ADAPTIVE NODE PARTICIPATION FOR STRAGGLER-RESILIENT FEDERATED LEARNING

Amirhossein Reisizadeh[1]*, Isidoros Tziotis*[2]*, Hamed Hassani*[3]*, Aryan Mokhtari*[2]*, Ramtin Pedarsani*[4]

[1]MIT, [2]UT Austin, [3]UPenn, [4]UC Santa Barbara

ABSTRACT

Federated learning is prone to multiple system challenges including system heterogeneity where clients have different computation and communication capabilities. Such heterogeneity in clients' computation speeds has a negative effect on the scalability of federated learning algorithms and causes significant slow-down in their runtime due to the existence of stragglers. In this chapter, we propose a novel straggler-resilient federated learning method that incorporates statistical characteristics of the clients' data to adaptively select the clients in order to speed up the learning procedure. The key idea of our algorithm is to start the training procedure with faster nodes and gradually involve the slower nodes in the model training once the statistical accuracy of the data corresponding to the current participating nodes is reached. The proposed approach reduces the overall runtime required to achieve the statistical accuracy of data of all nodes, as the solution for each stage is close to the solution of the subsequent stage with more samples and can be used as a warm-start. Our numerical experiments demonstrate significant speedups in wall-clock time of our straggler-resilient method compared to other federated learning benchmarks.

1. INTRODUCTION

Federated learning is a distributed framework whose objective is to train a model using the data of many clients (nodes), while keeping each node's data local. In contrast with centralized learning, the federated learning architecture allows for preserving the clients' privacy as well as reducing the communication burden caused by transmitting data to a cloud. Nevertheless, as we move towards deploying federated learning in practice, it is becoming apparent that several major challenges still remain and the existing frameworks need to be rethought to address them. Important among these challenges is system (device) heterogeneity due to existence of *straggling nodes* – slow nodes with low computational capability – that significantly slow down the model training [1, 2].

In this paper, we focus on system heterogeneity in federated learning and we leverage the interplay between statistical accuracy and system heterogeneity to design a straggler-resilient federated learning method that carefully and adaptively selects a subset of available nodes in each round of training. Federated networks consist of thousands of devices with a wide range of computational, communication, battery power, and storage characteristics. Hence, deploying traditional federated learning algorithms such as `FedAvg` [3] on such a highly heterogeneous cluster of devices results in significant and unexpected delays due to existence of slow clients or stragglers. In most of such algorithms, *all* the *available* clients participate in the model training –regardless of their computational capabilities. Consequently, in each communication round of such methods, the server has to wait for the slowest node to complete its local updates and upload its local model which significantly slows down the training process.

In this paper, we aim to mitigate the effect of stragglers in federated learning based on an adaptive node participation approach in which clients are selected to participate in different stages of training according to their computation speed. We call our straggler-resilient scheme a **F**ederated **L**earning method with **A**daptive **N**ode **P**articipation or `FLANP`. The key idea of this scheme is to start the model training procedure with only a few clients which are the fastest among all the nodes. These participating clients continue to train their shared models while interacting with the parameter server. Note that since the server waits only for the participating nodes, it takes a short time for the participating (and fast) clients to promptly train a shared model. This model is, however, not accurate as it is trained over only a fraction of samples. We next increase the number of participating clients and include the next fastest subset of nonparticipating nodes in the training. Note that the model trained from the previous stage can be a warm-start initialization for the current stage.

To discuss our main idea more precisely, consider a federated network of N available nodes each storing s data samples and suppose that we start the learning procedure with only m clients. Once we solve the empirical risk minimization (ERM) problem corresponding to $m \times s$ samples of these nodes up to its statistical accuracy, we geometrically increase the number of participating nodes to $n = \alpha m$ where $\alpha > 1$, by adding the next $n - m$ fastest clients in the network. By doing so, the new ERM problem that we aim to solve contains the samples from the previous stage as well as the samples of the newly participating nodes. Moreover, the solution for the ERM problem at the previous stage (with m clients) could be used as a warm-start for the ERM problem at the current stage with

978-1-6654-0540-9/22/$31.00 ©2022 IEEE

$n = \alpha m$ nodes. This is due to the fact that all samples are drawn from a common distribution, and as a result, the optimal solution of the ERM problem with fewer samples is not far from the optimal solution of the ERM problems with more samples, as long as the larger set contains the smaller set.

