Adaptive Regularization and Resilient Estimation in Federated Learning

Md Palash Uddin, Yong Xiang, *Senior Member, IEEE*, Yao Zhao, Mumtaz Ali,Yushu Zhang, *Senior Member, IEEE*, Longxiang Gao, *Senior Member, IEEE*

Abstract—Federated Learning (FL) is an emerging research area that produces a globally trained model using numerous local users' data and maintains their privacy. Heterogeneous or non-Independent and Identically Distributed (non-IID) data affect the global model's convergence and, therefore, cause high communication costs. These are because traditional FL approaches often disregard an adaptive regularized objective for the user-side training and utilize conventional arithmetic mean on the locally trained models for the server-side aggregation. To alleviate these issues, we propose a novel FL scheme in this paper. In particular, we propose an adaptive regularization approach to add to the classical objective function of the users' local models during training and a resilient estimation approach to the locally trained models during aggregation. The adaptive regularization approach is derived using the users' local and global performance diversification while the resilient estimation scheme uses a modified geometric mean aggregation over the local models' parameters. We provide consolidated theoretical results and perform extensive experiments on the IID and non-IID settings of MNIST, CIFAR-10, and Shakespeare datasets with various deep networks. The results manifest that our FL scheme outperforms the state-of-the-art approaches in terms of communication speedup, test-set performance, training convergence stability, and resiliency against attacks.

Index Terms—Distributed Learning, Federated Learning, Parallel Optimization, Data Parallelism, Communication Efficiency, Adaptive Regularization, Resilient Aggregation.

1 Introduction

The rise in popularity of smartphones, tablets, and wearable users has resulted in an overwhelming of rich data, such as images taken and text inputted by users. On the one hand, there is a potential scope to improve user experience and empower intelligent programs (e.g., analysis of user's activity, analysis of user's sentiment, analysis of image content, clinical care analysis of diabetes and heart disease, burglary analysis in a smart residential area, etc.). On the other hand, it poses privacy concerns, where a user's personal information has to be kept concealed. In this scenario, the users can be prepared with additional data computations rather than direct sharing of the private raw data as they are usually built with large processing and storage resources. As such, Federated Learning (FL) has been recently introduced to meet this purpose [1], [2], [3], [4].

In FL, each user locally uses a stochastic learning method to learn from its data, and a coordinating server globally updates its model parameters based on an element-wise weighted Arithmetic Mean (AM) of the model parameters from all of the local users. Local training and global aggregation phases are replicated until a predetermined training convergence is achieved, illustrated in Fig. 1. In this way, a user's private data do not leave its device and the individual model parameters are not shown to other users [1], [2], [5], [6], [7]. Due to its privacy protection virtue, FL is being utilized as a successful privacy-preserving distributed-edge machine learning platform on a broad scale [8], [9], [10] and is likely to be implemented in several privacy-sensitive real-life applications [10], [11], [12].

Heterogeneous or non-Independent and Identically Distributed (non-IID) data issue among FL users is the most critical barrier to developing a decent global model toward real-life application development. Non-IID data is mainly divided into two categories: class imbalance and size imbalance. In FL, users' local data are often acquired based on their usage preferences. As an outcome, each user's local collection will not be fundamentally descriptive of all users' overall data distribution, resulting in the class imbalance issue. Likewise, the users acquire their data in different degrees using their applications or services. In this instance, the quantity of the gathered training data can differ dramatically from one user to the next, resulting in the size imbalance issue. Such non-IID data greatly affect the training progress of the local model and the aggregation of the global model. In particular, non-IID data create more strictness to the convergence of the training model and, therefore, demand more communication rounds between the users and the server to produce the final decent global model. Because the communication network can be

- *M. P. Uddin, Y. Xiang, and Y. Zhao are with the School of Information Technology, Deakin University, Geelong, VIC 3220, Australia. E-mail: {m.uddin, yong.xiang, cnr}@deakin.edu.au.*
- *M. Ali is with the School of Agricultural Computational and Environmental Sciences, University of Southern Queensland, Toowoomba, Queensland, Australia. E-mail: Mumtaz.Ali@usq.edu.au.*
- *Y. Zhang is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China. E-mail: yushu@nuaa.edu.cn.*
- *L. Gao is with Qilu University of Technology (Shandong Academy of Sciences) and Shandong Computer Science Center (National Supercomputer Center in Jinan), China. E-mail: gaolx@sdas.org.*
- *Y. Xiang is the corresponding author.*

- *This work was supported in part by the Australian Research Council under grant DP220100983.*

unreliable with limited upload rates, and the associated expenses (computation, management, etc.) are still high, the communication rounds should be reduced to a minimum. Note that although the FL and the classical data-center distributed machine learning scheme [13], [14] formulate a very close objective, traditional optimization approaches of data-center distributed machine learning [13], [14], [15], [16] are unworkable in FL setup as thoroughly analyzed in [1], [6], [17], [18]. The key reasons are that these techniques presume IID data partitioning and a very limited number of users (workers). As such, Federated Averaging (FedAvg) [1], the baseline FL algorithm, was proposed that performs the conventional Stochastic Gradient Descent (SGD) optimization to train the local model on the user-side, then applies the AM on the locally trained models' parameters to produce the aggregated global model on the server-side. Along with FedAvg, Federated Stochastic Variance Reduced Gradient (FedSVRG) [2] is considered as another baseline FL method that employs Stochastic Variance Reduced Gradient (SVRG) optimizer for local model training on the user-side and AM to aggregate the local models as the updated global model on the server-side. These two benchmark studies take a broad look at communication costs and non-IID data issues. Subsequently, Momentum FedAvg (MomFedAvg) was proposed in [19] that substitutes traditional Gradient Descent (GD) by the momentum GD for local model training. Federated Proximal, FedProx in short, was presented in [20] that adds a proximal term (regularizer) to the local objective function during local model training. Both MomFedAvg and FedProx utilize the same AM scheme of FedAvg for server-side aggregation. [21] presented Federated Matched Averaging (FedMatAvg) via improving the previous work [22] that keeps the traditional SGD-based local model training of FedAvg but proposes a layer-wise model averaging strategy instead of the coordinate-wise model averaging scheme of FedAvg for the global model aggregation. [23] recently proposed a Robust FedAvg (RFedAvg) technique that preserves the same FedAvg training procedure for the local models while trimming a little portion of the top and lowest weights at each model node before AM-based aggregation. However, these FL works are not formulated enough to alleviate the above FL difficulties significantly. The key reasons are they (i) do not consider an adaptive regularized objective function for local model training; (ii) utilize the classical AM scheme for the global model aggregation; and (iii) contribute to either local model training module (e.g., [2], [19], [20] etc.) or global model aggregation module (e.g., [21], [22], [23] etc.) at a time, rather than considering to both of them in parallel.

To address the above-discussed FL challenges (non-IID data and communication overhead), we propose an adaptive and resilient FL scheme in this paper, based on the following observations. (i) The regularization (proximal term) added to the local objective function during the training phase regulates the convergence of the global model, and (ii) the classical AM of the local models during the aggregation phase seizes the stable convergence of the global model. In specific, first, it is observed that adding a regularization term weighting by a regularization coefficient to the objective function usually leads to faster convergence of the deep models [24], [25]. However, an improperly fixed regularization coefficient can hinder the progress of faster convergence in the FL setup. As such, we propose to add an adaptive regularization term to the objective function of the local model in the FL training module that deals with non-IID data issues by restricting local updates such that they are closer to the initial (global) model. Secondly, due to the obvious heterogeneous or malicious data among the clients, the local models' training results can vary, and some of their weights can be outliers in terms of the overall training objective. The conventional AM aggregator can be prone to improper and irregular updates for aggregation. Therefore, we propose a resilient averaging scheme for the aggregation of the local models that minimizes the impact of imbalanced model updates. We obtain the theoretical results of the proposed local model training and global model aggregation schemes and conduct a series of experiments to assess them on both IID and non-IID settings of diverse datasets (MNIST, CIFAR-10, and Shakespeare). The results show that the developed FL approach outperforms the investigated techniques in terms of communication speedup, testing accuracy enhancement, and training convergence. The following are our most important technological contributions in this paper.

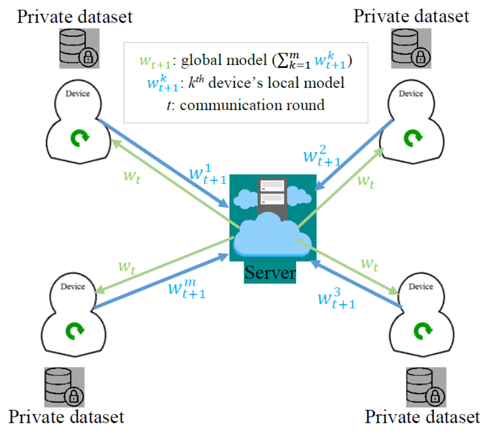


Fig. 1: Traditional FL scheme. The server selects m users at each round, t for training the global model, w_t on their private data. Each user uploads the updated local model, w_{t+1}^{k} to the server for deriving the new aggregated global model. Then, the server resends the aggregated model to other m users for retraining. Such training and aggregation phases are repeated until convergence.

- We propose a federated optimization approach that combines an adaptively regularized local model update scheme and a resilient aggregation scheme.
- We derive an optimal regularization coefficient tuning mechanism for local objective regularization during local model training.
- We present a novel resilient model aggregation strategy by minimizing the consequence of imbalanced local model updates.
- We provide a theoretical result analysis of the proposed local model training and global model aggregation schemes.

The remainder of this paper is structured in the following manner. Section 2 discusses the related FL works. After that, we explain our proposed FL approach in Section 3. In Section 4, the experimental results are examined and compared, and the work is summarized in Section 5.

2 Related Work

Although FL has emerged recently, it has attracted a lot of attention. We go through the most essential methods related to our work in this section. FedAvg, the baseline FL method, leverages SGD parallelism for local model training and coordinate-wise weighted AM for global model aggregation [1]. FedSVRG utilizes SVRG for local model updates and the same AM for global model aggregation, like FedAvg.

