FogFL: Fog-Assisted Federated Learning for Resource-Constrained IoT Devices

Rituparna Saha, Sudip Misra, *Senior Member, IEEE*, and Pallav Kumar Deb, *Graduate Student Member, IEEE*

***Abstract*—In this article, we propose a fog-enabled federated learning framework—FogFL—to facilitate distributed learning for delay-sensitive applications in resource-constrained IoT environments. While federated learning (FL) is a popular distributed learning approach, it suffers from communication overheads and high computational requirements. Moreover, global aggregation in FL relies on a centralized server, prone to malicious attacks, resulting in inefficient training models. We address these issues by introducing geospatially placed fog nodes into the FL framework as local aggregators. These fog nodes are responsible for defined demographics, which help share location-based information for applications with similar environments. Furthermore, we formulate a greedy heuristic approach for selecting an optimal fog node for assuming a global aggregator's role at each round of communication between the edge and cloud, thereby reducing the dependence on the execution at the centralized server. Fog nodes in the FogFL framework reduce communication latency and energy consumption of resource-constrained edge devices without affecting the global model's convergence rate, thereby increasing the system's reliability. Extensive deployment and experimental results corroborate that, in addition to a decrease in global aggregation rounds, FogFL reduces energy consumption and communication latency by 92% and 85%, respectively, as compared to state of the art.**

***Index Terms*—Communication latency, energy consumption, federated learning (FL), fog computing, Internet of Things.**

I. Introduction

THE DISTRIBUTED learning approach, federated learning (FL), enables IoT devices to learn an artificial intelligence (AI) model collaboratively without exposing local data. As this learning approach "brings the code to the data, instead of the data to the code" [1], FL enhances data privacy. However, conventional FL techniques suffer from the following challenges: 1) *communication overhead* due to the continuous exchange of model data among the plethora of edge devices and the cloud [1]. Poor network connectivity further increases this challenge by inducing higher latencies; 2) *sophisticated computation routines* necessary for training mandate the need for devices with high-performance configurations [2]. However, the edge devices in the IoT system are usually resource-constrained concerning computation power, storage, and energy; and 3) primary dependency on a *centralized entity* in each epoch. Such centralized entities are vulnerable to issues related to bottlenecks and single point of failure, which results in inefficient global model [3]. Due to these challenges, the FogFL model's deployment in the resource-constrained IoT environment mandates the need for a distributed architecture and strategic global aggregation routines.

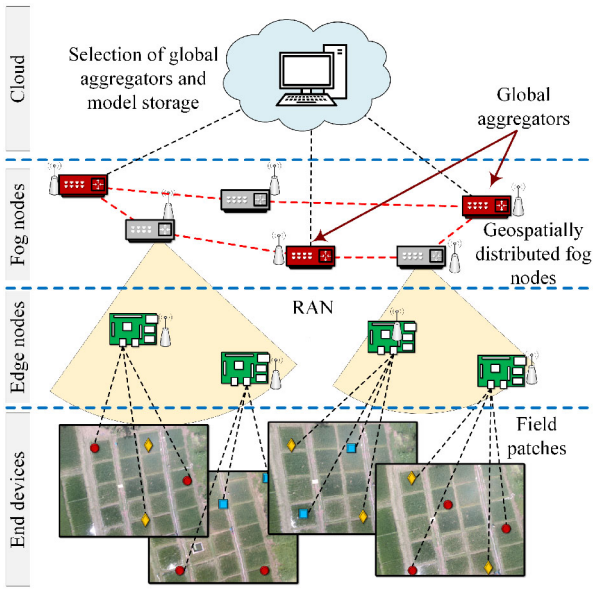

Fig. 1. Overview of the proposed FogFL framework.

In this work, we propose and implement a fog computing-based distributed FL framework—FogFL—for resource-constrained IoT environments. As shown in Fig. 1, we consider the geospatial placement of the resource-rich fog nodes for partially aggregating the locally updated models from the edge nodes present within its vicinity. We assume that the devices use radio access network (RAN) technologies such as WiFi for establishing communication among one another. After a certain number of communication rounds and local aggregations among the edge and fog nodes, the cloud selects an optimal fog node for global aggregation. The introduction of fog nodes in the device-to-cloud continuum facilitates the FL framework by reducing the possibility of bottlenecks and global aggregations at each round. Such reduced collection results in a

Manuscript received July 26, 2020; revised October 17, 2020 and November 30, 2020; accepted December 10, 2020. Date of publication December 22, 2020; date of current version May 7, 2021. *(Corresponding author: Sudip Misra.)*

Rituparna Saha is with the Advanced Technology Development Centre, Indian Institute of Technology Kharagpur, Kharagpur 721302, India (e-mail: rituparnasaha@iitkgp.ac.in).

Sudip Misra and Pallav Kumar Deb are with the Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur 721302, India (e-mail: sudipm@iitkgp.ac.in; pallv.deb@iitkgp.ac.in).

Digital Object Identifier 10.1109/JIOT.2020.3046509

2327-4662 © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA. Downloaded on July 29,2024 at 06:50:52 UTC from IEEE Xplore. Restrictions apply.

decrease in communication latency and energy consumption of the edge devices operating in IoT environments. Furthermore, it increases the system's reliability by reducing the dependency on a centralized entity at each epoch. It also ensures location-aware training of models, explicitly meant for only a subset of edge devices.

Example Scenario: For illustration, consider an irrigation scheduling application in agricultural fields as shown in Fig. 1. Each edge node collects data from the deployed sensors (soil moisture, temperature, humidity, and others) from a particular field. These fields vary in demographics and topology, and each edge device updates its local model using the on-device local data. The edge devices forward these local models to the fog nodes within its vicinity (geospatially), which perform local aggregations. After a certain number of local aggregations, the cloud server selects an optimal fog node that interacts with other fog nodes to update the model globally. This process continues until an efficient irrigation scheduling model is learned globally in that area.

