Exploring Improved Asynchronous Federated Learning with Fresh Information in Contested Environment

Danda B. Rawat
DoD Center of Excellence in AI/ML
Dept of Electrical Engineering & Computer Science
Howard University, Washington, DC 20059, USA
Email: db.rawat@ieee.org

***Abstract*—To tackle weaknesses of traditional synchronous federated learning (FL), asynchronous FL has emerged as a solution. However, asynchronous FL also faces challenges, particularly in identifying optimal clients capable of contributing to a global model with minimal latency and loss with high accuracy, while prioritizing information freshness. This paper explores client selection strategies in asynchronous FL that takes account of computing and communication delays, freshness of the data/information being used in generating local models, and the loss while providing the best global model with high efficiency in the contested environment. Furthermore, we present formal mathematical analysis and performance evaluation using numerical results. Results show that the proposed improved asynchronous FL outperforms the other FL approaches.**

***Index Terms*—Federated learning, asynchronous federated learning, improved asynchronous federated learning.**

I. Introduction

Federated Learning (FL) was proposed as a distributed collaborative learning in 2016 for communication efficiency and data privacy [2]. In FL, multiple clients $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, ...\mathcal{C}_k, ...\mathcal{C}_K\}$ train a machine learning model locally $\{m_1, m_2, ...m_h, ...m_K\}$ without sharing their training data[1], $\mathcal{D}_k$. We assume that $\mathcal{D}_k$ are allowed to be different from clients to clients (statistical heterogeneity) and $\{\mathcal{D}_k\}_{k=1}^{K} \sim (\mathcal{X}, \mathcal{Y})$. In FL, initial global model is distributed by a FL server $\mathcal{S}$ to all FL clients. Periodically, FL clients use their own local data to train their local models and report their local model parameters to the FL server. Then the FL server aggregates those reported local models $\{m'_k\}_{\forall k}$ to update the global model. The most popular aggregation algorithm for the global model is FedAvg [3] where the global model $\mathcal{G} = \frac{1}{|\mathcal{C}|}\sum_{k=1}^{|\mathcal{C}|} m_k^l$ is iteratively computed until the global model converges and exhibits satisfactory performance for the given task. Note that when new or fresh information $\mathcal{X}_n$ which is unseen before is available at the client, we have that $\mathcal{X} \cap \mathcal{X}^* = \emptyset$ or $\mathcal{D}_k \cap \mathcal{D}_k^* = \emptyset, \forall k$, indicating that the previous dataset did not have the new information that one can have (which is common in the contested battlefield environment).

In federated learning approach, we assume that any clients can join and leave/withdraw from the system dynamically at any time including the times when FL training takes longer time than users' delay requirements or demands of clients is not met. Next, we assume that there are two types of FL clients: 1) FL clients who actively precipitate in training and use the global model for inferencing and 2) FL clients who just use the available global model and do not participate in local model training. Our analysis focuses on the former one where FL clients participate in local training and report their local models to their FL server and use the global model for inferencing. Moreover, to tackle weaknesses of traditional synchronous FL, asynchronous FL has emerged as a solution where FL server can compute global parameters without waiting to get model parameters from all clients [6]. However, asynchronous FL also suffers from many problems including unnecessary updates for global models which might not lead to the best model [1], [9].

Our goal is to investigate the improved asynchronous FL for client selection strategies that takes account of computing and communication delays, freshness of the data/information being used in generating local models, and the loss while providing the best global model with high efficiency in the contested environment.

The remainder of the paper is organized as follows. System model for general FL is presented in Section II followed by background and problem statement in Section III. Formal mathematical analysis and algorithms are presented in Section IV. We presents performance evaluation in Section V and conclusion in VI.

II. System Model

A system model for a typical FL is given in Fig. 1 where FL clients train their machine learning models locally without sharing their training data but they share their local model parameters to the FL server. Then the FL server aggregates the local models to find a global model and broadcasts the global model to its clients. We assume that clients can join and leave at any time because of their interest or because of their communication issues in the contested environment. Furthermore, there could be FL clients who do not want to participate in learning model but be interested to use the available global models which are not shown in Fig. 1.