FedProx enhances FedAvg by adding a proximal term to the user's objective function with a fixed regularized coefficient but keeps the same AM of FedAvg for the aggregation [20]. In contrast to FedAvg, MomFedAvg increases model training speed by using momentum GD rather than conventional GD but uses the same aggregation scheme as FedAvg [19]. Astraea seeks to alleviate global and local imbalance by downsampling data augmentation and use of mediators but employs the FedAvg's AM for aggregation [26]. Although FedMatAvg [21] improves on [22] by averaging the local learned models layer-wise rather than coordinate-wise, as FedAvg does, it finally adopts the same direct AM scheme of FedAvg on the layers' parameters. Mutual Information directed FL (MIFL) updates local objective by adding a regularizer, derived for the optimal use of the user's existing local model to train the global model and picks up the top effective local models for aggregation [27]. However, it also accomplishes the direct AM of FedAvg on the selected local models' parameters. All of these approaches, however, (i) do not adaptively regularize the local objective function during user-side training; (ii) do not effectively minimize the impact of the outlier model updates during server-side aggregation; and (iii) develop either a local model training approach or a global model aggregation approach at a time, rather than taking both into account simultaneously.

On the other hand, a robust aggregation method, known as Robust Federated Aggregation (RFA) [28], is designed based on the geometric median to enhance the resiliency of FL in scenarios where a portion of the clients might transmit flawed updates to the server. The Krum approach, proposed in [29], compares FL clients' model outputs, excludes outliers, and selects the model closest to others as the key model for robust aggregation. Another aggregation rule, denoted as Trimmed Mean (TM), is introduced in [30], which modifies the version of the classical arithmetic mean rule, notable for selectively trimming certain values before performing the averaging process. Lastly, RFedAvg employs the same training method as FedAvg but trims a small portion of the largest and smallest weights at each model node before aggregation [23]. However, this direct removal of the top and lowest weights at each model node may potentially hinder the intended robust aggregation process.

3 Proposed Approach

3.1 Approach Overview

Local model update and global model aggregation are the two stages of our proposed FL scheme. In the first stage, we suggest updating the local models by adding an adaptive regularization term to the local subproblem, which we name "Adaptive Regularization-based Update" ("ARU" in short). This is to achieve an effective and stable learning convergence by optimally adding the regularization term to the local objective function at each training round. In the second stage, we propose to aggregate the local models' weights by applying a resilient estimation against the outlier updates, which we call "Resilient Estimation-based Aggregation" ("REA" in short). This is to obtain a robust aggregation by formulating an Estimated Geometric Mean (EGM) using the Inverse Hyperbolic Sine (IHS) transformation on the local model parameters. This way of aggregation leads to a more stable training convergence of the global model. We call the overall FL paradigm "ARU-REA" driven FL, whose generalized working flow and implementable framework view are depicted in Fig. 2 and Fig. 3, respectively.

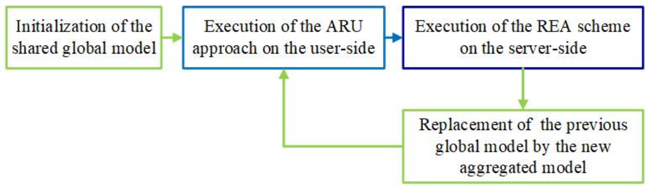

Initialization of the shared global model
Execution of the ARU approach on the user-side
Execution of the REA scheme on the server-side
Replacement of the previous global model by the new aggregated model

Fig. 2: An overview of the proposed ARU-REA FL scheme. To update the individual local model, each user executes the proposed ARU approach. The server then implements our REA approach to aggregate the locally trained models. The ARU and REA approaches are iterated until convergence is achieved.

3.2 Local Model Update

3.2.1 Objective Formulation

Our ARU scheme seeks to minimize the following cumulative objective function, $f(w)$ in the federated setting.

$$\min_{w\in\mathbb{R}^d} f(w) = \sum_{k=1}^{K} \frac{N_k}{N} l_k(w), \tag{1}$$

where $l_k(w)$ represents the k^{th} user's local objective function. Now, $l_k(w)$ is computed using Eq. (2) across the losses (denoted by $ls()$) of the samples i.e., how the predicted probabilities differ from the actual labels.

$$l_k(w) = \frac{1}{N_k} \sum_{i=1}^{N_k} ls(w, x(i)). \tag{2}$$

As defined in Eq. (1) and Eq. (2), the ARU scheme minimizes the cumulative loss function $f(w)$ by minimizing the weighted average of users' local losses $l_k(w)$ with the change of the model's weight, w.

3.2.2 Objective Regularization

The design of the local objective function is the key to the model convergence on different data behaviors among the users. In particular, we observe that individual users might differ considerably from one another in terms of diverse data characteristics, which limits the benefit of generalizing the objective (loss) across users. In the worst-case scenario,

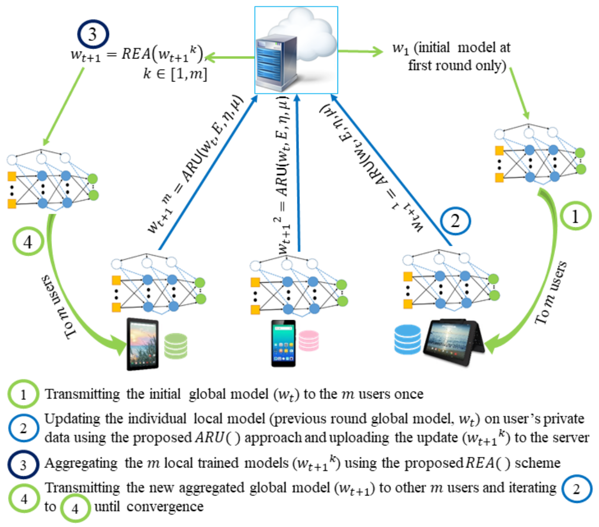

3
$w_{t+1} = REA(w_{t+1}{}^{k})$, $k \in [1,m]$
w_1 (initial model at first round only)
$w_{t+1}{}^{m} = ARU(w_t, E, \eta, \mu)$
$w_{t+1}{}^{2} = ARU(w_t, E, \eta, \mu)$
$w_{t+1}{}^{1} = ARU(w_t, E, \eta, \mu)$
2
1
4
To m users
To m users
1 Transmitting the initial global model (w_t) to the m users once
2 Updating the individual local model (previous round global model, w_t) on user's private data using the proposed $ARU(\,)$ approach and uploading the update ($w_{t+1}{}^{k}$) to the server
3 Aggregating the m local trained models ($w_{t+1}{}^{k}$) using the proposed $REA(\,)$ scheme
4 Transmitting the new aggregated global model (w_{t+1}) to other m users and iterating 2 to 4 until convergence

Fig. 3: An implementable framework view of the proposed ARU-REA FL paradigm. Each of the m selected users performs the proposed ARU method to train the server-sent global model over its N_k ($k \in [1, m]$) data samples at each round, t. After that, the server executes our REA scheme to generate a new aggregated global model using the uploaded local users' trained models over the N samples ($N = \sum_{k=1}^{m} N_k$). Other m users train the new aggregated model using ARU and the server accomplishes the aggregation of the newly uploaded locally trained models using REA. Such applications of ARU and REA are repeated until convergence.

sample distribution between users can be incredibly biased, necessitating the use of a significantly weighted regularized objective, which is difficult to tune by a classical approach. Therefore, as the model could favor divergence due to the severe data heterogeneity, regularization of the local objective by embedding a proximal term helps get rid of the divergence [20]. As such, we propose to add a proper proximal term, calculated using the previous and current model's gradients, to the k^{th} user's local objective function (denoted by l_{k_r}) for minimization as follows.

$$\underset{w \in \mathbb{R}^d}{min} l_{k_r}(w_{t+1}^k, w_t) = l_k(w_{t+1}^k) + \frac{\mu_k}{2}||w_{t+1}^k - w_t||^2, \quad (3)$$

where $\mu_k > 0$ is the regularizing coefficient that controls the impact of the proximal term during the loss minimization. This way of utilizing the regularization forces local model updates to be closer to the previous global model. In specific, the updated weights of the locally trained model could not be too far off from the weights of the preceding global model. However, larger μ_k gives high importance to the regularization but can decelerate the convergence while lower μ_k could not visualize any difference. As such, a proper μ_k can limit the trajectory of the iterates by confining the iterates to be closer to that of the global model, allowing for varying quantities of local updates while ensuring convergence. To do this, we propose to utilize the user's local model and the server's global model's progress to adaptively set a proper value to μ_k. Specifically, we evaluate the local model's progress after each local epoch over its previous progress and the global model's progress and set three different criteria to adaptively adjust μ_k. First, we look at the difference between the current training performance after every local epoch of the local model and its prior round's performance. If the user's current local loss exceeds its previous global round's or local epoch's loss (which indicates that the model is far from the convergence), then we increment μ_k to a somewhat large figure based on the current and prior loss diversification as follows.

$$\mu_{inc} = \mu_k + |\widetilde{l_c - l_p}| * \mu_k, \quad (4)$$

where l_c and l_p represent the user's current loss and previous loss respectively, and the operator $\widetilde{\ }$ refers to the normalizing to $[0, 1]$ operation.

