To Talk or to Work: Delay Efficient Federated Learning over Mobile Edge Devices

Pavana Prakash*, Jiahao Ding*, Maoqiang Wu†, Minglei Shu‡, Rong Yu†, and Miao Pan*
*Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77204
†School of Automation, Guangdong University of Technology, Guangzhou, China
‡Shandong Artificial Intelligence Institute, Qilu University of Technology (Shandong Academy of Sciences), Jinan, China

GLOBECOM 2021 - 2021 IEEE Global Communications Conference | 978-1-7281-8104-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/GLOBECOM46510.2021.9685793

***Abstract*—Federated learning (FL), an emerging distributed machine learning paradigm, in conflux with edge computing is a promising area with novel applications over mobile edge devices. In FL, since mobile devices collaborate to train a model based on their own data under the coordination of a central server by sharing just the model updates, training data is maintained private. However, without the central availability of data, computing nodes need to communicate the model updates often to attain convergence. Hence, the local computation time to create local model updates along with the time taken for transmitting them to and from the server result in a delay in the overall time. Furthermore, unreliable network connections may obstruct an efficient communication of these updates. To address these, in this paper, we propose a delay-efficient FL mechanism that reduces the overall time (consisting of both the computation and communication latencies) and communication rounds required for the model to converge. Exploring the impact of various parameters contributing to delay, we seek to balance the trade-off between wireless communication (to talk) and local computation (to work). We formulate a relation with overall time as an optimization problem and demonstrate the efficacy of our approach through extensive simulations.**

I. Introduction

Machine learning together with increased capabilities in mobile devices have led to a tremendous rise in the number of smart mobile devices and data generated at the edge network. About 80 billion devices are predicted to be connected to the Internet by 2025 [1]. Hence, computing networks are witnessing a paradigm shift from conventional cloud computing setting, by moving closer to the edge where data is produced, namely multi-access edge computing (MEC). However, utilizing centralized machine learning algorithms at the response-accelerated MEC is inefficient, since uploading and storing bulk data causes a large storage and communication bottleneck. Therefore, federated learning (FL) was introduced to solve these challenges where mobile devices jointly train a shared global model in a decentralized manner [2].

In an FL setup, user devices compute and transmit local model updates based on the local training data which are aggregated at the central server, facilitating users to learn collaboratively. With high-performance processors, modern mobile devices are equipped to handle such intensive computations, further aiding the implementation of FL in MEC. This has enabled its presence in a variety of delay-sensitive areas ranging from smart healthcare devices to predictive models from electronic health records. In particular, smart health applications have seen substantial success since they leverage the bulk data generated by tracking physical activities of its users from wearable devices such as smart watches, fitness trackers, and wristbands, to train quality learning models. Moreover, FL satisfies the privacy requirements of wearable computing by leaving personal data on the user devices [3].

For these extensive time-critical applications, the feasible offloading time has to be in the order of milliseconds [4]. In reality, without central availability of data, computing nodes need to communicate model updates often to attain convergence in FL. Communication of these updates may involve long round-trip times posing a limitation to this paradigm [5]. Moreover, unreliable and unpredictable network connections between the server and mobile devices could obstruct smooth transmission of updates. A large number of participants utilizing the constrained wireless bandwidth to upload model updates could add to uplink transmission delays. Therefore, given the nature of frequent exchange of updates in FL, over an expensive communication involving large number of mobile devices, reducing the overall time delay is crucial.

To address these challenges, many pioneering works analyze different aspects of the FL paradigm. Initial works such as [2] emphasizes on higher local computation to reduce the communication cost but lacks a theoretical model. Variants of FedAvg such as [6] and works on distributed optimization such as [7] aim to ease the communication burden. However, these works do not consider the limiting factors of wireless communication that can affect the performance of FL. Further, recent works including [8] formulate to reduce the time or energy consumption but do not contemplate the learning hyperparameters which significantly affect the training time.

While majority of the works focus on communication overhead, the latest surge of research in networks have paved way for the rapidly expanding 5G and the upcoming 6G networks which alleviate communication burdens [9]. To illustrate, a single-step of local computation on ResNet50 model over GPU consumes few hundreds of milliseconds [10], which is nearly comparable to the time taken to transmit over a wireless connection with transmission rate of 1 Gbps. Therefore, it is worthwhile to investigate the impacts of communication, local computation in conjunction with convergence over FL.

