Workie-Talkie: Accelerating Federated Learning by Overlapping Computing and Communications via Contrastive Regularization

Rui Chen[1], Qiyu Wan[1], Pavana Prakash[1], Lan Zhang[2], Xu Yuan[3], Yanmin Gong[4], Xin Fu[1], and Miao Pan[1]

[1]University of Houston, [2]Michigan Technological University, [3]University of Delaware, [4]University of Texas at San Antonio

Abstract

Federated learning (FL) over mobile edge devices is a promising distributed learning paradigm for various mobile applications. However, practical deployment of FL over mobile devices is very challenging because (i) conventional FL incurs huge training latency for mobile edge devices due to interleaved local computing and communications of model updates, (ii) there are heterogeneous training data across mobile edge devices, and (iii) mobile edge devices have hardware heterogeneity in terms of computing and communication capabilities.

To address aforementioned challenges, in this paper, we propose a novel "workie-talkie" FL scheme, which can accelerate FL's training by overlapping local computing and wireless communications via contrastive regularization (FedCR). FedCR can reduce FL's training latency and almost eliminate straggler issues since it buries/embeds the time consumption of communications into that of local training. To resolve the issue of model staleness and data heterogeneity co-existing, we introduce class-wise contrastive regularization to correct the local training in FedCR. Besides, we jointly exploit contrastive regularization and subnetworks to further extend our FedCR approach to accommodate edge devices with hardware heterogeneity. We deploy FedCR in our FL testbed and conduct extensive experiments. The results show that FedCR outperforms its status quo FL approaches on various datasets and models.

1. Introduction

Thanks to the hardware advance, edge devices are becoming capable of training deep neural networks (DNNs) on-device [8, 1]. Meanwhile, federated learning (FL) has been emerging as a powerful distributed learning framework which enables collaborative training without sharing the data. FL over edge devices is promising to provide various applications, including e-Healthcare [2], map construction for autonomous driving [36], smart farming in agricultural IoT, etc. One of the most important challenges hindering FL over edge devices from flying is the huge latency of FL training, which is mainly caused by wireless communications between FL server and edge devices, data heterogeneity, and hardware heterogeneity of edge devices.

Popular FL frameworks consider to interleave the computing of local model training and communications of local model updates for FL training [22, 16]. If such FL training is carried out across GPU clusters with Ethernet connection (50-100 Gbps [32]), the network latency ($\leq$1 ms) is negligible. Nevertheless, FL over edge devices are connected wirelessly, so the transmission bandwidth is limited and network latency is too big to ignore. To cut the network latency in wireless communications of model updates, recent research efforts in [38, 37] overlapped local training phase with the wireless transmission phase of model update. The update correction scheme [38] was also developed to handle the issue of model staleness [4], which means using an old version of global model to perform local training during FL training. As demonstrated in our empirical studies (Section 3.1), the existing update correction schemes face significant accuracy drops and fail to improve the system efficiency when the training data of edge devices are heterogeneous.

For most FL cases, local data distributions of edge devices are different from the overall global distribution. When FL clients conduct local training on their own data, the local model updates point to the direction towards the local optima, which may not be consistent with the objective of the aggregated global model. It is called *model drift* issue [13, 17, 18], and result in unstable FL training, which can severely slow down the FL convergence. To address data heterogeneity issue, existing approaches chose to share data across a subset of devices [6], or to correct the local updates via variance reduction [13]. However, those FL methods above cannot deal with the model staleness issue. Besides, they do not consider the hardware heterogeneity of edge devices.

FL clients vary a lot in terms of computing (i.e., CPU, GPU, memory, etc.), and wireless communication (i.e., channel conditions, wireless accessing technologies, etc.) capabilities. Such hardware heterogeneity of edge devices, or device heterogeneity for short, results in huge performance differences among participating edge devices, which occurs straggler problem in FL system, and thus causes huge FL training latency. To handle the straggler issue, some works in the literature [26] developed deadline based schemes that exclude the slow devices after a pre-set timestamp or allow partial updates from the stragglers, which may lead to biased gradient updates that affect FL accuracy.

Aiming to address aforementioned three associated challenges at one strike and reduce the latency of FL over edge devices, in this paper, we propose a novel "workie-talkie" FL scheme, which can accelerate FL's training by overlapping local computing and communications via constrastive regularization (FedCR). To address the *high communication latency* and *data heterogeneity* issues, FedCR overlaps computing and communications, and novelly integrates constrastive learning (CL) into FL local training to reduce the potential training accuracy loss. Since CL helps local models learn close to the global model, it can overcome the model drift issue caused by *data heterogeneity*. To deal with the model staleness caused by the overlapping scheme, we develop a novel class-wise contrastive loss in the model level to smooth the stale global model. In this way, FedCR can fully exploit the benefits of contrastive regularization to achieve a fast and stable FL training over edge devices. Moreover, to reduce corresponding computing overheads in CL and tackle *device heterogeneity* issue, we extend our FedCR to enable heterogeneous local model training over edge devices. That helps to reduce local computing time and the size of model updates for communications throughout FL training process. Our salient contributions are summarized as follows.

- To address high communication latency and data heterogeneity issues, we propose FedCR, a novel FL approach to accelerate FL's training over edge devices by overlapping computing and communications and integrating CL into local training to regularize local updates.
- To resolve model staleness issue, we propose a class-wise contrastive regularization method. By assigning class-wise temperature for each edge device, we aim to properly align the local models with the global model.
- To address device heterogeneity issue, FedCR allows edge devices to select and train a subnetwork of the original DNN based on their available resources. We develop a temperature scaling strategy to accommodate heterogeneous local model training in FedCR.
- We set up testbeds and conduct extensive experiments to verify the effectiveness of the proposed FedCR approach under various learning models, different data distributions across heterogeneous edge devices, and multiple wireless transmission settings.