III. Background

Synchronous FL: In synchronous FL, individual participating clients train their local models by using their local data and

[1]FL clients have non-independent and identically distributed (non-IID) data and the feature and task distributions of different clients are not identical [5], [8].

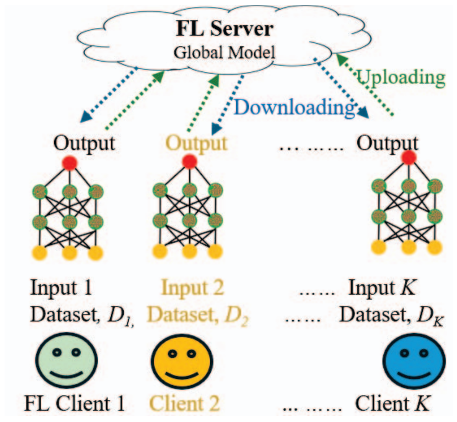

Fig. 1: A Typical Federated Learning Framework

share their model parameters to the FL server. In synchronous FL, server waits to receive updates from all participating clients before starting to aggregate local models for the updated global model, as shown in Fig 2. In FL, only parameters are transmitted to FL server without sharing the actual data to provide data privacy and communication efficiency (because of small size of parameters compared to actual data at clients) [4], [7]. In synchronous FL, overall performance is hugely impacted by high communication cost, coordination cost and waiting time for all clients to report their parameters as well as from selfish or uncooperative FL clients.

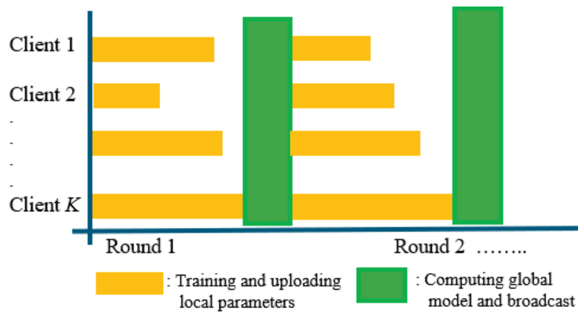
Client 1
Client 2
Client K
Round 1
Round 2 ....
: Training and uploading local parameters
: Computing global model and broadcast

Fig. 2: Synchronous Federated Learning Process

Asynchronous FL: In asynchronous FL, clients perform local model updates at their own pace and then communicate their model parameters asynchronously with a centralized FL server. The FL server incorporates the reported parameters into the global model and broadcasts the global model. These asynchronous pipeline process significantly reduces waiting time for FL server as the server can act upon receiving the model parameters from the clients to aggregate them into the global model, as shown in Fig. 3.

The asynchronous FL can handle contested environment with unresponsive clients (unresponsive because of the limited communication bandwidth and networking connectivity) or uncooperative clients better than the synchronous FL. However, in asynchronous FL, the FL server will have to incorporate the

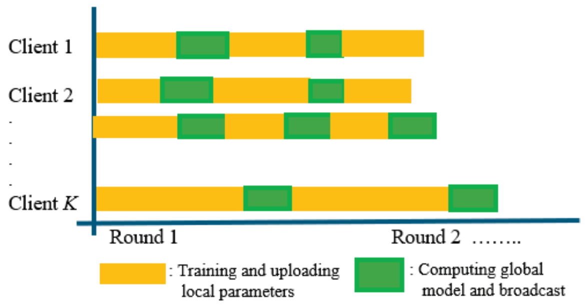
Client 1
Client 2
Client K
Round 1
Round 2 ....
: Training and uploading local parameters
: Computing global model and broadcast

Fig. 3: Asynchronous Federated Learning Process

reported local model parameters into the global model when an update from each client is received even if the updated local parameters might not result in a better global model.

