EFFICIENT AND ROBUST ASYNCHRONOUS FEDERATED LEARNING WITH STRAGGLERS

Anonymous authors
Paper under double-blind review

ABSTRACT

We address the efficiency issues caused by the straggler effect in the recently emerged federated learning, which collaboratively trains a model on decentralized non-i.i.d. (non-independent and identically distributed) data across massive worker devices without exchanging training data in the unreliable and heterogeneous networks. We propose a novel two-stage analysis on the error bounds of general federated learning, which provides practical insights into optimization. As a result, we propose a novel easy-to-implement federated learning algorithm that uses asynchronous settings and strategies to control discrepancies between the global model and delayed models and adjust the number of local epochs with the estimation of staleness to accelerate convergence and resist performance deterioration caused by stragglers. Experiment results show that our algorithm converges fast and robust on the existence of massive stragglers.

1 INTRODUCTION

Distributed machine learning has received increasing attention in recent years, e.g., distributed stochastic gradient descent (DSGD) approaches (Gemulla et al., 2011; Lan et al., 2017) and the well-known parameter server paradigm (Agarwal & Duchi, 2011; Li et al., 2013; 2014). However, these approaches always suffer from communication overhead and privacy risk (McMahan et al., 2017). Federated learning (FL) (Konečný et al., 2016) is proposed to alleviate the above issues, where a subset of devices are randomly selected, and training data in devices are locally kept when training a global model, thus reducing communication and protecting user privacy. Furthermore, FL approaches are dedicated to a more complex context with 1) non-i.i.d. (Non-independent and identically distributed), unbalanced and heterogeneous data in devices, 2) constrained computing resources with unreliable connections and unstable environments (McMahan et al., 2017; Konečný et al., 2016).

Typically, FL approaches apply weight averaging methods for model aggregation, e.g., FedAvg (McMahan et al., 2017) and its variants (Sahu et al., 2018; Wang et al., 2018; Kamp et al., 2018; Leroy et al., 2019; Nishio & Yonetani, 2019). Such methods are similar to the synchronous distributed optimization domain. However, synchronous optimization methods are costly in synchronization (Chen et al., 2018), and they are potentially inefficient due to the synchrony even when collecting model updates from a much smaller subset of devices (Xie et al., 2019b). Besides, waiting time for slow devices (i.e., stragglers or stale workers) is inevitable due to the heterogeneity and unreliability as mentioned above. The existence of such devices is proved to affect the convergence of FL (Chen et al., 2018). To address this problem, scholars propose asynchronous federated learning (AFL) methods (Xie et al., 2019a; Mohammad & Sorour, 2019; Samarakoon et al., 2018) that allow model aggregation without waiting for slow devices. However, asynchrony magnifies the straggler effect because 1) when the server node receives models uploaded by the slow workers, it probably has already updated the global model for many times, and 2) real-world data are usually heavy-tailed in distributed heterogeneous devices, where the rich get richer, i.e., the straggler effect accumulates when no adjustment operations in stale workers, and eventually it affects the convergence of the global model. Furthermore, dynamics in AFL brings more challenges in parameter tuning and speed-accuracy trade-off, and the guidelines for designing efficient and stale-robust algorithms in this context are still missing.

Contributions Our main contributions are summarized as follows. We first establish a new two-stage analysis on federated learning, namely training error decomposition and convergence analysis. To the best of our knowledge, it is the first analysis based on the above two stages that address the optimization roadmap for the general federated learning entirely. Such analysis provides insight into designing efficient and stale-robust federated learning algorithms.

By following the guidelines of the above two stages, we propose a novel FL algorithm with asynchronous settings and a set of easy-to-implement training strategies. Specifically, the algorithm controls model training by estimating the model consistency and dynamically adjusting the number of local epochs on straggle workers to reduce the impact of staleness on the convergence of the global model.

We conduct experiments to evaluate the efficiency and robustness of our algorithm on imbalanced and balanced data partitions with different proportions of straggle worker nodes. Results show that our approach converges fast and robust on the existence of straggle worker nodes compared to the state-of-the-art solutions.

Related Work Our work is targeting the AFL and staleness resilience approaches in this context. Straggler effect (also called staleness) is one of the main problems in the similar asynchronous gradient descent (Async-SGD) approaches, which has been discussed by various studies and its remedies have been proposed (Hakimi et al., 2019; Mitliagkas et al., 2016; Hadjis et al., 2016; Lian et al., 2015; Chen et al., 2016; Cui et al., 2016; Chai et al., 2019; Zheng et al., 2017; Dai et al., 2018; Zhou et al., 2018; Hakimi et al., 2019). However, these works are mainly targeting the distributed Async-SGD scenarios, which is different from FL as discussed in the previous section. Existing FL solutions that address the straggler effect are mainly *consensus-based*. Consensus mechanisms are introduced where a threshold metric (i.e., control variable) is computed, and only the workers who satisfy this threshold are permitted to upload their model (Chen et al., 2018; Smith et al., 2017; Nishio & Yonetani, 2019). Thus it significantly reduces the number of communications and updates model without waiting for straggle workers. However, current approaches are mainly focusing on synchronized FL. Xie et al. (2019a) propose an AFL algorithm which uses a mixing hyperparameter to adaptively control the trade-off between the convergence speed and error reduction on staleness. However, this work and above mentioned FL solutions only consider the staleness caused by network delay instead of imbalanced data size in each worker and only evaluate on equal size of local data, which is inconsistent with the real-world cases. Our approach is similar to (Xie et al., 2019a), but instead we adaptively control the number of local epochs combined with the approximation of staleness and model discrepancy, and prove the performance guarantee on imbalanced data partitions. We illustrate our approach in the rest of this paper.

2 Preliminaries and Definitions

We first summarize the general form of FL. Generally, an FL system consists of M distributed worker nodes (e.g., mobile phones) and a server node. The goal is training a global model across these worker nodes without uploading local data. Each worker node employs the same machine learning model, and an optimizer (e.g., stochastic gradient descent) to iteratively optimize the loss function of the local model. At t-th communication round, the server node uses an aggregation operator (e.g., averaging) to aggregate the local models uploaded by worker nodes, and broadcasts the aggregated global model to workers.