2. Background

2.1. Federated Learning

We consider an FL system consisting of one FL server and N edge devices. Each device has its own dataset with D_n samples. All edge devices attempt to jointly learn a global DNN model $\mathbf{w}$ under the coordination of the FL server. A standard global objective of FL is

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}) := \sum\nolimits_{n=1}^{N} \pi_n F_n(\mathbf{w}), \tag{1}$$

where $\pi_n = D_n/\sum_{n=1}^{N} D_n$ is the aggregation weight of device n and $F_n(\mathbf{w})$ is the *local* loss function of device n. FedAvg was proposed in [22] to solve the problem in Eqn. (1). In each round r, the FL server first broadcasts global model $\mathbf{w}_g^r$ to the participating devices. Then, each edge device n let $\mathbf{w}_n^{r,0} = \mathbf{w}_g^r$ and then perform K local training iterations with its local dataset. Last, the FL server aggregates the local models, $\{\mathbf{w}_n^{r,K}\}_{n=1}^{N}$ to produce the new global model, $\mathbf{w}_g^{r+1}$ and send it back to the devices. This procedure repeats until FL global model converges. The overall training time of *synchronous* FL is

$$T_{sync} = R_{sync} \cdot \max_{1\le n\le N}\{K\cdot T_n^{cp} + T_n^{cm}\}, \tag{2}$$

$$= R_{sync} \cdot (K\cdot T_n^{cp} + T_n^{cm} + T_n^{wait}), \forall n. \tag{3}$$

Here, R_{sync} is the total communication rounds, T_n^{cp} and T_n^{cm} are the average computing time for one training iteration and the average communication time per FL round of n-th device, respectively. Given straggler issues in synchronized FL systems, per-round training time is determined by the slowest device. In other word, besides the computing and communication time, participating devices also have waiting time $T_n^{wait} \ge 0$.

2.2. Computing and Communication Overlapping

An efficient scheme for reducing T_{sync} in Eqn.(2) is to overlap local model training and model aggregation to mask out the communication time and waiting time. Generally speaking, different from FedAvg, instead of waiting for the global model of the next communication round, every device n *immediately* starts performing local training using its local model $\mathbf{w}_n^{r,K}$. Specially, when edge devices receive the global model $\mathbf{w}_g^{r+1}$ in the $(t+1)$-th round, device n has already run additional S_n local training iterations and updated the model to $\mathbf{w}_n^{r,K+S_n}$. In other words, the resultant weights are typically computed with respect to outdated parameters (i.e., the global parameters from the previous round). Denote **staleness** of device n as the number of local training iterations performed during the model transmission, i.e., $S_n = \lceil (T_n^{cm} + T_n^{wait})/T_n^{cp} \rceil$. The **staleness of system** is determined by the device with largest staleness level, i.e., $S = \max_n\{S_n\}$.

The overall training time, denoted as T_{ol}, of those overlap FL training framework is given as,

$$T_{ol} = \max_{1\leq n\leq N}\{R_{ol}\cdot K\cdot T_n^{cp} + T_n^{cm}\}, \tag{4}$$

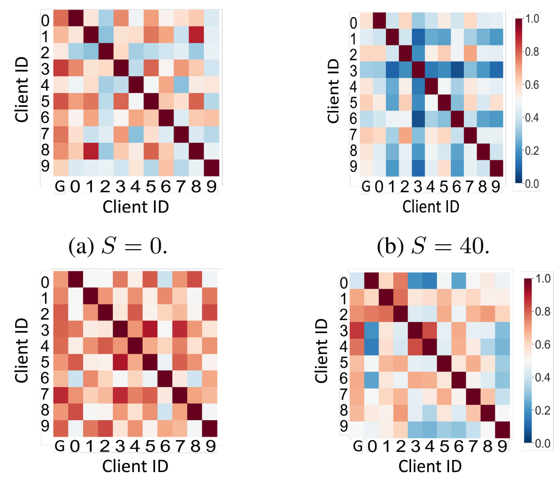
$$\approx \max_{1\leq n\leq N}\{(R_{ol}\cdot K + S_n)T_n^{cp}\}, \tag{5}$$

where R_{ol} is the total number of communication rounds run in the overlapping learning process. Hence, the system speedup can be derived as,

$$Ratio = \frac{T_{sync}}{T_{ol}} = \frac{R_{sync}\max_n\{(K+S_n)T_n^{cp}\}}{\max_n\{(R_{ol}K+S_n)T_n^{cp}\}}. \tag{6}$$

Overlapping learning can be achieved by either identifying the optimal gradient transfer order [9], using staled model weights [20], or using delayed model updates [27, 38]. In particular, DGA [38], the state-of-art overlapping scheme, aimed to correct the mismatch caused by the overlapping. DGA replaced the old local gradients from several iterations before with stale averaged gradients. The comparison of training timeline between synchronization and different overlapping schemes is shown in *Supplementary Material*.

However, existing overlapping learning frameworks have three main weaknesses. 1) Many of these methods incur substantial overheads especially in memory. For instance, in DGA, each edge device has to locally store multiple copies of its old local model updates for update correction. Thus, the memory consumption increases linearly with the staleness level (See *Supplementary Material* for more details). It becomes difficult to implement FL algorithm on resource limited edge devices in wireless network environment. 2) Existing overlapping designs fail to consider data heterogeneity issue. Hence, when data heterogeneity and severe staleness issues co-exist, it would extremely slowdown convergence and fail to reach the desired accuracy. Thus, it is possible that the overlapping designs based on model update correction perform worse than FedAvg (i.e., $Ratio < 1$ in Eqn.(6)). 3) Since the correction design in the existing overlapping scheme requires the local model to have the same model architecture as the global model, it is difficult for these low-end devices to participate in the large model training. Even if they can participate, those low-end devices will easily become stragglers, which may aggravate the staleness issue in existing overlapping approaches.



(a) $S = 0$. (b) $S = 40$. (c) $S = 0$. (d) $S = 40$.

Figure 1: CKA similarities vs staleness. Client ID G represents the global model. (a)-(b): using DGA for FL training; (c)-(d): using contrastive learning for local training in FL.

3. Overlapping FL with Contrastive Regularization

3.1. Motivation

The memory inefficiency and accuracy degradation of DGA root from the fact that its design of *one-step compensation for the local gradients after the local training* cannot effectively reduce the huge model divergence issue. Theoretically, the divergence between the local updates and the aggregated updates in DGA is upper bounded with a factor of staleness level S (as stated in Lemma 2.2 in [38]). A large S value leads to a loose upper bound, indicating slow convergence. More importantly, this upper bound becomes more loose under data heterogeneity scenarios. However, the gradient compensation cannot further reduce the model divergence.

We next empirically examine how staleness affects the feature space of the local and global models by training a ResNet20 on CIFAR-10 dataset (please see *Supplementary Material* for more experiment details). Here, we use Centered Kernel Alignment (CKA) [14] to measure the similarity of the output features between local models and the global model, given the same testing datatset. The results are shown in Fig. 1. CKA outputs a similarity score between 0 (not similar at all) and 1 (identical). From Figs. 1a and 1b, we can observe that the average similarity between the local models and the global model greatly decreases (from 0.72 to 0.51) as the staleness level increases (from $S = 0$ to $S = 40$). It suggests that the gradient correction in DGA fails to reduce the model drifts issue when a large staleness occurs. Thus, it results in a poor global model performance.