In the proposed FLANP algorithm, as time progresses, we gradually increase the number of participating clients until we reach the full training set and all clients are involved. Note that in this procedure, the slower clients are only used towards the end of the learning process, where the model is already close to the optimal model of the aggregate loss. Another essential observation is that since the model trained in previous rounds already has a reasonable statistical accuracy and this model serves as the initial point of the next round of the iterative algorithm, the slower nodes are only needed to contribute in the final rounds of training, leading to a smaller wall-clock time. This is in contrast with having all nodes participate in training from the beginning, which leads to computation time of each round being determined by the slowest node.

Related Work. System (device) heterogeneity challenge, which refers to the case that clients have different computational, communication and storage characteristics, has been studied in the literature. Asynchronous methods have demonstrated improvements in distributed data centers. However, such methods are less desirable in federated settings as they rely on bounded staleness of slow clients [4, 5]. The active sampling approach is another direction in which the server aims for aggregating as many local updates as possible within a predefined time span [6]. More recently, [7] proposed a normalized averaging method to mitigate stragglers in federated systems and the objective inconsistency due to mismatch in clients' local updates. Deadline-based computation has also been studied to mitigate stragglers in decentralized settings [8]. The idea of adaptive sample size training in which we solve a sequence of geometrically increasing ERM problems has been used previously for solving large-scale ERM problems. In particular, it has been shown that this scheme improves the overall computational cost of both first-order [9, 10] and second-order [11, 12, 13] methods for achieving the statistical accuracy of the full training set. In this paper, we exploit this idea to develop FLANP for a different setting to address the issue of device heterogeneity in federated learning. As mentioned, FLANP is a general meta-algorithm that can be employed with any federated learning subroutine studied in the literature [3, 14, 15, 16, 17, 18].

2. FEDERATED LEARNING SETUP

In this section, we state our setup. Consider a federated architecture where N nodes interact with a central server, and each node $i \in [N] = \{1, \cdots, N\}$ has access to s data samples denoted by $\{z_1^i, \cdots, z_s^i\}$. These samples are drawn at the beginning of the training process, and nodes cannot draw new samples during training. Further, define $\ell(\cdot, \cdot) : \mathbb{R}^d \times \mathcal{Z} \to \mathbb{R}$ as a loss function where $\ell(\mathbf{w}, z_j^i)$ indicates how well the model $\mathbf{w}$ performs with respect to the sample z_j^i. Also, define the empirical loss of node i as $L^i(\mathbf{w}) := \frac{1}{s}\sum_{j=1}^{s} \ell(\mathbf{w}, z_j^i)$. For any $1 \leq n \leq N$, we denote by $L_n(\mathbf{w})$ the collective empirical risk corresponding to samples of all nodes $\{1, \cdots, n\}$, which is defined as

$$L_n(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^{n} L^i(\mathbf{w}). \tag{1}$$

$L_n(\mathbf{w})$ represents the average loss over the $n \times s$ samples stored at nodes $\{1, \cdots, n\}$. We let $\mathbf{w}_n^*$ denote the optimal minimizer of the loss $L_n(\mathbf{w})$, i.e., $\mathbf{w}_n^* = \arg\min_{\mathbf{w}} L_n(\mathbf{w})$. We assume that the samples z_j^i are i.i.d. realizations of a random variable Z with probability distribution $\mathcal{P}$. The problem of finding a global model for the aggregate loss of all available N nodes, which can be considered as the empirical risk minimization (ERM) in (1) for $n = N$, i.e.,

$$\min_{\mathbf{w}} L_N(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} L^i(\mathbf{w}) = \frac{1}{Ns}\sum_{i=1}^{N}\sum_{j=1}^{s} \ell(\mathbf{w}, z_j^i) \tag{2}$$

is a surrogate for the expected risk minimization problem $\min_{\mathbf{w}} L(\mathbf{w}) := \mathbb{E}_{Z\sim\mathcal{P}}[\ell(\mathbf{w}, Z)]$. Our ultimate goal is to find the optimal solution of the expected risk $\mathbf{w}^* = \arg\min_{\mathbf{w}} L(\mathbf{w})$; however, since distribution $\mathcal{P}$ is unknown and we only have access to a finite number of realizations of the random variable Z, i.e., $\{z_1^i, \cdots, z_s^i\}_{i=1}^N$, we settle for solving problem (2).