Second, we utilize the user's P ($1 < P \leq 5$) local losses and the server's P global losses. If P prior local losses of the user and P previous global losses drop sequentially (which indicates that the model is converging at a decent pace), then we decrement μ_k to a reasonably tiny value depending on the current and previous rounds' performances, as specified below.

$$\mu_{dec} = \mu_k - |\widetilde{\bar{l_l} - \bar{l_g}}| * \mu_k, \quad (5)$$

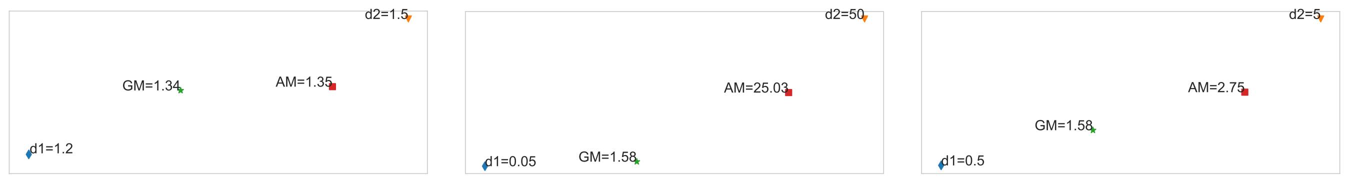
where l_l and l_g denote the local loss and global loss respectively, and the operator $\bar{\ }$ represents the AM operation. At last, if none of the aforementioned conditions apply (indicating that the model is convergent on average), we assign μ_k to a value that is halfway between them (Eq. (4) and Eq. (5)), as formulated below.

$$\mu_{ave} = \frac{\mu_{inc} + \mu_{dec}}{2}. \quad (6)$$

These criteria are for allocating a decent amount of regularization (μ_k) in the user's local objective function, based on the user's current and previous loss dissimilation, or the user's previous average loss and previous global average loss dissimilation, so that the local model does not drift too far from the previous global model and, thus, any potential divergence is avoided. We now present the pseudocode for the ARU scheme with the adaptive computing mechanism of μ_k in Algorithm 1. It is worth noting that the proximal term comes from the conventional machine learning paradigm [24], [25], and it is used to adaptively reformulate the k^{th} user's local objective. In comparison to traditional machine learning, the presented usage of the proximal term varies in that we aim to handle data heterogeneity with various local users in our ARU-REA FL paradigm.

3.2.3 Theoretical Result

Let ϕ be a representative of the local epochs (E) and be measured in the scale of [0, 1] of E. Then, given the l_{k_r} and ∇l_{k_r}, we can state that $w^\star$ is the ϕ-approximate solution of $\min_w l_{k_r}(w_{t+1}^k, w_t)$ if $||\nabla l_{k_r}(w^\star, w_t)|| \leq \phi||\nabla l_{k_r}(w_t, w_t)||$. Again, let $A(w_{t+1}^k) = \sqrt{\frac{\mathbb{E}_k[||\nabla l_k(w_{t+1}^k)||^2]}{||\nabla f(w_{t+1}^k)||^2}}$ for $||\nabla f(w_{t+1}^k)|| \neq 0$ and $\mathbb{E}_k[.]$ denotes the expectation over the users with $\frac{N_k}{N}$ and $\sum_{k=1}^{K} \frac{N_k}{N} = 1$. Then, we call $l_k(w_{t+1}^k)$ at w_{t+1}^k to be A-locally asymmetrical if $\mathbb{E}_k[||\nabla l_k(w_{t+1}^k)||^2] \leq ||\nabla f(w_{t+1}^k)||^2 A^2$. Now, we assume the following in context of FL [20]: (i) l_k is non-convex and L-Lipschitz smooth, and there exists $L\_ > 0$ such that $\nabla^2 l_k \succeq -L\_\mathbf{I}$ with



(a) (b) (c)

Fig. 4: AM vs GM. (a) AM and GM on two balanced inputs ($d1 = 1.2$ and $d2 = 1.5$). AM (=1.35) and GM (=1.34) behave similarly on such balanced inputs. (b) and (c): AM and GM on two sets of unbalanced but proportioned inputs ($d1 = 0.05$ and $d2 = 50$; and $d1 = 0.5$ and $d2 = 5$). AM produces 25.03 for the first set of $d1 = 0.05$ and $d2 = 50$ and 2.75 for the second set of $d1 = 0.5$; and $d1 = 5$ while GM produces 1.58 for both sets of inputs. As such, it is obvious that AM misbehaves with the unbalanced input sets but GM still offers a robust and resilient aggregation on them.

Algorithm 1 ARU Approach.

```
    Input: $w_t$ (server-sent global model at global round $t$ for $k^{th}$ user), $E$ (local epochs), $\eta$ (learning rate), $\mu$ (regularization coefficient)
    Output: $w^k_{t+1}$ (updated local model)
 1: procedure ARU ($w_t, E, \eta, \mu$) :        ▷ Run on $k^{th}$ FL user
 2:     Set user's local model, $w^k_{t+1}$ :=$w_t$
 3:     Set $\mu_k = \mu$
 4:     Bring previous local losses and assign them to $l_l$
 5:     Bring previous global losses and assign them to $l_g$
 6:     Assign last value of $l_l$ to $l_p$
 7:     ß:= Split local training set into batches of size $B$
 8:     for each local epoch $i$ from 1 to $E$ do
 9:         for each batch $b \in$ ß do
10:             $w := \nabla_{w^k_{t+1}}(ls(w^k_{t+1}, b) + \frac{\mu_k}{2}||w^k_{t+1} - w_t||^2)$
11:             $w^k_{t+1} := w^k_{t+1} - \eta w$
12:         end for
13:         Assign current epoch loss to $l_c$
14:         if $l_c$>$l_p$ then
15:             $\mu_k = \mu_{inc} = \mu_k + |\widetilde{l_c - l_p}| * \mu_k$
16:         else if $l_l$'s and $l_g$'s last $P$ values are in decrea-
                sing order then
17:             $\mu_k = \mu_{dec} = \mu_k - |\widetilde{\bar{l_l} - \bar{l_g}}| * \mu_k$
18:         else
19:             $\mu_{inc} = \mu_k + |\widetilde{l_c - l_p}| * \mu_k$
20:             $\mu_{dec} = \mu_k - |\widetilde{\bar{l_l} - \bar{l_g}}| * \mu_k$
21:             $\mu_k = \mu_{ave} = \frac{\mu_{inc}+\mu_{dec}}{2}$
22:         end if
23:         Update $l_p$ by $l_c$
24:     end for
25:     return $w^k_{t+1}$
26: end procedure
```

$\bar{\mu} = \mu - L\_ > 0.$, (ii) l_k is A-locally asymmetrical i.e., $A(w_t) \leq A$ and w_t is an unstationary solution, and (iii) A_ϵ exists for $\epsilon > 0$ such that $\{w| \, ||\nabla f(w^k_{t+1})||^2 > \epsilon\}$, $A(w^k_{t+1}) \leq A_\epsilon$. As such, the anticipated decrease of the global objective function can be determined as follows [20].

$$\mathbb{E}_U[f(w_{t+1})] \leq f(w_t) - \gamma||\nabla f(w_t)||^2, \tag{7}$$

where U are the chosen users at round t, $\gamma = (\frac{1}{\mu} - \frac{\phi A}{\mu} - \frac{A(1+\phi)\sqrt{2}}{\bar{\mu}\sqrt{K}} - \frac{LA(1+\phi)}{\mu\bar{\mu}} - \frac{L(1+\phi)^2A^2}{2\bar{\mu}^2} - \frac{LA^2(1+\phi)^2}{\bar{\mu}^2K}(2\sqrt{2K}+2)) > 0$ on the chosen K and ϕ, and the dynamically adjusted μ.

Algorithm 2 REA Approach.

```
    Input: $w^k_{t+1}$ ($m$ local models)
    Output: $w_{t+1}$ (aggregated model)
 1: procedure REA ($w^k_{t+1}$) :        ▷ Run on FL server
 2:     Assign total number of layers of $w^k_{t+1}$ to $t_l$
 3:     for each layer $l$ from 1 to $t_l$ do
 4:         Set total number of nodes (including bias) to $t_n$
 5:         for each node $n$ from 1 to $t_n$ do
 6:             for each model $k$ from 1 to $m$ do
 7:                 $w := w^k_{t+1}$
 8:                 $ws_{nl} := w_{nl}$
 9:                 $ws^{egm}_{nl,k} := \log(ws_{nl,k} + \sqrt{ws_{nl,k}{}^2 + 1})$
10:             end for
11:             $w^{egm}_{nl(t+1)} := \sum_{k=1}^{m} \frac{N_k}{N} ws^{egm}_{nl,k}$
12:             $w^{\star}_{nl(t+1)} := \frac{e^{w^{egm}_{nl(t+1)}} - e^{-w^{egm}_{nl(t+1)}}}{2}$
13:         end for
14:     end for
15:     Organize all $w^{\star}_{nl(t+1)}$ and set them to $w_{t+1}$
16:     return $w_{t+1}$
17: end procedure
```

Algorithm 3 ARU-REA Meta Algorithm.

```
    Input: $T_{acc}$ (targeted test accuracy)
    Output: $w_{t+1}$ (final global model)
 1: procedure ARU-REA($T_{acc}$) :
 2:     Initialize $w_1$, $\eta$, $\mu$ and $E$
 3:     for each communication round $t$ from 1 to $T$ do
 4:         if $T_{acc}$ has not been achieved then
 5:             Select $m := \lceil(C \times K)\rceil$ users, $0 < C \leq 1$
 6:             Set $U := m$ randomly picked users
 7:             for each user $k \in U$ in parallel do
 8:                 $w^k_{t+1}$ := ARU ($w_t, E, \eta, \mu$)
                                        ▷ Training of the local models
 9:             end for
10:             $w_{t+1}$ := REA($w^k_{t+1}$)
                                        ▷ Aggregation of the local models
11:         else
12:             $w_{t+1}$ is the final aggregated model
13:             break
14:         end if
15:     end for
16: end procedure
```

For the full proof, refer to Appendix A.1 [20]. The crucial stages consist of employing the locally asymmetrical assumption and our idea of ϕ-approximate for each subproblem while allowing for just m users to be active at each round. In particular, the last step introduces $\mathbb{E}_U$, an expectation on the selection of users, U, at round t.

