Client Selection Based on Diversity Scaling for Federated Learning on Non-IID Data

Yuechao Ren(✉), Atul Sajjanhar, Shang Gao, and Seng Loke

Deakin University, 221 Burwood Hwy, Burwood, VIC 3125, Australia
renyue@deakin.edu.au

Abstract. In a wireless Federated Learning (FL) system, clients train their local models over local datasets on IoT devices. The derived local models are uploaded to the FL server which generates a global model, then broadcasts the model back to the clients for further training. Due to the heterogeneous feature of clients, client selection plays an important role in determining the overall training time. Traditionally, maximum number of clients are selected if they can derive and upload their local models before the deadline in each global iteration. However, selecting more clients not only increases the energy consumption of the clients, but also might not be necessary as having fewer clients in early global iterations and more clients in later iterations have been proved better for model accuracy. To address the issue, this paper proposes a client selection scheme which dynamically adjusts and optimizes the trade-off between maximizing the number of selected clients and minimizing the total communication cost between the clients and the server. By comparing the data diversity of clients, this scheme can select the most suitable clients for global convergence. A Diversity Scaling Node Selection framework (FedDS) is implemented to dynamically change the selecting weights of each node based on the degree of non-i.i.d data diversity. Results has shown that the proposed FedDS can speed up the FL convergence rate compared to FedAvg with random node selection.

Keywords: Federated Learning · Diversity Scaling · Convergence · client Selection

1 Introduction

Due to the increasing growth of processing capabilities at mobile edge nodes, networks are undergoing a paradigm shift from traditional cloud computing to a Mobile Edge Computing (MEC) system [1,2]. Federated learning (FL) is an emerging distributed machine learning (DML) technology that enables edge devices to jointly train a shared machine-learning model without the direct transmission of private data [3]. A fundamental challenge for FL is data heterogeneity. In contrast to previous optimisations in DML, where algorithms operate on independent and identically distributed (i.i.d.) data samples partitioned

W. Wang and J. Wu (Eds.): BROADNETS 2023, LNICST 511, pp. 126–137, 2023.
https://doi.org/10.1007/978-3-031-40467-2_8

from a huge dataset [4–6], in FL, models are trained on local clients by using local data, which are usually client-specific and have dissimilarities between each other. Furthermore, data samples on clients may not be independent and identical (non-I.I.D.). Thus, picking clients at random to take part in each training round can't show how the data is distributed globally. For example, under heterogeneous settings, random node selection has been proven to achieve extremely long training latency [4]. Training on clients with non-i.i.d. datasets will result in biased model updates, which will lower the overall convergence rate and accuracy of the model as well as necessitate additional rounds of communication for resource-constrained edge devices [5,6].

Due to the non-i.i.d Date issue, network uncertainty, bandwidth limitation, the straggler effect, etc. [7], participating clients (nodes) will have a substantial impact on the performance of FL. It is necessary to optimize the client selection procedure for performance purposes. In this paper, FedDS is a node selection strategy developed to increase the convergence rate of FL with non-iid nodes. By utilising a selection strategy based on selecting weights, it considers local nodes' diversity scaling to aggregate and determine the suitable subset of nodes for local training and global aggregation. Our main contributions are as follows:

- We propose FedDS, a node selection strategy that allows the server to dynamically adjust the selecting weights of each node in each round based on the diversity of participating nodes. FedDS favourably chooses the nodes that can help improve the model convergence.
- We evaluate how different local updates affect model training and suggest an aggregation algorithm that uses diversity scaling to evaluate how different the local nodes' updates are.
- We evaluate the performance of FedDS by conducting experiments on real datasets under varying heterogeneity settings. Results illustrate the efficacy of FedDS in accelerating the convergence rate of the FL model in comparison to the widely used FedAvg method [3].