Let us further clarify the data heterogeneity model used in our setting. Data samples $\{z_1^i, \cdots, z_s^i\}_{i=1}^N$ are i.i.d. (data homogeneous); however, the realized samples are fixed through the learning process. More precisely, empirical risk functions L^i are realized and fixed which yields that the local gradient directions ∇L^i are *not* necessarily an unbiased estimator for the aggregate loss gradient ∇L_N. In other words, nodes do not have access to unbiased estimators of the expected loss and therefore federated learning methods designed merely for i.i.d. data settings would not be useful here. This is indeed a significant challenge in designing fast convergent optimization methods in such settings which we will highlight in Section 3.

Statistical Accuracy. The difference of expected and empirical risks $L_n(\mathbf{w}) - L(\mathbf{w})$ is referred to as the estimation error and can be bounded by a function of the sample size. In particular, since $L_n(\mathbf{w})$ captures ns samples, we assume that there exists a constant V_{ns} bounding the estimation error with high probability, $\sup_{\mathbf{w}} |L_n(\mathbf{w}) - L(\mathbf{w})| \leq V_{ns}$.

The estimation error V_{ns} has been deeply studied in the statistical learning literature [19, 20]. In particular, it has been shown that for strongly convex functions the estimation error is proportional to the inverse of sample size [21, 22]. In this work, we also assume that $V_{ns} = \frac{c}{ns}$ for a constant c. Note that for the loss function L_n once we find a point $\tilde{\mathbf{w}}$ that has an optimization error of V_{ns}, i.e., $L_n(\tilde{\mathbf{w}}) - L_n(\mathbf{w}_n^*) \leq V_{ns}$, there

is no gain in improving the optimization error as the overall error with respect to the expected risk L would not improve. Hence, when we find a point $\tilde{\mathbf{w}}$ such that $L_n(\tilde{\mathbf{w}}) - L_n(\mathbf{w}_n^*) \leq V_{ns}$, we state that it has reached the statistical accuracy of L_n. Our goal is to find a solution $\mathbf{w}_N$ that is within the statistical accuracy of the ERM problem of the full training set in (2).

System Heterogeneity Model. As mentioned earlier, federated clients attribute a wide range of computational powers leading to significantly different processing times for a fixed computing task such as gradient computation and local model update. To be more specific, for each node $i \in [N]$, we let T_i denote the (expected) time to compute one local model update. The time for such update is mostly determined by the computation time of a fixed batch-size stochastic gradient of the local empirical risk $L_i(\mathbf{w})$. Clearly, larger T_i corresponds to slower clients or stragglers. Without loss of generality, we assume that the nodes are sorted from faster to slower, that is, $T_1 \leq \cdots \leq T_N$ with node 1 and N respectively identifying the fastest and slowest nodes in the network.

3. ADAPTIVE NODE PARTICIPATION APPROACH

Several federated learning algorithms have been proposed to solve the ERM problem in (2) such as `FedAvg` [3], `FedProx` [23], `SCAFFOLD` [24], `DIANA` [25], and `FedGATE` [26]. These methods consist of several rounds of local computations by the clients and communication with the server. As explained earlier, federated clients operate in a wide range of computational characteristics, and therefore, the server has to wait for the slowest node in each communication round to complete its local computation task. All in all, the slowest nodes determine the overall runtime of such federated algorithms which causes significant slow-down.

`FLANP`: A Straggler-Resilient FL Meta-Algorithm. Our proposal to address the device heterogeneity and mitigate the stragglers is as follows.The server first solves the ERM problem corresponding to n_0 fastest nodes, where n_0 is much smaller than the total number of available nodes N. To identify the n_0 fastest nodes, the server first broadcasts a short handshake message to all nodes and waits for the first n_0 nodes that respond. These n_0 nodes will participate in the training process in the first stage. Using `Federated_Solver` which is a federated learning subroutine of choice, e.g., `FedAvg` or `FedGATE`, the n_0 participating nodes proceed to minimize the empirical risk corresponding to their data points, which we denote by $L_{n_0}(\mathbf{w})$ as defined in (1). This continues until the n_0 nodes reach their corresponding statistical accuracy, that is, they reach a global model $\mathbf{w}_{n_0}$ such that $L_{n_0}(\mathbf{w}_{n_0}) - L_{n_0}(\mathbf{w}_{n_0}^*) \leq V_{n_0 s}$. Note that at this stage the server has to wait only for the slowest client among the participating ones, i.e., node n_0, which is potentially much faster than the network's slowest node N.