We use $X^{(i)} = \{x_1^i, x_2^i, ..., x_{m_i}^i\}$ to present local data points in worker node i, where m_i is the size of data points in this worker. The whole dataset $\chi = \bigcup_i X^{(i)}$, where $i \in \{1, 2, 3, ..., M\}$. We assume that $X^{(i)} \bigcap X^{(j)} = \emptyset$ for $i \neq j$, and apparently, the total size of data $m = \sum_{i=1}^{M} m_i$. We denote the model in worker node i by $\omega_i \in \mathbb{R}^d$, and the objective function of worker node i by

$$F_i(\omega_i) = \frac{1}{m_i} \sum_{j=1}^{m_i} f(x_j^i; \omega_i), \tag{1}$$

where $f(\cdot) : \chi \to \mathbb{R}$ is user-defined loss function. Then the objective function of the global model is

$$F(\omega) = \frac{1}{m}\sum_{j=1}^{m} f(x_j;\omega) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{m_i} f(x_j^i;\omega)}{m} = \sum_{i=1}^{M}\frac{m_i}{m}F_i(\omega), \tag{2}$$

where ω is the aggregated global model. The overall goal is to find a model ω^* with:

$$\omega^* = \arg\min F(\omega). \tag{3}$$

Usually, we can use gradient-descent methods to solve equation 3. At the communication round $t+1$, the global model can be formulated as

$$\omega^{t+1} = \omega^t - \eta^t \nabla F(\omega^t), \tag{4}$$

where η^t is the learning rate for model ω^t. From equation 2 and equation 4, we can get

$$\omega^{t+1} = \omega^t - \eta^t \sum_{i=1}^{M}\frac{m_i}{m}\nabla F_i(\omega^t) = \sum_{i=1}^{M}\frac{m_i}{m}(\omega^t - \eta^t\nabla F_i(\omega^t)) = \sum_{i=1}^{M}\frac{m_i}{m}\omega_i^{t+1}. \tag{5}$$

Note that equation 5 is efficient only for reliable environment. To extend equation 5 to unreliable and heterogeneous FL, we give a more general form as

$$\omega^{t+1} = \omega^t - \eta^t g(\omega^t, \xi_t) = \omega^t + \sum_{i=1}^{M} h(\omega_i^{t+1}, \tau_i) \tag{6}$$

where $g(\cdot)$ is the user-defined aggregation function, and ξ_t is a vector which describes the settings of activated workers, such as worker ID, number of local epochs, and the learning rate. Here and thereafter, we use $g(\omega^t)$ to represent $g(\omega^t, \xi_t)$ for convenience. We denote *update term* as $h(\cdot)$, a user-defined function which represents the model parameter differences between the collected models from activated worker nodes and previous global model, and $-\eta^t g(\omega^t) = \sum_{i=1}^{M} h(\omega_i^{t+1}, \tau_i)$. Here τ_i is the time when worker node i received the global model ω^{τ_i}. When $h(\omega_i^{t+1}, \tau_i) = \frac{m_i}{m}(\omega_i^{t+1} - \omega^t)$, we get FedAvg (McMahan et al., 2017) as a special case of equation 6.

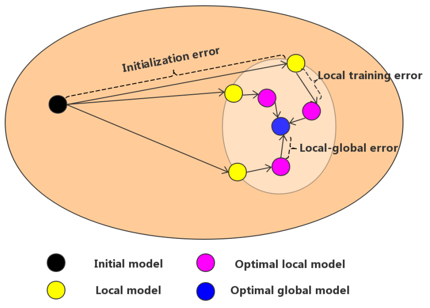


Figure 1: Illustration of initialization error, local training error and local-global error.

3 Methodology

In this section, we aim to design an efficient and robust FL algorithm. To do so, we first establish a two-stage analysis, and finally, propose our new FL algorithm by combining the insights provided by the two stages.

Stage 1: Traning Error Decomposition. We first discuss the main errors of the general FL. We assume that each worker node has a local optimal model $\omega_i^* = \arg\min F_i(\omega)$. Then at the communication round t, we define the global error as

$$\underbrace{\|\omega^t - \omega^*\|}_{\text{global error}} \le \underbrace{\sum_{i=1}^{M}\frac{m_i}{m}\|\omega_i^t - \omega_i^*\|}_{\text{initialization and local error}} + \underbrace{\sum_{i=1}^{M}\frac{m_i}{m}\|\omega_i^* - \omega^*\|}_{\text{local-global error}}, \tag{7}$$

where $\|\cdot\|$ is L^2 norm. For worker node i, two terms in the right-hand side of inequality 7 respectively represent 1) *initialization and local error*: the error between the local model at communication round

t and the optimal local model (the well known empirical risk). Here, the initialization error (i.e., the error between the initial model and local model at communication round t) partially contributes to the first term. 2) *local-global error*: the error between optimal local models and optimal global solution, which is a constant given a specific learning task. Figure 1 illustrates these errors. Usually, the error between the initial model and the optimal global model is greater than the local-global error, and thus at the early stage of training, the first term is greater than the second term in the right-hand side of inequality 7. Therefore, reducing the initialization error and the local error at the beginning of model training can reduce the global error $\|\omega^t - \omega^*\|$. Afterward, when initialization and local error is minimized, the local-global error dominates the global error. However, as we mentioned previously, the local-global error is a constant that can not be solved directly. Therefore, we need a more sophisticated analysis to reach ω^* since 7 is no longer appropriate to guide the optimization other than the early stage of FL training. Following the above analysis, we analyze the convergence bounds of the general FL (Eq. 6) on the rest of the training stages other than the early stage.

Stage 2: Convergence Analysis. First, we make the following assumptions on the objective functions:

Assumption 1. ***Smoothness***. *For all i in $\{1, 2, 3, ..., M\}$ and given constant β, the objective function $F(\omega)$ and $F_i(\omega)$ are β-smooth, i.e.,*

$$\|\nabla F(\omega) - \nabla F(\omega')\| \leq \beta\|\omega - \omega'\|,$$
$$\|\nabla F_i(\omega) - \nabla F_i(\omega')\| \leq \beta\|\omega - \omega'\|.$$

Assumption 2. ***The first and second moment conditions***. *The objective function $F(\omega)$ and the aggregation operation $g(\omega^t)$ satisfy the following:*

(a) The objective function $F(\omega)$ is bounded by a scalar $F_{inf} = F(\omega^*)$.

(b) There exist scalars $\delta_G \geq \delta > 0$ such that $\nabla F(\omega^t)^\top \mathbb{E}(g(\omega^t)) \geq \delta\|\nabla F(\omega^t)\|^2$ and $\|\mathbb{E}(g(\omega^t))\| \leq \delta_G\|\nabla F(\omega^t)\|$.

(c) There exist scalars $L \geq 0$ such that $\mathbb{V}(g(\omega^t)) = \mathbb{E}(\|g(\omega^t)\|^2) - \|\mathbb{E}(g(\omega^t)\|^2 \leq L + \|\nabla F(\omega^t)\|^2$. $\mathbb{E}(\cdot)$ is abbreviation of $\mathbb{E}_{\xi_t}(\cdot)$ which denotes the expected value w.r.t. the distribution of the random variable ξ_t given t.