The work of P. Prakash, J. Ding, and M. Pan was supported in part by the U.S. National Science Foundation under grants CNS-1801925, CNS-2029569, and CNS 2107057. The work of M. Wu and R. Yu was supported by National Natural Science Foundation of China (No. 61971148).

978-1-7281-8104-2/21/$31.00 ©2021 IEEE

Intuitively, if a user performs more local computation to achieve a high local accuracy, frequent communication can be avoided due to decrease in the number of model updates. However, in case of data that is not representative of the overall distribution, this leads to local overfitting, adding to the convergence delay [11]. On the contrary, to reduce computation, we can perform single-step updates which consumes lesser time to compute and communicate each update. However, it results in additional communications to update the current model, in order to attain a targeted global accuracy, increasing the overall time. As a result, the trade-off between wireless communication (to talk) and local computation (to work) of mobile devices needs to be balanced.

In this paper, we mainly aim to realize a balance between the two, by carefully studying the effect of various parameters, constraining overall time as the principal factor. We observe that for FL on each mobile device, the 'talking' (i.e., global communication) time is determined by the local update size as well as wireless parameters such as transmission power, channel gain, bandwidth and background noise. Correspondingly, the 'working' (i.e., local computation) time of each mobile device is influenced by the training data size and hyperparameters, together with the processor capabilities such as number of cycles and frequency scales. The overall time is further conditioned by the preset accuracies and the number of connected mobile devices. Capturing this motivation, our salient contributions can be summarized as follows,

- We build a theoretical model for FL on edge GPUs over wireless networks that considers the impact of both computation and communication models on the overall time of training. To this end, we formulate an optimization problem to minimize the overall time consumed and reduce the number of communication rounds required to achieve FL convergence.
- Based on this model, we propose a delay-efficient FL solution mechanism by optimizing the influencing parameters to reduce the overall time. To realize this, we further consider the trade-off between local computation (to work) and wireless global communication (to talk). We demonstrate the theoretical convergence of the model and further define computational values based on leveraging the frequency of GPUs.
- We verify the effectiveness of our solution mechanism through extensive simulations over real-world datasets and illustrate the influence of each parameter on the overall time delay. We demonstrate that our solution significantly reduces the overall time in comparison with the baseline methods, while still achieving high accuracy.

II. Delay-Efficient Federated Learning (DEFL) and Model Description

A. Federated Learning over Mobile Edge Computing

We consider an MEC-assisted FL system consisting of one edge (parameter) server and a set of $\mathcal{M}$ of M mobile devices. Each mobile device m has a local dataset $\mathcal{D}_m$ of size D_m, constituting a set of input samples and labels, $\{x_i^m, y_i^m\}_{i=1}^{D_m}$ with d features. The loss function F with respect to model parameters $\mathbf{w}$ on m's dataset is given by,

$$\mathtt{F}_m(\mathbf{w}) = \frac{1}{D_m} \sum_{i \in \mathcal{D}_m} \mathtt{f}_i(\mathbf{w}), \tag{1}$$

where $\mathtt{f}_i(\mathbf{w}) = \mathtt{f}_i(\mathbf{w}; x_i^m, y_i^m)$ is the loss on data point i. The objective of minimizing the global loss is of the form,

$$\min_{\mathbf{w} \in \mathbb{R}^d} \mathtt{F}(\mathbf{w}) = \sum_{m=1}^{M} \frac{D_m}{D} \mathtt{F}_m(\mathbf{w}), \tag{2}$$

where $D = \sum_{m=1}^{M} D_m$ is the total data size.

B. Computation Model

Typically, CPUs incur high computation costs [8] and in contrast, with increased processing power and memory bandwidth, GPUs lower the computational costs. Furthermore, its massively parallel architecture can efficiently handle compute-intensive manipulations making it most suitable for high performance deep learning models. Hence in our work, we build a model for FL over edge GPUs whose frequency $f_m \in \mathbb{R}^m$, can be given as,

$$f_m = \frac{1}{a_s + \frac{a_c}{f_c} + \frac{a_M}{f_M}}, \tag{3}$$

where a_s, a_c and a_M are constants related to static, core frequency f_c (including all of GPU's cores) and memory frequency f_M, respectively [12]. G_m is the number of GPU cycles required for local computation by a mobile device and can be measured offline. We use mini-batch stochastic gradient descent (SGD) in which the computation is conditioned by the given batch size b. The local computation time taken to execute a single iteration of GPU-accelerated mini-batch SGD at the m-th mobile device can be given by,