2 Related Work

In general synchronous FL, nodes are randomly selected to participate in local training(e.g., FedAvg [3], FedProx [4],CMFL [8]). However, FL may have nodes with discrepancies in data quality and distribution, computational ability (e.g., CPU or GPU, memory size), network connectivity (e.g., traffic transmission speed), and constrained resources (e.g., limited bandwidth and energy budget). A well-designed node selection strategy is essential for FL performance enhancement. There are prior studies on node selection strategy, focusing on system heterogeneity and network connectivity [9,10]. For example, The authors in [9] presented a node selecting scheme by considering the resource conditions of nodes. Amiria et al. [10] introduced an algorithm to schedule nodes by calculating the $\ell 2$-norm of local updates and transmission channel condition. The authors in [11] proposed a different strategy to achieve a faster convergence by choosing node

with higher local loss. However, measuring the local loss of node in run time leads to additional communication and computation cost.

Several other works have focused on probabilistic-based node selection strategy, where every node has a chance of contributing to the global model. [12–16]. Particularly, Chen et al. [12] evaluated the contribution of each node based on the norm of local updates, hence calculating the chosen probability for each node. When bandwidth resources are restricted, nodes having a higher norm of local updates have a greater probability of getting selected, hence boosting the convergence rate. The authors in [13] utilised Artificial Neural Networks (ANNs) to estimate the model updates of nodes that are not given the bandwidth for transmission. After analysing the successfully transmitted updates, the additional received updates can further speed up the model convergence.

Ren et al. [14] presented a probabilistic selection strategy that takes into account the importance of local update and transmission latency. Chen et al. [15] proposed a sampling strategy which aims at selecting more important nodes. It used the local update norm to determine the chance of node being selected in the node sampling process, which in turn optimises the variance of local gradients for aggregation. Similarly, the authors in [16] utilised importance sampling method to choose nodes on the server side. Similar to [15], the node selecting strategy is to minimise the local gradient variance bound.

In contrast to previous studies, this paper analyses the effects of data heterogeneity, and proposes a scheme based on node selecting weights for participating node selection. It examines the diversity of local updates and their relationship with the global update before adjusting the selection weights of each node based on diversity scaling. Hence, when training on heterogeneous data samples across nodes, the diversity of local updates directly contributes to the upper bound of the global model change and, consequently, convergence rate, as demonstrated theoretically in Sect. 3.1.

The remaining sections are organised as follows. Section 3 provides the preliminary including the concept of Diversity Scaling and the challenge of non-i.i.d. data. In Sect. 4, the proposed aggregation algorithms and node selection scheme are discussed. Evaluation and results are discussed in Sect. 5, followed by the conclusion in Sect. 6.

3 Preliminaries

In this section, we first introduce the key concept (Diversity Scaling) used in our work, then discuss the challenge to FL on heterogeneous data.

3.1 Diversity Scaling

A general FL training round contains the following steps:

– Global model initialisation: The central server initialises a global model and broadcasts it to each node.

- Local node update: Each selected node performs local training using local data samples by mainly calculating its local updates based on the global model (e.g., stochastic gradient descent (SGD)).
- Global Model Aggregation: After local node finishes their local training, all participated nodes send their own updates to the central server for further aggregation. The central server aggregates these updates based on the predefined aggregation algorithm, then updates the former global model. In FedAvg [3], updates from participated nodes are averaged and added to the global model as defined in Eq. (1). The global model is updated at the end of round.

$$\begin{aligned} \Delta_{avg}^{t} &= \frac{1}{K}(\Delta_1^t + \Delta_2^t + ... + \Delta_K^t) \\ w^{t+1} &= w^t + \Delta_{avg}^t \end{aligned} \tag{1}$$

- The above steps form one round of FL training. These steps are usually running several rounds until convergence is reached and/or a satisfactory global model accuracy is obtained.