Assumption 3. ***Strong convexity***. *For all i in $\{1, 2, 3, ..., M\}$ and given constant c, the objective function $F(\omega)$ and $F_i(\omega)$ are c-strong convex, i.e.,*

$$F(\omega') \geq F(\omega) + \nabla F(\omega)^\top(\omega' - \omega) + \frac{1}{2}c\|\omega' - \omega\|^2,$$
$$F_i(\omega') \geq F_i(\omega) + \nabla F_i(\omega)^\top(\omega' - \omega) + \frac{1}{2}c\|\omega' - \omega\|^2.$$

Theorem 1. ***Convergence for strongly-convex problems***. *When c and β in assumption 1 and 3 satisfy $c \leq \beta$, we can set the step size $\eta^t = \bar{\eta}$, where $0 < \bar{\eta} \leq \frac{\delta}{\beta L_G}$, and $L_G = 1 + \delta_G^2$. With $\bar{\eta}$, the upper error bound of global model satisfies:*

$$\mathbb{E}(F(\omega^t) - F(\omega^*)) \leq \frac{\bar{\eta}\beta L}{2c\delta} + (1 - \bar{\eta}c\delta)^{t-1}\left(F(\omega^1) - F(\omega^*) - \frac{\bar{\eta}\beta L}{2c\delta}\right). \quad (8)$$

The proof of theorem 1 is provided in appendix A.1. Theorem 1 gives an error bound for the general form of model aggregation without assuming that $g(\omega^t)$ should come from $\nabla F(\omega^t)$. Note that the scalars δ and δ_G are equal to 1 when $g(\omega^t)$ is the unbiased estimation of $\nabla F(\omega^t)$. However, current convergence bound in theorem 1 is too loose, and it can be further optimized by introducing controlled local epoch settings.

We assume that $\nabla F_i(\omega_i^{t+1}) = \gamma\nabla F_i(\omega_i^t) + R_i^t$ with step size $\bar{\eta} \leq \frac{\nu}{\beta}$. Here γ is the projection length of $\nabla F_i(\omega_i^{t+1})$ on $\nabla F_i(\omega_i^t)$, $v = 1 - \gamma$, and R_i^t is the remainder term which is perpendicular to $\nabla F_i(\omega_i^t)$. Then given a local epoch $\bar{E}$, we have

$$\delta \propto \bar{E} - o(\nu\bar{E}), L \propto \bar{E}^2 + o(\nu\bar{E}^2). \quad (9)$$

Then we can extend theorem 1 with the local epoch $\bar{E}$.

Theorem 2. ***Convergence with selected local epoch**. We use δ_0 and L_0 to represent the scalar δ and L in assumption 2 when $\bar{E} = 1$, and all worker nodes are assumed to participate in model training, then under 9, 8 can be rewritten as*

$$\mathbb{E}(F(\omega^t) - F(\omega^*)) \leq \frac{\bar{\eta}\beta L_0 \bar{E}}{2c\delta_0} + (1 - \bar{\eta}c\delta_0\bar{E})^{t-1}\left(F(\omega^1) - F(\omega^*) - \frac{\bar{\eta}\beta L_0\bar{E}}{2c\delta_0}\right). \quad (10)$$

The theorem 2 gives us the error bound of FL with the selected number of local epochs for strongly-convex problems. The proof of theorem 2 is provided in appendix A.2. The right-hand side of theorem 2 implies the dynamics of hyper-parameters in local models for efficiency and robustness trade-off. In a general FL algorithm, e.g., FedAvg, the model settings in worker nodes are always predefined and kept fixed, which may lead to a higher variance. We now discuss such dynamics and practical insights on designing efficient and robust FL algorithms.

Selection of local epochs. We discuss how to reduce the global error and communication round simultaneously for general FL. From the second term of the right-hand side of 10, we can see that theorem 2 yields to linear convergence when $\mathbb{E}(F(\omega^t) - F(\omega^*)) \gg \frac{\bar{\eta}\beta L_0\bar{E}}{2c\delta_0}$. In this condition, to quickly reduce the global error, we can reduce the second term of the right-hand side of 10 by increasing the local epoch $\bar{E}$ while reducing the communication round t. Therefore, we can dynamically assign each worker with a bigger number of local epoch while reducing the communication round.

Asynchronous acceleration with stragglers. We discuss why asynchronous strategies are needed in FL. We rearrange 10 as:

$$\mathbb{E}(F(\omega^t) - F(\omega^*)) \leq \left(1 - (1 - \bar{\eta}c\delta_0\bar{E})^{t-1}\right)\frac{\bar{\eta}\beta L_0\bar{E}}{2c\delta_0} + (1 - \bar{\eta}c\delta_0\bar{E})^{t-1}\left(F(\omega^1) - F(\omega^*)\right). \quad (11)$$

When t increases, and we fix $\bar{E}$ and $\bar{\eta}$, the global error only depends on L_0. L_0 can be controlled by sampling more worker nodes within a communication round. Specifically, we compare n-worker participation with M-worker participation for model aggregation at the server node. When we select n workers out of M, L_0 increases according to assumption 2(c) since the variance increases. Thus, to get the same precision, we decrease $\bar{\eta}$, while it significantly slows the convergence speed. However, in practice, waiting for all the workers to be ready is time-consuming. Thus, we can introduce asynchronous mechanisms in model aggregation without waiting for all workers.

Robust training on straggle workers. We discuss how to reduce the global error for FL on the existence of stragglers. As we mentioned above, asynchronous strategies can accelerate model training by reducing the waiting time at each communication round. However, the straggler effect is magnified by asynchrony, as discussed in section 1. Stale workers accumulate their staleness, which increases the variances and affects the convergence of the global model. A practical strategy to tame such effect is increasing the number of local epoch under the considerations that when the distributions of local data are far away from the global data, we use more epochs to train from the local data. However, the divergence of these local epoch numbers between stale and non-stale workers may affect variance adversely, and we can adjust the number of local epoch with the normalized epochs from all workers to reduce such variance.

Theorem 3. ***Convergence for non-convex problems** With the Assumption 1 and 2, we can select a step size $\eta^t = \bar{\eta}$ with $0 < \bar{\eta} \leq \frac{\delta}{\beta L_G}$, $L_G = 1 + \delta_G^2$. The expected error bound satisfies:*

$$\mathbb{E}\left(\frac{1}{t}\sum_{i=1}^{t}\|\nabla F(\omega^i)\|^2\right) \leq \frac{\bar{\eta}\beta L_0\bar{E}}{\delta_0} + \frac{2(F(\omega^1) - F_{inf})}{t\bar{\eta}\delta_0\bar{E}}. \quad (12)$$

Theorem 3 is similar to theorem 2 that the first term of the right-hand side of (12) does not decrease by iterative training. Note that the above remarks are also applicable to theorem 3. We provide proof of theorem 3 in appendix A.3.

Proposed Algorithm. Under the guidance of the above analysis and the practical insights discussed above, we propose a fast and stale-robust AFL algorithm. Algorithm 1 and 2 illustrate the processes in worker nodes and the server node, respectively. $H(t)$ is a predefined function at communication round t which determines how long should the server node waiting for the updated models from workers. $H(t)$ can be used to control the accuracy-speed trade-off. The training processes on the server node can be divided into two stages, i.e., the initial stage and the converging stage. We switch the stages by estimating the consistency of model updates.