A. Motivation

FL entirely depends on a centralized entity for selecting participants, device configuration, and evaluation of the global aggregation at each epoch [1], which opens the scope for bottlenecks and issues about the single point of failure. Such conditions often lead to distortion of the global model [3]. Moreover, FL uses additional computation in modern smartphones with high-performance configurations to reduce communication rounds during model training, which is infeasible for resource-constrained edge devices [2]. On the other hand, since FL offers multiple advantages over the traditional distributed learning approaches, such as the use of nonindependent and identically distributed (non-IID) training data, minimum data transmission from clients, and privacy protection [2], it acts as a motivation for our work. Fog computing [11], [12] at the edge network is an optimal solution in the IoT system to alleviate the problems mentioned above. Therefore, we design FogFL intending to reduce the frequency of global aggregations, communication cost, and energy consumption of resource-constrained edge nodes. Additionally, while most of the current research focuses on the convergence time of FL instead of reliable global aggregation, FogFL increases the FL framework's reliability without compromising the model's accuracy.

B. Difference of FogFL From Existing Solutions

In the edge computing-based IoT environment, FL requires multiple edge nodes as participants that download the global model and update it based on local data, at each iteration. The edge nodes then upload the locally updated model to the cloud, responsible for the global aggregation of the locally updated models from each client edge node [2]. Next, the globally updated model is stored in the cloud and send to all the edge nodes, which is then used as the starting point by each client for the next round and this process is repeated until the global model reaches the target accuracy. Existing research works implement the FL framework by placing the central server either at the cloud or edge. Cloud-based FL framework accesses more clients but faces high communication overhead and latency while the edge-based FL framework faces efficient communication with less number of clients [9].

The proposed FogFL framework differs from the cloud-based and edge-based FL framework primarily by employing the fog nodes into the FL framework as local aggregators and a variable global aggregator node at each round. Moreover, in FogFL, each global aggregation is executed after each ε number of local aggregations of total N communication rounds, instead of executing at each round that helps to reduce the communication cost in the proposed framework than the traditional methods. Furthermore, FogFL differs from the client-edge-cloud hierarchical architecture by building a two-level architecture without making any centralized global aggregator node. Additionally, at each N/ε global aggregation round, a fog node is selected as a global aggregator node based on the minimum workload and latency than the other fog nodes, which makes the system more reliable than the state of the art.

C. Contributions

We design and implement a fog computing-based FL framework—FogFL—for resource-constrained edge devices. The FogFL algorithm trains a model in a distributed and decentralized fashion without depending on the cloud server to complete an epoch. We exploit the fog computing paradigm for computation, storage, and network resources for serving on-demand IoT applications. Toward this, the primary contributions of this work are as follows.

1) *FogFL Framework:* We design and implement the FogFL framework to reduce communication overheads while facilitating a decentralized and distributed FL framework across the resource-constrained edge devices. To reduce the possibility of issues due to the single point of failure, we design FogFL without depending on a centralized global aggregator.
2) *Variable Global Aggregator Node:* We formulate and adopt a greedy heuristic strategy to select an optimal global aggregator fog node at the end of an epoch to increase the reliability of the system.
3) *Evaluation:* Through extensive evaluation of the implemented FogFL system as well as real-world emulation, we present a discussion based on the test accuracy, communication latency, and energy consumption of the edge devices. We also present a comparison of FogFL with existing state-of-the-art FL frameworks.

II. Related Works

As mentioned in Section I-B, FL is widely categorized as: 1) cloud-based FL and 2) edge-based FL. In cloud-based FL, research works (e.g., [4] and [5]) proposed algorithms to reduce communication cost using data compression technique and proposed a resource-aware FL framework, respectively. However, the cloud-based FL framework suffers from a communication overhead, which reinforce edge-based FL frameworks using limited number of clients.

Authorized licensed use limited to: NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA. Downloaded on July 29,2024 at 06:50:52 UTC from IEEE Xplore. Restrictions apply.

TABLE I
COMPARISON OF FOGFL WITH EXISTING STATE OF THE ART

Research Work	Architecture	Communication Overhead	Energy Consumption	Global Aggregator
McMahan *et al.* [2], Konečnỳ *et al.* [4]	Cloud-based	Yes	No	Fixed
Xu *et al.* [5]	Cloud-based	Yes	Yes	Fixed
Mills *et al.* [6], Wang *et al.* [7]	Edge-based	Yes	No	Fixed
Tran *et al.* [8]	Edge-based	Yes	Yes	Fixed
Kim *et al.* [3]	Edge-based	Yes	No	Not Fixed
Liu *et al.* [9], Abad *et al.* [10]	Cloud-Edge-based	Yes	No	Fixed
FogFL (Proposed)	Fog-based	Yes	Yes	Not Fixed

In edge-based FL, Wang *et al.* [7] theoretically analyzed the convergence rate of distributed gradient descent, and determined the global aggregation frequency to reduce resource consumption. Wang *et al.* [13] integrated deep reinforcement learning and FL techniques with mobile-edge computing (MEC) and proposed an edge-AI framework to optimize the MEC, caching, and communication. In the same context, Mills *et al.* [6] adopted Adam optimization-based FedAvg, and novel data compression to reduce the convergence rounds. These researches mainly focused on minimizing convergence time without considering the uncertainty of the wireless network, energy limitation of the end nodes, and local data sizes. To address these lacunae, Tran *et al.* [8] evaluated FL over wireless networks considering these limitations. Kim *et al.* [3] proposed a blockchain-based FL approach, where the local updates are accumulated in the blockchain. On the other hand, Liu *et al.* [9] and Abad *et al.* [10] proposed a client-edge-cloud hierarchical FL system to reduce communication cost than traditional FL.