3.3 Global Model Aggregation

The local models' training outcomes might differ because of the obvious heterogeneity of the data, and some of their updates may be outliers in terms of the overall training goal. As such, the AM used at the aggregation phase in the classical FL algorithms might be prone to such outlier updates and, therefore, can hamper the convergence of the global model. To better interpret, let $ws_{nl} = [w^1_{nl(t+1)}, w^2_{nl(t+1)}, ..., w^m_{nl(t+1)}]$ be the weight-vector at node n of layer l for aggregation in which $w^k_{nl(t+1)}$ represents the weight at node n of l^{th} layer for the k^{th} user. Then, the AM is applied to calculate the aggregated weight, $w^*_{nl(t+1)}$ on ws_{nl} as follows.

$$w^*_{nl(t+1)} = \sum_{k=1}^{m} \frac{N_k}{N} ws_{nl,k}. \tag{8}$$

From Eq. (8), it can be seen that the traditional AM treats every weight in ws_{nl} indiscriminately, which can, therefore, be vulnerable to erroneous and irregular updates. As a result, we propose a robust and resilient averaging strategy, called REA, for the aggregation of the local models that reduces the impact of imbalanced model updates. To do this, we propose to reformulate the classical Geometric Mean (GM) strategy, as defined in Eq. (9), for the node-wise aggregation.

$$w^{gm}_{nl(t+1)} = e(\frac{\sum_{k=1}^{m} \frac{N_k}{N} \log ws_{nl,k}}{\sum_{k=1}^{m} \frac{N_k}{N}}) \tag{9}$$

A typical pictorial analysis of the difference between AM and GM is illustrated in Fig. 4, where it can be seen that GM offers a more robust and resilient central tendency over AM on the imbalanced inputs during aggregation. However, the conventional GM cannot cope with the negative and zero values and the model updates can contain a mixture of positive, negative, and zero values. As such, we modify it, called EGM, to cope with any levels of positive, negative, and zero values using the IHS conversion [31] on ws_{nl}, as defined in Eq. (10).

$$ws^{egm}_{nl,k} = IHS(ws_{nl,k}) = \log(ws_{nl,k} + \sqrt{ws_{nl,k}{}^2 + 1}) \quad for\ 1 \le k \le m. \tag{10}$$

Now, the average of the $ws^{egm}_{nl,k}$ is accomplished using Eq. (11) and then the Hyperbolic Sine (HS) transformation (Eq. (12)) is applied on the averaged result to get the final aggregation value ($w^{\bigstar}_{nl(t+1)}$) in the original space.

$$w^{egm}_{nl(t+1)} = \sum_{k=1}^{m} \frac{N_k}{N} ws^{egm}_{nl,k}. \tag{11}$$

$$w^{\bigstar}_{nl(t+1)} = HS(w^{egm}_{nl(t+1)}) = \frac{e^{w^{egm}_{nl(t+1)}} - e^{-w^{egm}_{nl(t+1)}}}{2}. \tag{12}$$

We provide the pseudocode for our entire EGM-based REA scheme in Algorithm 2. The new aggregated global model, w_{t+1} is sent to users who are selected for immediate training using the ARU algorithm (Algorithm 1). New users are chosen at random as a tiny fraction of the entire user base, just like FedAvg. Then, the aggregation is accomplished via the REA approach. The ARU and REA operations are performed until the desired learning convergence is achieved, as illustrated in the ARU-REA FL meta-algorithm (Algorithm 3).

3.3.1 Theoretical Analysis

Assumption. Each chosen user transmits its learned model to the server for aggregation within the pre-defined timeframe.

Proposition. Let $w^{\bigstar}_{t+1}$ and w^*_{t+1} be the EGM and GM-based aggregated model at round t, respectively. Considering the aggregation output is in $conv\{w^1, w^2, ..., w^K\}$ for all $w^1, w^2, ..., w^K \in \mathbb{R}^d$ and the aforesaid assumption holds, the output (i.e., $w^{\bigstar}_{t+1}$) of REA (i) satisfies the following inequality: $w^{\bigstar}_{t+1} < w^*_{t+1}$; and (ii) assures that local users would not be drawn to various attractions that are far from the global model.

Proof. Let $\omega_k = \frac{N_k}{N} > 0$, $\omega = \sum_{k=1}^{m} \omega_k$, and x_k be the non-negative weight-vector at node n of layer l for aggregation using the classical AM and our proposed REA scheme for convenience. Then, the aggregated weight at node n of layer l using an AM-based aggregation scheme and our REA approach can be calculated in Eq. (13) and Eq. (14), respectively.

$$w^*_{nl(t+1)} = \sum_{k=1}^{m} \omega_k x_k. \tag{13}$$

$$w^{\bigstar}_{nl(t+1)} = e(\frac{\sum_{k=1}^{m} \omega_k \log x_k}{\sum_{k=1}^{m} \omega_k}) = \sqrt[\omega]{x_1^{\omega_1} x_2^{\omega_2} \cdots x_m^{\omega_m}} \tag{14}$$

The following can be assumed to be true since an x_k with $\omega_k = 0$ does not affect the inequality. All x_k must be equal for equality to hold. If they are not all equal, then it is still necessary to demonstrate strict inequality. As the natural logarithm is strictly concave, the finite form of Jensen's inequality and the functional equations of the natural logarithm indicate the following: $\log\left(\frac{\omega_1 x_1 + \cdots + \omega_m x_m}{\omega}\right) > \frac{\omega_1}{\omega} \log x_1 + \cdots + \frac{\omega_m}{\omega} \log x_m = \log \sqrt[\omega]{x_1^{\omega_1} x_2^{\omega_2} \cdots x_m^{\omega_m}}$. Now, due to the strict monotonic behavior of natural logarithm, we can write $\sqrt[\omega]{x_1^{\omega_1} x_2^{\omega_2} \cdots x_m^{\omega_m}} < \frac{\omega_1 x_1 + \cdots + \omega_m x_m}{\omega}$ i.e., $w^{\bigstar}_{t+1} < w^*_{t+1}$, which proves the Proposition (i).

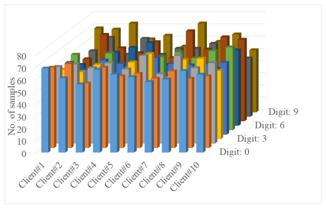
Now, it is noticeable that including a proximal component in the user's local objective function forces the local model update to be closer to the original (global) model [20]. In particular, the locally trained model's updated weights could not deviate too much from the weights of the preceding global model. As a result, the global model aggregated with the parameters of these local models stays closer to the prior global model, which is not the case when using no proximal component. The global model ($w^{\bigstar}_{t+1}$), which is obtained by using the reformulated EGM in the REA scheme, is comparable to the results of using this regularization term in the user's local objective function. Therefore, $w^{\bigstar}_{t+1}$ might constrain the local updates to be more similar to the original global model and aid in minimizing the effects of data heterogeneity. This establishes Proposition (ii) and, therefore, we can conclude that REA offers reliable stability of the convergence of the global model.

4 Experimental Results

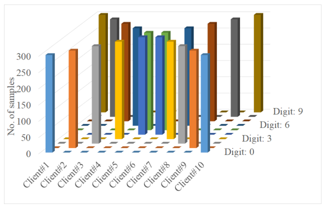
4.1 Experimental Setting

We choose to improve the usage of mobile users by creating high-quality global models for tasks like image classification and language modeling. To further understand the benefits of our proposed ARU-REA FL technique, we replicate the benchmark FedAvg [1] approach on the MNIST image classification and the Shakespeare (SP) language modeling challenge, and we conduct more experiments on the CIFAR-10 image classification task than FedAvg. We use three types of DNN models, identical to FedAvg, namely the Multi-Layer Perceptron (MLP), Convolutional Neural Network (CNN), and Long Short Term Memory (LSTM) neural networks. We apply our proposed scheme in three different manners: ARU (where the model update module of FedAvg is replaced with our proposed ARU scheme and the same aggregation strategy as FedAvg is maintained), REA (where the same model update strategy as FedAvg is held and the aggregation module of FedAvg is substituted with our proposed REA scheme), and ARU-REA (where both the model update and aggregation modules of FedAvg are substituted by our ARU and REA approaches, respectively). We carry out two types of experiments to demonstrate the advantage of the FL paradigm: (i) increasing user parallelism (increasing user participation from 10% to 100%), and (ii) increasing computation per user (varying number of local epochs and local batch size per user) and extensively make a comparison with FedAvg [1] and FedProx [20]. At last, we compare our proposed ARU-REA to the significant classical FL works to broaden the scope of the evaluation, and to the attack-robust FL approaches to demonstrate the resiliency of our method against attacks. Notice that Appendix A of the supplemental document has a full description of the local users and server resources utilized in the experiments.

Dataset and data distribution. MNIST is a typical dataset for recognizing 10 handwritten digits (0-9), whereas CIFAR-10 is a more complicated dataset for classifying 10 real-world vehicles and animals. The next dataset is SP's character prediction for language modeling. We take both IID and non-IID data distributions into account to demonstrate the validity of the FL techniques, as recommended in [1]. We do this by creating IID and non-IID MNIST, IID CIFAR-10, and non-IID SP datasets. We shamble the 60,000/50,000 training samples and assign 600/500 instances to each of the 100 users to prepare IID MNIST/IID CIFAR-10. In the case of non-IID MNIST, we split the training data into 200 chunks of size 300 after categorizing them depending on the number of classes (digits). Then, we allocate two chunks to each of the 100 users, ensuring that each user only receives samples from two classes. In Fig. 5, we show IID and non-IID MNIST data configurations for 10 users. The SP dataset for language modeling is built using *The Complete Works* of William Shakespeare. Each speaking position in each play is assigned to a user dataset, resulting in a spontaneous non-IID partition. The users with fewer than 10K data points are then filtered away, and 66 users are chosen at random. We use 80 percent of the data for training and combine the rest into a global test set, resulting in a training set of 1,117,274 characters and a test set of 118,967 characters. Table 1 contains significant information about the experimental datasets (MNIST, CIFAR-10, and SP).



(a)



(b)

Fig. 5: Data distribution for 10 typical users using the MNIST dataset. (a) IID partitioning, and (b) Non-IID partitioning.

TABLE 1: Dataset information.