After R rounds of training, the magnitude of weights w changing on the global model is shown in Eq. (2), t denote the t^{th} training round.

$$\begin{aligned} \|w^R - w^0\| &= \left\| \sum_{t=0}^{R-1} (w^{t+1} - w^t) \right\| \\ &\leq \sum_{t=0}^{R-1} \|w^{t+1} - w^t\| = \sum_{t=0}^{R-1} \left\| \Delta_{avg}^t \right\| \end{aligned} \tag{2}$$

From the above Equation, it can be seen that the upper bound weight changing on the global model is directly determined by the value of $\left\|\Delta_{avg}^t\right\|$, so as the convergence rate. Considering one extreme example, if all the participated nodes have the exact same updates, then it can be argued that there is no diversity in this given round of training. Therefore, the multitude of nodes can be replaced by a single node. Another extreme example is that if the updates of participated nodes can perfectly cancel each other after averaging, then $\left\|\Delta_{avg}^t\right\|$ will be 0. By analyzing these two extreme examples, it's more likely that nodes' updates won't be either perfectly coherent or decoherent. Instead, they'll be somewhere in between.

Diversity coefficient can be used to quantify the diversity in node updates and measures how different the updates to each node are, as shown in Eq. (3).

$$\gamma^t = \frac{\frac{1}{K}\sum_k \|\Delta_k^t\|}{\left\|\Delta_{avg}^t\right\|} \tag{3}$$

A larger value of γ means higher degree of dissimilarity between node updates. This is very likely when training on heterogeneous data samples. To tackle the data heterogeneity issue (non-i.i.d.) in FL, we propose FedDS, which is an modification to FedAvg. Our FedDS uses diversity scaling to account for the effect of node update diversity, as demonstrated in Section IV-A Aggregation Algorithm via Diversity Scaling.

3.2 Challenges to Non-i.i.d. Data Distribution

FedAvg can achieve an acceptable convergence rate by using random node selection strategy and a simple weight averaging design; however, when it comes to partial node selection and non-i.i.d. training data, the convergence rate is relatively slow [17], as also demonstrated in [5,7,11], and [18]. It has been proven that increasing the amount of local computing (i.e., more local updates) can reduce the number of communication rounds (a major bottleneck in FL [3,7]) required for convergence. However, even with improved local computing [17,19], unsatisfactory model performance on non-iid datasets still exist. This is due to the strong relationship between the data distribution and the local objective loss function.

Simple selection strategies such as the random selection result in data samples being distributed differently on the chosen nodes. Due to training instability, a FL model has trouble converging when dealing with a non-i.i.d. context because the model is more optimal to the local objective rather than the global objective.

From a data heterogeneity point of view, it is important to understand and analyse the non-trivial node selection strategies, finding and choosing the nodes that contribute more to model convergence. By quantifying the diversity in node updates and measuring how different the updates to each node are, nodes with updates that are adversely affect the global update could be identified. Further, by dynamically changing the selecting weights for those potential adverse local nodes, it can be expected that the nodes helping reduce global loss the most will have a better chance of being chosen in the next training round.

4 The Proposed Algorithms

To improve the convergence rate of federated learning, we design a node selection scheme by incorporating two algorithms. An aggregation algorithm (FedDS) takes node update dissimilarity into account via diversity scaling. It can identify those nodes with higher degree of dissimilarity to further reduce the expected decrement of global loss in each round. Then, based on the result of aggregation algorithm, a node selection algorithm is applied to dynamically adjust the selecting weights for each node to be selected for the next round. Consequently, the server can preferentially select nodes that propel a faster model convergence.

4.1 Aggregation Algorithm via Diversity Scaling

The proposed aggregation algorithm (FedDS) is an extension method to FedAvg. It takes γ into account via diversity scaling, as shown in Algorithm 1. A similar notion called gradient diversity can be found in [20]. It has been demonstrated that high similarity between concurrent gradient updates degrades the performance of mini-batch stochastic gradient descent algorithms. However, update diversity in this paper is distinct from the gradient diversity, and we employ it to speed up the federated training.