Synthesis: From the above discussions, we notice that most of the existing literature focuses on designing novel FL frameworks with probable convergence time, using a centralized aggregator node, which creates a lacuna in securing the aggregator's vulnerability of issues due to the single point of failure. Furthermore, there are limited research works on implementing FL for applications using resource-constrained IoT devices. We propose FogFL to address these issues, while reducing communication latency and energy consumption, without considering the centralized entity for global aggregation. In Table I, we summarize the fundamental difference of FogFL with the existing works.

III. SYSTEM ARCHITECTURE

In this section, we describe the FogFL framework and formulate a greedy heuristic approach for selecting the global aggregator fog node at each round.

A. System Model

We consider K edge nodes as clients, F fog nodes as the regional server and responsible for both local aggregation and global aggregation, and a cloud server as the coordinator node. The edge nodes of a particular region communicate with the regional fog node using a wireless channel interface, and the fog nodes communicate with each other via a dedicated wireless channel, where the number of edge nodes is greater than the number of fog nodes ($K > F$). The set of geographically closed edge nodes $\mathcal{K}$ map to the set of fog nodes $\mathcal{F}$ with many-to-one mapping. All the important key notations used to describe the system are listed in Table II. Each round of the FogFL framework comprises six phases: 1) selection; 2) configuration; 3) local update; 4) local aggregation; 5) global aggregation; and 6) reporting. At the beginning of each round, the cloud node selects and configures a fraction of $C \in (0, 1)$ clients. The selected clients $M = \lceil C \cdot K \rceil$ locally update the model parameter w. We assume the local data set $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K$ across K edge nodes follow the non-IID and unbalanced property. A training data sample $a \in \mathcal{D}_k$ for client $k \in \mathcal{K}$ consists of two parts (X_a, Y_a), where X_a denotes the input vector and Y_a denotes the corresponding desired output. The loss function across client k for each data sample $a \in \mathcal{D}_k$ is represented as $l_a(X_a, Y_a, w)$ or $l_a(w)$. The local loss function across each client k is formulated as

$$L_k(w) = \frac{1}{n_k} \sum_{a \in D_k} l_a(w) \tag{1}$$

where $n_k = |D_k|$ and local accuracy $0 \leq \theta \leq 1$, which is upper bounded by $O(\log(1/\theta))$ [8]. Each client k updates the

TABLE II
KEY NOTATIONS

Notations	Description
K	Total number of edge devices or clients
F	Total number of fog nodes
C	Fraction of clients selected in each round, $C \in (0, 1)$
M	Selected clients at each round i.e. $M = \lceil C \cdot K \rceil$
D_k	Data sample size at client or edge node k
n_k	Number of data samples i.e. $n_k = \lvert D_k \rvert$
s	Model size across all clients
u_k	Number of local updates at edge node k
τ_1	Datarate across edge nodes
τ_2	Datarate across fog nodes
q_f	Total number of clients associated with a fog node
z_1	CPU cycles for training on data sample D_k
z_2	CPU cycles for local aggregation of local model
ρ_k^e	CPU cycle frequency at k^{th} edge node
ρ_j^f	CPU cycle frequency at j^{th} fog node
d^f	Distance between two fog node
d^e	Distance between an edge and its associated fog node
$\delta'_{i,j}$	Edge to fog communication delay for i^{th} global and j^{th} local aggregation round
λ_i	Fog to fog communication delay across i^{th} fog node
Ω_i	Workload at i^{th} fog node
G	Global aggregator node
N	Total number of communication rounds for training
ε	Number of local aggregation rounds
$\mathcal{T}$	Deadline time to perform download, update and upload for a client
$\mathcal{V}$	Total number of successfully attempted clients

Authorized licensed use limited to: NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA. Downloaded on July 29,2024 at 06:50:52 UTC from IEEE Xplore. Restrictions apply.

local parameters w_k as follows:

$$w_k(t) = w_k(t) - \eta \nabla L_k(w_k(t, B_k)) \tag{2}$$

where $t = 0, 1, 2, \ldots, N$ denotes the iterations, B_k denotes the minibatch size, and $\eta \geq 0$ is the learning rate. Unlike FL [2], we consider the number of local updates u_k as follows:

$$u_k = \frac{E_k n_k}{B_k} \frac{e_k m_k P_k}{J_k} \tag{3}$$

where device conditions, such as residual energy e_k, available processing memory m_k, available CPU capacity P_k, and number of jobs running on device J_k, are considered with the number of local examples n_k, training passes E_k, and minibatch size B_k across the kth client. After u_k local updates, all the client nodes under each fog node $f \in \mathcal{F}$ send the updated parameters to the adjacent fog node f to evaluate the local aggregation. The local aggregation parameter w_f over tth epoch is calculated as follows:

$$w_f(t) = \frac{\sum_{k=1}^{q_f} n_k w_k(t)}{n_f}. \tag{4}$$

The number of clients managed by fog node f is denoted by $q_f = \sum_{k=1}^{K} x_{f,k}$, where $x_{f,k} \in \{0, 1\}$ and $n_f = \sum_{k=1}^{q_f} |D_k|$. After ε rounds of selection, configuration, local update, and local aggregation operations, the cloud server selects a fog node as a global aggregator $G \in \mathcal{F}$ among all the fog nodes based on two parameters latency and workload to perform the global aggregation. The global aggregation parameter $\tilde{w}(t)$ is evaluated by taking the average of local aggregation parameter $w_f(t)$ from each fog f which is defined as follows:

$$\tilde{w}(t) = \frac{\sum_{f=1}^{F} w_f(t)}{F}. \tag{5}$$

In the next step, the global aggregator node G circulates the final model among all fog nodes and stores the final model in a cloud server for the next round of communication. FogFL seeks to optimize the global parameters and repeats this process until the global model reaches the target accuracy of $0 \leq \Phi \leq 1$. The steps of the FogFL framework are presented in Algorithm 1. We consider u_k local updates and ε local aggregations before performing one global aggregation. Therefore, FogFL executes total N/ε global aggregations for total $N/\varepsilon \times \varepsilon \simeq N$ iterations.