$$T_m^{cp} = \frac{G_m b}{f_m}. \tag{4}$$

The proposed model can also be used with CPUs or other processors where f_m in (4) is replaced by the given processor's frequency value. Since GPUs are capable of parallel execution and process the whole-batch samples simultaneously [13], in our work, we assume a synchronous model implying parallel local computation by mobile devices. Hence, the computation time during each communication round depends on the value of the slowest computation i.e., the highest time consumed by any mobile device given by,

$$T_{cp} = \max_m T_m^{cp}. \tag{5}$$

C. Communication Model

The downlink bandwidth used by the server to broadcast the updated global model is much larger than the uplink bandwidth used by the mobile devices to transmit their local updates. Since this leads to a minimal downlink time versus uplink time [8], we consider only the uplink time as the communication time. Further, we assume that the

local model update size s to be fixed and the same for all mobile devices. Considering the transmission bandwidth B, transmission power of the mobile device m as p_m, h_m being the channel gain of the link between the mobile device and the server, N_o the background noise, the communication time of one model update from each mobile device to the parameter server can be given by,

$$T_m^{cm} = \frac{s}{B \log_2 \left(1 + \frac{p_m h_m}{N_o}\right)}. \tag{6}$$

Assuming a synchronous model for communication, the communication time per communication round is given by,

$$T_{cm} = \max_m T_m^{cm}. \tag{7}$$

D. Overall Time

The total computation time per communication round depends on the number of local iterations V and the overall time depends on communication together with computation time. Hence, the total time consumed by the system for one communication round can be defined as,

$$T = T_{cm} + V T_{cp}. \tag{8}$$

E. To Talk or To Work

Both communication and computation-intensive networks can significantly benefit from reduced communication as communication is expensive. In addition, factors such as slow speed, poor communication channel, congestion in networks further challenge the efficient communication of model updates. Thus, reducing communication is a necessity in comparison with computation. In this aspect, when mobile devices perform more local computation to reach a high preset local accuracy, the number of local updates is reduced, indeed reducing the frequency of communicating with the server. This suggests fewer communication rounds implying savings in communication cost and time. Correspondingly, when functions across users share some similarity, taking local steps can lead to faster convergence [14]. Moreover, since the recent mobile edge devices are equipped with fast processors, increasing local computation does not burden or compromise the computation time. Adding parallel computing capabilities with the utilization of GPUs further aids in speeding up computation as described in Section II-B. Hence, we reduce the 'talking' over 'working' when balancing the trade-off.

F. DEFL Algorithm

Our methodology of FL named DEFL (**D**elay **E**fficient **F**ederated **L**earning), is described in Algorithm 1. The problem is formulated at the system-level and the computed values from the proceeding sections are utilized in our algorithm.

Algorithm 1 DEFL

Inputs: $\mathbf{w}_0$, preset global convergence error ϵ, computed values of b^* and $\theta^* \in [0, 1]$.

1: Initialize $\mathbf{w}_0$
2: **for** 1 to H communication rounds for achieving ϵ, **do**
3: **Local Computation**: Each mobile device m performs local training to compute stochastic gradient on mini-batch sized b^*, and solves (2) in V local rounds to achieve θ^*-approximate solution.
4: **Wireless Communication**: Every participating mobile device m transmits the local model update w_v^m to the edge server through the communication channel.
5: **Aggregation and Broadcast**: The parameter server aggregates the received updates to obtain the global model, and broadcasts it to the mobile devices.
6: **end for**

III. Theoretical and Convergence Analysis

To present the theoretical analysis, we first state the following standard assumptions on the local loss function $\mathtt{F}_m$.

Assumption 1. *The loss function $\mathtt{F}_m$ is L-smooth, that is for all $\mathbf{v}$ and $\mathbf{w}$, we have $\mathtt{F}_m(\mathbf{v}) \leq \mathtt{F}_m(\mathbf{w}) + (\mathbf{v} - \mathbf{w})^T \nabla \mathtt{F}_m + \frac{L}{2}\|\mathbf{v} - \mathbf{w}\|^2$.*

Assumption 2. *Let ξ_k^m be sampled from the m-th device's local data uniformly at random. The variance of stochastic gradients in each device is bounded, i.e., $\mathbb{E}\|\nabla \mathtt{F}_m(\mathbf{w}_k^m, \xi_k^m) - \nabla \mathtt{F}_m(\mathbf{w}_k^m)\| \leq \sigma^2$.*

The convergence bound of the model can be given by the following theorem using $\mathbf{w}_*$ as a fixed minimizer of $\mathtt{F}$.