Note that when training for the first round, we still need to randomly select a subset of nodes S^t from all the nodes, denoted as K, since all the nodes by default have the same selecting weights. After each node performs local update, each node's update dissimilarity Δ_k^t is sent to the server for averaging and diversity coefficient γ calculation. Larger γ value indicates higher degree of dissimilarity between node updates in current round. For the purpose of speeding up the convergence rate, it can be observed that a lower diversity coefficient γ value (i.e., lower degree of dissimilarity between node updates) could lead to a faster model convergence. Hence, we choose the minimum value of γ. γ_{max} is adopted to keep the change of diversity coefficient in a certain range. In practise, setting γ_{max} to $\sqrt{K}$ works well. Since γ can be calculated directly from trainer updates after one round. Due to its specialty, γ can be then utilised in Algorithm 2 as a coefficient for updating the selecting weights of nodes.

In the following training rounds, the server needs to select a subset of nodes S^t based on the labeled selecting weights of each node, calculated by Algorithm 2 and denoted as P. Node with a higher P value, a node will have a higher chance of being selected for the next round.

4.2 Node Selection

To enhance the convergence rate of a global model, one can aim to preferentially select the nodes with a higher contribution when training the global model on a variety of nodes (e.g., nodes with i.i.d. dataset, as observed in [3,18]). As a result, we propose a node selection strategy that can dynamically modify the selecting weights for each node in each communication round. The node selecting weights can be reduced if their local updates share significant dissimilarity and slow model convergence. It is done based on their update diversity and data distribution heterogeneity, which can be distinguished by the output value γ of the proposed aggregation algorithm (FedDS). Specifically, according to Eq. (4), the selecting weights for nodes labelled by Algorithm 2 are decreased, while the selecting weights for all other nodes are increased.

$$\Delta p_i^t = p_i^t \times min[\beta^\gamma, 1], i \in S^t \tag{4}$$

where p_i^t denotes the selecting weights for node i in the t-th round, Δp_i^t denotes the selecting weights changing in the next round. min function returns the minimum value among two arguments. β value is adopted to keep the rate of selecting weights change in a stable range. Based on the experiment results, setting β to 0.7 is a good choice of balancing the tradeoff.

After getting the selecting weights change for the participating nodes in the current round and previous round, the selecting weights for the rest nodes also need be changed accordingly, as shown in Eq. (5)

$$p_i^{t+1} = \begin{cases} p_i^t - \Delta p_i^t & i \in S^t \\ p_i^t + \dfrac{\sum_{S^t} \Delta p_i^t}{|K - S^t|} & i \in K - S^t \end{cases} \tag{5}$$

Algorithm 1. Aggregation via Diversity Scaling

Input: P, S, K

Initialisation: global model w^0, accelerated global model $w^0_{acc} = w^0$.
for round $t \leftarrow 0, 1, 2, ...$ **do**
  Server
  if $t == 0$ **then**
    Select S^t of K nodes based on P^0
    Send w^0 to nodes in S^t
  else
    Select a subset S^t with highest P of K nodes
    Send w^{t-1} to nodes in S^t
  end if
  for each node k *(in parallel)* in S^t **do**
    $w^t_k \leftarrow w^t_{acc}$
    $w^t_k \leftarrow w^t_k - \eta * \nabla f_k(w)$
    Send $\Delta^t_k = w^t_k - w^t_{acc}$ to server
  end foreach
  Server
  Update global model
    $w^{t+1} = w^t_{acc} + \Delta^t_{avg}$
  Update accelerated global mode
    $w^{t+1}_{acc} = w^t_{acc} + min(\gamma^t, \gamma_{max}) * \Delta^t_{avg}$
  Where
    $\Delta^t_{avg} = \frac{1}{K} \sum_k \Delta^t_k$ and $\gamma^t = \frac{\frac{1}{K} \sum_k \|\Delta^t_k\|}{\|\Delta^t_{avg}\|}$
end for
return $min(\gamma^t, \gamma_{max})$ for Algorithm 2