Algorithm 1 FogFL Algorithm

INPUTS: $\mathcal{K}, C, \mathcal{F}, \varepsilon, N$
OUTPUT: $\tilde{w}(t)$
PROCEDURE:

```
for each round t = 1, 2, ..., N do
    Select C fraction of clients where C ∈ (0, 1)
    Initialize w_k(t) at each selected client k ∈ K
    for each fog f ∈ F in parallel do
        for each client k ∈ K in parallel do
            Calculate local updates w_k(t), u_k times using (2)
        end for
        Calculate local aggregation w_f(t) using (4)
    end for
    if t is an integer multiple of ε then
        Select global aggregator node G
        Calculate global parameter w̃(t) using (5)
    end if
end for
```

B. Global Aggregator Node Selection

In each round, the cloud server selects a global aggregator node $G \in \mathcal{F}$ based on the two parameters workload and communication latency among all the fog nodes.

1) Workload Parameter: For each fog node $i \in \mathcal{F}$, workload $\Omega_i(t)$ denotes the data processing request in bits coming from the associated clients at the tth epoch, which is formulated as follows:

$$\Omega_i(t) = \sum_{k=1}^{K} x_{i,k} s \quad \forall i \in \mathcal{F} \tag{6}$$

where $x_{i,k} \in \{0, 1\}$ is a binary variable, denotes the association of the kth client with the ith fog node and s denotes the locally updated model size, which is assumed to be same across all clients.

2) Communication Latency Parameter: Each one-hop link between the ith and jth fog nodes has the transmission delay $\lambda_{i,j}^{Tr}(t) = (s/\tau_2)$ and propagation delay $\lambda_{i,j}^{Pr}(t) = [(d_{i,j}^{f} s)/(c)]$, $\forall i, j \in \mathcal{F}, i \neq j$, respectively, where we consider the model size s and datarate τ_2 are same at all fog nodes, $d_{i,j}^{f}$ denotes the distance between the ith and jth fog nodes, and c denotes the speed of light. The processing delay at fog node $i \in \mathcal{F}$ is $\lambda_i^C(t) = \sum_{k=1}^{K} x_{i,k}(z_2/\rho_i^f)$, where z_2 denotes the number of CPU cycles required for processing model size s and ρ_i^f denotes the CPU frequency of the ith fog node. Inspired by the work of Misra and Saha [14], we consider the task arrivals at a fog node $i \in \mathcal{F}$ follow the Poisson process, and is defined as $\gamma_i = \sum_{k=1}^{K} x_{i,k}$. We assume that each fog node $i \in \mathcal{F}$ follows M/M/1 queuing model, and calculate queuing delay as $\lambda_i^{que}(t) = [(1)/(\mu_i - \gamma_i)]$, where μ_i denotes the service rate of the ith fog node. Therefore, the total delay across the ith fog node at tth epoch is defined as follows:

$$\lambda_i(t) = \lambda_i^D(t) + \lambda_i^C(t) + \lambda_i^{que}(t) \tag{7}$$

where $\lambda_i^D(t) = max(\lambda_{i,j}^{Tr}(t) + \lambda_{i,j}^{Pr}(t))\ \forall i, j \in \mathcal{F}, i \neq j$. In each round, the selection of the global aggregator node $G \in \mathcal{F}$ is determined by selecting a fog node $i \in \mathcal{F}$ having minimum workload $\Omega_i(t)$ and delay $\lambda_i(t)$. So, the utility function $\mathcal{U}_i$ of the ith fog node is defined as follows:

$$\mathcal{U}_i = \alpha \lambda_i(t) + (1 - \alpha)\Omega_i(t) \quad \forall i \in \mathcal{F} \tag{8}$$

where constant α is used to control the relative importance of the workload and delay, respectively. We formulate an optimization problem to find a fog node as the global aggregator node, which aims to minimize the utility function $\mathcal{U}_i$, $i \in F$ subject to minimum latency and workload constraints. Mathematically, we formulate our problem (**P1**) as follows:

$$\arg\min_{i \in \mathcal{F}} \ \mathcal{U}_i \tag{9a}$$

$$\text{s.t.} \ \ \lambda_i(t) \leq \lambda^{\max} \quad \forall i \in \mathcal{F} \tag{9b}$$

$$\Omega_i(t) \leq \Omega^{\max} \quad \forall i \in \mathcal{F} \tag{9c}$$

Authorized licensed use limited to: NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA. Downloaded on July 29,2024 at 06:50:52 UTC from IEEE Xplore. Restrictions apply.