Algorithm 1 Process at the server node

```
Input: Function $H(t)$, model $F(\omega)$, initial epoch $E_{init}$, initial waiting time $\Delta t$, threshold of communication rounds $T_2$.
Output: Updated model $F(\omega)$
 1: Initialize $\omega^t$ as a constant or a random vector, $t \leftarrow 0$.
 2: Broadcast $\omega^t$ and $E_{init}$ to each worker.
 3: repeat   // optimizing the initialization and local error.
 4:     Set $t \leftarrow t + 1$.
 5:     During $\Delta t$ time, receive the triplet $(\omega_i, \tau_i, E_i)$ from any worker $i$.
 6:     Update $\omega^t$ with $\omega_i$ with $\tau_i = t - 1$ using 5.
 7:     Broadcast $(\omega^t, t)$ to each worker.
 8:     Calculate $U$ using 13.
 9: until $U \leq 0.1$
10: Broadcast start flag to each worker.
11: repeat   // the converging stage.
12:     Set $t \leftarrow t + 1, \Delta t \leftarrow H(t)$.
13:     During $\Delta t$ time, receive the triplet $(\omega_i, \tau_i, E_i)$ from any worker $i$.
14:     Update $\omega^t$ by 6 and 14.
15:     Update $mean(E)$ by 15.
16:     Broadcast the triplet $(\omega^t, t, mean(E))$ to each worker
17: until $t > T_2$
```

Algorithm 2 Process at the worker node

```
Input: Number of local epoch $E_{init}$, communication round $t$, global model $\omega^t$, batch size $B_i$, average global epochs $mean(E)$, local optimizer $opt$, communication time $\Delta t_i$, start flag sent by the server node
Output: Triplet $(\omega_i, \tau_i, E_i)$
 1: Estimate staleness $s$ using $s \leftarrow \left\lfloor \frac{\Delta t_i}{H(t)} \right\rfloor$.
 2: Set $E_i' \leftarrow E_{init}$.
 3: if start flag is True then
 4:     Set $E_i' \leftarrow mean(E) * s$.
 5: end if
 6: Set $\tau_i \leftarrow t$, $E_i \leftarrow 0$.
 7: for $e$ in $1, 2, 3, ..., E_i'$ do
 8:     $E_i \leftarrow E_i + 1$.
 9:     Randomly divide $X^{(i)}$ with batch size $B_i$.
10:     Update $\omega_i$ by using $opt$ for each batch.
11:     if $e = E_i'$ then
12:         if communication to the server is available then
13:             Send triplet $(\omega_i, \tau_i, E_i)$ to the server.
14:         else
15:             $e \leftarrow e - 1$.
16:         end if
17:     end if
18: end for
```

Definition 1. ***Update consistency****. The model update consistency of* n *worker nodes is the similarities between worker models at communication round* t*, i.e.,*

$$U = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} \cos(\omega_i^{t+1} - \omega^t, \omega_j^{t+1} - \omega^t). \tag{13}$$

U is consistent with the global error in inequality 7, and in algorithm 1 we empirically set 0.1 as the threshold to switch from the initial stage and the converging stage of the global model training. At the initial stage of global model training, we use a bigger local epoch to accelerate training time as discussed above, and repeat this process until $U \leq 0.1$. After the initial stage, we define the update term as

$$h(\omega_i^{t+1}, \tau_i) = \frac{\varphi_i}{\varphi} \left(\omega_i^{t+1} - \omega^t\right). \tag{14}$$

$\frac{\varphi_i}{\varphi}$ is the above mentioned normalized local epoch $\bar{E}$ with $\varphi_i = \frac{mean(E')m_i}{E_i d\left(\omega_i^{t+1}\right)}$, and

$$mean(E') = \frac{1}{n^t} \sum_{j=1}^{n^t} \frac{E_j}{t + 1 - \tau_j}. \tag{15}$$

φ is the regularization term where $\varphi = \sum_{i=1}^{n} \varphi_i$. Finally, we define a stale-related penalty function of φ_i as:

$$d\left(\omega_i^{t+1}\right) = \begin{cases} e^{t-\tau_i}, \cos(\omega_i^{t+1} - \omega_i^t, \omega_{fresh}^{t+1} - \omega^t) \leq -0.1 \\ t - \tau_i + 1, others \end{cases}. \tag{16}$$

Here, ω_{fresh}^{t+1} is the average model of worker nodes with $\tau_i = t$. The key processes of worker nodes are 1) estimating its staleness level, and 2) assign the number of local epoch using $mean(E)$ in the received triplet from the server node and the previously estimated staleness level. In the next section, we evaluate the performance of our algorithms.

4 EXPERIMENTS

We evaluate the performance of our approach on both imbalanced and balanced data partitions with the existence of stale worker nodes.

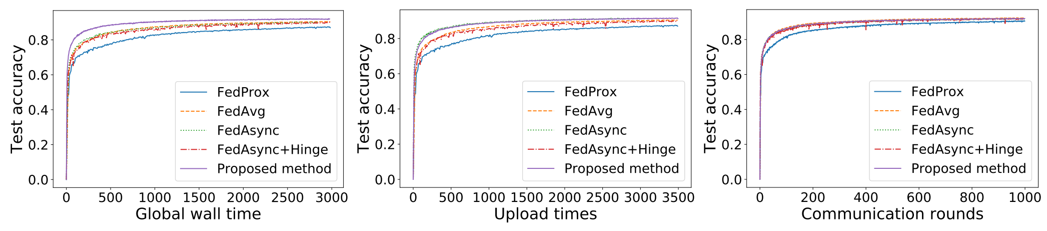
Experiment Settings. We conduct experiments on Fashion-MNIST (Xiao et al., 2017) and CIFAR-10 (Krizhevsky et al., 2009) to test the accuracy of our approach on 100 simulated workers, where 60 workers are stale. We use 55,000 on Fashion-MNIST and 50,000 on CIFAR-10 for training and 10,000 for testing. $[0, 1]$ normalization is used in the data preprocessing. We conduct all experiments on CPU devices. We use a light-weight convolutional neural network (CNN) model, which is suitable for mobile edge devices. It has 4 convolutional layers that use 3×3 kernels with the size of $32, 64, 64, 128$. Rectified linear unit (ReLU) is used in each convolutional layer, and every two convolutional layers are followed by a 2×2 max-pooling layer and a dropout of 50%. Finally, we use a 512-unit dense layer with ReLU and a dropout of 50% and an output layer with softmax. We use an SGD optimizer with a learning rate of 0.01. We set batch size as 50, and the initial number of local epochs E as 50. We randomly split the data size in each worker node ranging from 2 to 2122 with a standard deviation of 480 on CIFAR-10, and 9 to 2157 with a standard deviation of 540 on Fashion-MNIST. For the balanced cases we randomly assign each worker with 500 samples. The communication speed of nodes is divided into ten levels ranging from 100 milliseconds to 1 second, and the 60 stale workers are assigned with bigger levels (6-10). Finally, we set $H(t) = 0.4s$.