Instead, we want to regulate the local model during the local training by introducing a feature-aligned regularization term. A fascinating attribute of feature-aligned local training is the preservation of global information that is absent from the local data distribution during local training. From Figs. 1a and 1b, FedCR achieves much higher pairwise CKA similarity compared with DGA at different staleness levels. As expected, the feature learn from local models is more aligned with the global feature space, which will reduce model drift. It is especially beneficial in FL with overlapping designs.

3.2. FedCR Design

The goal of FedCR is to reduce the training latency by overlapping communication and computation in FL training. To deal with model divergence when data heterogeneity and staleness issues co-exist, we introduce a class-wise contrastive regularization (CR) into local training. This class-wise CR provides guidance to local training so that the representations of local models to be well aligned to the global model. As such, edge devices are no longer required to store multiple copies of old model updates, which saves memory footprint in the case of high network latency or severe straggler issues. In the following sections, we present the network architecture, the class-wise CR design, and the learning procedure. Then, we describe how to incorporate subnetwork scheme in FedCR to address device heterogeneity issue.

Network architecture. The deep neural network in FedCR is decoupled into three parts: a network encoder $f_{enc} : \mathcal{X} \to \mathcal{H}$ that maps the input space $\mathcal{X}$ to the representation space $\mathcal{H}$, a projection head $f_{pjt} : \mathcal{H} \to \mathcal{Z}$ that maps the high-dimensional representation to a low-dimensional embedding, and a linear classifier $f_{cls} : \mathcal{Z} \to \mathcal{Y}$ to produce classification output at the target space $\mathcal{Y}$. Note that given the limited resource of edge devices, we consider a simple and *fixed* projection head that outputs a sub-vector of the representation vector. Recent study in [12] shows that such fixed low-rank diagonal projector can outperform a linear trainable projector.

Let $\boldsymbol{z}^{glob}_{j,n} = \boldsymbol{z}^{glob}_{j} = f_{pjt}(f_{enc}(\mathbf{w}^r_g, x_j)), \forall n$ be the projected representation of the input x_j in <u>glob</u>al model $\mathbf{w}^r_g$, $\boldsymbol{z}^{cur}_{j,n} = f_{pjt}(f_{enc}(\mathbf{w}^{r,k}_n, x_j))$ be the representation vector of the input x_j in <u>cur</u>rent local model $\mathbf{w}^{r,k}_n$ of device n, and $\boldsymbol{z}^{prev}_{j,n} = f_{pjt}(f_{enc}(\mathbf{w}^{r,K}_n, x_j))$ be the representation vector of the input x_j in <u>prev</u>ious local model $\mathbf{w}^{r,K}_n$ of device n. All embeddings are ℓ_2-normalized for inner product.

Class-wise CR design. The *goal* of CR is to enforce the representation of local model (i.e., $\boldsymbol{z}^{cur}_{j,n}$) be close to the one of the global model (i.e., $\boldsymbol{z}^{glob}_{j,n}$) to handle the model drift issue. Inspired by MOON [18], we define feature-align

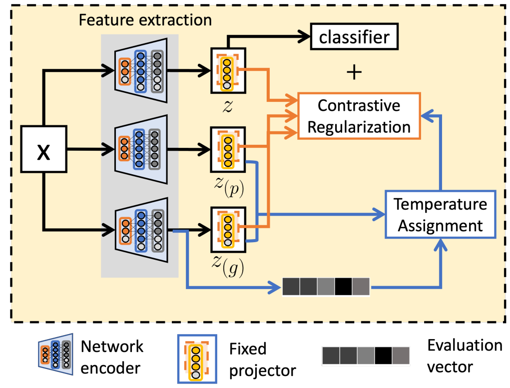


Figure 2: Local training in FedCR.

regularization of one data sample x_j as follows,

$$\mathcal{R}_n(x_j, \tau_n) = -\log \frac{\exp(\boldsymbol{z}^{glob}_{j,n} \cdot \boldsymbol{z}^{cur}_{j,n} / \tau_n)}{\sum_{i \in \{glob, prev\}} \exp(\boldsymbol{z}^{i}_{j,n} \cdot \boldsymbol{z}^{cur}_{j,n} / \tau_n)}, \quad (7)$$

where τ_n is temperature parameter of device n. The numerator in Eqn. (7) aims to maximize the similarity between current local model and global model and the denominator in Eqn. (7) minimize the similarity between current local model and previous local model. Different from MOON that uses a constant and identical temperature τ to all the edge device, we consider adaptive temperature assignment to capture the difference between the local data distribution of each device and the global distribution, and the change of feature learned by the global model.

In Eqn. (7), τ_n controls the strength of encouraging the feature learned by the device n's local model to be similar to that of the global model. When the local model lacks of sufficient training samples to learn the comprehensive representation and the global model is reliable, we can adjust the τ_n to encourage the local models learn from the more established global model. Hence, we define the class-wise temperature t_c in class $c \in \mathcal{Y}$ as

$$\tau^c_n = p_{c,g} \cdot \frac{D_n}{|\mathcal{S}^c_n| + \alpha} \quad (8)$$

where $\mathcal{S}^c_n$ denotes the set of indices satisfying $y = c$ in device n's local dataset. α is a large non-zero value. Here, $\mathbf{p}_g = [p_{1,g}, \cdots, p_{|\mathcal{Y}|,g}]$ is validation accuracy evaluated on the auxiliary dataset $\mathcal{A}$ in the server side and the $\mathbf{p}$ is send to the devices along with the global model. It reflects the performance of the global model $\mathbf{w}^r_g$ on class c. When $|\mathcal{S}_c|$ is small and accuracy of global model in class c, $p_{c,g}$ is high, the penalty strength of feature dissimilarity with large τ^c_n are down-scaled, pulling the local model closer to the global one. On the contrary, the penalty strength of feature dissimilarity with small τ^c_n are up-scaled, thus less encouraged to approach the global model.