Dataset	Training samples	Testing samples	Classes	Channels	Resolution or features
MNIST	60,000	10,000	10	1	28×28
CIFAR-10	50,000	10,000	10	3	32×32
SP	1,117,274	118,967	80	1	8

Model architecture. On the MNIST dataset, we effectuate (i) a two hidden layers MLP neural network (represented as 2NN) with a size of 784 - FC(200, 200, ReLU) - FC(200, 200, ReLU) - FC(200, 10, softmax) (parameters: 199,210); and (ii) a CNN with a structure 28×28 - CONV(5×5, 32, ReLU) - POOL(2×2, max) - CONV(5×5, 64, ReLU) - POOL(2×2, max) - FC(7*7*64, 128, ReLU) - FC(128, 10 softmax) (parameters: 454,922). For the CIFAR-10 dataset, we implement a CNN with an architecture of 32×32×3 - CONV(3×3, 32, ReLU) - POOL(2×2, max) - CONV(3×3, 64, ReLU) - POOL(2×2, max) - CONV(3×3, 64, ReLU) - POOL(2×2, max) - FC(3*3*64, ReLU) - FC(64, 10, softmax) (parameters: 122,570). On the SP dataset, we build a stacked character-level 1-layer LSTM of structure Encoder(80×8)-LSTM(8-256-1, RNN)-Decoder(256×80) (parameters: 293,584). The LSTM predicts every next character upon reading each character in a line from the SP dataset. The model receives a group of characters and encapsulates each character in an 8-dimensional space that it has learned. The encapsulated characters are then passed through the LSTM layer with 256 nodes, and the LSTM layer's result is finally sent to a softmax output layer with one node for each of the 80 characters.

Hyperparameter. The user fraction (C), local epochs (E), and local batch size (B) are the existing crucial FL scheme-centric hyperparameters, while the regularizing coefficient (μ) is the added local objective-centric hyperparameter in our proposed ARU-REA FL paradigm. We conduct experiments on C using distinct user percentages (10%-100%) of the overall user population and different combinations of E and B to contribute to the global learning stage, identical to FedAvg. In our ARU-REA FL scheme, we dynamically adjust μ from its initial value (0.01) for each selected local user after every local epoch. We assign η to 0.1 for images

TABLE 2: Impact of extending user fraction (C) over 2NN MNIST. The cell values show the highest degree of accuracy that the relevant techniques are capable of attaining as well as the number of communication rounds required to get the requisite test-set accuracy (i.e., maximum accuracy achieved by FedAvg for the specific experimental setup). The communication speedups are shown by the figures in parentheses.

Dataset	B	Method	Testing accuracy (%)				Communication round (speedup)			
			C				C			
			0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
IID MNIST	∞	FedAvg	85.90	85.96	85.90	85.91	1790	1786	1781	1778
		FedProx	88.13	88.10	88.11	701	681 (2.63×)	713 (2.50×)	691 (2.58×)	701 (2.54×)
		ARU	93.63	93.60	93.61	510	508 (3.52×)	517 (3.45×)	507 (3.51×)	510 (3.49×)
		REA	86.52	86.58	86.52	1368	1333 (1.34×)	1342 (1.33×)	1358 (1.31×)	1368 (1.30×)
		ARU-REA	95.38	95.35	95.36	354	351 (5.10×)	359 (4.97×)	346 (5.15×)	354 (5.02×)
	10	FedAvg	96.52	96.81	96.78	96.86	97	95	96	100
		FedProx	96.68	96.91	96.95	97.01	89 (1.09×)	90 (1.06×)	89 (1.08×)	86 (1.16×)
		ARU	96.88	97.11	97.15	97.22	74 (1.31×)	78 (1.22×)	72 (1.33×)	77 (1.30×)
		REA	97.06	97.31	97.38	97.27	69 (1.41×)	68 (1.40×)	61 (1.57×)	64 (1.56×)
		ARU-REA	97.71	97.86	97.92	97.92	51 (1.90×)	53 (1.79×)	52 (1.85×)	51 (1.96×)
Non-IID MNIST	∞	FedAvg	86.05	86.01	85.93	85.91	1789	1769	1775	1778
		FedProx	88.30	88.24	88.17	88.06	642 (2.79×)	601 (2.94×)	665 (2.67×)	701 (2.54×)
		ARU	93.81	93.74	93.67	93.56	328 (5.45×)	325 (5.44×)	446 (3.98×)	510 (3.49×)
		REA	86.67	86.63	86.55	86.53	1320 (1.36×)	1273 (1.39×)	1305 (1.36×)	1368 (1.30×)
		ARU-REA	95.55	95.49	95.42	95.31	285 (6.28×)	294 (6.02×)	347 (5.12×)	354 (5.02×)
	10	FedAvg	97.50	97.54	97.41	97.43	633	647	481	647
		FedProx	97.90	97.80	97.67	97.61	393 (1.61×)	404 (1.60×)	389 (1.24×)	385 (1.68×)
		ARU	97.98	97.88	97.82	97.84	371 (1.71×)	361 (1.79×)	287 (1.68×)	272 (2.38×)
		REA	98.07	98.05	97.94	97.92	329 (1.92×)	251 (2.58×)	195 (2.47×)	193 (3.35×)
		ARU-REA	98.71	98.65	98.58	98.57	215 (2.94×)	172 (3.76×)	165 (2.92×)	156 (4.15×)

TABLE 3: Effect of increasing computation (E and/or B) in each user. The value of $u = (NE)/(KB)$ indicates the expected number of local updates per CR. (a) Over IID and non-IID MNIST datasets using CNN. (b) over Non-IID SP dataset using LSTM.

Data distribution	*E, B, u*	Testing accuracy (%)					Communication round (speedup)				
		FedAvg	FedProx	ARU	REA	ARU-REA	FedAvg	FedProx	ARU	REA	ARU-REA
IID	1, ∞, 1	97.68	95.21	98.13	98.11	98.68	388	113 (-)	340 (1.14×)	360 (1.08×)	306 (1.27×)
	5, ∞, 5	98.49	98.49	98.54	98.66	98.74	364	372 (0.98×)	280 (1.30×)	279 (1.30×)	125 (2.91×)
	1, 50, 12	98.73	98.74	99.32	99.15	99.53	226	372 (0.61×)	129 (1.75×)	141 (1.60×)	93 (2.43×)
	20, ∞, 20	98.52	98.64	98.84	98.57	98.82	395	291 (1.36×)	179 (2.21×)	336 (1.18×)	167 (2.37×)
	1, 10, 60	98.66	98.82	99.02	98.89	99.20	75	70 (1.07×)	53 (1.42×)	62 (1.21×)	42 (1.79×)
	5, 50, 60	98.67	98.94	99.04	99.05	99.19	93	52 (1.79×)	47 (1.98×)	45 (2.06×)	30 (3.10×)
	20, 50, 240	98.69	98.88	99.08	99.11	99.27	67	51 (1.31×)	39 (1.72×)	34 (1.97×)	28 (2.39×)
	5, 10, 300	98.64	98.64	98.87	98.70	99.10	59	26 (2.27×)	20 (2.95×)	25 (2.36×)	14 (4.21×)
	20, 10, 1200	98.84	99.03	99.11	99.13	99.56	59	57 (1.04×)	37 (1.59×)	36 (1.64×)	13 (4.54×)
Non-IID	1, ∞, 1	98.53	97.76	98.96	98.95	99.87	1077	372 (-)	881 (1.22×)	833 (1.29×)	590 (1.83×)
	5, ∞, 5	98.35	98.71	99.02	99.70	99.95	1036	534 (1.94×)	374 (2.77×)	221 (4.67×)	203 (5.10×)
	1, 50, 12	97.54	97.71	98.92	98.76	99.24	1068	578 (1.85×)	141 (7.57×)	161 (6.63×)	112 (9.54×)
	20, ∞, 20	98.55	96.03	98.92	98.74	99.14	1077	152 (-)	542 (1.99×)	872 (1.24×)	371 (2.90×)
	1, 10, 60	96.66	95.83	96.83	96.85	97.42	384	134 (-)	284 (1.35×)	266 (1.44×)	100 (3.84×)
	5, 50, 60	98.65	98.75	98.95	99.08	99.28	385	315 (1.22×)	218 (1.77×)	151 (2.55×)	134 (2.87×)
	20, 50, 240	98.51	98.77	99.17	98.92	99.48	375	184 (2.04×)	89 (4.21×)	151 (2.48×)	71 (5.28×)
	5, 10, 300	95.59	95.71	96.08	96.19	96.21	375	288 (1.30×)	92 (4.07×)	165 (2.27×)	79 (4.75×)
	20, 10, 1200	89.69	98.97	99.15	98.85	99.54	41	39 (1.05×)	33 (1.24×)	35 (1.17×)	28 (1.46×)

(a)

E, B, u	Testing accuracy (%)					Communication round (speedup)				
	FedAvg	FedProx	ARU	REA	ARU-REA	FedAvg	FedProx	ARU	REA	ARU-REA
1, ∞ (500), 33.86	55.04	55.47	55.82	55.69	56.038	290	207 (1.40×)	165 (1.76×)	147 (1.97×)	127 (2.28×)
5, ∞ (500), 169.28	52.71	52.34	52.91	52.81	52.96	267	68(-)	213 (1.25×)	226 (1.18×)	101 (2.64×)
1, 50, 338.58	46.42	46.55	46.68	46.52	46.67	225	250 (0.9×)	168 (1.34×)	183 (1.23×)	121 (1.86×)
1, 10, 1692.84	47.22	44.09	47.32	47.42	47.52	225	16 (-)	183 (1.23×)	148 (1.52×)	114 (1.97×)
5, 50, 1692.84	46.36	46.48	47.06	46.46	46.61	143	95 (1.51×)	48 (2.98×)	50 (2.86×)	35 (4.09×)
5, 10, 8464.20	47.22	47.06	47.26	47.31	47.41	143	25 (-)	71 (2.01×)	61 (2.34×)	42 (3.40×)

(b)

datasets and 0.8 for language modeling dataset, based on the naive-strategy suggestion and few promising approaches [32], [21], [27] and [23]. We perform all experiments using Pytorch with SGD optimizer and classical cross-entropy loss function.