Baselines. We compare the performance of our proposed method with four approaches: 1) **FedAvg** (McMahan et al., 2017) (synchronized). We set the sampling rate $C = 0.1$ **FedProx** (Sahu et al., 2018) (synchronized). We set $C = 0.1, \mu = 1$ as the best parameters provided in their paper. **FedAsync** (Xie et al., 2019a) (asynchronized). We set $\gamma = 0.1, \rho = 0.005, t - \tau \leq 4$. **FedAsync + Hinge** (Xie et al., 2019a) (asynchronized). We set $a = 4, b = 4, \gamma = 0.1, \rho = 0.005, t - \tau \leq 10$ as shown in their paper. All the baselines besides FedProx use the same strategy of local epoch decay with local epoch E decreases by 1 per 50 global communication rounds until it decays to 1. All the four approaches use the same batch size of 50 and initial local epochs of 10.

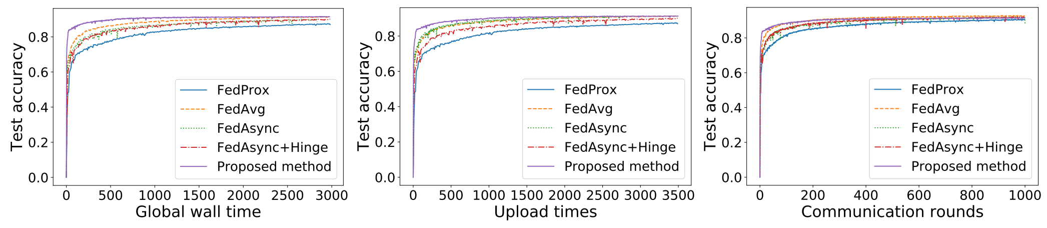
Results and Analysis. Figure 2 shows the performance of our proposed algorithm and four baselines. Our method converge faster compared to all the baselines, and the convergence is promised with 60% stale workers. Furthermore, the whole upload times of our method do not increase with the same level of accuracy. From the experiment results on Fashion-MNIST, we can see that our method has the same accuracy level on test data compared with synchronized approach such as FedAvg. We can also see that on imbalanced data partitions (i.e., more realistic FL scenarios), our method is faster and more stable compared to other baselines. Finally, we can clearly see the stage transition from the initial training stage to the converging stage (e.g., the transitions in imbalanced cases in figure 2(b) and (d)), which validates the efficiency of our approach. Figure 3 shows the performance of our method with different proportion of stale nodes in 1,000 global communication rounds. Our method outperforms the AFL baseline (i.e., FedAsync) in both accuracy and loss, and when the proportion of stale workers is less than 80%, our method outperforms the synchronized FL baseline (i.e., FedAvg).

5 CONCLUSIONS AND FUTURE WORK

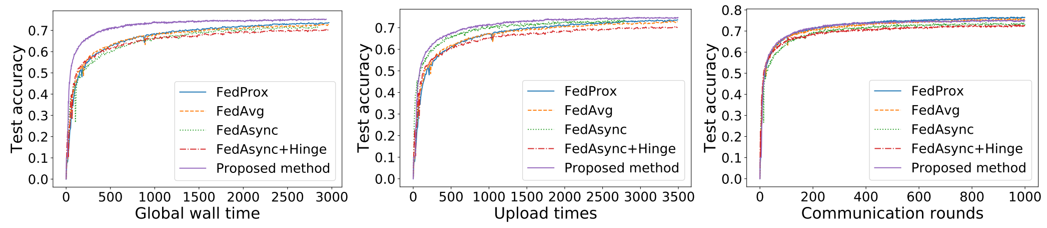
In this paper, we propose a new two-stage analysis on federated learning, and inspired by such analysis, we propose a novel AFL algorithm that accelerates convergence and resists performance deterioration caused by stragglers simultaneously. Experimental results show that our approach converges two times faster than baselines, and it can resist the straggler effect without sacrificing accuracy and communication. As a byproduct, our approach improves the generalization ability of neural network models. We will theoretically analyze it in future work. Besides, while not the focus of our work, security and privacy are essential concerns in federated learning, and as the future work, we can apply various security methods to our approach. Furthermore, besides the stale-



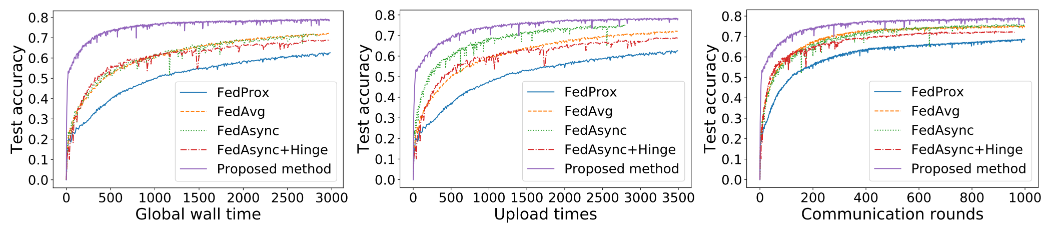
(a) Accuracy on Fashion-MNIST (balanced case)



(b) Accuracy on Fashion-MNIST (imbalanced case)



(c) Accuracy on CIFAR-10 (balanced case)



(d) Accuracy on CIFAR-10 (imbalanced case)

Figure 2: Accuracy on test data of CIFAR-10 and Fashion-MNIST with imbalanced and balanced data partitions.

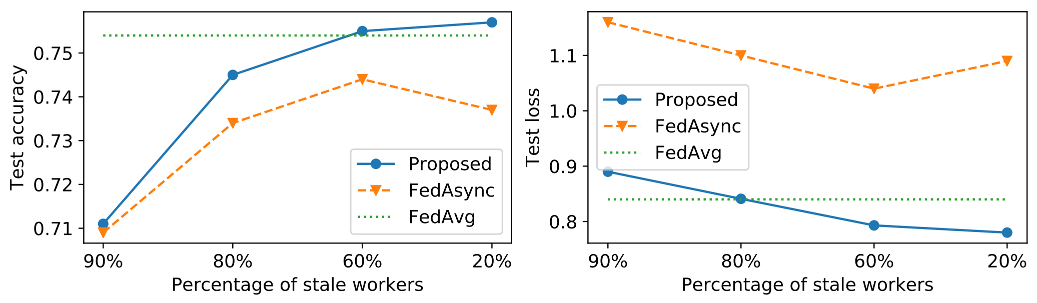


Figure 3: Test accuracy and loss with different proportion of stale workers on CIFAR-10 dataset in 1,000 communication rounds. We respectively test the performance with 20%, 60%, 80%, and 90% of stale workers. The green dotted line is FedAvg which waits all selected workers.

resistance ability, the discrepancy estimation in our method also has the potential ability to resist malicious attacks to the worker nodes such as massive Byzantine attacks, which has been addressed in (Bagdasaryan et al., 2018; Li et al., 2019; Muñoz-González et al., 2019). We will analyze and evaluate such ability in future work.

REFERENCES

Alekh Agarwal and John C Duchi. Distributed delayed stochastic optimization. In *Advances in Neural Information Processing Systems*, pp. 873–881, 2011.

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. *arXiv preprint arXiv:1807.00459*, 2018.

