionally well in a variety of circumstances, demonstrating its efficacy in datasets with disparate features.

Its excellent accuracy and performance are a result of the dynamic tuning approach and the freedom in selecting AI local models. Subsequently, the model has been compared with those already in use.

Federated learning approaches, considering communication methods, loss functions, and other key parameters as shown in Table VI. The model, referred to as Alpha-FedAvg, operates synchronously., incorporates deep learning loss functions, and dynamically adjusts iteration parameters during local training, aligning with the trends observed in recent literature. Following the identification of the ideal control parameter and the conclusion that the LSTM model produces the best results, the emphasis shifted to improving the training dynamics through an investigation of the effects of dynamic alpha. Experiments were carried out using the LSTM model and the control parameter set to 0.0005 to determine the optimal maximum value for dynamic alpha. The study entailed contrasting a fixed alpha technique with dynamic alpha that had variable maximum values (e.g., 100, 1000). Dynamic alpha was consistently preferred by the results, which showed better loss values and prediction accuracy.

TABLE V. Control parameter experiments

Control parameter	0.000025	0.00005	0.00025	0.0005	0.025	0.5
Predection	0.8747	0.8702	0.9427	0.9745	0.8468	0.839

TABLE VI. A comparison with existing federated learning approaches

Papers	Communi-cation	Loss Function dependency	Loss Func-tion of Deep Learning	Weighted-average of each participant's data size is used for aggregate	Selection of Client Participants Considering the Volume of Local Data	Selection of Client Participants Using the Local Loss Function Value	Dynamic tuning iteration in each local training
(Liu et al., 2020) [5]	Async	yes	yes	no	no	no	no
(Xie et al., 2020) [6]	Async	yes	no	no	no	no	no
(Fantacci & Picano, 2020) [7]	Async	yes	no	no	no	no	no
(Lu et al., 2020) [3]	Sync	yes	yes	no	no	no	no
(Qin et al., 2020) [2]	Sync	yes	no	yes	no	yes	yes
(Chen et al., 2018) [8]	Sync	yes	no	no	no	no	None
(S. Li et al., 2019) [9]	Sync	yes	n/a	no	yes	no	None
(Arivazhagan et al., 2019) [10]	Sync	yes	yes	yes	no	no	None
(Ye et al., 2020) [11]	Sync	yes	no	no	no	no	None
(Nishio & Yonetani, 2019)[4]	Sync	yes	yes	no	no	no	None
(Huang et al., 2020) [12]	Sync	yes	yes	no	no	no	None
(T. Li et al., 2020) [13]	Sync	yes	no	no	no	no	None
(Wang et al., 2019) [14]	Sync	yes	no	yes	no	yes	None
Alpha-FedAvg	Sync	yes	yes	yes	yes	yes	yes

```
Summary:
 In UTC time 1638828530.422045, the device (IP: 192.168.0.128, Port: 80)
 experienced an attack.
 The attacker's IP and port are (IP: 192.168.0.170, Port: 58902)
 according to protocol 6.
 The MAC address of the attacker is d8:f2:ca:8e:17:69,
 and the MAC address of the victim is dc:a6:32:fb:69:b5.
```