Algorithm 2 Global Aggregator Node Selection

INPUTS: $\mathcal{F}, \Omega^{\max}, \lambda^{\max}$
OUTPUT: G
PROCEDURE:

```
for each fog i ∈ F do
    Calculate workload Ω_i(t) using (6)
    for each fog j ∈ F and i ≠ j do
        Calculate latency λ_i(t) using (7)
    end for
    if Ω_i(t) ≤ Ω^max and λ_i(t) ≤ λ^max then
        Calculate U_i using (8)
    end if
end for
G = arg min_{i∈F} U_i
```

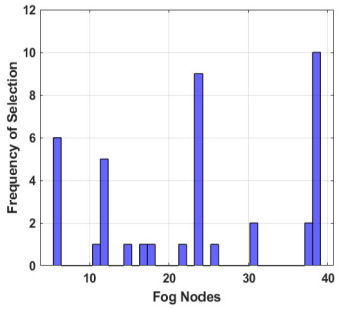

Fig. 2. Histogram of global aggregator node selection.

where (9a) ensures that the maximum delay at a fog node in the fog network should not exceed a threshold $\lambda^{\max}$. Equation (9b) ensures that the maximum workload allowed at a fog node does not exceed the threshold $\Omega^{\max}$. In this article, we solve the optimization problem in (9) using a greedy heuristic approach, as shown in Algorithm 2. The result in Fig. 2 manifests the frequency of selection of different fog node as a global aggregator node at each round. It ensures that the proposed greedy heuristic approach brings dynamicity in the selection process by selecting different fog nodes with minimum workload and latency at each round. However, in the worst case, if no fog nodes meet the workload and latency constraints, the cloud server will take the responsibility to do the global aggregation. The worst case complexity of the Algorithm 2 is $O(F(F-1)) \simeq O(F^2)$.

Lemma 1: Problem **P1** is a convex optimization problem.

Proof: We convert the multivariate function $\mathcal{U}_i(\lambda_i(t), \Omega_i(t)) = \alpha\lambda_i(t) + (1-\alpha)\Omega_i(t)$ to univariate and represent it as $g(\alpha) = \mathcal{U}_i(\lambda_i(t), \Omega_i(t)) = \alpha\lambda_i(t) + (1-\alpha)\Omega_i(t)$. For any two variables $x, y \in \mathbb{R}$, $g(\psi x + (1-\psi)y) = (\psi x + (1-\psi)y)\lambda_i + (1-(\psi x + (1-\psi)y))\Omega_i = \psi g(x) + (1-\psi)g(y)$. Since $g(\psi x + (1-\psi)y) = \psi g(x) + (1-\psi)g(y)$, $g(\alpha)$ is an affine function. Considering the bounds of its convexity, we conclude that $\mathcal{U}_i(\lambda_i(t), \Omega_i(t))$ is a convex function. ■

IV. System Analysis

In this section, we analyze the proposed FogFL framework based on: 1) delay; 2) energy consumption of the resource-constrained edge nodes; and 3) reliability.

A. Delay Model

The FL time is a function of both the computation and communication time [8]. The computation time depends on data size and execution time during local processing. On the other hand, the communication time depends on the network channel bandwidth and distance between the participating nodes. Therefore, the computation delay of the kth edge node is evaluated as $\delta_k^C = (z_1/\rho_k^e)$, where z_1 and ρ_k^e denote the CPU cycles for data D_k and CPU cycle-frequency of the kth edge node, respectively. We assume that the model size s and the datarate τ_1 are same for all the edge nodes. Therefore, the uplink transmission delay $\delta_k^T = (s/\tau_1)$ is constant throughout the system. The uplink propagation delay of the kth edge node is calculated as $\delta_k^P = [(d_{k,f}^e s)/(c)]$, where $d_{k,f}^e$ denotes the distance between the kth edge node and its corresponding fog node $f \in \mathcal{F}$, and c is the constant denoting the speed of light. Moreover, the transmission and propagation delays of downlink communication can be evaluated using the same approach. However, as per the research of Tran *et al.* [8], the downlink transmission time is negligible compared to the uplink transmission time (uploading time) due to the high downlink bandwidth and transmission power of the high-performance configuration fog nodes than the resource-constrained edge nodes. We assume that FogFL executes ε number of local aggregations before performing one global aggregation to globally train the model for total N number of communication rounds. Thus, at the ith global and jth local aggregation round, the total delay at kth edge node is $\delta_{i,j,k} = (\delta_k^P + \delta_k^T + \delta_k^C)$ and total edge to fog communication delay is computed as $\delta'_{i,j} = \max_{k\in\mathcal{K}}(x_{i,j,k}\delta_{i,j,k})$, where $x_{i,j,k} \in \{0,1\}$ denotes the selection of the edge node $k \in \mathcal{K}$ at the ith global aggregation and the jth local aggregation round.

For each round, the average delay of the proposed system is defined as follows:

$$\delta'' = \frac{1}{N}\sum_{i=1}^{N/\varepsilon}\sum_{j=1}^{\varepsilon}\delta'_{i,j}. \tag{10}$$

B. Energy Consumption Model

The resource-constrained edge nodes mainly affect the evaluation of the energy consumption. Following the work of Burd and Brodersen [15], the total energy consumption across K edge nodes at the ith global aggregation round and the jth local aggregation round is evaluated as follows: $E_{i,j}^{comp} = \sum_{k=1}^{K}((\beta_k/2)z_1\rho_k^2 n_k x_{i,j,k})$, where β_k, z_1, ρ_k^e, n_k, and $x_{i,j,k}$ denote the effective capacitance coefficient, CPU cycles for data sample D_k, CPU cycle frequency of the kth edge node, number of data samples, and binary variable, respectively. On the other hand, we adopt the energy model of Heinzelman *et al.* [16] to calculate the total communication energy consumption of the K edge nodes at the ith global and jth local aggregation round. Mathematically, $E_{i,j}^{comm} = \sum_{k=1}^{K} x_{i,j,k}(E_{i,j,k}^{elec}s + E_{i,j,k}^{amp}sd_{k,f}^e)$, where $E_{i,j,k}^{elec}$ and $E_{i,j,k}^{amp}$ denote the transmission and amplification energies of the kth edge node, respectively. s and $d_{k,f}^e$ denote the model size and distance of the kth edge node from the corresponding fog node $f \in \mathcal{F}$, respectively. Thus, the average energy consumption E