Zheng Chai, Hannan Fayyaz, Zeshan Fayyaz, Ali Anwar, Yi Zhou, Nathalie Baracaldo, Heiko Ludwig, and Yue Cheng. Towards taming the resource and data heterogeneity in federated learning. In *2019 {USENIX} Conference on Operational Machine Learning (OpML 19)*, pp. 19–21, 2019.

Jianmin Chen, Xinghao Pan, Rajat Monga, Samy Bengio, and Rafal Jozefowicz. Revisiting distributed synchronous sgd. *arXiv preprint arXiv:1604.00981*, 2016.

Tianyi Chen, Georgios Giannakis, Tao Sun, and Wotao Yin. Lag: Lazily aggregated gradient for communication-efficient distributed learning. In *Advances in Neural Information Processing Systems*, pp. 5050–5060, 2018.

Henggang Cui, Hao Zhang, Gregory R Ganger, Phillip B Gibbons, and Eric P Xing. Geeps: Scalable deep learning on distributed gpus with a gpu-specialized parameter server. In *Proceedings of the Eleventh European Conference on Computer Systems*, pp. 4. ACM, 2016.

Wei Dai, Yi Zhou, Nanqing Dong, Hao Zhang, and Eric P Xing. Toward understanding the impact of staleness in distributed machine learning. *arXiv preprint arXiv:1810.03264*, 2018.

Rainer Gemulla, Erik Nijkamp, Peter J Haas, and Yannis Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 69–77. ACM, 2011.

Stefan Hadjis, Ce Zhang, Ioannis Mitliagkas, Dan Iter, and Christopher Ré. Omnivore: An optimizer for multi-device deep learning on cpus and gpus. *arXiv preprint arXiv:1606.04487*, 2016.

Ido Hakimi, Saar Barkai, Moshe Gabel, and Assaf Schuster. Taming momentum in a distributed asynchronous environment. *arXiv preprint arXiv:1907.11612*, 2019.

Michael Kamp, Linara Adilova, Joachim Sicking, Fabian Hüger, Peter Schlicht, Tim Wirtz, and Stefan Wrobel. Efficient decentralized deep learning by dynamic model averaging. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 393–409. Springer, 2018.

Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Guanghui Lan, Soomin Lee, and Yi Zhou. Communication-efficient algorithms for decentralized and stochastic optimization. *Mathematical Programming*, pp. 1–48, 2017.

David Leroy, Alice Coucke, Thibaut Lavril, Thibault Gisselbrecht, and Joseph Dureau. Federated learning for keyword spotting. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6341–6345. IEEE, 2019.

Liping Li, Wei Xu, Tianyi Chen, Georgios B Giannakis, and Qing Ling. Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 1544–1551, 2019.

Mu Li, Li Zhou, Zichao Yang, Aaron Li, Fei Xia, David G Andersen, and Alexander Smola. Parameter server for distributed machine learning. In *Big Learning NIPS Workshop*, volume 6, pp. 2, 2013.

Mu Li, David G Andersen, Alexander J Smola, and Kai Yu. Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems*, pp. 19–27, 2014.

Xiangru Lian, Yijun Huang, Yuncheng Li, and Ji Liu. Asynchronous parallel stochastic gradient for nonconvex optimization. In *Advances in Neural Information Processing Systems*, pp. 2737–2745, 2015.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

Ioannis Mitliagkas, Ce Zhang, Stefan Hadjis, and Christopher Ré. Asynchrony begets momentum, with an application to deep learning. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 997–1004. IEEE, 2016.

Umair Mohammad and Sameh Sorour. Adaptive task allocation for asynchronous federated mobile edge learning. *arXiv preprint arXiv:1905.01656*, 2019.

Luis Muñoz-González, Kenneth T Co, and Emil C Lupu. Byzantine-robust federated machine learning through adaptive model averaging. *arXiv preprint arXiv:1909.05125*, 2019.

Takayuki Nishio and Ryo Yonetani. Client selection for federated learning with heterogeneous resources in mobile edge. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pp. 1–7. IEEE, 2019.

Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet Talwalkar, and Virginia Smith. On the convergence of federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

Sumudu Samarakoon, Mehdi Bennis, Walid Saady, and Merouane Debbah. Distributed federated learning for ultra-reliable low-latency vehicular communications. *arXiv preprint arXiv:1807.08127*, 2018.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pp. 4424–4434, 2017.

Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K Leung, Christian Makaya, Ting He, and Kevin Chan. When edge meets learning: Adaptive control for resource-constrained distributed machine learning. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pp. 63–71. IEEE, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Cong Xie, Sanmi Koyejo, and Indranil Gupta. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*, 2019a.

Cong Xie, Sanmi Koyejo, and Indranil Gupta. Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In *International Conference on Machine Learning*, pp. 6893–6901, 2019b.

Shuxin Zheng, Qi Meng, Taifeng Wang, Wei Chen, Nenghai Yu, Zhi-Ming Ma, and Tie-Yan Liu. Asynchronous stochastic gradient descent with delay compensation. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 4120–4129. JMLR. org, 2017.

Zhengyuan Zhou, Panayotis Mertikopoulos, Nicholas Bambos, Peter Glynn, Yinyu Ye, Li-Jia Li, and Li Fei-Fei. Distributed asynchronous optimization with unbounded delays: How slow can you go? In *International Conference on Machine Learning*, pp. 5965–5974, 2018.

A Proofs

A.1 Proof of Theorem 1

Lemma 1. *Under the assumption 1, we can get:*

$$\mathbb{E}(F(\omega^{t+1})) - F(\omega^t) \leq -\eta_t \nabla F(\omega^t)^\top \mathbb{E}(g(\omega^t)) + \frac{1}{2}\eta_t^2 \beta \mathbb{E}(\|g(\omega^t)\|^2). \tag{17}$$

Proof: Under the assumption 1, for any ω and ω', we have:

$$\begin{aligned}
F(\omega) &= F(\omega') + \int_0^1 \frac{\mathrm{d}F(\omega' + t(\omega - \omega'))}{\mathrm{d}t}\mathrm{d}t \\
&= F(\omega') + \int_0^1 \nabla F(\omega' + t(\omega - \omega'))^\top (\omega - \omega')\mathrm{d}t \\
&= F(\omega') + \nabla F(\omega')^\top (\omega - \omega') + \int_0^1 [\nabla F(\omega' + t(\omega - \omega')) - \nabla F(\omega')]^\top (\omega - \omega')\mathrm{d}t \\
&\leq F(\omega') + \nabla F(\omega')^\top (\omega - \omega') + \int_0^1 \beta \|t(\omega - \omega')\| \|\omega - \omega'\| \mathrm{d}t \\
&= F(\omega') + \nabla F(\omega')^\top (\omega - \omega') + \frac{1}{2}\beta \|\omega - \omega'\|^2.
\end{aligned} \tag{18}$$

Then using 18, we have:

$$\begin{aligned}
F(\omega^{t+1}) - F(\omega^t) &\leq \nabla F(\omega^t)^\top (\omega^{t+1} - \omega^t) \\
&\quad + \frac{1}{2}\beta \|\omega^{t+1} - \omega^t\|^2 \\
&= -\nabla F(\omega^t)^\top \eta_t g(\omega^t) + \frac{\beta}{2}\|\eta_t g(\omega^t)\|^2.
\end{aligned} \tag{19}$$