Figure 5. The final report that present to court of law

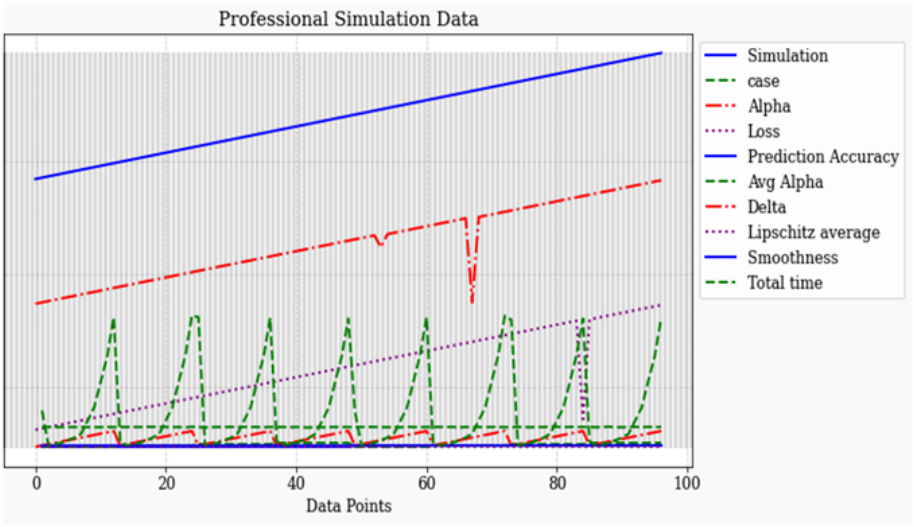

Figure 6. A visual representation of various parameters

The ideal maximum value for dynamic alpha was determined by painstaking investigation, striking a balance between model accuracy and training efficiency. This investigation highlights how important dynamic alpha tuning is for improving the learning flexibility and performance of the LSTM model. The fixed alpha examples in this Table VII, with values of 1, 3, 10, and 1000, showed clear trends in prediction accuracy and loss value between the first and last rounds. Interestingly, the dynamic alpha configurations (maximum value of 100) performed better than the fixed alpha configurations every time, resulting in lower loss values and higher prediction accuracies in both rounds.

Based on the last round's smallest loss value of 0.113 and highest prediction (on real time data) accuracy of 0.9951, the dynamic alpha configuration with a maximum value of 100 was found to be the ideal search space. This suggests that extending the maximum value past 100 might not provide any more advantages in identifying the best course of action. In comparison to fixed alpha configurations, the dynamic alpha technique found to be successful in improving the learning dynamics of the LSTM model, resulting in better performance.

After selecting a dynamic alpha of 100 in Alpha-FedAvg algorithm, Figure 6 provides a visual representation of how various parameters evolve across different iterations. The dynamic alpha, set at a value of 100. Taking note of the lines in Figure 6, each line corresponds to a specific parameter, and the x-axis represents the iterations while the y-axis represents the value of alpha. The green line, symbolizing the average alpha, plays a pivotal role as a key indicator for identifying local iteration entry points. By closely observing these entry points, Important

information is obtained about how the local convergence of individual parameters is shaped by the dynamic alpha of 100. It becomes evident that they tend to align with the best optimization direction. The interplay between the dynamic alpha and these parameters guides the algorithm towards optimal convergence, emphasizing the effectiveness of Alpha-FedAvg approach. This observation further underscores the intricate relationship between dynamic alpha, local convergence, and the overall optimization process in the algorithm.

TABLE VII. Alpha Tuning Experiment

Simulation Case	Alpha Setup	Loss Value	Prediction Accuracy	Avg Alpha	Delta average	Lipschitz average	Smoothness average	Total Global iteration	Total Time
First round	Fixed 1	0.245	0.844	1	0.0067	0.130	0.053	8	90.14
	Fixed 3	0.230	0.861	2.985	0.0133	0.124	0.053	8	15.01
	Fixed 10	0.224	0.871	9.956	0.0025	0.130	0.056	8	15.05
	Fixed 1000	0.224	0.872	912.232	0.0044	0.078	0.017	8	15.10
	Dynamic (Max 100)	0.223	0.872	27.675	0.0027	0.131	0.053	96	15.07
	Dynamic (Max 1000)	0.223	0.873	98.648	0.0011	0.145	0.057	48	15.14
Last round	Fixed 1	0.244	0.8456	1	0.0071	0.139	0.054	8	15.11
	Fixed 3	0.235	0.8549	2.985	0.0129	0.129	0.052	8	15.04
	Fixed 10	0.247	0.8439	9.969	0.0025	0.124	0.057	8	15.00
	Fixed 1000	0.3276	0.7622	897.434	0.0031	0.109	0.015	8	14.99
	Dynamic (Max 100)	0.113	0.9951	99.334	0.0011	0.129	0.055	96	14.66
	Dynamic (Max 1000)	0.1019	0.9003	98.077	0.0009	0.110	0.046	48	15.08

6. Conclusions and Future Work

In this paper, The Alpha-FedAvg is presented, an innovative federated learning algorithm designed specifically for decentralized environments on edge devices. One of the algorithm's unique features is the adaptive learning rate mechanism, which uses the Lipschitz and Smoothness parameters to dynamically modify the learning rate for each node. In a series of extensive trials, Alpha-FedAvg shows higher convergence and performance through the use of federated averaging, in addition to achieving effective model aggregation. These experiments span a wide range of topics, like simulate various network assaults to actual scenarios employing ESP32 microcontrollers coupled to sensors like sound detectors. The program is notable for its ability to protect privacy; it can detect and neutralize network threats without exposing user information. Analysis through comparison with current federated learning techniques highlights the superiority of the algorithm, demonstrating its superiority in terms of communication effectiveness, loss function optimization, and model accuracy under different conditions. The experiment's outcomes demonstrate how useful the Alpha-FedAvg algorithm is in edge IoT contexts, especially when it comes to protecting user privacy and identifying network threats. Nevertheless, there are certain restrictions that could affect generalizability, such as the exclusive focus on the Alpha-FedAvg algorithm and the extent of simulated assault scenarios. The reliability and scalability of the approach across a variety of edge device environments and network situations require more investigation.