Improved asynchronous FL: To address aforementioned issues, in improved asynchronous FL, the centralized FL server waits for a couple of clients to report their local model parameters before starting to aggregate to produce a global model subject to the loss to be within a given limit and delay being below the provided threshold. The improved asynchronous FL process is depicted in Fig. 4.

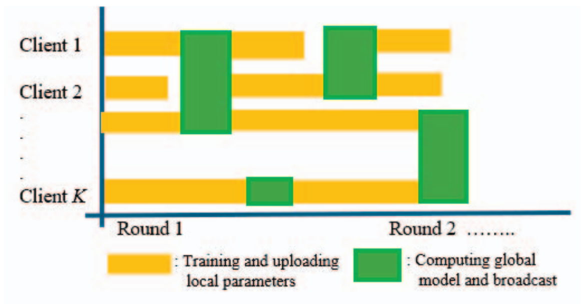
Client 1
Client 2
Client K
Round 1
Round 2 ....
: Training and uploading local parameters
: Computing global model and broadcast

Fig. 4: Improved Asynchronous Federated Learning Process

Furthermore, it is important to include new data (or fresh data or less aged data) in training to generate updated local models to incorporate the recent change in the environment such as in battlefield which is contested and changing dynamically. In the next section, we present a problem statement and the formal analysis for improved asynchronous FL which improves the overall performance in the contested environment subject to given constraints.

IV. Problem Statement and Formal Analysis for Improved Federated Learning

This section presents the problem statement and formal analysis for improved federated learning.

A. Problem statement

For federated learning, our goal is to optimize the asynchronous FL through client selection strategies that takes account of computing and communication delays, freshness of the data/information being used in generating local models, and the loss while providing the best global model. The optimization problem can be expressed as the minimization of expected value that is a function of loss, time taken for training locally and uploading model parameter (which depends on computing resources and communication link quality) to the FL server $\mathcal{S}$ and freshness of the information, that is,

$$\min_{w\in\mathbb{R}^d} \sum_{k=1}^{n\leq|\mathcal{C}|} \mathbb{E}_{(x,y)\sim\mathcal{D}_k}\left[\ell_k(\mathcal{M}(x,w),y), \Delta_k(\mathcal{M}(.),\mathbb{E}[T_w])\right] \quad (1)$$

subject to

$$Co1: \Delta_k(\mathcal{M}(.),\mathbb{E}[T_w]) \leq \overline{\Delta} \leftarrow \text{Training \& uploading delay}$$

$$Co2: \ell_k(\mathcal{M}(x,w),y) \leq \epsilon \quad \leftarrow \text{Loss optimized on client } k$$

$$Co3: \mathcal{D}_k \neq \emptyset \text{ and } \mathcal{D}_k \cap \mathcal{D}_k^* = \emptyset \quad \leftarrow \mathcal{D}_k^* \text{ denotes new info.}$$

where d represents the number of model parameters, $\mathcal{M}(x,w)$ is the machine learning (such as deep neural network) prediction function and $\mathcal{D}_k$ is the training data at client k. The constraint $Co1$ denotes the total delay caused by i) learning at FL client (learning model and resources related) and ii) uploading local parameters to the FL server (communication related), the constraint $Co2$ denotes the loss function of client k, and the $Co3$ indicates that the client has new information for generating new model parameters to reflect dynamically changing battlefield environment. Next, the $\overline{\Delta}$ represents the max delay that improved asynchronous can tolerate for each client and ϵ denotes the loss limit. In improves asynchronous FL, clients who will be reporting to FL server is $n \leq |\mathcal{C}|$ where $|\mathcal{C}|$ is the cardinal of $\mathcal{C}$ representing the total number clients in FL. When $n = |\mathcal{C}|$, the asynchronous FL become the synchronous FL.