Authorized licensed use limited to: NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA. Downloaded on July 29,2024 at 06:50:52 UTC from IEEE Xplore. Restrictions apply.

of the proposed system is as follows:

$$E = \frac{1}{N}\sum_{i=1}^{N/\varepsilon}\sum_{j=1}^{\varepsilon}(E_{i,j}^{comp} + E_{i,j}^{comm}). \tag{11}$$

C. Reliability Model

We analyze the reliability of the system by evaluating the number of clients who successfully attempt the training process (local update and upload) within the deadline time. Let $V_{i,j}$ denotes the number of successfully attempted clients at each ith global aggregation round and jth local aggregation round, which is evaluated as

$$V_{i,j} = \sum_{k\in\mathcal{K}} \mathbf{f}(x_{i,j,k}\delta_{i,j,k} - \mathcal{T}) \tag{12}$$

where $\mathbf{f}(.)$ is a unit step function, $x_{i,j,k}$ is a binary variable, $\delta_{i,j,k}$ denotes the communication delay across the kth client, and $\mathcal{T}$ denotes the deadline time (in seconds). We set $\mathbf{f}(\delta_{i,j,k}-\mathcal{T}) = 1$ when the kth client successfully executes the operations within deadline $\mathcal{T}$, i.e., $(\delta_{i,j,k} - \mathcal{T}) > 0$, and 0 otherwise. Following the work of Yao and Ansari [17], we evaluate the ratio of number of successfully attempted clients to the total number of participants in the full training operation i.e., $\mathcal{V}/\mathcal{M}$, where $\mathcal{V} = \sum_{i=1}^{N/\varepsilon}\sum_{j=1}^{\varepsilon} V_{i,j}$ denotes the total number of successfully attempted clients and $\mathcal{M} = \sum_{i=1}^{N/\varepsilon}\sum_{j=1}^{\varepsilon} M_{i,j}$ denotes the total number of participant, where $M_{i,j}$ is the number of participants at the ith global and jth local aggregation round.

V. Performance Evaluation

We evaluate the performance of FogFL using both hardware prototype and simulation environments. In this section, we describe each of these setups along with the reference model and data set. We also describe the benchmark solution against, which we compare the proposed FogFL framework.

A. Benchmark Solutions

To evaluate the FogFL framework's performance, we compare it with the FedAvg algorithm [2] and hierarchical FL framework [10]. Through the FedAvg algorithm, the authors proposed a decentralized training approach for FL. They used a central server to perform aggregation at each round after collecting the local updates from the clients. On the other hand, compare with another hierarchical client-fog-cloud-based FL framework (HFL), following the work of [10]. To make compatible with the proposed framework for comparison, we implement HFL by employing the fog nodes as small cell base stations (SBS) and centralized server as macrocell base station (MBS). In the subsequent section, we present our observations concerning delay, energy, and test accuracy.

B. Prototype and Simulation Setup

In prototype setting, we deploy six Raspberry Pi devices with 1.5-GHz clock cycle as resource-constrained edge nodes, two personal computers—one with Intel i5 processor 2.5-GHz clock cycle, 8-GB RAM and the other with Intel i3 processor 2.0 GHz, 4-GB RAM as fog nodes. These devices communicate using RAN technologies such as WiFi. We also consider a system with Intel i5 processor 3.2 GHz, 8-GB RAM, for assuming the role of a cloud server. The prototype follows at most 3 Raspberry device connection under each fog node, connected with the cloud. For FedAvg, HFL, and FogFL, we consider the same configurations for the edge nodes and servers. However, we exclude the layer of fog nodes in the case of FedAvg. We perform the FL operation at most ten times for each of the prototype setting and collect the corresponding results regarding energy consumption and processing time of edge nodes during training and incorporate the average results in the simulation setup.

We mimic the features of the prototype setting and design a simulation environment for a large number of working nodes by varying the client fractions, fog nodes, and a number of local aggregations using MATLAB R2018a. In all three environments, we randomly distribute the working nodes within a 5 km $\times$ 5 km coverage area. For all environments, we design the experiment with the total number of edge nodes K as 1000 and varying client fractions C as 10%, 20%, and 30%. Moreover, in the FogFL and HFL frameworks, we experiment with varying number of fog nodes ν as 0.02 and 0.06 fraction of total edge nodes, under each client fractions. Moreover, for simplification, we consider the same number of local updates u_k for each edge device k.

C. Reference Model and Data set

In all three frameworks, we couple the edge nodes with local data sets to perform on-device training. We conduct our experiment using a multilayer perceptron (MLP) with two hidden layers (2NN), where each layer consists of 400 neurons using the ReLU activation function. We set the hyperparameters of the MLP model, such as learning rate and batch size as 0.001 and 50, respectively. We train and test the MLP model on proxy data set MNIST with 70 000 instances (60 000 for training, 5000 for testing, and 5000 for validation) for the proof of concept. We consider a non-IID partition of the data set across each client following the work of McMahan *et al.* [2]. For FedAvg and HFL, we run the experiment for a total number of communication rounds $N = 200$. On the other hand, in FogFL, we perform a total N/ε rounds of global aggregations, where one global aggregation is performed after every ε rounds of local aggregations. We present a comparative study by varying the local aggregation numbers of ε with 10 and 20.