Per our discussion in Section 2, a more accurate solution than $\mathbf{w}_{n_0}$ would not help improving the optimality gap. Therefore, once statistical accuracy is achieved, the procedure is terminated and we increase the number of participating nodes from n_0 to $2n_0$. To select the $2n_0$ fastest nodes, we repeat the hand-shaking communication protocol that we discussed. Then, the selected nodes use `Federated_Solver` to find the minimizer of the loss corresponding to $2n_0$ participating nodes, while using the solution of the previous stage $\mathbf{w}_{n_0}$ as their starting point.

Again, in this stage, the training process terminates when we find a point $\mathbf{w}_{2n_0}$ within the statistical accuracy of the loss corresponding to $2n_0$ participating nodes, i.e., $L_{2n_0}(\mathbf{w}_{2n_0}) - L_{2n_0}(\mathbf{w}_{2n_0}^*) \leq V_{2n_0 s}$. In the stage with $2n_0$ participating nodes, the computation delay is determined by the slowest participating node among $2n_0$ nodes, which is slower than the previous stage with n_0 participating nodes, but still faster than the slowest node of the network. The procedure of geometrically increasing the number of participating nodes continues till the set of participating nodes contains all the available N nodes, and these nodes find the final global model $\mathbf{w}_N$ within the statistical accuracy of the global loss function $L_N(\mathbf{w})$. Algorithm 1 summarizes the straggler-resilient meta-algorithm.

Algorithm 1: `FLANP`

Initialize fast-to-slow nodes $\{1, \cdots, N\}$, $n = n_0$
participating nodes with initial global model $\mathbf{w}_{n_0}$
while $n \leq N$ **do**
  while $L_n(\mathbf{w}_n) - L_n(\mathbf{w}_n^*) > V_{ns}$ **do**
    nodes $\{1, \cdots, n\}$ are participating and update local models via `Federated_Solver`
    server aggregates local models from nodes $\{1, \cdots, n\}$ and updates global model $\mathbf{w}_n$
  end
  $n \leftarrow \min\{2n, N\}$ `% doubling participants`
end

Remark 1 *In our scheme, clients' computation speeds are not needed, and the parameter server figures out the fastest n nodes only by following the handshaking protocol.*

From a high-level perspective, Algorithm 1 exploits faster nodes in the beginning of the learning procedure to promptly reach a global model withing their statistical accuracy. By doing so, the server avoids waiting for slower nodes to complete their local updates; however, the optimality gap of such models are relatively large since only a fraction of data samples have contributed in the global model. By gradually increasing the number of participating nodes and activating slower nodes, the quality of the global model improves while the synchronous computation slows down due to slower nodes. The key point is that slower nodes join the learning process towards the end.

The criterion in Algorithm 1, that is $L_n(\mathbf{w}_n) - L_n(\mathbf{w}_n^*) > V_{ns}$, verifies that the current global model satisfies the statistical accuracy corresponding to n participating nodes