Taking expectations in 19 w.r.t the distribution of ξ_t, we complete the proof. □

Lemma 2. *Under the assumption 1 and 2, we can get:*

$$\mathbb{E}(F(\omega^{t+1})) - F(\omega^t) \leq -\eta_t(\delta - \frac{\eta_t \beta L_G}{2})\|\nabla F(\omega^t)\|^2 + \frac{1}{2}\eta_t^2 \beta L. \tag{20}$$

Proof: Using assumption 2(b) and 2(c), we have:

$$\begin{aligned}
\mathbb{E}(\|g(\omega^t)\|^2) &\leq \|\mathbb{E}(g(\omega^t))\|^2 + L + \|\nabla F(\omega^t)\|^2 \\
&\leq L + (1 + \delta_G^2)\|\nabla F(\omega^t)\|^2 \\
&= L + L_G \|\nabla F(\omega^t)\|^2.
\end{aligned} \tag{21}$$

Then using 21, assumption 2(b) and lemma 1, we have:

$$\begin{aligned}
\mathbb{E}(F(\omega^{t+1})) - F(\omega^t) &\leq -\eta_t \nabla F(\omega^t)^\top \mathbb{E}(g(\omega^t)) \\
&\quad + \frac{1}{2}\eta_t^2 \beta \mathbb{E}(\|g(\omega^t)\|^2) \\
&\leq -\eta_t \delta \|\nabla F(\omega^t)\|^2 + \frac{\eta_t^2}{2}\beta(L + L_G\|\nabla F(\omega^t)\|^2).
\end{aligned} \tag{22}$$

We can easily get Lemma 2 by rearranging 22. □

Then we prove Theorem 1 under the assumption 1, 2, 3. First, we define

$$\mathcal{F}(\hat{\omega}) = F(\omega^t) + \nabla F(\omega^t)^\top (\hat{\omega} - \omega^t) + \frac{c}{2}\|\hat{\omega} - \omega^t\|^2. \tag{23}$$

Function $\mathcal{F}$ is a quadratic model relevant to $\hat{\omega}$. Then it has the minimal value when all the partial derivatives are 0. That is

$$\frac{\partial \mathcal{F}(\hat{\omega})}{\partial \hat{\omega}} = \nabla F(\omega^t) + c(\hat{\omega} - \omega^t) = \vec{0}.$$

Then, when we select $\hat{\omega} = \hat{\omega}^* = \omega^t - \frac{\nabla F(\omega^t)}{c}$ for 23, we get the minimal of 23, which is

$$\mathcal{F}_{min} = F(\omega^t) - \frac{\|\nabla F \omega^t\|^2}{2c}. \tag{24}$$

From assumption 3, we have

$$F(\omega^*) \geq \mathcal{F}(\omega^*) \geq \mathcal{F}_{min} = F(\omega^t) - \frac{\|\nabla F \omega^t\|^2}{2c}, \tag{25}$$

which is equivalent to

$$2c(F(\omega^t) - F(\omega^*)) \leq \|\nabla F \omega^t\|^2. \tag{26}$$

Then from Lemma 2, when a fixed $\bar{\eta} \leq \frac{\delta}{\beta L_G}$ is selected, we have:

$$\mathbb{E}(F(\omega^{t+1})) - F(\omega^t) \leq -\eta_t \frac{\delta}{2} \|\nabla F(\omega^t)\|^2 + \frac{1}{2}\eta_t^2 \beta L. \tag{27}$$

And using 26, we have

$$\mathbb{E}(F(\omega^{t+1})) - F(\omega^t) \leq -\eta_t \delta c(F(\omega^t) - F(\omega^*)) + \frac{1}{2}\eta_t^2 \beta L. \tag{28}$$

Subtracting $F(\omega^*)$ from both sides and moving $F(\omega^t)$ from left to right, we get

$$\mathbb{E}(F(\omega^{t+1})) - F(\omega^*) \leq -\eta_t \delta c(F(\omega^t) - F(\omega^*)) + (F(\omega^t) - F(\omega^*)) + \frac{1}{2}\eta_t^2 \beta L. \tag{29}$$

Taking the whole expectations and rearranging 29, we obtain

$$\mathbb{E}(F(\omega^{t+1}) - F(\omega^*)) \leq (1 - \eta_t \delta c)\mathbb{E}(F(\omega^t) - F(\omega^*)) + \frac{1}{2}\eta_t^2 \beta L. \tag{30}$$

Substracting the constant $\frac{\bar{\eta}\beta L}{2c\delta}$ from both sides of 30, we have

$$\mathbb{E}(F(\omega^{t+1}) - F(\omega^*)) - \frac{\bar{\eta}\beta L}{2c\delta} \leq (1 - \eta_t \delta c)(\mathbb{E}\left(F(\omega^t) - F(\omega^*)) - \frac{\bar{\eta}\beta L}{2c\delta}\right). \tag{31}$$

The left hand side of 31 is a geometric series with common ratio $1 - \eta_t \delta c$, then we complete the proof. $\square$

A.2 Proof of Theorem 2

We first prove 9. Assume $\nabla F_i(\omega_i^{t+1}) = (1-\nu)\nabla F_i(\omega_i^t) + Res_i^t$, we have

$$\begin{aligned}
\|Res_i^t\| &= \|\nabla F_i(\omega_i^{t+1}) - (1-\nu)\nabla F_i(\omega_i^t)\| \\
&= \|(1-\nu+\nu)\nabla F_i(\omega_i^{t+1}) - (1-\nu)\nabla F_i(\omega_i^t)\| \\
&\leq \|(1-\nu)\left(\nabla F_i(\omega_i^{t+1}) - \nabla F_i(\omega_i^t)\right)\| + \|\nu \nabla F_i(\omega_i^{t+1})\| \\
&\leq (1-\nu)\beta\|\omega_i^{t+1} - \omega_i^t\| + \nu\|\nabla F_i(\omega_i^{t+1})\| \\
&= (1-\nu)\beta\eta_t\|\nabla F_i(\omega_i^t)\| + \nu\|\nabla F_i(\omega_i^{t+1})\| \\
&\leq (1-\nu)\nu\|\nabla F_i(\omega_i^t)\| + \nu\|\nabla F_i(\omega_i^{t+1})\|.
\end{aligned} \tag{32}$$

Let $\|\nabla F_i(\omega_i^t)\| = a$ and $\|\nu \nabla F_i(\omega_i^{t+1})\| = b$, we define

$$h(\nu) = a(1-\nu)\nu + b\nu = \frac{\nu}{a}(\frac{a+b}{a} - \nu). \tag{33}$$

Since $\frac{a+b}{a} > 1$ and $\nu \leq 1$, we have $h(\nu)_{min} = h(0) = 0$, and $a \approx b$ when η_t is small. We know that the smaller ν is, the smaller $\|Res_i^t\|$ is.