Evaluation metric. We assess the FL approaches in terms of training communication rounds to attain the specified test-set accuracy and training losses to produce a steady convergence behavior, similar to FedAvg. We investigate communication round reduction in particular to obtain the targeted test-set accuracy for demonstrating the alleviation of communication overhead for any data split (IID or non-IID); in the meanwhile, we tacitly alleviate the influence of heterogeneous (non-IID) data. As an example of the hurdles with heterogeneous data, the non-IID data arrangement poses an imbalance in data allocation among users by adding more difficulty to the training procedures and needing more training communication rounds than IID data to reach convergence. Providing faster learning convergence is a realistic way of proving that the heterogeneous (non-IID) data challenge has been handled. As a result, the decrease in communication rounds serves the goal of reducing data heterogeneity as well. Furthermore, the training loss graphs depict the global model's convergence behavior for the proposed and current FL approaches. Because the proposed approaches ARU and REA improve the local model training and global model aggregation processes of the benchmark FedAvg method, our proposed ARU-REA approach has no additional overall burden when compared to the standard FedAvg. As a result, our ARU-REA retains the convergence rate of up to $\mathcal{O}(\frac{1}{T})$ [33]. Nevertheless, according to our observations, the modified model training and aggregation in our ARU-REA scheme require a few additional operations, which may incur some negligible computing.

4.2 Increasing Parallelism

First, we investigate with the user fraction C, which establishes the level of multi-user parallelism for our ARU-REA technique and its two variations (ARU and REA), and the FedAvg and FedProx benchmark methods. For 2NN MNIST, we run 1800 communication rounds on both IID and non-IID datasets with B=∞ (full local batch size, i.e., 600 samples per user) and $E = 1$, and 100 and 700 communication rounds on IID and non-IID data settings, respectively, with B=10 and $E = 1$. For CNN MNIST, we run 400 and 1100 communication rounds on IID and non-IID datasets, respectively, with B=∞ and $E = 5$, and 100 and 400 communication rounds on IID and non-IID data settings, with B=10 and $E = 5$. The total number of communication rounds required to obtain the desired test-set accuracy (i.e., FedAvg's highest accuracy possible under a given experimental configuration) is then noted. Table 2 shows the quantitative results of expanding C in terms of test-set accuracies and communication speedups for 2NN MNIST, while the results using CNN MNIST are provided in Table 1 of Appendix B of the supplemental document. Fig. 6 shows the corresponding test-set accuracies vs communication rounds plots. Additionally, the training losses versus communication rounds charts in Fig. 7 show how the global model convergence behaves. In Appendix B of the

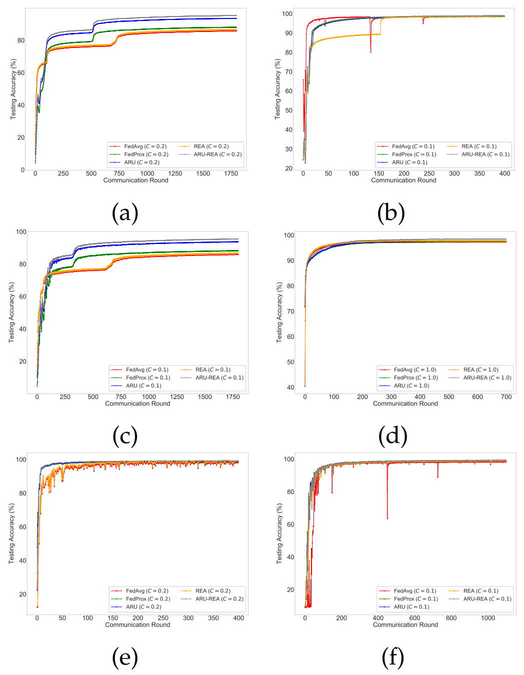


Fig. 6: Test-set accuracies versus communication rounds on increasing user parallelism. (a) $C = 20$ with $B = \infty$ over 2NN IID MNIST. (b) $C = 10$ with $B = \infty$ over CNN IID MNIST. (c) $C = 10$ with $B = \infty$ over 2NN non-IID MNIST. (d) $C = 100$ with $B = 10$ over 2NN non-IID MNIST. (e) $C = 20$ with $B = 10$ over CNN non-IID MNIST. (f) $C = 10$ with $B = \infty$ over CNN non-IID MNIST.

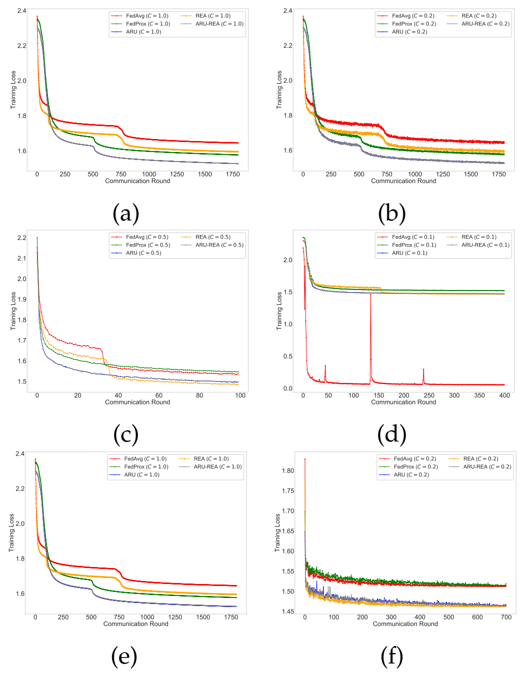


Fig. 7: Training losses versus communication rounds on increasing user parallelism. (a) $C = 100$ with $B = \infty$ over 2NN IID MNIST. (b) $C = 20$ with $B = \infty$ over 2NN IID MNIST. (c) $C = 50$ with $B = 10$ over 2NN IID MNIST. (d) $C = 10$ with $B = \infty$ over CNN IID MNIST. (e) $C = 100$ with $B = \infty$ over 2NN non-IID MNIST. (f) $C = 20$ with $B = 10$ over 2NN non-IID MNIST.

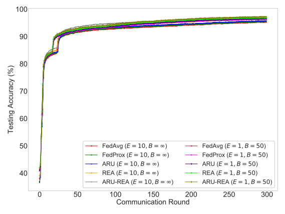
supplemental document, there are additional result graphs (testing accuracies and training losses vs communication rounds plots).

It is obvious that our proposed ARU-REA scheme and its two variations: ARU and REA perform better than the traditional FedAvg and FedProx in terms of lowering communication rounds, improving test-set accuracy, and better stabilizing the convergence of the global model for both large (i.e., B=∞) and small (i.e., B=10) batch sizes over the varied C. In particular, the proposed ARU-REA obtains communication speedups using 2NN and CNN over the IID MNIST dataset of up to 5.15× and 5.82×, respectively, as well as 6.28× and 4.40×, respectively, over the non–IID MNIST dataset. In light of these results, we set C to 0.1 for the remaining MNIST and CIFAR-10 experiments since it reaches a good level of parallelism and strike a nice balance between convergence and computation. However, we choose $C = 1$ for the SP experiments to permit the highest parallelism.

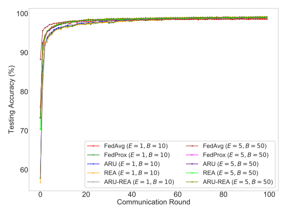
4.3 Increasing Computation Per User

In this experiment, we raise the number of SGD operations ($u = (\mathbb{E}[N_k]/B)E = (NE)/(KB)$ specifies the amount of SGD updates) per user at every communication round and set C to 0.1 for the MNIST dataset and 1.0 for the SP dataset. To do this, we either increase E, decrease B, or do both. For IID MNIST, we execute more communication rounds (400 for CNN and 1800 for 2NN) on the first four lower values of u than its other values (100 for both CNN and 2NN). For non-IID MNIST, the number of communication rounds on the first four u values is 1100 for CNN and 1800 for 2NN, while 400 for CNN and 1000 for 2NN on the remaining values of u. For the non-IID SP dataset, we perform 300 global rounds for the first three u values and 150 for its remaining values.

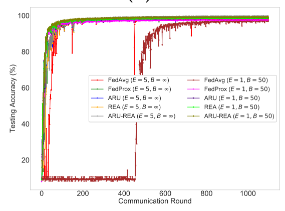
In Table 3, we present the quantitative communication speedups and test-set accuracies using CNN MNIST and LSTM SP, while the results using 2NN MNIST are provided in Table 2 of Appendix C of the supplemental document. We present the test-set accuracies vs communication rounds plots in Fig. 8, and the convergence behavior of the global model using the training losses vs communication rounds in Fig. 9. In Appendix C of the supplemental document, additional testing accuracy, and training loss plots are provided. The results show how altering B and E to boost SGD updates u per user is advantageous in reducing communication costs and stabilizing the convergence of the global model. In particular, the communication speedups realized by our ARU-REA approach are up to 4.61× and 5.56× over IID and non-IID MNIST, respectively, using the 2NN, and 4.54× and 9.54× over IID and non-IID MNIST, respectively, using the CNN. The communication speedup using ARU-REA for the LSTM non-IID SP experiment is up to 4.09×. Additionally, with rising u, ARU and REA separately beat the baselines FedAvg and FedProx. It can now be determined that our proposed ARU-REA has a good possibility of being implemented in real-world applications based on these encouraging communication speedups and accuracy enhancements.



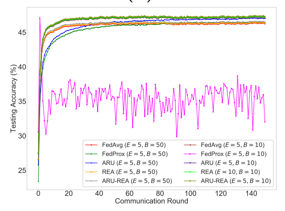
(a)



(b)

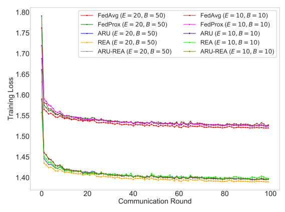


(c)

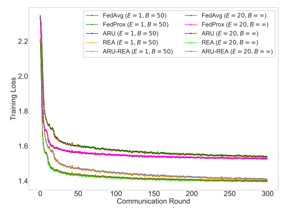


(d)

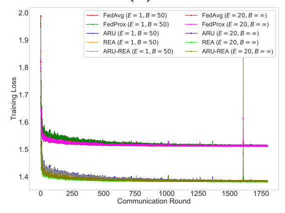
Fig. 8: Test-set accuracies versus communication rounds on extending computation per user. (a) Over 2NN IID MNIST. (b) Over CNN IID MNIST. (c) Over CNN non-IID MNIST. (d) Over LSTM non-IID SP.