Algorithm 2. Node Selecting Weights

Input: $S, K, \gamma, \beta, p^t_i, i \leftarrow 1, 2, ..., K$

Initialisation: $p^0_i = \frac{1}{K}$, w^0, $w^0_{acc} = w^0$
for round $t \leftarrow 1, 2, ...$ **do**
  Server
    Sampling S^t nodes according to P^{t-1}_i
    $w^{t+1}, w^{t+1}_{acc}, \gamma \leftarrow$ ALGORITHM 1
    Updating the selecting weights $p^t_i, i \leftarrow 1, 2, ..., K$, by (4), (5) for use in the next round
end for
return p^t_i for Algorithm 1

5 Evaluation and Analysis

We implement FedDS on different tasks with different hyper parameters and compare the results with FedAvg. Firstly, we demonstrated FedDS on speeding up the convergence rate of global model. Then, we tested FedDS under different level of data heterogeneity. We analyze the communication cost and complexity

compared with FedAvg, as shown in Sect. 5.3. The adopted dataset, learning model and experiment settings are listed as follows:

In the experiments, we consider non-convex classification problem with MNIST [21] using a convolutional neural network (CNN) model. Specifically, this CNN model has 7 layers with two Convolutional layers ($5\times5\times10, 5\times5\times20$), each of which is followed by 2×2 Max pooling. The second convolutional layer has 50% dropout, and two Fully Connected layers ($320 \times 50, 50 \times 10$), followed by Softmax at the end. ReLu activation maps all of the Convolutional and Fully connected layers.

The initial learning rate, learning-rate decay, local batch size, and local epoch number are set to be 0.01, 0.995, 20 and 1, respectively. Although adjusting these parameters may improve the performance, it is out of scope of this paper, as we mainly focus on how to improve the training accuracy under non-i.i.d. dataset at client sites.

To implement different levels of data heterogeneity, we introduce two parameters μ and ν. Specifically, μ denotes the ratio of nodes which are equipped with i.i.d. dataset. ν denotes how many labels the nodes have. For example, if the number of total nodes $K = 50$, $\mu = 0.2$ means that $\mu K = 10$ nodes are i.i.d., $(1\text{-}\mu)K = 40$ nodes are equipped with non-i.i.d. dataset. $\nu = 3$ means that the data samples on these nodes evenly belong to 3 labels. As such, smaller μ and ν indicate a higher data heterogeneity.

5.1 Performance

We evaluate the test accuracy performance of the proposed FedDS by comparing with the baseline algorithm FedAvg. The level of data heterogeneity is $\mu = 0.5$, $\nu = 2$. In each training round, the server selects $S^t = 10$ nodes out of total 50 nodes randomly (selecting fraction $c = 0.2$) to join the training. For fair comparison, the selected nodes number in each round are the same for both FedDS and FedAvg.

As shown in Fig. 1, FedDS can achieve higher accuracy and lower training loss comparing to FedAvg. FedDS has faster convergence speed in several initial rounds due to the high diversity of local updates that caused by data heterogeneity. Thus for the next round training, reducing the selecting weights of those nodes whose local updates share significant dissimilarity is effective to speed up the model convergence.

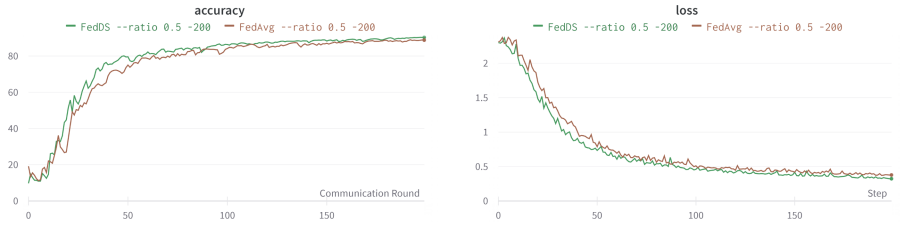


Fig. 1. Performance of the proposed FedDS vs FedAvg with –ratio(μ) = 0.5 - round ($T = 200$).