D. Results and Discussions

Using the prototype and simulation environments described in Section V-B, we present our observations, such as variance in accuracy, delay, and energy consumption on executing FogFL with a varying number of clients.

Impact of Multiclient Parallelism: We analyze the impact of multiclient parallelism with varying numbers of local aggregations ε as 10 and 20 for FedAvg, HFL, and FogFL from the accuracy plots as depicted in Fig. 3(a) and (b), respectively. The figures manifest that the global test accuracy for FogFL and HFL is approximately the same and outperform

Authorized licensed use limited to: NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA. Downloaded on July 29,2024 at 06:50:52 UTC from IEEE Xplore. Restrictions apply.

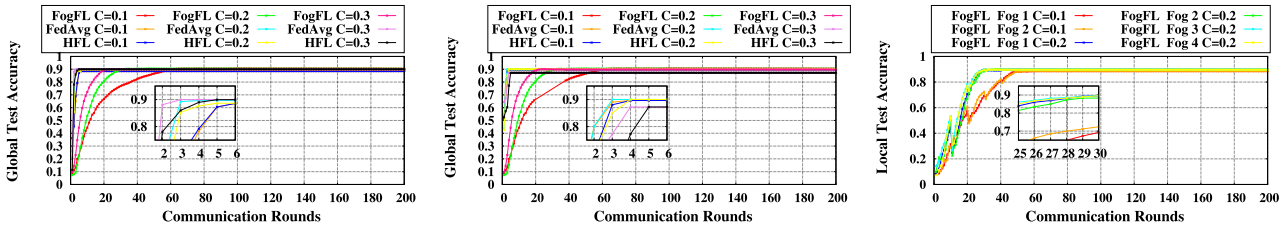

Fig. 3. Test accuracy versus communication rounds. (a) Number of local aggregations $\varepsilon = 10$. (b) Number of local aggregations $\varepsilon = 20$. (c) Number of local aggregations $\varepsilon = 10$.

TABLE III
TOTAL NUMBER OF GLOBAL COMMUNICATION ROUNDS FOR FEDAVG, HFL, AND FOGFL

Parameters	FedAvg	HFL & FogFL	
		$\varepsilon = 10$	$\varepsilon = 20$
C = 0.1	50	5	3
C = 0.2	25	3	2
C = 0.3	17	2	2

FedAvg. From the results, we also calculate the number of communication rounds to reach the target accuracy (0.85) for different client fractions in all three frameworks and listed in Table III. Table III indicates that FogFL demonstrates significant improvement for $C \geq 0.1$ over HFL and converges much faster than the FedAvg with reasonable accuracy in each case of client fractions. It implies that the introduction of intermediate fog nodes in FogFL and HFL distributively offloads the task of aggregations from the central cloud and helps to converge faster with small number of global aggregations.

Reduction of Communication Latency: We analyze the average delay at each round of FedAvg, HFL, and FogFL for varying number of local aggregations and fog nodes, as shown in Figs. 4(a) and 5(a), respectively. The results manifest that FogFL outperforms the benchmarks for all cases of client fractions. It occurs due to the employment of intermediary fog nodes as a global aggregator node into the FogFL framework, which reduces the transmission delay. From Fig. 4(a), we also observe that for both FogFL and HFL with $\varepsilon > 10$ results in high communication latency in all cases of client fractions. It implies that the increase in the number of local aggregations significantly increases the number of communications between the edge and fog nodes. On the other hand, Fig.5(a) describes that both FogFL and HFL reduces communication latency with the increase in the number of fog nodes. Overall, we conclude that the proposed framework FogFL reduces delay by 85% and 68% than both benchmarks FedAvg and HFL, respectively, in all cases of client fractions.

Reduction of Energy Consumption: In the same manner, we also analyze the total energy consumption of the edge devices at each round for all the three frameworks with varying client fractions, as outlined in Figs. 4(b) and 5(b). The results manifest that FogFL outperforms the FedAvg and approximately results the same with HFL, in all cases of client fractions. Moreover, in all three frameworks with the increasing number of clients $C \geq 0.1$, the total energy consumption for each round also increases linearly. It implies that an increase in

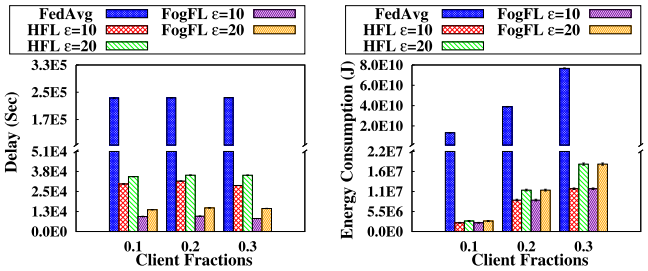

Fig. 4. Impact of different local aggregations. (a) Delay versus client fractions. (b) Energy consumption versus client fractions.

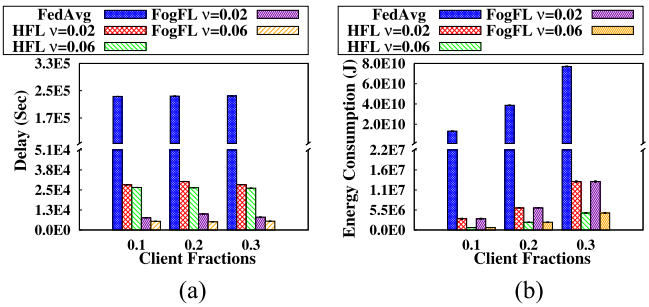

Fig. 5. Impact of different number of fog nodes. (a) Delay versus client fractions. (b) Energy consumption versus client fractions.

the number of clients enriches the number of operations at each round and resulting in high energy consumption. Overall, FogFL reduces the total energy consumption by 92% than FedAvg and performs equally as HFL, in all types of client fractions. Based on the results of delay and energy consumption, we fix the local aggregation ε as 10 for the remaining experiments.