Theorem 1 ([7]). *Suppose Assumptions 1 and 2 hold, and a constant stepsize η such that $\eta = \frac{\sqrt{M}}{4L\sqrt{K}}$ is chosen and the FL algorithm is run on identical data, then we have,*

$$\mathbb{E}\left[\mathtt{F}(\bar{\mathbf{w}}_K) - \mathtt{F}(\mathbf{w}_*)\right] \leq \frac{8\|\mathbf{w}_0 - \mathbf{w}_*\|^2}{\sqrt{MK}} + \frac{\sigma^2}{2L\sqrt{MK}} + \frac{\sigma^2 M(V-1)}{LK}, \tag{9}$$

where $\bar{\mathbf{w}}_K = \frac{1}{K}\sum_{k=1}^K \hat{\mathbf{w}}_k$ and $\hat{\mathbf{w}}_k = \frac{1}{M}\sum_{m=1}^M \mathbf{w}_k^m$. Additionally, the number of gradient steps is K, local rounds is V, and mobile devices is M.

Remark 1. *The result of Theorem 1 is based on each user only computing a single stochastic gradient in each global iteration. However, in our FL setting, each mobile device computes a mini-batch of size b in each communication round. Thus, we present the following corollary to show the convergence of DEFL.*

Corollary 1. *Suppose Assumptions 1 and 2 hold, and a constant stepsize η such that $\eta = \frac{\sqrt{M}}{4L\sqrt{K}}$ is chosen, with $K \geq M$ and the batch size equals b, then we have,*

$$\mathbb{E}\left[\mathtt{F}(\bar{\mathbf{w}}_K) - \mathtt{F}(\mathbf{w}_*)\right] \leq \frac{8\|\mathbf{w}_0 - \mathbf{w}_*\|^2}{\sqrt{MK}} + \frac{\sigma^2}{2bL\sqrt{MK}} + \frac{\sigma^2 M(V-1)}{bLK}. \tag{10}$$

Proof. Mini-batch SGD is conditioned by the given batch size b. Using this in (9), we hence obtain this corollary. □

Remark 2. *From Corollary 1, we can observe that when each mobile device considers a mini-batch size b in each iteration, it reduces the variance by a factor of b.*

We now use the convergence properties of DEFL, to estimate the number of communication rounds required to complete training of the mobile devices in coordination with the edge server. We hence present the following corollary.

Corollary 2. *The number of communication rounds for achieving an ϵ-global model convergence, i.e, satisfying $\mathbb{E}\left[\mathtt{F}(\bar{\mathbf{w}}_K) - \mathtt{F}(\mathbf{w}_*)\right] \leq \epsilon$ is given by,*

$$H = \mathcal{O}\left(\frac{1}{b^2\epsilon^2 MV} + \frac{M}{b\epsilon}\right), \tag{11}$$

where $\mathcal{O}$ is the big-$\mathcal{O}$ notation.

Proof. Since the system satisfies $\mathbb{E}\left[\mathtt{F}(\bar{\mathbf{w}}_K) - \mathtt{F}(\mathbf{w}_*)\right] \leq \epsilon$ to achieve an ϵ-accuracy, this is easily seen to be true by setting the right term in (10) to ϵ. Further, considering the relation of number of communication rounds, $H = K/V$ to solve for H and using the big-$\mathcal{O}$ notation in (10), we thus obtain (11). □

Remark 3. *At the user level, for achieving a θ-accuracy locally in SGD, i.e., $\mathbb{E}\|\mathbf{w}_V - \mathbf{w}_*\|_2^2 \leq \theta$, the number of local rounds required for a mobile device's local model is $V = \nu \log \frac{1}{\theta}$ [15], where ν is a constant related to step size and gradient noise. Then, substituting in* (11) *and using the term c to approximate the big-$\mathcal{O}$ notation we have,*

$$H = \frac{c}{b^2\epsilon^2 M\nu \log\frac{1}{\theta}} + \frac{cM}{b\epsilon}. \tag{12}$$