Local objective. The class-wise CR of device n with local dataset $\mathcal{D}_n$ is formulated as

$$\mathcal{R}_n = \sum_{c \in \mathcal{Y}} \sum_{j \in \mathcal{S}_n^c} - \log \frac{\exp\left(\boldsymbol{z}_{j,n}^{glob} \cdot \boldsymbol{z}_{j,n}^{cur} / \tau_n^c\right)}{\sum_{i \in \{glob, prev\}} \exp\left(\boldsymbol{z}_{j,n}^{i} \cdot \boldsymbol{z}_{j,n}^{cur} / \tau_n^c\right)}. \tag{9}$$

The local objective of device n in FedCR is formulated as

$$F_n(\mathbf{w}_n) = \mathcal{L}_n^{CE}(\mathbf{w}_n) + \lambda \mathcal{R}_n, \tag{10}$$

where λ is to control the weight of contrastive regularizer. The first part is a cross-entropy loss term in supervised learning denoted as $\mathcal{L}_n^{CE}$. The second part is our proposed CR term denoted as $\mathcal{R}_n$. The consequent benefits are obvious: it *saves massive memory footprint*, since edge devices no longer need to store multiple copies of old local model updates. It comes at the cost of additional but constant computation overheads regardless of staleness levels. In particular, since the local model trained with one edge device's own dataset is better aligned with the global model that aggregates the knowledge from all edge devices' data, it also *mitigates model drift problems* caused by data heterogeneity. The overall framework of local training is shown in Fig. 2.

Heterogeneous subnetwork training. FedCR requires additional requirement of storing three full-size models, which may be overwhelming for low-end edge devices. To solve this issue, a straightforward solution is to train customize models on edge devices based on their available resources. Luckily, FedCR does not require the network architecture to be exactly the same on different devices since FedCR uses a fixed projector to maps the feature representation of an arbitrary shape into a fixed dimension. In this way, FedCR can be easily extended to support heterogeneous local model training.

Following [11], we consider to use a sub-structure of the original network encoder as a new encoder that can satisfy the computing and memory requirement of one edge device. The new encoder along with the projection head and the classifier are defined as a subnetwork. The subnetwork design is parameterised w.r.t the partition rate $p \in (0, 1]$ per layer. More specifically, given a specific partition rate p, for each layer l with width d_l, the neurons with index $\{0, 1, \dots, \lceil p \cdot d_l - 1 \rceil\}$ kept in layer l and drops out the rest neurons. Partition is not applied on the input, the fixed projector for CR, and last layer to maintain the same dimensionality. Under such nested order subnetwork partitioning, the left-most features are always used for all the devices during training. As such, our fixed projector head in CR can help the small local model learn the representation be to aligned with that of the global model.

There is a potential issue if we use smaller local subnetwork. Intuitively, large models learn better representation than small ones. In the FL scenario, a subnetwork with

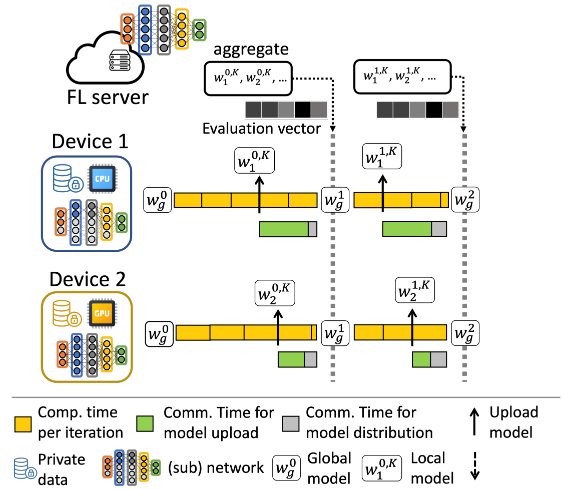


Figure 3: Overall framework of FedCR.

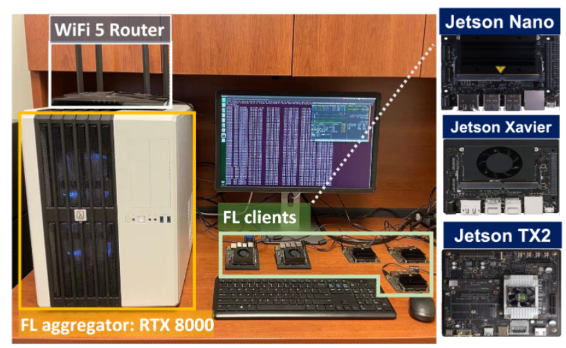


Figure 4: FedCR testbed in our lab.

smaller size should pay more attention to penalize its local model, if it's far away from the large global model. Here, we reweight the temperature strategy of device n by multiplying τ_n with the partition ratio p. Therefore, τ_n is designed to decrease when the devices choose a smaller local model.

4. Implementation

Figure 3 illustrates the overall framework of FedCR. Specifically, after an edge device n completes K local updates, it sends its updated local model $\mathbf{w}_n^{0,K}$ to the FL server. During the model uploading, device n will not stop and wait, but continue local training until the new global model is received. After the first communication round, each device utilize the evulation of the global model and the proposed class-wise CR to guide the local model training. On the FL server side, he aggregates the global model using weighted averaging and evaluate the aggregated global model using a public dataset. The the new global model and evaluation vector of global model will be sent back to the edge devices.

The FedCR is implemented on testbed. As illustrated in Fig. 4, it consists of an FL aggregator and several FL clients. On FL edge server side, we use a NVIDIA RTX 8000 with 4 GPUs. The FL server and FL clients are wirelessly connected via a Wi-Fi 5 router. On FL edge device side, we consider three types of edge devices: NVIDIA Jetson Xavier NX, NVIDIA Jetson TX2, and NVIDIA Jetson Nano. Their profiles are summarized in *Supplementary Material*. For the communications between FL aggregator and clients, we follow the WebSocket [7] communication protocol and use bmon [23] to measure wireless transmission speed.

5. Evaluation

5.1. Experiment Setup

Datasets and models. Our experiments consider three image classification tasks, three CNN models: CNN, ResNet20, and ResNet34 [10], and three open datasets: CIFAR-10, CIFAR-100, and Tiny-ImageNet[1]. For CIFAR-10, we use a CNN network which has two 5x5 convolution layers followed by 2x2 max pooling and two fully connected layers with ReLU activation. For CIFAR-100, we use ResNet20 model as the base encoder. For Tiny-ImageNet, we use ResNet34 model as the base encoder. The auxiliary dataset in the FL server side consists a small number of data samples with size of 32 for different classes. In our experiments, we follow the strategy from [31] to sample auxiliary data for CIFAR-10 and CIFAR-100, Tiny-ImageNet. The number of local training iterations $K = 20$ for all the datasets. Other hyper-parameter settings are provided in *Supplementary Material*.