B. Response/Wait Time Analysis for FL Server

As we discussed in previous sections, the FL clients report their local model parameter to the FL server. This process can be modeled as a Poisson process with parameter λ (which is arrival rate of reports from clients at the FL server). Before calculating a global model and broadcasting it to the FL clients, the FL server observes (or waits) until it can see that the FL clients who reported their models gives the best possible global model that optimizes (1). The *time* that FL server has to wait until it optimizes the Eq. (1) and no additional clients are needed to improve the global models is denoted as δ units. Now, our goal is to find out the expected time the FL server has to wait to satisfy the constraints before calculating the global model and broadcasting the global model to the FL clients.

Let us assume that the FL server starts the aggregation process at time 0 and let $T_1, T_2, \cdots, T_n$ denotes the FL model update inter-arrival times (which are independent events). Let us further consider that W be the random variable that counts the FL clients that will report that optimizes (1) before the FL server can compute the aggregate model. Then, we can write

$$\begin{aligned}\Pr[\{W=n\}] &= \Pr\left[\{T_1\leq\delta\}\cap\cdots\cap\{T_b\leq\delta\}\cap\{T_{b+1}>\delta\}\right]\\ &= \Pr\left[\{T_1\leq\delta\}\right]\cdot\Pr\left[\{T_n\leq\delta\}\right]\cdot\Pr\left[\{T_{n+1}>\delta\}\right]\\ &= \left(1-e^{-\lambda\delta}\right)^n e^{-\lambda\delta}\end{aligned}$$

Thus, the expected number of FL clients that report their model parameters before FL server can compute a global model is

$$\begin{aligned}\mathbb{E}[W] &= \sum_{n\geq 0} n\left(1-e^{-\lambda\delta}\right)^n e^{-\lambda\delta}\\ &= \left(1-e^{-\lambda\delta}\right)e^{-\lambda\delta}\sum_{n\geq 0} n\left(1-e^{-\lambda\delta}\right)^{n-1}\\ &= \left(1-e^{-\lambda\delta}\right)e^{-\lambda\delta}e^{2\lambda\delta}\\ &= e^{\lambda\delta}-1\end{aligned}$$

Finally, the expected time the FL server has to wait to calculate the global model that optimizes (1) and broadcast it to its FL clients is

$$\mathbb{E}[T_w] = \mathbb{E}[W]\mathbb{E}[T] = \frac{e^{\lambda\delta}-1}{\lambda}$$

C. New Information at Clients for Improved Asynchronous FL

For a given FL client, let Q be the random variable that denotes the amount of time a FL client will have to wait to use new information (when training is ongoing with old data) to generate the model parameters and send those updated parameters to FL server. Note that if a FL client gets new information (new dataset or data sample) with a rate of α for training when there are already f datasets or samples of datasets in pipeline to use to train to generate local model parameters, then the total waiting time of the new/fresh info arrival is the convolution of $f+1$ independent exponential random variables with a common parameter μ (which represents the use of the data for training to generate local parameters which is also known as service rate for the data). Next, let us consider that F be the random variable that counts the number of data samples to be used to train the model that the new arrival finds in the system. Our goal is to find the probability distribution of Q

$$\begin{aligned}\Pr[\{Q\leq t\}] &= \sum_{g=0}^{\infty}\Pr[\{Q\leq t\}\mid\{F=f\}]\Pr[\{F=f\}]\\ &= \sum_{n=0}^{\infty}\Pr[\{Q\leq t\}\mid\{F=f\}]\cdot\Pr\left[P_f\right]\\ &= \sum_{f=0}^{\infty}\int_0^t\frac{(\mu x)^f}{f!}\mu e^{-\mu x}\left(\frac{\alpha}{\mu}\right)^f\left(1-\frac{\alpha}{\mu}\right)dx\\ &= \int_0^t \mu e^{-\mu x}\left(1-\frac{\alpha}{\mu}\right)\sum_{f=0}^{\infty}\frac{(\alpha x)^f}{f!}dx\\ &= \int_0^t \mu e^{-\mu x}\left(1-\frac{\alpha}{\mu}\right)e^{\alpha x}dx\\ &= \int_0^t \mu e^{-\mu x\left(1-\frac{\alpha}{\mu}\right)}\left(1-\frac{\alpha}{\mu}\right)dx\\ &= 1-e^{-\mu t\left(1-\frac{\alpha}{\mu}\right)} = 1-e^{-(\mu-\alpha)t}\end{aligned}$$

where $Pr[f] = Pr[\text{new info/data is available to the client}]$ denotes the probability. From above, we can conclude that the total waiting time is exponentially distributed with parameter $\mu-\alpha$ which indicates the process is dependent on the arrival rate of the new information and use rate of the same to generate local model parameters which are reported for the global model.