5.2 Data Heterogeneity

We evaluate the performance of the proposed algorithm in different data heterogeneity scenarios by using different fraction of non-i.i.d. nodes and labels for the participating nodes (i.e., $\mu = 0.3, 0.5, 0.7$ and $\nu = 1$).

As can be seen from Fig. 2, FedDS converges faster and achieves a higher test accuracy, compared with FedAvg regardless of different levels of data heterogeneity. Furthermore, as shown in Table 1, as the data becomes more heterogeneous, the performance improvement is greater (i.e., mu changes from 0.7 to 0.3).

FedDS performs better when the number of i.i.d. nodes is limited and the non-i.i.d. nodes have highly skewed data samples (e.g., $mu = 0.3, 0.5$), demonstrating FedDS's effectiveness in identifying nodes that have more contributions to the global model.

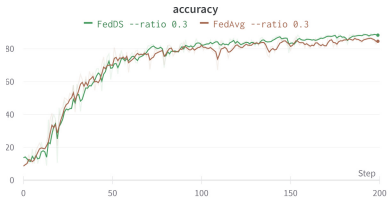
FedDS and FedAvg perform similarly in scenarios with the least amount of data heterogeneity (i.e., $mu = 0.7$). The reason is that non-i.i.d. nodes may not be selected to join the training when a large number of i.i.d. nodes are available.

5.3 Communication Cost

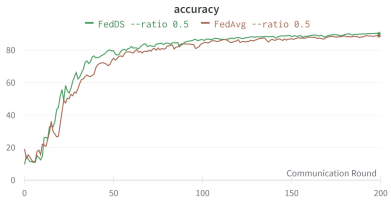
We evaluate the communication cost of the proposed algorithm and compare it with FedAvg. During the training process, a fraction of n nodes are selected to train a model whose size is N with T iterations to achieve a desired model. In each round, every node sends the local model with size of N to the server and receives the global model with size of N from the server, so the communication cost of each node is $2N \times T$. In general, training on the non-i.i.d data requires comparatively more training rounds to reach the convergence than the i.i.d data. Thus, a potential way to make comparison and to prove that the non-i.i.d data heterogeneity issue has been mitigated is to speed up the training convergence by reducing the communication round when setting a target accuracy. The comparison with FedAvg is shown in Table 2. It can be seen that our algorithm supports more efficient communications. The communication cost are reduced for all of the data heterogeneity settings to a target test accuracy (80%).

Table 1. Test accuracy comparison of FedDS and FedAvg with different settings of data heterogeneity.

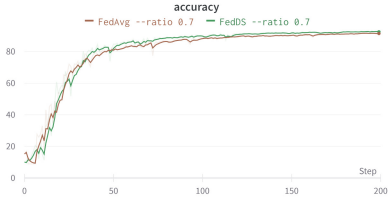
Algorithm	$\mu = 0.3$	$\mu = 0.5$	$\mu = 0.7$
FedAvg	83.62%	88.52%	90.56%
FedDS	88.44%	90.79%	92.52%
Difference	4.82%	2.27%	1.96%



(a) $\mu = 0.3$ and $\nu = 1$



(b) $\mu = 0.5$ and $\nu = 1$



(c) $\mu = 0.7$ and $\nu = 1$

Fig. 2. Test accuracy over communication rounds of FedDS and FedAvg with different data heterogeneity settings.

Table 2. Communication round to achieve target test accuracy, comparison between FedDS and FedAvg with different settings of data heterogeneity.

Algorithm	$\mu = 0.3$, acc $= 80\%$	$\mu = 0.5$, acc $= 80\%$	$\mu = 0.7$, acc $= 80\%$
FedAvg	$T = 91$	$T = 82$	$T = 48$
FedDS	$T = 70$	$T = 48$	$T = 40$

6 Conclusion

In this paper, we have presented a node selection strategy, FedDS. It can select suitable nodes and speed up the convergence rate of model with non-i.i.d. datasets. FedDS can change the selecting weights of node in each training round dynamically based on the diversity of their data. It can sort out the suitable local updates from nodes by measuring the dissimilarity between node updates in the current round. The experimental results under different data heterogeneity settings have shown that FL training with the proposed FedDS can speed up the model convergence and gain higher test accuracy.