We can hence define the overall time for convergence as a product of the number of communication rounds required H, and the total time for one communication round T as,

$$\mathcal{T} = HT. \tag{13}$$

IV. Problem Formulation

From our theoretical analysis, we can deduce the impact of batch size (shown in Remark 2), number of communication rounds and time, preset accuracies and the number of participating mobile devices on the convergence rate. We hence achieve our objective of reducing the overall time by optimizing these variables. Accordingly, the optimization problem can be formulated using (13) with values from (12) and (8) as follows,

$$\underset{b,\theta,T_{cp}}{\text{minimize}} \left(\frac{c}{b^2\epsilon^2 M\nu \log\frac{1}{\theta}} + \frac{cM}{b\epsilon}\right) * \left(T_{cm} + \nu \log\frac{1}{\theta} T_{cp}\right) \tag{14}$$

$$\text{subject to} \quad b \in \{2^n | n = 0, 1, ...\} \tag{15}$$

$$0 \leq \theta \leq 1 \tag{16}$$

$$\max_m \frac{G_m b}{f_m} = T_{cp} \tag{17}$$

Constraint (16) defines the relative local accuracy that each mobile device attains on solving its local sub-problem. Here, $\theta = 0$ corresponds to the exact solution and $\theta = 1$ implies no improvement; hence we aim to achieve a lower value of θ for higher accuracy. This is also in accordance with (12), which indicates that 'working' more to achieve higher local accuracy results in smaller number of communication rounds. Although, this is in line with achieving our objective, (14) indicates that an inverse dependence on θ along with the relation with other parameters imply that we can only benefit a certain level by achieving a full relative accuracy of close to 0. Hence, this control helps in avoiding local overfitting condition that otherwise delays convergence. Constraint (15) sets a range of the most commonly used effective batch size values starting from 1, which is the case of SGD. For a given target global accuracy, a larger b leads to smaller number of communication rounds as per (12). Further, since we 'work' more to achieve a preset local accuracy to balance the trade-off, computation time determined by the slowest computation is defined by constraint (17).

V. Solution

The formulated problem to relieve the communication bottleneck by allowing more distributed computation is difficult to solve and involves a mix of integers and continuous variables. Hence, firstly, we introduce an auxiliary variable $\alpha = \log(1/\theta)$ to aid the optimization process, where $\alpha \in [0, +\infty)$ since $\theta \in [0, 1]$. Second, since constraint (17) is non-convex, we can transform it to convex to alleviate solving. Third, we relax the constraint of b in (15) from an integer to continuous; (14) can be reformulated as,

$$\underset{b,\alpha,T_{cp}}{\text{minimize}} \quad \left(\frac{c}{b^2\epsilon^2 M\nu\alpha} + \frac{cM}{b\epsilon}\right) * (T_{cm} + \nu\alpha T_{cp}) \tag{18}$$

$$\text{subject to} \quad b \geq 1 \tag{19}$$

$$\alpha \geq 0 \tag{20}$$

$$T_{cp} \geq \frac{G_m b}{f_m}, \quad \forall m \in \mathcal{M} \tag{21}$$

Proof. We use Karush-Kuhn-Tucker (KKT) conditions to solve the delay minimization problem (18). We first write the Lagrangian of (18) as follows,

$$\mathcal{L}(b, \alpha, T_{cp}, \lambda, \mu) = \left(\frac{cT_{cm}}{b^2\epsilon^2 M\nu\alpha} + \frac{cMT_{cm}}{b\epsilon} + \frac{cT_{cp}}{b^2\epsilon^2 M} + \frac{cM\nu\alpha T_{cp}}{b\epsilon}\right) - \lambda_1(b-1) - \lambda_2\alpha - \sum_{m=1}^{M} \mu_m \left(T_{cp} - \frac{G_m b}{f_m}\right), \tag{22}$$

where λ_1, λ_2, and $\{\mu_m\}_{m=1}^M$ are non-negative dual variables.

We take the first order derivatives of (22) with respect to the dual and optimization variables giving the stationary conditions from Eqs. (23)-(25) and list the rest of the KKT conditions as in Eqs. (26)-(28) shown by,

$$\frac{\partial \mathcal{L}}{\partial b} = \frac{-2cT_{cm}}{b^3\epsilon^2 M\nu\alpha} - \frac{cT_{cm}M}{b^2\epsilon} - \frac{2cT_{cp}}{b^3\epsilon^2 M} - \frac{cMT_{cp}\nu\alpha}{b^2\epsilon} - \lambda_1 + \frac{\mu_m G_m}{f_m} = 0, \quad \forall m \in \mathcal{M}, \tag{23}$$