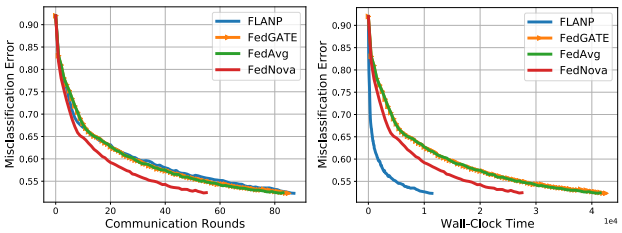


Fig. 1: NN on CIFAR10

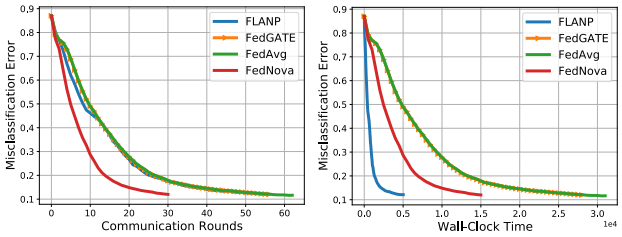


Fig. 2: NN on MNIST

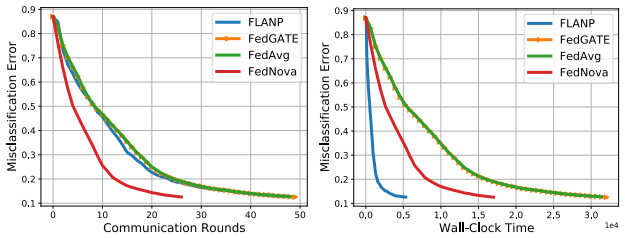
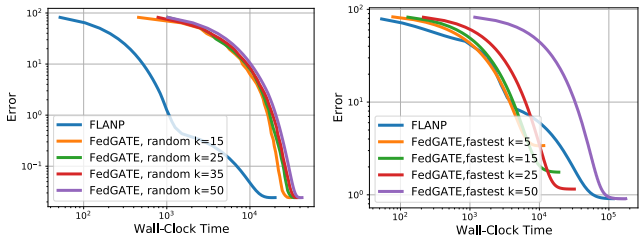


Fig. 3: NN on MNIST



(a) k randomly picked. (b) k fastest picked.

Fig. 4: Partial node participation

$\{1, \cdots, n\}$. This condition, however, is not easy to check since the optimal solution $\mathbf{w}_n^*$ is unknown. A sufficient and computationally feasible criterion is to check if $\|\nabla L_n(\mathbf{w}_n)\|^2 \leq 2\mu V_{ns}$, when ℓ is μ-strongly convex.

4. NUMERICAL EXPERIMENTS

We conduct various numerical experiments for convex and nonconvex risks and evaluate the performance of the proposed method versus other benchmarks. We focus on a particular case of `FLANP` where the federated learning subroutine is picked to be `FedGATE` proposed by [26].

Benchmarks. Bellow is a brief description for multiple federated learning benchmarks that we use to compare with the proposed `FLANP` with `FedGATE` as its inner sub-routine. Note that in all these benchmarks all the available N nodes participate in the training process.

• `FedAvg` [3]. Nodes update their local model using a SGD rule for τ local iterations before uploading to the server.

• `FedGATE` [26]. Here we consider it as a benchmark running with all the available N nodes.

• `FedNova` [7]. In each round, each node i updates its local model for τ_i iterations where τ_is vary across the nodes. To mitigate the heterogeneity in τ_is, the server aggregates normalized updates (w.r.t. τ_i) and updates the global model.

We compare the performance of `FLANP` (using `FedGATE`) with such benchmarks in terms of communication rounds and wall-clock time. We examine `FLANP` against the benchmarks under both full and partial node participation scenarios and highlight its practicality. We consider computation speeds that are uniformly distributed and exponentially distributed.

Uniform computation speeds.

Data and Network. We use three main datasets for different problems: MNIST ($60,000$ training, $10,000$ test samples), CIFAR10 ($50,000$ training, $10,000$ test samples) and synthetic ($10,000$ samples) datasets. To implement our algorithm, we employ a federated network of $N \in \{20, 50, 100\}$ heterogeneous clients and in order to model the device heterogeneity, we realize and then fix the computation speed of each node i, i.e. T_i from the interval $[50, 500]$ uniformly at random.

Neural Network Training. We train a fully connected neural network with two hidden layers with 128 and 64 neurons and compare with three other benchmarks including `FedNova` which is stragglers-resilient. We conduct two sets of experiments over CIFAR10 and MNIST on a network of $N = 20$ clients, as demonstratd in Figures 1 and 2 where `FLANP` accelerates `FedNova` by up to $3\times$.

Random exponential computation speeds. Using the same setup described earlier, we here pick the clients' computation speed to be i.i.d. random exponential variables. We train a fully connected neural network with two hidden layers with 128 and 64 neurons on MNIST and compare with `FedAvg`, `FedGATE` and `FedNova` as demonstrated in Figure 3.