References

1. Chiang, M., Zhang, T.: Fog and IoT: an overview of research opportunities. IEEE Internet Things J. **3**, 854–864 (2016)
2. Xiong, Z., Zhang, Y., Niyato, D., Wang, P., Han, Z.: When mobile blockchain meets edge computing. IEEE Commun. Mag. **56**, 33–39 (2018)
3. McMahan, H.B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of 20th International Conference on Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)
4. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. arXiv:1812.06127 [cs, stat] (2020)
5. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-IID data. arXiv:1806.00582 (2018)
6. Wang, H., Kaplan, Z., Niu, D., Li, B.: Optimizing federated learning on Non-IID data with reinforcement learning. In: IEEE INFOCOM 2020 - IEEE Conference on Computer Communications, pp. 1698–1707 (2020)
7. Wang, S., et al.: Adaptive federated learning in resource constrained edge computing systems. arXiv:1804.05271 [cs, math, stat] (2019)
8. Wang, L., Wang, W., Li, B.: CMFL: mitigating communication overhead for federated learning. In: 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), pp. 954–964 (2019)
9. Nishio, T., Yonetani, R.: Client selection for federated learning with heterogeneous resources in mobile edge. In: Proceedings of the IEEE International Conference on Communications (ICC) (2019)
10. Amiri, M.M., Gündüz, D., Kulkarni, S.R., Poor, H.V.: Convergence of update aware device scheduling for federated learning at the wireless edge. IEEE Trans. Wireless Commun. **20**, 3643–3658 (2021)
11. Cho, Y.J., Wang, J., Joshi, G.: Client selection in federated learning: convergence analysis and power-of-choice selection strategies. arXiv:2010.01243 (2020)
12. Chen, M., Shlezinger, N., Poor, H.V., Eldar, Y.C., Cui, S.: Communication-efficient federated learning. Proc. Natl. Acad. Sci. U. S. A. **118** (2021). https://doi.org/10.1073/pnas.2024789118
13. Chen, M., Poor, H.V., Saad, W., Cui, S.: Convergence time optimization for federated learning over wireless networks. IEEE Trans. Wireless Commun. **20**, 2457–2471 (2021)
14. Ren, J., He, Y., Wen, D., Yu, G., Huang, K., Guo, D.: Scheduling for cellular federated edge learning with importance and channel awareness. arXiv:2004.00490 (2020)
15. Chen, W., Horvath, S., Richtarik, P.: Optimal client sampling for federated learning. arXiv:2010.13723 (2020)
16. Rizk, E., Vlaski, S., Sayed, A.H.: Optimal importance sampling for federated learning. In: ICASSP 2021–2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3095–3099 (2021)
17. Li, X., Huang, K., Yang, W., Wang, S., Zhang, Z.: On the convergence of FedAvg on non-IID data. In: Proceedings of the International Conference on Learning Representations (ICLR) (2020)
18. Wu, H., Wang, P.: Fast-convergent federated learning with adaptive weighting. IEEE Trans. Cogn. Commun. Netw. **7**, 1078–1088 (2021)

19. Stich, S.U.: Local SGD converges fast and communicates little. In: Proceedings of the International Conference on Learning Representations (ICLR) (2019)
20. Yin, D., Pananjady, A., Lam, M., Papailiopoulos, D., Ramchandran, K., Bartlett, P.: Gradient diversity: a key ingredient for scalable distributed learning. In: Storkey, A. and Perez-Cruz, F. (eds.) Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics, pp. 1998–2007. PMLR (2018)
21. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**, 2278–2324 (1998)