Data partition. We consider FL training with non-IID cases. We use Dirichlet distribution with four different concentration parameters $\beta \in \{0.1, 0.5, 1, 5\}$. A small β indicates a high data heterogeneous level. Without specific explanations, we consider $\beta = 0.5$ in the following experiments.

Device distribution. The proportion of these three device categories is $0.1 : 0.4 : 0.5$, which is representative of in-the-field system performance distribution [33]. According to the testbed configuration, we consider Xavier NX as high-end devices, TX2 as medium-end devices, and Nano as low-end devices. We consider the high-end devices to train the model with $p = 1$ (full size), the medium-end devices with $p = 0.6$, and the low-end devices with $p = 0.2$. Hence, only 10% clients train the network with $p \leq 1$, 40% clients train with $p \leq 0.6$ and the rest clients train with $p \leq 0.2$.

Wireless transmission settings. We consider two different wireless transmission scenarios: *indoor* and *outdoor* environments, where the transmission speeds are 100Mbps and 20Mbps, respectively. According to the model size and devices used in our experiments, we estimate the corresponding staleness levels are 10 and 40 for CIFAR-10, respectively. Without specific explanations, we consider outdoor environment in the following experiments.

Peer FL designs for comparison. We compare FedCR with DGA [38], LGC [34], Overlap-Prox [19], Overlap-MOON, and Overlap-HeteroFL. LGC [34] allows straggler devices to postpone their (partial) model updates by one communication round. In Overlap-Prox,Overlap-MOON and Overlap-HeteroFL, we slightly modify original FedProx [18], MOON [18], and HeteroFL [5] by overlapping the computing and communications, respectively. For Overlap-Prox, there is a hyperparameter μ_{prox} to control the weight of its proximal regularizer (i.e., $F_{prox} = \mathcal{L}^{CE} + \mu_{prox}\mathcal{R}_{prox}$). We set μ_{prox} to 0.1 by default like [19]. For local model in Overlap-MOON, we follow the original MOON [18] and use 2-layer MLP as a trainable projector head to construct positive and negative pairs. The output dimension of the projection head is set to 256 and the temperature parameter τ is 0.5 for all devices and λ is 5.

5.2. FL Performance Comparison

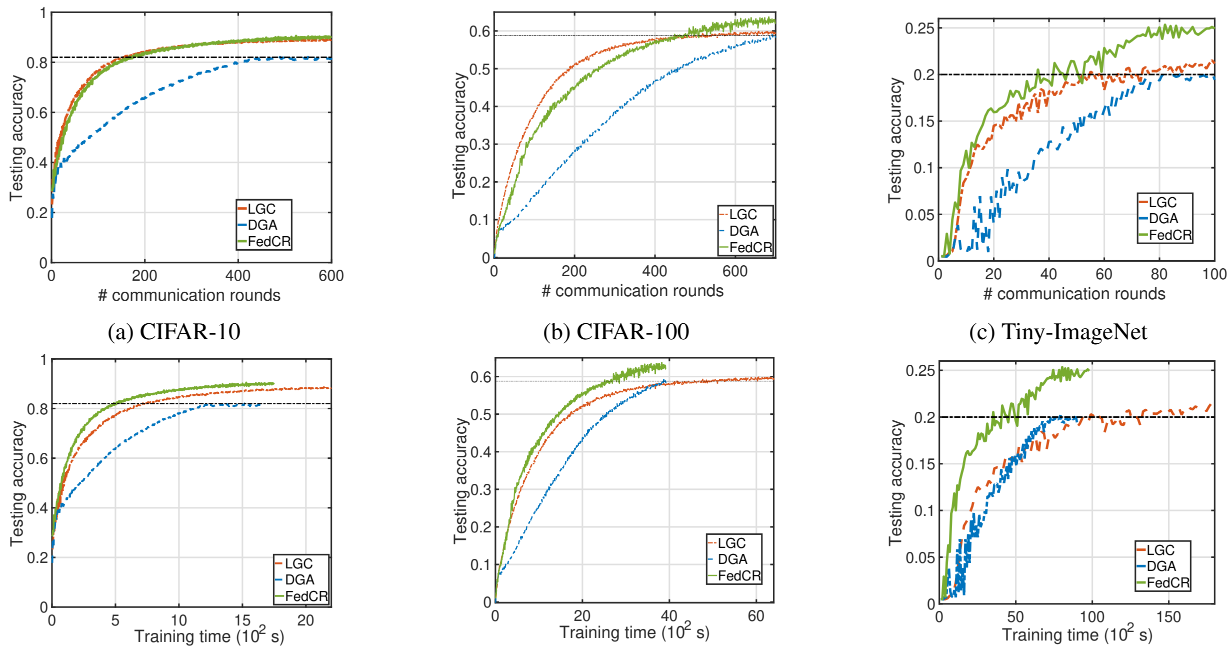
Convergence rate vs overall training time. Figure 5 shows the comparison of different FL schemes in terms of testing accuracy vs communication rounds/FL training time. In general, we observe that FedCR outperforms its peer FL schemes across different image classification tasks. Take CIFAR-100 as an example in Fig.5b and Fig.5e, FedCR speedups the training process by $1.52\times$ and $2.01\times$ for reaching target accuracy of 58.8%, compared to DGA and LGC, respectively. FedCR increases the testing accuracy by 5.8% and 2.8% with 700 rounds, compared with DGA and LGC, respectively. FedCR uses $1.65\times$ and $2.67\times$ less communication rounds on Tiny-ImageNet for reaching target accuracy of 20% as shown in Fig. 5a, compared to DGA and LGC, respectively.

As expected, FedCR outperforms DGA. As discussed in Sec. 3.1, when DGA fails to tightly control the model divergence (See Fig. 1), it makes the latency reduction of overlapping communication with computing less effective. By contrast, FedCR can mitigate the model divergence to reduce the total number of FL rounds and training time.

FedCR outperforms LGC, since LGC aims to overcome the straggler issue but still cannot handle the big latency of wireless communications. The devices still need to wait for the new aggregated global model to start the local training for the next round. FedCR, on the other hand, buries the communication latency into the local computing time to address both communication latency and straggler issues.