Comparison with partial node participation methods. Thus far, we have compared the `FLANP` method with federated benchmarks in which *all* of the available nodes participate in training in every round. To demonstrate the resiliency of `FLANP` to partial node participation methods, we consider two different scenarios. First, we compare the wall-clock time of a neural network training of `FLANP` with partial node participation `FedGATE` in which only k out of $N = 50$ nodes are randomly picked and participate in each round. As demonstrated in Figure 4(a), `FLANP` is significantly faster than `FedGATE` with partial node participation. Second, we consider the case that the k participating nodes are not randomly picked, but are the fastest clients. As shown in Figure 4(b), although partial participation methods with k fastest nodes begin to outperform `FLANP`, towards the end of the training, they suffer from higher training error saturation as the data samples of *only* k nodes contribute in the learned model and hence the final model is significantly inaccurate.

References

[1] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *arXiv preprint arXiv:1908.07873*, 2019.

[2] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, pp. 1273–1282, PMLR, 2017.

[4] S. U. Stich, "Local sgd converges fast and communicates little," in *ICLR 2019 ICLR 2019 International Conference on Learning Representations*, no. CONF, 2019.

[5] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint arXiv:1903.03934*, 2019.

[6] T. Nishio and R. 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.

[7] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *arXiv preprint arXiv:2007.07481*, 2020.

[8] A. Reisizadeh, H. Taheri, A. Mokhtari, H. Hassani, and R. Pedarsani, "Robust and communication-efficient collaborative learning," in *Advances in Neural Information Processing Systems*, pp. 8388–8399, 2019.

[9] A. Mokhtari and A. Ribeiro, "First-order adaptive sample size methods to reduce complexity of empirical risk minimization," in *NeurIPS*, 2017.

[10] A. Mokhtari, A. Ozdaglar, and A. Jadbabaie, "Efficient nonconvex empirical risk minimization via adaptive sample size methods," in *AISTATS*, 2019.

[11] A. Mokhtari, H. Daneshmand, A. Lucchi, T. Hofmann, and A. Ribeiro, "Adaptive Newton method for empirical risk minimization to statistical accuracy," in *NeurIPS*, 2016.

[12] M. Eisen, A. Mokhtari, and A. Ribeiro, "Large scale empirical risk minimization via truncated adaptive Newton method," in *AISTATS*, 2018.

[13] M. Jahani, X. He, C. Ma, A. Mokhtari, D. Mudigere, A. Ribeiro, and M. Takác, "Efficient distributed hessian free algorithm for large-scale empirical risk minimization via accumulating sample strategy," in *AISTATS*, 2020.

[14] J. Wang and G. Joshi, "Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms," *arXiv preprint arXiv:1808.07576*, 2018.

[15] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *International Conference on Learning Representations*, 2019.

[16] Z. Huo, Q. Yang, B. Gu, L. C. Huang, *et al.*, "Faster on-device training using new federated momentum algorithm," *arXiv preprint arXiv:2002.02090*, 2020.

[17] G. Malinovsky, D. Kovalev, E. Gasanov, L. Condat, and P. Richtarik, "From local sgd to local fixed point methods for federated learning," *arXiv preprint arXiv:2004.01442*, 2020.

[18] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[19] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.

[20] O. Bousquet, *Concentration inequalities and empirical processes theory applied to the analysis of learning algorithms*. PhD thesis, École Polytechnique: Department of Applied Mathematics Paris, France, 2002.

[21] P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe, "Convexity, classification, and risk bounds," *Journal of the American Statistical Association*, vol. 101, no. 473, pp. 138–156, 2006.

[22] R. Frostig, R. Ge, S. M. Kakade, and A. Sidford, "Competing with the empirical risk minimizer in a single pass," in *Conference on learning theory*, pp. 728–763, 2015.

[23] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *arXiv preprint arXiv:1812.06127*, 2018.

[24] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for on-device federated learning," *arXiv preprint arXiv:1910.06378*, 2019.

[25] K. Mishchenko, E. Gorbunov, M. Takáč, and P. Richtárik, "Distributed learning with compressed gradient differences," *arXiv preprint arXiv:1901.09269*, 2019.

[26] F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi, "Federated learning with compression: Unified analysis and sharp guarantees," *arXiv preprint arXiv:2007.01154*, 2020.