Impact of the Increasing Number of the Fog Nodes: We analyze the average delay and energy consumption result of the FogFL algorithm at each round with fog node variations ν as 0.02 and 0.06 fraction of total edge nodes with local aggregation ε as 10 under various client fractions, as depicted in Fig. 5(a) and (b), respectively. Both figures manifest that with the increasing number of fog nodes under each client fraction, significantly decrease the both metrics. It implies that the rising number of fog nodes per client fractions substantially distributes the load among the fog nodes, which results in less energy consumption and delay per round.

Impact of Performing Fog-Label Global Aggregation: The FogFL framework is equipped with an optimum global aggregator node selection process, which is designed based on the

Authorized licensed use limited to: NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA. Downloaded on July 29,2024 at 06:50:52 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
SUCCESSFULLY ATTEMPTED CLIENTS RATIO AT EACH ROUND DURING TRAINING FOR FEDAVG, HFL, AND FOGFL

Time (Sec)	FedAvg Ratio	HFL & FogFL Ratio
250	0	0.58
1000	0	0.98
22000	0.59	1
23000	0.98	1

heuristic approach. The proposed selection process appoints a fog node with minimum latency and workload than all other fog nodes as global aggregator at each round, as shown in Fig. 2. This particular feature introduces dynamicity in the system. Additionally, Fig. 3(c) shows the test accuracy of different fog nodes for varying client fractions $C = 0.1$ and 0.2 with fog nodes 0.02 fraction of edge nodes. The curve strokes in Fig 3(c) denote the variation in accuracy at each global aggregations after $\varepsilon = 10$ local aggregations in fog nodes. The figure manifests that the test accuracy of different fog nodes under the same client fraction is quite same, due to the distribution of updated global model among all fog nodes after completion of each global aggregation round. This ensures that in FogFL, the damage of any fog node or global aggregator node at a particular round will not hamper the accuracy of the model. Moreover, we calculate the successfully attempted client ratio at each round during training as listed in Table IV. It depicts that FogFL and HFL take less deadline time than FedAvg due to the intermediary local aggregation at fog node. These features conclude that the fog-label global aggregation is much more efficient than the state of the art.

VI. CONCLUSION

In this article, we designed and deployed the FogFL framework for resource-constrained edge devices without considering any fixed central entity as a global aggregator. We achieved this by introducing geospatially placed fog nodes to collect local updates and then perform global aggregations in this layer. Such use of fog nodes also allows slicing and sharing of models based on the area demographics. Furthermore, we formulated a greedy heuristic approach for selecting an optimum fog node as a global aggregator at each round, increasing the system's efficiency. Through extensive experimental and simulation results, we presented a comparative study of FogFL against state-of-the-art solutions. The results show that FogFL reduces delay by 85% and 68% than both FedAvg and HFL, respectively, and reduces energy consumption by 92% than FedAvg. Subsequently, we also analyzed that at different client fractions with local aggregation number, $\varepsilon = 10$ performs better than $\varepsilon > 10$.

We intended to extend our work by addressing the client selection problem and optimizing the training model over the edge devices to learn the model collaboratively. The selection of a large number of clients results in bottleneck across cloud servers due to inconsistent network channel conditions, which necessitates determining an appropriate group of clients for learning a large-scale distributed FL.

REFERENCES

[1] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," 2019. [Online]. Available: arXiv:1902.01046.
[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Stat.*, 2017, pp. 1273–1282.
[3] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Commun. Lett.*, vol. 24, no. 6, pp. 1279–1283, Jun. 2019.
[4] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016. [Online]. Available: arXiv:1610.05492.
[5] Z. Xu, Z. Yang, J. Xiong, J. Yang, and X. Chen, "Elfish: Resource-aware federated learning on heterogeneous edge devices," 2019. [Online]. Available: arXiv:1912.01684.
[6] 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.
[7] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.
[8] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE INFOCOM*, 2019, pp. 1387–1395.
[9] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–6.
[10] M. S. H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning ACROSS heterogeneous cellular networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2020, pp. 8866–8870.
[11] H. Shah-Mansouri and V. W. S. Wong, "Hierarchical fog-cloud computing for IoT systems: A computation offloading game," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3246–3257, Aug. 2018.
[12] T. Wang, J. Zhou, A. Liu, M. Z. A. Bhuiyan, G. Wang, and W. Jia, "Fog-based computing and storage offloading for data synchronization in IoT," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4272–4282, Jun. 2019.
[13] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Netw.*, vol. 33, no. 5, pp. 156–165, Sep./Oct. 2019.
[14] S. Misra and N. Saha, "Detour: Dynamic task offloading in software-defined fog for IoT applications," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1159–1166, May 2019.
[15] T. D. Burd and R. W. Brodersen, "Processor design for portable systems," *J. VLSI Signal Process. Syst. Signal Image Video Technol.*, vol. 13, nos. 2–3, pp. 203–221, 1996.
[16] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. IEEE Conf. Syst. Sci.*, 2000, pp. 1–10.
[17] J. Yao and N. Ansari, "Fog resource provisioning in reliability-aware IoT networks," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8262–8269, Oct. 2019.

Authorized licensed use limited to: NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA. Downloaded on July 29,2024 at 06:50:52 UTC from IEEE Xplore. Restrictions apply.