Fixed and Dynamic Regularization. We evaluate the effectiveness of the proposed class-wise CR. We compare FedCR with Overlap-Prox, DGA, and Overlap-MOON. Overlap-Prox aims to align the weights of local model with those of the global one and has a small memory overhead (only store

[1]https://www.kaggle.com/c/tiny-imagenet



(a) CIFAR-10 (b) CIFAR-100 (c) Tiny-ImageNet

(d) CIFAR-10 (e) CIFAR-100 (f) Tiny-ImageNet

Figure 5: The convergence of testing accuracy over iterations and time. (a)-(c) illustrate the accuracy after training for a given round (i.e., 600 rounds for CIFAR-10, 700 rounds for CIFAR-100, and 100 rounds for Tiny-ImageNet). (d)-(f) show the overall training time for reaching the target accuracy (i.e., 81% for CIFAR-10, and 58.8% for CIFAR-100, 21% for Tiny-ImageNet)

.

Table 1: The number of parameters are counted for each communication round (# Para). The 'M' after metric values means $\times 10^6$. The speedup is evaluated using Eqn. (6). The testing accuracy is on CIFAR-100. We run three trials and report the mean and standard derivation.

Methods	Acc	#Para	Speedup
DGA	$58.8\% \pm 1.1\%$	163M	**2.82**
Overlap-Prox	$56.1\% \pm 0.4\%$	**86M**	**2.82**
Overlap-MOON	**$63.6\% \pm 0.8\%$**	127M	2.18
FedCR	$62.3\% \pm 0.5\%$	122M	2.58

two models locally). FedCR and Overlap-MOON have both demonstrated significant improvements in accuracy compared to Overlap-Prox and DGA. It indicates that using CR is more effective than using gradient correction and weight alignment in the situations of large network latency. Overlap-MOON with the trainable projector has better testing accuracy, while the training time is longer, compared with the proposed FedCR. In Table 1, it is observed that the accuracy gain of Overlap-MOON is $1.02\times$, while the training time per iteration increases by $1.2\times$. This is because although Overlap-MOON introduces a trainable projector in CR, it extracts good representation of learned models at the cost of higher model complexity, leading to longer training time per local iteration and more memory usage. By contrast, FedCR uses a fixed and non-trainable projector, and class-wise CR to accelerate local training. More experiments related to the impact of training budget can be found in *Supplementary Material*. Compared with the peer schemes, FedCR achieves the best trade-off among the learning performance, memory size and training time speedup.

Table 2: The test accuracy with different temperature assignments.

Methods	CIFAR-10		CIFAR-100	
	$\beta = 0.5$	$\beta = 5$	$\beta = 0.5$	$\beta = 5$
Fixed τ	82.2%	87.2%	61.3%	64.7%
Avg τ_n	82.7%	87.8%	61.9%	65.1%
Class-wise τ_n	**83.1%**	**88.2%**	**62.3%**	**65.4%**

Temperature Design. Here we compare different temperature strategies. 'Fixed τ' assigns the fixed and same temperature to all the devices; 'Avg τ_n' is the average value of τ_n^c in Eqn. 8 and does not consider the class-aware information; 'Class-wise τ_n' is our proposed scheme in FedCR. For 'Fixed τ', we tune τ from $\{0.2, 0.5, 1\}$ and select the

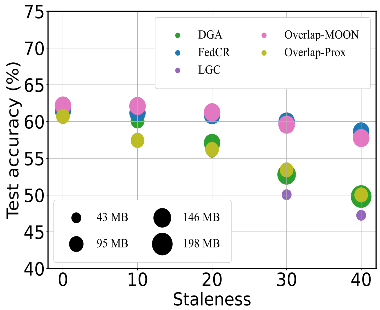


Figure 6: The test accuracy and memory usage with varying stalenessß.

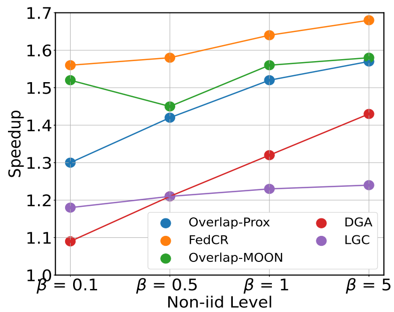


Figure 7: Training speedup with varying β values.

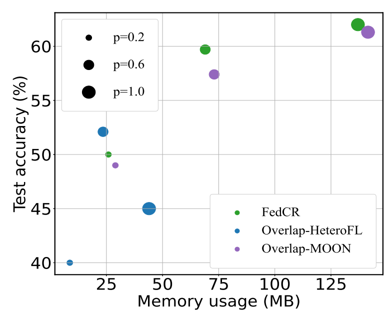


Figure 8: The testing accuracy and memory usage under different p-subnetwork.

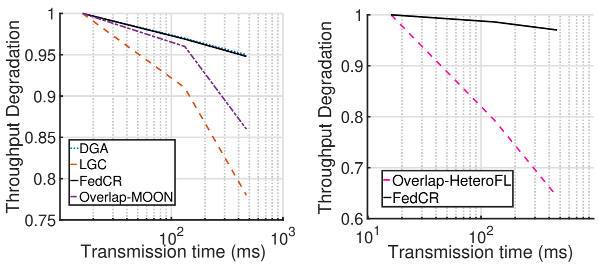
one with highest testing accuracy. The results are shown in Table 2. We can observe that FedCR benefits from the class-wise temperature assignment. As β becomes smaller, the accuracy improvement of class-wise temperature assignment becomes larger. It indicates that the class-wise scheme is effective, especially when the client data distribution is highly class-imbalanced.

Impact on Staleness According to the staleness definition in Sec. 2.2, we denote the the comm./comp. ratio as $\frac{S}{K} = \frac{T_{cm}+T_{wait}}{KT_{cp}}$. For different bandwidth and learning task settings, we can generally categorize the ratio into three cases: (1) $\frac{S}{K} = 0$, (2) $0 < \frac{S}{K} \leq 1$, and (3) $\frac{S}{K} > 1$. For Case 1, FedCR is the same as FedAvg. For Cases 2 & 3, FedCR can save a lot of memory spaces during FL training process since it uses CR with fixed projector. Fig. 6 show the testing accuracy under the given time budget (i.e., 2500s). We can observe that the memory usage of DGA linearly grows as staleness increases due to its gradient correction design. Moreover, FedCR can achieve higher testing accuracy with smaller memory usage compared with DGA and Overlap-MOON. Since LGC is not a overlapping scheme, it has the least memory usage while suffering from low testing accuracy. Similar trend can be observed in Overlap-Prox. Compared with Overlap-MOON, FedCR has slightly lower accuracy with less memory usage when the staleness levels is small. We later show that Overlap-MOON incurs much longer training time than FedCR to achieve the same accuracy.