D. Algorithms for Computing Global Model and FL Client Selection

This section presents two algorithms: **Algorithm 1** for FL clients to generate local models and report to the FL server and **Algorithm 2** for FL server to compute global model by optimizing the expected value in Eq. (1) and broadcast the updated global model to the clients.

Algorithm 1 Clients in Improved Asynchronous FL

```
1:  Input: Set of clients $\mathcal{C}$, local data, training model, and other algorithmic initial parameters and constants, and an initial global model.
2:  Output: Get an updated model and send the updated local model to the FL server.
3:  Start with an initial global model and step 0.
4:  repeat for each FL client, $k$
5:      $t = t + 1$                                   ▷ Increment iteration.
6:      Train local model using local data.
7:      Upload the model parameters to the FL server $\mathcal{S}$.
8:      if $\mathcal{D}_k \cap \mathcal{D}_k^* = \emptyset$ OR $\ell_k(.) \leq \epsilon$ then ▷ No new info is available and model meets the loss criteria.
9:          Do not train local model with old data.
10:         Go to Step 13 and Exit.
11:     else
12:         Go to Step 5 to start the training with new/fresh info and send updated model parameters to the FL server.
13: until Convergence criteria are met or no fresh info is available.
14: return Upload local model to the FL server $\mathcal{S}$.
```

Algorithm 2 Client selection in Improved Asynchronous FL

```
1:  Input: Number of clients $|\mathcal{C}|$, local models, and other algorithmic initial parameters and constants, and an initial global model.
2:  Output: Get updated global model and send it to all FL clients.
3:  Start with an initial global model and client $k = 0$.
4:  repeat
5:      $k = k + 1$                                   ▷ Increment iteration
6:      if $\sum \ell_k(.) \leq \epsilon$ and $\sum \Delta_k(.) \leq \overline{\Delta}$ then ▷ Model meets the loss and delay criteria
7:          Aggregate local models received from clients to get an updated global model.
8:          Go to Step 4 for the next FL client.
9:      else
10:         Go to Step 11.
11: until Convergence criteria are met for the global model.
12: return Updated global model and broadcast it to all FL clients.
```

V. Performance Evaluation

We consider a closed environment for evaluating the performance of the proposed approach where we assume 10 FL clients train their local models and share the learned model parameters to their FL server which computes the global model and broadcasts it to all of its clients periodically. We have considered three different scenarios: synchronous FL, asynchronous FL and improved asynchronous FL.

In a *synchronous FL*, we assume that not all clients will be able to communicate to their FL server (even though they are able to train their local models) because of communication issues caused by jamming or communication outage in the dynamically changing contested environment. Thus, if 80% or more FL clients' model parameters are received, FL server computes the global model and broadcasts the updated model.

In an *asynchronous FL*, whenever one client's parameters are received, FL server incorporates those parameters into the global model and broadcasts the updated model.

In an *improved asynchronous FL*, whenever FL server has local model parameters from clients that optimize (1), it computes the global model and broadcasts the updated model to its clients.