Then we consider the situation that $E_i = E = 1$, and define

$$\begin{aligned}
\nabla F(\omega^t)^\top \mathbb{E}(g_{E=1}(\omega^t)) &\geq \delta_0 \|\nabla F(\omega^t)\|^2 \quad and \\
\mathbb{V}(g_{E=1}(\omega^t)) = \mathbb{E}(\|g_{E=1}(\omega^t)\|^2) &- \|\mathbb{E}(g_{E=1}(\omega^t))\|^2 \\
&\leq L_0 + \|\nabla F(\omega^t)\|^2.
\end{aligned}$$

When $\nabla F_i(\omega_i^{t+1}) = (1-\nu)\nabla F_i(\omega_i^t) + Res_i^t$ is satisfied, we have

$$\begin{aligned}&\mathbb{E}(g_{E=\bar{E}}(\omega^t))\\&= \mathbb{E}(g_{E=1}(\omega^t)(1+(1-\nu)+,...,+(1-\nu)^{\bar{E}-1} + Res'^t))\\&= \frac{1-(1-\nu)^{\bar{E}}}{\nu}\mathbb{E}(g_{E=1}(\omega^t) + Res'^t)).\end{aligned} \tag{34}$$

Based on Taylor expansion, we can get

$$(1-\nu)^{\bar{E}} = 1 - \bar{E}\nu + o(\bar{E}\nu) \tag{35}$$

Then equation 34 can be written as

$$\mathbb{E}(g_{E=\bar{E}}(\omega^t)) = \bar{E}\mathbb{E}(g_{E=1}(\omega^t) + Res'^t), \tag{36}$$

Res'^t is the sum of all Res_i^t, and it is perpendicular $g_{E=1}(\omega^t)$. Then we have

$$\begin{aligned}&\nabla F(\omega^t)^\top \mathbb{E}(g_{E=\bar{E}}(\omega^t))\\&\geq \bar{E}\nabla F(\omega^t)^\top \mathbb{E}(g_{E=1}(\omega^t) + Res'^t)\\&\geq \bar{E}\delta_0\|\nabla F(\omega^t)\|^2 = \delta\|\nabla F(\omega^t)\|^2.\end{aligned} \tag{37}$$

Variance is the second moment of expectation. Based on 37, we have 9. Then using 9 and 8, we complete the proof. □

A.3 Proof of Theorem 3

When assumption 1, 2 are satisfied, and lemma 2 is hold. If $\eta_t = \bar{\eta} \leq \frac{1}{\beta L_G}$ holds and we take expectation at both sides of 20, we have

$$\mathbb{E}(F(\omega^{t+1})) - \mathbb{E}(F(\omega^t)) \leq -\eta_t\frac{\delta}{2}\mathbb{E}(\|\nabla F(\omega^t)\|^2) + \frac{1}{2}\eta_t^2\beta L. \tag{38}$$

Then sum all the form of 38 from 1 to t. We have

$$\begin{aligned}&\mathbb{E}(F(\omega^{t+1})) - \mathbb{E}(F(\omega^1)) =\\&\mathbb{E}(F(\omega^{t+1})) - F(\omega^1) \leq -\eta_t\frac{\delta}{2}\sum_{i=1}^{t}\mathbb{E}(\|\nabla F(\omega^i)\|^2)\\&\qquad + \frac{1}{2}t\eta_t^2\beta L.\end{aligned} \tag{39}$$

Besides, we can easily understand that $F_{inf} \leq \mathbb{E}(F(\omega^{t+1}))$, because F_{inf} is the minimal value of F. Then we have

$$F_{inf} - F(\omega^1) \leq \mathbb{E}(F(\omega^{t+1})) - F(\omega^1) \leq -\eta_t\frac{\delta}{2}\sum_{i=1}^{t}\mathbb{E}(\|\nabla F(\omega^i)\|^2) + \frac{1}{2}t\eta_t^2\beta L. \tag{40}$$

By rearrange 40, we have

$$\mathbb{E}(\sum_{i=1}^{t}\|\nabla F(\omega^i)\|^2) \leq \frac{t\eta_t\beta L}{\delta} + \frac{2(F(\omega^1) - F_{inf})}{\eta_t\delta}. \tag{41}$$

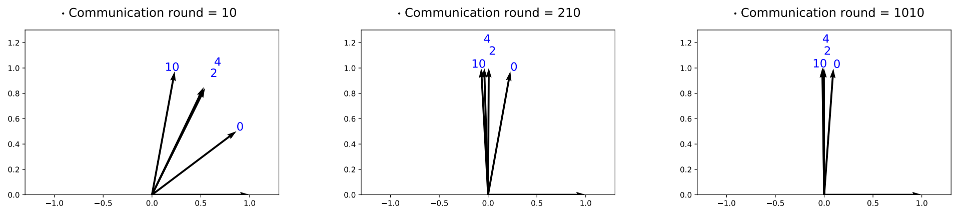
Dividing t from both sides of 41, we get

$$\mathbb{E}(\frac{1}{t}\sum_{i=1}^{t}\|\nabla F(\omega^i)\|^2) \leq \frac{\eta_t\beta L}{\delta} + \frac{2(F(\omega^1) - F_{inf})}{t\eta_t\delta}. \tag{42}$$

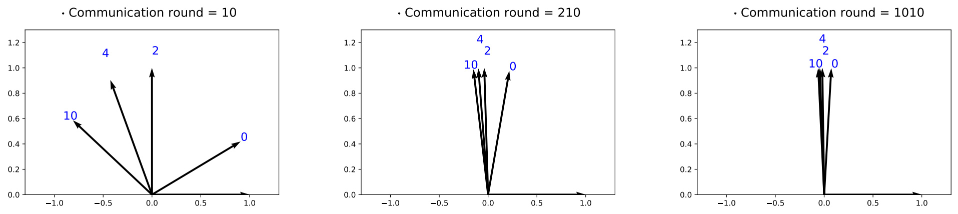
Then using equation 9, we complete the proof. □

B Additional Experiments on Stale-Robustness

We conduct additional experiments to evaluate stale-robustness of our algorithm on CIFAR-10 based on the settings in section 4. We visualize the impact of different staleness levels at different communication rounds with cosine angles (i.e., discrepancies) between the update terms (i.e., update directions of local models) of stale workers and fresh workers in figure 4. The results show that our method (in the first row) effectively adjusts the update direction of the reversed stale nodes while angles of stale nodes reverse with FedAvg compared to our algorithm, which shows the robustness of our method.



(a) Proposed method



(b) FedAvg

Figure 4: Impact visualization of different levels of staleness using cosine angles between the update terms defined in section 2 of fresh nodes (40 out of 100) and stale nodes (60 out of 100 worker nodes) on CIFAR-10 at different communication round. The blue numbers represent the staleness levels by using the differences of version numbers of models between the stale nodes and the fresh nodes. E.g., the staleness level is 10 at this communication round means that the fresh nodes has updated 10 more versions compared to the stale nodes.