Data Heterogeneity We show the training speedup of different schemes to reach the same testing accuracy in Fig. 7. We study the impacts of data heterogeneity by varying β from 0.1 to 5 on CIFAR-100. For a smaller β, the partition will be more unbalanced. Compared with the peer schemes, FedCR achieves much better speedup among different levels of data heterogeneity. The regularization methods, FedCR, Overlap-MOON, and Overlap-Prox, are more time-efficient, compared with DGA and LGC. Compared with Overlap-Prox, the contrastive regularizor can further speedup the training time. Compared with Overlap-MOON, FedCR use fixed and non-learnable projection to reduce the training time to achieve the same testing accuracy.

Device heterogeneity. Next, we consider FL with heterogeneous model training. Since the peer schemes except for Overlap-HeteroFL cannot support heterogeneous model training, we consider FL training for each p-subnetwork for those peer schemes. In this experiment, we only compare with Overlap-MOON since it has the best testing accuracy in previous experiments. From Fig. 8, we can observe that as p increases, FedCR can improve the testing accuracy of p-subnetwork. Compared with Overlap-HeteroFL, FedCR increases the test accuracy by 23%, 15%, 37% on different p-subnetworks, respectively. Surprisingly, the subnetwork training improves the testing accuracy of global model in our FedCR. Our global model with $p = 1$ has the best accuracy with 61.9%, which is better than Overlap-MOON with 60.3%. It is worth a further analysis and we would like to leave it as future work. Besides, FedCR also improves the accuracy of small network (i.e., $p = 0.2$).

Training Throughput Comparison In Fig. 9, we report the throughput degradation of training a ResNet20 with different wireless transmission speeds. Fig. 9a shows local model training with full local model size. It is observed that when wireless transmission speed gradually decreases (transmission time increases), LGC's performance degrades sharply. On the other hand, DGA and our proposed FedCR yields a relatively stable speed. Both of them has similar and good performance. However, DGA cannot support the heterogeneous local model training. By contrast, our proposed FedCR enables heterogeneous training to further boost the training throughput. Fig. 9b shows the training throughput of FL training with heterogeneous local model sizes. It is observed that the training throughput of FedCR remains high



(a) Homogeneous local model. (b) Heterogeneous local model.

Figure 9: Training throughput degradation under different transmission times, which is calculated based on the measured transmission speed.

Table 3: The accuracy with 64 devices and 128 devices (sample fraction = 0.2) on CIFAR-100 under two time budgets ($10 \times 10^2 s$ and $25 \times 10^2 s$).

Methods	N=64		N=128	
	10	25	10	25
LGC	42.65%	37.82%	41.65%	48.82%
Overlap-Prox	46.12%	52.79%	45.57%	52.12%
DGA	42.89%	51.98%	41.56%	50.12%
Overlap-MOON	51.12%	58.01%	50.47%	56.14%
FedCR	**56.27%**	**64.68%**	**55.47%**	**61.58%**

and steady even under slow wireless communications. More experiments regarding a dynamic communication condition can be found in *Supplementary Material*.

Impacts of Device Numbers We evaluate the impact of the number of participating edge devices in each communication round on FedCR performance. Specifically, we consider two different simulation settings: (1) We partition the dataset into 64 devices and all devices participate in each round. (2) We partition the dataset into 128 devices and randomly sample 20% devices to participate in each round. As shown in Table 3, using CR in local training is more effective than using gradient correction and proximal regularization, FedCR reaches the highest testing accuracy given a target time budget, especially when it has large number of devices in FL system.

6. Related Work

Recognizing that training large-scale FL models over edge devices is a time consuming task, several research efforts have been made on decreasing these costs via device scheduling [15], network optimization [35, 25] and resource utilization optimization [30, 28, 29, 24]. For example, Lai et al. in [15] proposed a client selection scheme for FL training based on both the hardware capabilities and the training quality of clients. Luo et al. in [21] studied the tradeoffs between FL training latency and energy and proposed cost-efficient design to determine the design the number of local updates for minimizing both training latency and energy. Rui et al. in [3] and Shi et al. in [28] have made efforts on time-efficient FL design on how to determine the optimal learning parameters (e.g., quantization level and the number of local updates). However, the previous works only consider a FL setting with interleaved computing and communication. Our study is orthogonal to them and potentially can be combined with these techniques to further boost the training efficiency as our design proposes to overlap computing and communication in FL. Close to our work, DGA [38] and CoCoD-SGD [27] also consider a overlapping scheme to reduce the training latency. As stated before, there is limited performance gain when directly applying these techniques to non-iid data distribution across devices. Besides, they do not deal with device heterogeneity. Such limited performance gain motivates us to rethink the computing and communication overlapping framework design for federated training over edge devices that takes data and device heterogeneity into account.

7. Conclusion

In this paper, we have presented a novel FL approach, FedCR, which accelerates FL training over edge device while maintaining high training accuracy. Our key idea is to overlap local model communications (i.e., transmitting local model updates) with local training to reduce the latency of wireless communications, and introduces CR into local training to address data heterogeneity issue. Furthermore, we have developed class-wise CR and heterogeneous sub-network training methods to deal with the model staleness and device heterogeneity issues in FedCR, respectively. Extensive experimental results have demonstrated that FedCR not only effectively reduces FL training latency, but also achieves higher training accuracy and smaller memory footprints than the state-of-the-art solutions.

Acknowledgement

The work of R. Chen and M. Pan was supported in part by the National Science Foundation Grants (NSF) under Grant CNS-2107057. The work of Q. Wan and X. Fu was partially supported by NSF under Grants CCF-2130688, CCF-1900904, and CNS-2107057. The work of L. Zhang was supported in part by NSF CCF-2106754, CCF-2221741, CCF- 2153381, and CCF-2151238. The work of X. Yuan was supported in part by NSF under Grant 2146447. The work of Y. Gong was supported in part by the US National Science Foundation under grant CNS-2047761 and CNS-2106761.

References

[1] Apple. Coreml on device training, mar 2021.

[2] Theodora S Brisimi, Ruidi Chen, Theofanie Mela, Alex Olshevsky, Ioannis Ch Paschalidis, and Wei Shi. Federated learning of predictive models from federated electronic health records. *International journal of medical informatics*, 112:59–67, 2018.

[3] Rui Chen, Dian Shi, Xiaoqi Qin, Dongjie Liu, Miao Pan, and Shuguang Cui. Service delay minimization for federated learning over mobile devices. *arXiv preprint arXiv:2205.09868*, 2022.

[4] 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.

[5] Enmao Diao, Jie Ding, and Vahid Tarokh. Heterofl: Computation and communication efficient federated learning for heterogeneous clients. *arXiv preprint arXiv:2010.01264*, 2020.