Fig. 5 shows number of clients reporting local model parameters to their server in synchronous FL, asynchronous FL and improved asynchronous FL where 80% or more clients report is needed in synchronous FL and at least one client needs to report its updated model parameters to the server in asynchronous FL (there could be 2 clients reporting their parameters to server could happen at the same time as depicted in Fig. 5). In the improved asynchronous FL, server waits for the subset of FL clients n from the set of all clients $n \leq |\mathcal{C}|$ (in our example $|\mathcal{C}| = 10$) to report their updated model parameters that minimize the expected value in Eq. (1). Note that in step 4 as shown in Fig. 5, four clients are needed to optimize Eq. (1) for improved asynchronous FL whereas only one client was reporting in traditional asynchronous FL as expected. Furthermore, we can see that the same number of clients (say in step 1 as in Fig. 5) reported their parameters to server where 2 clients could optimize Eq. (1) in improved asynchronous FL but it happened to have two clients reported at the same time in the traditional asynchronous FL.

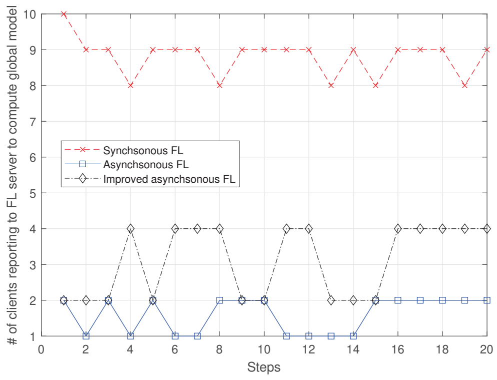

Fig. 5: Number of clients reporting their local parameters to the FL server for a global model for all three different scenarios.

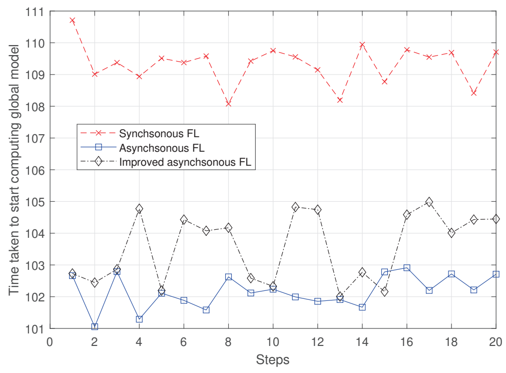

Fig. 6: Total wait time for the server to start computing global model while meting delay, loss requirements for three different scenarios.

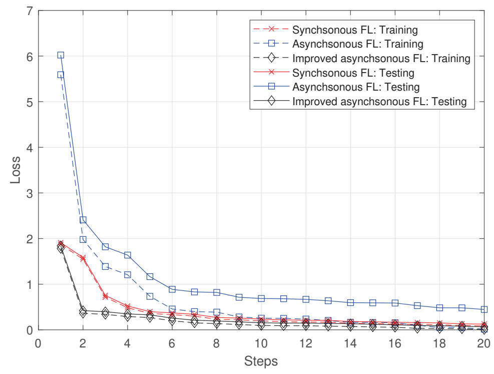

Fig. 7: Expected training and testing loss for three different scenarios.

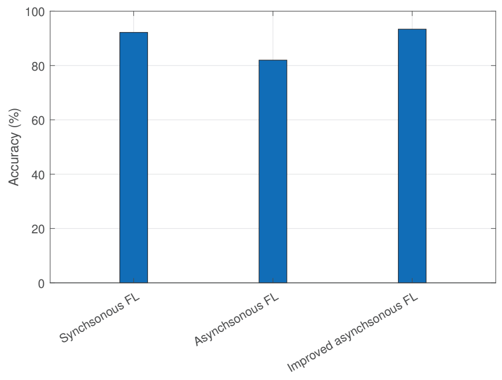

Fig. 8: Expected accuracy for three different scenarios.