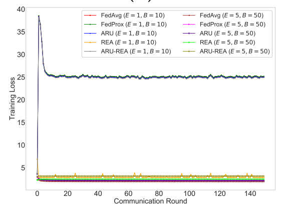
(a)



(b)

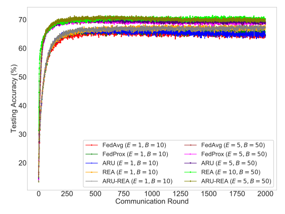


(c)

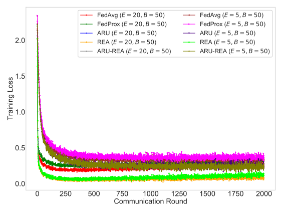


(d)

Fig. 9: Training losses versus communication rounds on extending computation per user. (a) Over 2NN IID MNIST. (b) Over 2NN IID MNIST. (c) Over 2NN non-IID MNIST. (d) Over LSTM non-IID SP.



(a)



(b)

Fig. 10: Extending computation per user on CIFAR-10. (a) Testing accuracies. (b) Training losses.

TABLE 4: Numbers of communication rounds required to reach a targeted test-set accuracy using the existing FL methods and our proposed ARU-REA over CNN MNIST (IID and non-IID).

Method	Testing accuracy (%)				Communication round (speedup)				Training wall-time (seconds)			
	E, B				*E, B*				*E, B*			
	1, 50		5, 10		1, 50		5, 10		1, 50		5, 10	
	IID	Non-IID	IID	Non-IID	IID	Non-IID	IID	Non-IID	IID	Non-IID	IID	Non-IID
FedAvg [1]	98.73	97.54	98.64	95.59	226	1068	59	375	1912	19892	2792	39610
FedSVRG [2]	98.75	97.59	98.69	95.02	180 (1.26×)	1250 (0.85×)	75 (0.79×)	250 (1.50×)	12501	22506	17808	96490
FedProx [20]	98.74	97.71	98.64	95.71	372 (0.61×)	578 (1.85×)	26 (2.27×)	288 (1.30×)	2508	36414	3259	46597
MomFedAvg [19]	98.05	97.59	98.20	94.59	219 (1.03×)	976 (1.09×)	65 (0.91×)	328 (1.14×)	2149	22712	3178	42789
FedMatAvg [21]	97.26	96.40	98.49	94.54	248 (0.91×)	1080 (0.99×)	74 (0.80×)	346 (1.08×)	17985	36971	24789	67950
MIFL [27]	98.71	97.62	98.66	94.69	203 (1.11×)	997 (1.07×)	53 (1.11×)	284 (1.32×)	2807	39741	3549	47923
ARU	99.32	98.92	98.87	96.08	129 (1.75×)	141 (7.57×)	20 (2.95×)	92 (4.07×)	2263	24125	2945	41258
REA	99.15	98.76	98.70	96.19	141 (1.60×)	161 (6.63×)	25 (2.36×)	165 (2.27×)	2052	20147	2895	41023
ARU-REA	99.53	99.24	99.10	96.21	93 (2.43×)	112 (9.54×)	14 (4.21×)	79 (4.75×)	2563	24785	3690	45623

TABLE 5: Test-set accuracy using our proposed ARU-REA and the existing robust FL methods for different percentages of attacks over 2NN IID MNIST.

Method	Testing accuracy (%)		
	Attack (%)		
	10	20	50
FedAvg [1] without Byzantine Attack	96.86		
FedAvg [1] with Byzantine Attack	84.87	80.98	78.91
RFA [28]	93.96	91.87	90.59
Krum [29]	92.89	91.16	90.12
TM [30]	92.88	90.94	89.59
RFedAvg [23]	93.05	91.98	90.47
ARU-REA	94.15	92.36	91.12

4.4 Experiment on CIFAR-10

After achieving notable results on the MNIST digit recognition dataset and the SP language modeling dataset, we apply our proposed ARU-REA approach to the complex CIFAR-10 image classification dataset. Compared to FedAvg, we take more combinations of E and B into account to determine how they regulate the test-set performance and communication overhead. For the first four u values, we perform 4,000 global rounds, and for the remaining u values, 2000. The test-set accuracies and corresponding communication speedups are summarised in Table 3 of Appendix C of the supplemental document. However, the test-set accuracies and training losses are shown in Fig. 10 while additional result graphs are provided in Appendix C of the supplemental document. These findings show that ARU-REA offers up to 10.54× communication speedups to reach the required test-set accuracies and improves convergence stabilization. ARU and REA also outperform FedAvg and FedProx in terms of communication speedup, test-set accuracy improvement, and stability of training loss for the global model.

4.5 Extensive Comparison

In addition to the previous experiments and results on our proposed ARU-REA, along with its two variants, and the vanilla FedAvg [1] and FedProx [20] on three different datasets (MNIST, CIFAR-10, and SP), we now extend the evaluation of the ARU-REA approach by comparing it with other four promising FL techniques: FedSVRG [2], MomFedAvg [19], FedMatAvg [21], and MIFL [27]. We still keep FedAvg and FedProx as the standard FL methods in this comparative study. For FedSVRG, we set $C = 1$ for all datasets as this technique necessitates complete user participation to compute the whole gradient at each training round. We set B and E to {1, 5, 10} and {10, 50}, respectively. The communication speedups needed to achieve the specified test-set accuracies using CNN MNIST are displayed in Table 4 while the results using 2NN MNIST, CNN CIFAR, and LSTM SP are provided in Table 4 of Appendix D of the supplemental document. We also provide the total wall-clock training times for the proposed and compared methods in these result tables using HPC system 1. As can be shown, our proposed ARU-REA paradigm and its two variants greatly outperform the investigated approaches for all three datasets in terms of communication rounds reduction and test-set accuracy improvement. When considering training time, we observe that our ARU and REA individually require nearly the same amount of training time as FedAvg. In contrast, the other investigated methods demand significantly more training time than FedAvg. Furthermore, our ARU-REA necessitates less training time than FedSVRG, FedProx, FedMatAvg, and MIFL, and only slightly more time than the MomFedAvg method, with the difference being negligible.

4.6 Robustness-Accuracy Tradeoff

Lastly, to validate the robustness of our proposed ARU-REA approach against Byzantine attacks, we compare our method with four potential robust FL approaches: RFA [28], Krum [29], TM [30], and RFedAvg [23]. As before, we use the *FedAvg method without attacks* as the baseline [28], [30], and we also consider FedAvg with attacks as another comparative method. In this case, we test the label-flipping attack, which is the most common Byzantine attack used in FL [34]. To generate the label-flipped local datasets, we follow the symmetric approach [35], where the original class label is flipped to any wrong class label with equal probability. Specifically, we consider flipping different proportions (10%, 20%, and 50%) of the labels in clients' local training datasets to the wrong labels while keeping the test dataset constant to observe the model's robustness. We conduct four different experiments using all previously mentioned models and datasets as follows: (i) 2NN IID MNIST with E=1, B=10, and C=1.0 for 100 communication rounds; (ii) CNN non-IID MNIST with E=5, B=10, and C=1.0 for 400 communication rounds; (iii) CNN IID CIFAR with E=5, B=10, and C=1.0 for 200 communication rounds; and (iv) LSTM non-IID Shakespeare with E=1, $B = \infty$ (500), and C=1.0 for 300 communication rounds. We present the results for the first type of experiment in Table 5, and the results for the other types of experiments can be found in Table 5, Table 6, and Table 7 of Appendix E in the supplemental document.

From these results, it is evident that our proposed ARU-REA approach offers better robustness by producing higher accuracy in the presence of attacks compared to the investigated robust FL algorithms while performing almost the same as the baseline *FedAvg without attacks*. This phenomenon underscores a critical aspect of our work—the ability to strike a balance between achieving high accuracy and maintaining robustness, even when subjected to malicious attacks. Furthermore, our approach consistently outperforms existing approaches as the level of attacks increases, thereby providing a compelling illustration of the accuracy-robustness tradeoff inherent in our FL framework. As the intensity of attacks rises, our method not only sustains its accuracy but also exhibits a remarkable capacity to adapt and resist the detrimental effects of malicious clients. This adaptability and resilience, demonstrated through our experiments, reaffirm the effectiveness of our approach in mitigating the impact of attacks while upholding accuracy.

5 Conclusion

This paper presents an adaptive regularized and outlier resilient FL paradigm. First, we have provided an adaptive local model method considering the dynamic regularization of the federated objective function at each local training round. The final local models that are updated in this way aid in the rapid and steady convergence of the global model and improve generalization performance during training on the non-IID data. Secondly, we have provided a resilient model aggregation approach considering the impact-minimization of the outlier weights from the locally trained models at each round of global aggregation. Along with the adaptive model update scheme, this approach of aggregating local models increases training convergence to improve the generalization capabilities of the global model. To prove the superiority of our FL technique, we have provided theoretical results and conducted extensive experiments on the IID and non-IID data partitioning from three different datasets in varied contexts. The findings manifest that our FL scheme outperforms the state-of-the-art FL approaches in reducing communication overhead, increasing test-set accuracy, effectively stabilizing the convergence of the global model, and producing better robustness against attacks.

From the results and findings, it can be inferred that the adaptive regularization-based local model update and resilient estimation-based global model aggregation could be a potential research area in the FL sector. Additionally, as in the baseline FL framework, the use of additional privacy-preserving methods, such as differential privacy [36], encryption strategy [37], [38], and secure multi-party computation [39] in the ARU-REA paradigm is still on the rise.

References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, (Florida, USA), 2017.

[2] J. Konecny, H. B. McMahan, and P. Richtarik, "Federated Optimization: Distributed Machine Learning for On-Device Intelligence," in *Computing Research Repository (CoRR)*, vol. Computing Research Repository (CoRR), 2016.

[3] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated Multi-Task Learning," in *International Conference on Neural Information Processing Systems (NIPS 2017)*, (CA, USA), 2017.