$$\frac{\partial \mathcal{L}}{\partial \alpha} = \frac{-cT_{cm}}{b^2\epsilon^2 M\nu\alpha^2} + \frac{cT_{cm}M\nu}{b\epsilon} - \lambda_2 = 0, \tag{24}$$

$$\frac{\partial \mathcal{L}}{\partial T_{cp}} = \frac{c}{b^2\epsilon^2 M} + \frac{cM\nu\alpha}{b\epsilon} - \mu_m = 0, \quad \forall m \in \mathcal{M}, \tag{25}$$

$$\lambda_1(b-1)=0, \quad \lambda_2(\alpha)=0, \tag{26}$$

$$\mu_m\left(T_{cp}-\frac{G_m b}{f_m}\right)=0, \quad \forall m\in\mathcal{M}, \tag{27}$$

$$\lambda_1\geq 0, \quad \lambda_2\geq 0, \quad \mu_m\geq 0, \quad \forall m\in\mathcal{M}. \tag{28}$$

Since the inequality constraints are nonlinear yet differentiable and lower-bounded with a non-negative duality gap, the KKT necessary conditions serve as the optimality conditions. Hence, considering the above dual feasibility and complementary slackness conditions to solve the derivatives, KKT points are obtained. We check all of the obtained points for feasibility of the problem to finally deduce the optimal values as,

$$\begin{cases} \alpha^* = \sqrt{\frac{T_{cm} f_m}{M^2 \epsilon \nu^2 G_m}}, & \forall m\in\mathcal{M}; \\ b^* = 2cM\sqrt{\frac{T_{cm} f_m \epsilon}{G_m}}, & \forall m\in\mathcal{M}; \\ T^*_{cp} = \max_m \frac{G_m b^*}{f_m}, & \forall m\in\mathcal{M}. \end{cases} \tag{29}$$

□

From these relations, theoretically, the computation time is vastly affected by loads from all the mobile devices and the processors' computational capabilities and speed. Further, the batch size has a direct impact on T_{cp} with larger b leading to higher computation and faster convergence. Both b and the relative local error θ are impacted by the set global convergence error ϵ, M, along with other parameters. A lower value of θ^* (which can be computed from α^*) implying higher local accuracy, results in more 'working' and less 'talking'.

VI. Performance Evaluation

A. Settings

To evaluate the proposed delay efficient FL, we perform simulations using image classification tasks on the widely used MNIST[1] and CIFAR-10[2] datasets using CNN. For the FL tasks, we consider 1 parameter server and 10 mobile devices with distributed data and a learning rate of 0.01. In accordance with our computational model in (3), we use Nvidia RTX8000 with the number of GPU cycles of 30 cycles/bit and following constraint (17), we consider an equal maximum computation capacity of $f_m = 2$ GHz for all the mobile devices. For communication model, we assume the bandwidth $B = 20$ MHz and noise $N_o = -174$ dBm/Hz.

B. Impact of optimization parameters over convergence

According to (29), the computed values of b^*, θ^* and in turn T^*_{cp} are conditioned by the relative global convergence error ϵ. Hence, we empirically choose a value which leads to both increased performance yet takes less overall time. From the values in Fig. 1(a), we thus set $\epsilon = 0.01$. The optimized variables computed from our solution are used in (12) to determine the number of communication rounds H, which can be empirically shown as in Fig. 1(d). We now study the impact of the parameters on the overall time as follows.

Batch size. Generally, larger batch size to train the model allows computational speedups from the parallelism of GPUs. However, too large a batch size may lead to lower generalization, resulting in more overall time. Whereas, smaller batch sizes are shown to have less computation but are not guaranteed to converge to the global optima. Theoretically, the value of b computed from (29) has a lower limit of 1 and can be rounded off to 32 (for MNIST data size) which also corresponds to a value from the initial constraint (15). Empirically, as shown in Fig. 1(b), to achieve the same target ϵ, while $b = 64$ has the shortest overall time, it has a lower test accuracy. On the other hand, $b = 16$ achieves the highest test accuracy but takes more time of about 200 seconds. Consequently, the computed value of $b = 32$ achieves a good trade-off between prediction performance and overall time.