In the future, tackling important issues like robustness, and user-friendly implementations could lead to additional breakthroughs in Alpha-FedAvg. Scalability issues will be essential for supporting varied edge device ecosystems and greater datasets as the algorithm develops. The efficacy of the method in dynamic and heterogeneous situations will be guaranteed by robustness testing conducted in real-world deployments. Furthermore, user-friendly solutions will be essential to promoting broad adoption and opening up Alpha-FedAvg to a larger range of developers and practitioners. Future research can help ensure that the algorithm remains relevant and effective in traversing the ever-changing technical landscapes and satisfying the demands of a wide range of application domains by addressing these issues.

References

[1] T. Sun, D. Li, and B. Wang, "Decentralized federated averaging," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 4289–4301, 2022.

[2] Y. Qin, H. Matsutani, and M. Kondo, "A selective model aggregation approach in federated learning for online anomaly detection," in *2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*. Rhodes, Greece: IEEE, 2020, pp. 684–691.

[3] X. Lu, Y. Liao, P. Lio, and P. Hui, "Privacy-preserving asynchronous federated learning mechanism for edge network computing," *IEEE Access*, vol. 8, pp. 48 970–48 981, 2020.

[4] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019-2019 IEEE international conference on communications (ICC)*. Rhodes, Greece: IEEE, 2019, pp. 1–7.

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

[6] C. Xie, O. Koyejo, and I. Gupta, "Asynchronous federated optimization," in *12th Annual Work shop on Optimization for Machine Learning*. NeurIPS, 2020, pp. 1–11.

[7] R. Fantacci and B. Picano, "Federated learning framework for mobile edge computing networks," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 1, pp. 15–21, 2020.

[8] T. Chen, G. Giannakis, T. Sun, and W. Yin, "Lag: Lazily aggregated gradient for communication-efficient distributed learning," in *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, vol. 31. Montréal, Canada: NeurIPS, 2018, pp. 1–11.

[9] S. Li, Y. Cheng, Y. Liu, W. Wang, and T. Chen, "Abnormal client behavior detection in federated learning," in *Proceedings of the 2nd International Workshop on Federated Learning for Data Privacy and Confidentiality (FL-NeurIPS 19)*, vol. 31. Vancouver, BC, Canada: NeurIPS, 2019, pp. 1–11.

[10] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, "Federated learning with personalization layers," *arXiv preprint arXiv:1912.00818*, 2019.

[11] Y. Ye, S. Li, F. Liu, Y. Tang, and W. Hu, "Edgefed: Optimized federated learning based on edge computing," *IEEE Access*, vol. 8, pp. 209 191–209 198, 2020.

[12] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu, "Loadaboost: Loss-based adaboost federated machine learning with reduced computational complexity on iid and non-iid intensive care data," *Plos one*, vol. 15, no. 4, p. e0230706, 2020.

[13] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[14] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE journal on selected areas in communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[15] V.-A. Stefanidis, Y. Verginadis, and G. Mentzas, "Multicloudfl: Adaptive federated learning for improving forecasting accuracy in multi-cloud environments," *Information*, vol. 14, no. 12, p. 662, 2023.

[16] T. Zhang, L. Gao, C. He, M. Zhang, B. Krishnamachari, and A. S. Avestimehr, "Federated learning for the internet of things: Applications, challenges, and opportunities," *IEEE Internet of Things Magazine*, vol. 5, no. 1, pp. 24–29, 2022.

[17] L. Gao, H. Fu, L. Li, Y. Chen, M. Xu, and C.-Z. Xu, "Feddc: Federated learning with non-IID data via local drift decoupling and correction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 112–10 121.

[18] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *ICLR 2020 Eighth International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020, p. Paper 261.

[19] M. Garin, T. Evgeniou, and N. Vayatis, "Weighting schemes for one-shot federated learning," *SSRN*, 2022.

[20] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating federated learning via momentum gradient descent," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 8, pp. 1754–1766, 2020.

Karam Muhammed Mahdi Salih Karam M. Salih Assistant lecturer in Computer Network Dept., Collage of Information technology, Ninevah university, Phd student in University of Mosul.

Najla Badie Ibraheem Assistant prof. in Department of Computer Science, Computer Science and Mathematics College, University of Mosul, Mosul, Iraq.