[4] Z. Ma, J. Ma, Y. Miao, X. Liu, K.-K. R. Choo, and R. Deng, "Pocket diagnosis: Secure federated learning against poisoning attack in the cloud," *IEEE Transactions on Services Computing*, pp. 1–13, 2021.

[5] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical Secure Aggregation for Federated Learning on User-Held Data," in *International Conference on Neural Information Processing Systems (NIPS 2016)*, (Barcelona, Spain), 2016.

[6] J. Konecny, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated Learning: Strategies for Improving Communication Efficiency," in *International Conference on Neural Information Processing Systems (NIPS 2016)*, (Barcelona, Spain), 2016.

[7] M. P. Uddin, Y. Xiang, X. Lu, J. Yearwood, and L. Gao, "Federated learning via disentangled information bottleneck," *IEEE Transactions on Services Computing*, pp. 1–14, 2022.

[8] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, H. B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards Federated Learning at Scale: System Design," in *2nd SysML Conference*, (CA, USA), 2019.

[9] W. A. Group, "Federated Learning White Paper V1.0," tech. rep., 2018.

[10] L. Huang, A. L. Shea, H. Qian, A. Masurkar, H. Deng, and D. Liu, "Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records," *Journal of Biomedical Informatics*, vol. 99, Nov. 2019.

[11] Z. Li, H. Zhou, T. Zhou, H. Yu, Z. Xu, and G. Sun, "ESync: Accelerating Intra-Domain Federated Learning in Heterogeneous Data Centers," *IEEE Transactions on Services Computing*, 2020.

[12] S. Truex, L. Liu, M. E. Gursoy, L. Yu, and W. Wei, "Demystifying Membership Inference Attacks in Machine Learning as a Service," *IEEE Transactions on Services Computing*, 2019.

[13] R. McDonald, K. Hall, and G. Mann, "Distributed training strategies for the structured perceptron," in *HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, (Los Angeles, California), pp. 456–464, 2010.

[14] J. Dean, G. S. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Y. Ng, "Large Scale Distributed Deep Networks," in *International Conference on Neural Information Processing Systems (NIPS 2012)*, (Lake Tahoe, Nevada), 2012.

[15] D. Povey, X. Zhang, and S. Khudanpur, "Parallel training of Deep Neural Networks with Natural Gradient and Parameter Averaging," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, (Reykjavik, Iceland), 2014.

[16] S. Zhang, A. Choromanska, and Y. LeCun, "Deep learning with Elastic Averaging SGD," in *International Conference on Neural Information Processing Systems (NIPS 2015)*, (Montreal, Canada), 2015.

[17] Y. Zhou, Q. Ye, and J. Lv, "Communication-efficient federated learning with compensated overlap-fedavg," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 1, pp. 192–205, 2022.

[18] A. Nilsson, S. Smith, G. Ulm, E. Gustavsson, and M. Jirstrand, "A Performance Evaluation of Federated Learning Algorithms," in *Second Workshop on Distributed Infrastructures for Deep Learning*, (Rennes, France), 2018.

[19] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating Federated Learning via Momentum Gradient Descent," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 8, pp. 1754–1766, 2020.

[20] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated Optimization in Heterogeneous Networks," in *MLSys Conference*, (TX, USA), 2020.

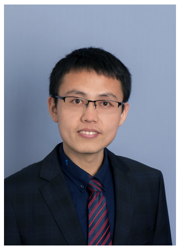
[21] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated Learning with Matched Averaging," in *International Conference on Learning Representations*, (Addis Ababa, Ethiopia), 2020.
[22] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, T. N. Hoang, and Y. Khazaeni, "Bayesian Nonparametric Federated Learning of Neural Networks," in *International Conference on Machine Learning*, (California, USA), 2019.
[23] M. P. Uddin, Y. Xiang, J. Yearwood, and L. Gao, "Robust Federated Averaging via Outlier Pruning," *IEEE Signal Processing Letters*, 2021.
[24] M. Li, T. Zhang, Y. Chen, and A. J. Smola, "Efficient mini-batch training for stochastic optimization," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 661–670, 2014.
[25] Z. Allen-Zhu, "How to make the gradients small stochastically: Even faster convex and nonconvex sgd," *Advances in Neural Information Processing Systems*, vol. 31, 2018.
[26] M. Duan, D. Liu, X. Chen, R. Liu, Y. Tan, and L. Liang, "Self-Balancing Federated Learning With Global Imbalanced Data in Mobile Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 1, pp. 59–71, 2020.
[27] M. P. Uddin, Y. Xiang, X. Lu, J. Yearwood, and L. Gao, "Mutual Information Driven Federated Learning," *IEEE Transactions on Parallel and Distributed Systems*, 2020.
[28] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1142–1154, 2022.
[29] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.
[30] T. Wang, Z. Zheng, and F. Lin, "Federated learning framew ork based on trimmed mean aggregation rules," *Available at SSRN 4181353*, 2022.
[31] J. B. Burbidge, L. Magee, and A. L. Robb, "Alternative transformations to handle extreme values of the dependent variable," *Journal of the American Statistical Association*, vol. 83, no. 401, pp. 123–127, 1988.
[32] H. Zhu and Y. Jin, "Multi-Objective Evolutionary Federated Learning," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2019.
[33] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the Convergence of FedAvg on Non-IID Data," in *Eighth International Conference on Learning Representations*, (Addis Ababa, Ethiopia), 2020.
[34] X. Fang and M. Ye, "Robust federated learning with noisy and heterogeneous clients," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10062–10071, 2022.
[35] B. van Rooyen, A. Menon, and R. C. Williamson, "Learning with symmetric label noise: The importance of being unhinged," in *Advances in Neural Information Processing Systems*, vol. 28, 2015.
[36] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Differentially Private Asynchronous Federated Learning for Mobile Edge Computing in Urban Informatics," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2134–2143, 2020.
[37] L. Lyu, J. Yu, K. Nandakumar, Y. Li, X. Ma, J. Jin, H. Yu, and K. S. Ng, "Towards Fair and Privacy-Preserving Federated Deep Models," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 11, pp. 2524–2541, 2020.
[38] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6532–6542, 2020.
[39] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical Secure Aggregation for Privacy-Preserving Machine Learning," in *Conference on Computer and Communications Security*, (Texas, USA), 2017.

Md Palash Uddin received a Ph.D. degree in Information Technology from Deakin University, Australia in 2023. He also received a B. Sc. degree in Computer Science and Engineering from Hajee Mohammad Danesh Science and Technology University (HSTU), Bangladesh, and an M. Sc. degree in Computer Science and Engineering from Rajshahi University of Engineering & Technology, Bangladesh. He is currently working as a Postdoctoral research fellow at the School of Information Technology, Deakin University, Australia. He is also an academic faculty member at HSTU, Bangladesh. His research interests include machine learning, federated learning, blockchain, and remote sensing image analysis.

Yong Xiang received a Ph.D. degree in Electrical and Electronic Engineering from The University of Melbourne, Australia. He is a Professor at the School of Information Technology, Deakin University, Australia. His research interests include distributed computing, cybersecurity and privacy, machine learning and AI, and communications technologies. He has published 7 monographs, over 220 refereed journal articles, and over 100 conference papers in these areas. Professor Xiang is the Senior Area Editor of IEEE Signal Processing Letters, the Associate Editor of IEEE Communications Surveys and Tutorials, and the Associate Editor of Computer Standards and Interfaces. He was the Associate Editor of IEEE Signal Processing Letters and IEEE Access, and the Guest Editor of IEEE Transactions on Industrial Informatics, IEEE Multimedia, etc. He has served as Honorary Chair, General Chair, Program Chair, TPC Chair, Symposium Chair and Track Chair for many conferences, and was invited to give keynotes at numerous international conferences.

Yao Zhao is currently pursuing the Ph.D. degree at the School of Information Technology, Deakin University. She has completed an M.S. degree in Computer Science and Technology at Nanjing University of Aeronautics and Astronautics in 2019. She worked for Suning Finance as the blockchain product manager. Besides, she used to be a blockchain developer in Dazhong News Group in 2020. Her research direction includes blockchain, IoT, and edge computing.

Mumtaz Ali received the MSc and MPhil degrees in Mathematics from $Quaid-i-azam$ University, Islamabad, Pakistan, in 2012 and 2014, and the Ph.D. degree in Data Science from the University of Southern Queensland, QLD, Australia, in 2020. He held a Postdoctoral Research Fellow position in Data Science and Alfred Deakin Postdoctoral Research Fellow with the School of IT, Deakin University, VIC, Australia. He is currently a Lecturer with the UniSQ College, University of Southern Queensland, QLD, Australia. He is also an adjunct professor at the University of Prince Edward Island, Canada. His research interests include Machine learning, Deep learning, Data Science, AI, and Decision Support Systems.



Yushu Zhang (SM'23) received the B.A. degree in information and computing science from North University of China, Taiyuan, China, 2010, and the Ph.D. degree in computer science from Chongqing University, Chongqing, China, in 2014. He held various research positions with the City University of Hong Kong, Southwest University, University of Macau, and Deakin University. He is currently a Professor at the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China. His research interests include multimedia processing and security, artificial intelligence, and blockchain. He is an Associate Editor of Signal Processing and Information Sciences.

Longxiang Gao (SM17) received his PhD in Computer Science from Deakin University, Australia. He is currently a Professor at the Qilu University of Technology and Shandong Computer Science Center in China. He was a Senior Lecturer at the School of Information Technology, Deakin University, and a post-doctoral research fellow at IBM Research & Development, Australia. His research interests include Fog/Edge computing, Blockchain, data analysis, and privacy protection. Dr. Gao has over 100 publications, including patents, monographs, book chapters, journals, and conference papers. Some of his publications have been published in top venues, such as IEEE TMC, IEEE TPDS, IEEE TSC, IEEE IoTJ, IEEE TDSC, IEEE TVT, IEEE TCSS, IEEE TII, and IEEE TNSE. He has been Chief Investigator (CI) for more than 20 research projects (the total awarded amount is over $5million), from pure research projects to contracted industry research. He is a Senior Member of IEEE.