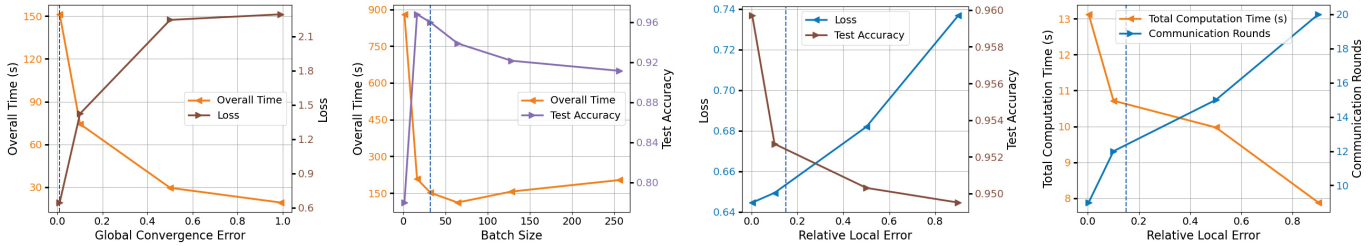
Relative Local Error. A lower value of relative local error θ (i.e., higher local accuracy), induces the model to 'work' more to achieve θ-accurate solution locally. This implies that fewer communication rounds is necessary according to (12) and consequently, lesser communication time than the original FedAvg algorithm. This behavior is captured in Fig. 1(d), where the theoretically calculated $\theta \approx 0.15$ from (29) has a higher computation time (due to 'working' more), but smaller H due to reduced number of model updates. Conversely, higher θ is undesired since lower computation results in 'talking' more with larger number of H and higher overall time. Further, as shown in Fig. 1(c), θ is just as low as to achieve a better performance in terms of reduced training loss at the same overall time while avoiding local overfitting.

Computation Time. The computed batch size influences the computation time since the training dataset is processed batch-wise, subject to device capabilities. Accordingly, increasing b implies taking advantage of the available computational resources of the mobile devices. As seen in Fig. 1(d), higher computation leads to reduced number of communication rounds which in turn leads to reduced overall time.

Comparison with Baseline. For evaluation, we use Federated Averaging (FedAvg) from [2] as a baseline to compare the performance of our proposed solution. For FedAvg on MNIST IID data using CNN, we set the parameter values as recommended by the authors through their experiments as $b = 10$ and $V = 20$. We then choose random values of $b = 16$ and $V = 15$ for MNIST and $b = 64$ and $V = 30$ for CIFAR-10 to test the effect of parameters as a whole, marked by 'Rand.'. For our work marked as 'DEFL', we choose values as per our delay-efficient optimized solution from Section V and as verified in Section VI. With a preset θ ensuring more computation, along with the optimized b and fixed ϵ, we observe from Fig. 2 that, although we achieve nearly the same test accuracy, DEFL significantly outperforms the baseline in terms of the overall time. Comparatively, we reduce the overall time by nearly 70% compared with FedAvg for MNIST and by 18% for CIFAR. Similarly, there is a reduction of around 38% comparing with 'Rand.' for MNIST and 75% for CIFAR.

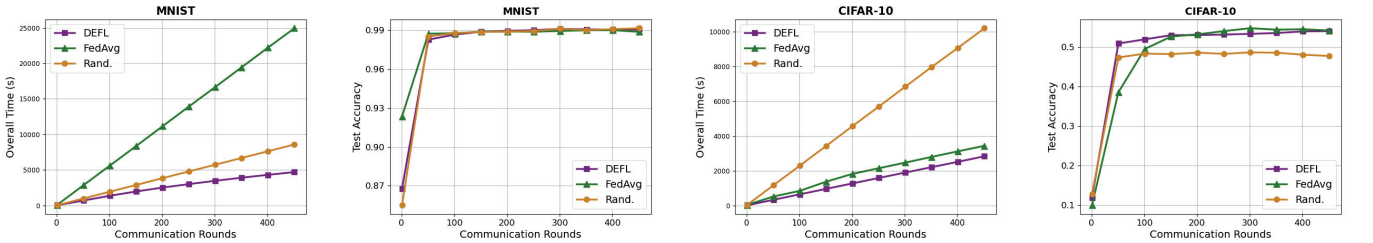
[1]Downloaded from: http://yann.lecun.com/exdb/mnist

[2]Downloaded from: http://www.cs.toronto.edu/~kriz/cifar.html



(a) Impact of ϵ on $\mathcal{T}$ and convergence. (b) Impact of b with the chosen ϵ. (c) Impact of θ with b^* and chosen ϵ. (d) Impact of θ, T_{cp} on H.