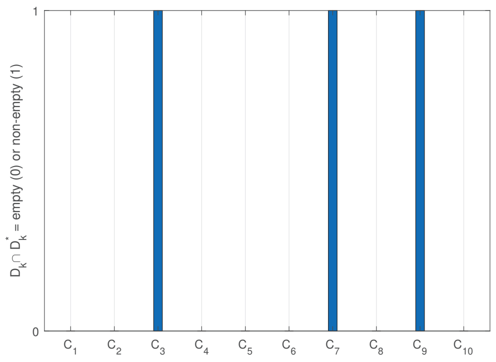

Fig. 9: State of the clients for one instance about new or fresh information $\mathcal{D}_k \cap \mathcal{D}_k^* = \emptyset$ or non-empty. For a given instance we plotted here, only clients 3, 7 and 9 had new information and all other users had no new information.

Next, Fig. 6 shows the expected wait time for FL server to compute a global model based on reported local models from clients in all scenarios. Delay is higher in synchronous FL as FL server has to wait until 80% or more clients need to report, asynchronous FL takes less time as FL server starts computing after receiving local parameters from a client. However, in improved asynchronous FL, server has to wait a bit more time than that of asynchronous FL to compute a global model as it has to wait until it meets the criteria to optimize Eq. (1).

Furthermore, Fig. 7 shows the variation of excepted values of training and testing losses for different scenarios where improved asynchronous FL outperforms as it optimizes based on loss, delay and new information available that satisfies the criteria to optimize (1).

Expected value of accuracy is plotted in Fig. 8 where accuracy of the synchronous FL and improved asynchronous FL are comparable. The asynchronous FL has the lowest performance since it considers sending global models after each client's update which might not reflect the overall dynamic system.

We also took a snapshot at one of the instances for FL clients to see which ones have new information and which ones do not. We can see that, for a given time instance, only clients 3, 7 and 9 had new information and all other clients had no new information as shown in Fig. 9. Note that when new information is incorporated in training and testing, FL model gives better results in terms of loss and accuracy.

VI. Conclusion

This paper studied the improved asynchronous federated learning approach that enhances the overall performance while mitigating weaknesses of the both synchronous FL and traditional asynchronous FL. Specifically, this paper explored client selection strategies in asynchronous FL that takes account of computing and communication delays, freshness of the data/information being used in generating local models, and the loss while providing the best global model with high efficiency in the contested environment. We presented formal mathematical analysis and performance evaluation using numerical results to support our analysis.

Our future work includes detailed formal analysis and extensive experimentation with all possible cases for the contested environment.

Acknowledgments

This work was supported by the DoD Center of Excellence in AI/ML at Howard University under Contract W911NF-20-2-0277 with the US ARL. However, any opinions, findings, conclusions, or recommendations expressed in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the funding agency.

References

[1] Bimal Ghimire, Danda B Rawat, and Abdul Rahman. Efficient information dissemination in blockchain-enabled federated learning for iov. *IEEE Internet of Things Journal*, 2023.

[2] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[3] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[4] Jianyang Ren, Wanli Ni, and Hui Tian. Toward Communication-Learning Trade-Off for Federated Learning at the Network Edge. *IEEE Communications Letters*, 26(8):1858–1862, August 2022.

[5] Lixu Wang, Yang Zhao, Jiahua Dong, Ating Yin, Qinbin Li, Xiao Wang, Dusit Niyato, and Qi Zhu. Federated learning with new knowledge: Fundamentals, advances, and futures. *arXiv preprint arXiv:2402.02268*, 2024.

[6] Chenhao Xu, Youyang Qu, Yong Xiang, and Longxiang Gao. Asynchronous federated learning on heterogeneous devices: A survey. *Computer Science Review*, 50:100595, 2023.

[7] Liang Zeng, Wenxin Wang, and Wei Zuo. A Federated Learning Latency Minimization Method for UAV Swarms Aided by Communication Compression and Energy Allocation. *Sensors*, 23(13), 2023.

[8] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated Learning with Non-IID Data. *arXiv preprint arXiv:1806.00582*, 2018.

[9] Hongbin Zhu, Yong Zhou, Hua Qian, Yuanming Shi, Xu Chen, and Yang Yang. Online client selection for asynchronous federated learning with fairness consideration. *IEEE Transactions on Wireless Communications*, 22(4):2493–2506, 2022.