[6] Moming Duan, Duo Liu, Xianzhang Chen, Renping Liu, Yujuan Tan, and Liang Liang. Self-balancing federated learning with global imbalanced data in mobile systems. *IEEE Transactions on Parallel and Distributed Systems*, 32(1):59–71, 2020.

[7] Ian Fette and Alexey Melnikov. The websocket protocol, 2021.

[8] Andrew Hard, Chloé M Kiddon, Daniel Ramage, Francoise Beaufays, Hubert Eichner, Kanishka Rao, Rajiv Mathews, and Sean Augenstein. Federated learning for mobile keyboard prediction, 2018.

[9] Sayed Hadi Hashemi, Sangeetha Abdu Jyothi, and Roy Campbell. Tictac: Accelerating distributed deep learning with communication scheduling. In *Proceedings of Machine Learning and Systems*, volume 1, pages 418–430, Stanford, CA, March 2019.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. arxiv 2015. *arXiv preprint arXiv:1512.03385*, 2015.

[11] Samuel Horvath, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos Venieris, and Nicholas Lane. Fjord: Fair and accurate federated learning under heterogeneous targets with ordered dropout. *Advances in Neural Information Processing Systems*, 34:12876–12889, 2021.

[12] Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. In *Proceedings of the 10th International Conference on Learning Representations (ICLR'22)*, Baltimore MD, July 2022.

[13] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

[14] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning (ICML'19)*, pages 3519–3529, Long Beach, California, June 2019. PMLR, JMLR.

[15] Fan Lai, Xiangfeng Zhu, Harsha V Madhyastha, and Mosharaf Chowdhury. Oort: Efficient federated learning via guided participant selection. In *15th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 21)*, pages 19–35, 2021.

[16] Liang Li, Dian Shi, Ronghui Hou, Hui Li, Miao Pan, and Zhu Han. To talk or to work: Flexible communication compression for energy efficient federated learning over heterogeneous mobile edge devices. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, pages 1–10, Virtual Conference, May 2021.

[17] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. *arXiv preprint arXiv:2102.02079*, 2021.

[18] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'21)*, pages 10713–10722, Virtual, June 2021. IEEE.

[19] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine learning and systems*, volume 2, pages 429–450, 2020.

[20] Youjie Li, Mingchao Yu, Songze Li, Salman Avestimehr, Nam Sung Kim, and Alexander Schwing. Pipe-sgd: A decentralized pipelined sgd framework for distributed deep net training. In *Advances in Neural Information Processing Systems(NeurIPS'18)*, volume 31, Montréal, Canada, December 2018.

[21] Bing Luo, Xiang Li, Shiqiang Wang, Jianwei Huang, and Leandros Tassiulas. Cost-effective federated learning design. In *IEE Conference on Computer Communications*, pages 1–10. IEEE, June 2021.

[22] 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 (AISTATS'17)*, Ft. Lauderdale, FL, April 2017.

[23] Mahesh Pawar, Anjana Panday, Ratish Agrawal, and Sachin Goyal. Developing a big-data-based model to study and analyze network traffic. In *Big Data and Knowledge Sharing in Virtual Organizations*, pages 198–223. IGI Global, 2019.

[24] Pavana Prakash, Jiahao Ding, Rui Chen, Xiaoqi Qin, Minglei Shu, Qimei Cui, Yuanxiong Guo, and Miao Pan. Iot device friendly and communication-efficient federated learning via joint model pruning and quantization. *IEEE Internet of Things Journal*, 9(15):13638–13650, 2022.

[25] Pavana Prakash, Jiahao Ding, Maoqiang Wu, Minglei Shu, Rong Yu, and Miao Pan. To talk or to work: Delay efficient federated learning over mobile edge devices. In *2021 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2021.

[26] 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*, 3:3, 2018.

[27] Shuheng Shen, Linli Xu, Jingchang Liu, Xianfeng Liang, and Yifei Cheng. Faster distributed deep net training: Computation and communication decoupled stochastic gradient

descent. In *International Joint Conference on Artificial Intelligence (IJCAI'19)*, Macao, China, August 2019.

[28] Dian Shi, Liang Li, Rui Chen, Pavana Prakash, Miao Pan, and Yuguang Fang. Towards energy efficient federated learning over 5G+ mobile devices. *arXiv preprint arXiv:2101.04866*, 2021.

[29] Wenqi Shi, Sheng Zhou, Zhisheng Niu, Miao Jiang, and Lu Geng. Joint device scheduling and resource allocation for latency constrained wireless federated learning. *IEEE Transactions on Wireless Communications*, 2020.

[30] Tung T Vu, Duy T Ngo, Nguyen H Tran, Hien Quoc Ngo, Minh N Dao, and Richard H Middleton. Cell-free massive mimo for wireless federated learning. *IEEE Transactions on Wireless Communications*, 19(10):6377 – 6392, 2020.

[31] Lixu Wang, Shichao Xu, Xiao Wang, and Qi Zhu. Addressing class imbalance in federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10165–10173, 2021.

[32] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. *Advances in neural information processing systems*, 30, 2017.

[33] Carole-Jean Wu, David Brooks, Kevin Chen, Douglas Chen, Sy Choudhury, Marat Dukhan, Kim Hazelwood, Eldad Isaac, Yangqing Jia, Bill Jia, et al. Machine learning at facebook: Understanding inference at the edge. In *2019 IEEE international symposium on high performance computer architecture (HPCA)*, pages 331–344. IEEE, 2019.

[34] Jian Xu, Shao-Lun Huang, Linqi Song, and Tian Lan. Live gradient compensation for evading stragglers in distributed learning. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2021.

[35] Kai Yang, Tao Jiang, Yuanming Shi, and Zhi Ding. Federated learning via over-the-air computation. *IEEE Transactions on Wireless Communications*, 19(3):2022–2035, 2020.

[36] Dongdong Ye, Rong Yu, Miao Pan, and Zhu Han. Federated learning in vehicular edge computing: A selective model aggregation approach. *IEEE Access*, 8:23920–23935, 2020.

[37] Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning. *arXiv preprint arXiv:2110.06848*, 2021.

[38] Ligeng Zhu, Hongzhou Lin, Yao Lu, Yujun Lin, and Song Han. Delayed gradient averaging: Tolerate the communication latency for federated learning. In *Advances in Neural Information Processing Systems (NeurIPS'21)*, volume 34, New Orleans, Louisiana, December 2021.