Fig. 1. Studying the impact of different parameters on the overall time and performance.



(a) Overall time vs. comm. rounds. (b) Test accuracy vs. comm. rounds. (c) Overall time vs. comm. rounds. (d) Test accuracy vs. comm. rounds.

Fig. 2. Performance evaluation of DEFL over MNIST and CIFAR-10 datasets.

Hence, DEFL can be useful in accelerating the FL process on mobile edge devices such as wearable devices.

VII. Conclusion

In this paper, we introduced a delay efficient FL mechanism suitable for mobile edge devices such as wearable devices, by studying the trade-off between wireless communication (to talk) and local computation (to work) with respect to the overall time. With careful consideration of this prevailing balance, we interpreted the effects of the learning model, wireless communication and hyper parameters in conjunction over the total time consumed. Guided by this theoretical model, we demonstrated the impact of these parameters through extensive simulations. Empirical evaluations have shown that DEFL can reduce the overall time delay while achieving high performance accuracy, implying that FL can be accommodated in delay-sensitive applications suitable for mobile devices.

References

[1] D. Reinsel, J. Gantz, and J. Rydning, "The Digitization of the World from Edge to Core," *IDC White Paper*, 2018.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Artificial Intelligence and Statistics*, Fort Lauderdale, FL, Apr 2017, pp. 1273–1282.

[3] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A Federated Transfer Learning Framework for Wearable Healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[4] A. Al-Shuwaili and A. Lawey, "Achieving Low-Latency Mobile Edge Computing by Uplink and Downlink Decoupled Access in HetNets," *arXiv preprint arXiv:1809.04717*, 2018.

[5] H. Trinh, P. Calyam, D. Chemodanov, S. Yao, Q. Lei, F. Gao, and K. Palaniappan, "Energy-aware mobile edge computing and routing for low-latency visual data processing," *IEEE Transactions on Multimedia*, vol. 20, no. 10, pp. 2562–2577, 2018.

[6] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated Learning: Strategies for Improving Communication Efficiency," in *NIPS Workshop on Private Multi-Party Machine Learning*, Barcelona, Spain, Dec 2016.

[7] A. Khaled, K. Mishchenko, and P. Richtárik, "Tighter Theory for Local SGD on Identical and Heterogeneous Data," in *The 23rd International Conference on Artificial Intelligence and Statistics (AISTATS 2020)*, Sicily, Italy, Jun 2020, pp. 4519–4529.

[8] N. H. Tran, W. Bao, A. Zomaya, M. N. Nguyen, and C. S. Hong, "Federated Learning over Wireless Networks: Optimization Model Design and Analysis," in *IEEE Conference on Computer Communications (INFOCOM)*. Paris, France: IEEE, Apr 2019, pp. 1387–1395.

[9] D. Shi, L. Li, R. Chen, P. Prakash, M. Pan, and Y. Fang, "Towards Energy Efficient Federated Learning over 5G+ Mobile Devices," *arXiv preprint arXiv:2101.04866*, 2021.

[10] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour," *arXiv preprint arXiv:1706.02677*, 2017.

[11] D. Rothchild, A. Panda, E. Ullah, N. Ivkin, I. Stoica, V. Braverman, J. Gonzalez, and R. Arora, "FetchSGD: Communication-Efficient Federated Learning with Sketching," in *Thirty-seventh International Conference on Machine Learning*, Virtual, Jul 2020.

[12] Y. Abe, H. Sasaki, S. Kato, K. Inoue, M. Edahiro, and M. Peres, "Power and Performance Characterization and Modeling of GPU-Accelerated Systems," in *2014 IEEE 28th International Parallel and Distributed Processing Symposium*, Phoenix, AZ, May 2014, pp. 113–122.

[13] L. Li, D. Shi, R. Hou, H. Li, M. Pan, and Z. Han, "To Talk or to Work: Flexible Communication Compression for Energy Efficient Federated Learning over Heterogeneous Mobile Edge Devices," in *IEEE International Conference on Computer Communications (INFOCOM'21)*, Virtual Conference, May 2021.

[14] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic Controlled Averaging for Federated Learning," in *Thirty-seventh International Conference on Machine Learning*, Virtual, Jul 2020.

[15] J. Konečnỳ, Z. Qu, and P. Richtárik, "Semi-stochastic Coordinate Descent," *Optimization Methods and Software*, vol. 32, no. 5, pp. 993–1005, 2017.