Fair Resource Allocation in Federated Learning

Abstract

Federated learning involves jointly learning over massively distributed partitions of data generated on remote devices. Naively minimizing an aggregate loss function in such a network may disproportionately advantage or disadvantage some of the devices. In this work, we propose q-Fair Federated Learning (q-`FFL`), a novel optimization objective inspired by resource allocation strategies in wireless networks that encourages a more *fair* accuracy distribution across devices in federated networks. To solve q-`FFL`, we devise a scalable method, q-`FedAvg`, that is suited to federated networks. We validate both the improved fairness and flexibility of q-`FFL` and the efficiency of q-`FedAvg` through simulations on a suite of federated datasets.

1. Introduction

With the growing prevalence of IoT-type devices, data is frequently collected and processed outside of the data center and directly on distributed devices, such as wearable devices or mobile phones. *Federated learning* is a promising learning paradigm in this setting that pushes statistical model training to the edge (McMahan et al., 2017).

The number of devices in federated networks is generally large—ranging from hundreds to millions. While one can naturally view federated learning as a multi-task learning problem where each device corresponds to a task (Smith et al., 2017), the focus is often instead to fit a single global model over these distributed devices/tasks via some empirical risk minimization objective (McMahan et al., 2017). Naively minimizing the average loss via such an objective may disproportionately advantage or disadvantage some of the devices, which is exacerbated by the fact that the data are often heterogeneous across devices both in terms of size and distribution. In this work, we therefore ask: Can we devise an efficient optimization method to encourage a *more fair distribution* of the model performance across devices in federated networks?

There has been tremendous recent interest in developing fair methods for machine learning. However, current methods that could help to improve the fairness of the accuracy distribution in federated networks are typically proposed for a much smaller number of devices, and may be impractical in federated settings due to the number of involved constraints (Cotter et al., 2018). Recent work that has been proposed specifically for the federated setting has also only been applied at small scales (2-3 groups/devices), and lacks flexibility by optimizing only the performance of the single worst device (Mohri et al., 2019).

In this work, we propose q-`FFL`, a novel optimization objective that addresses fairness issues in federated learning. Inspired by work in fair resource allocation for wireless networks, q-`FFL` minimizes an aggregate *reweighted* loss parameterized by q such that the devices with higher loss are given higher relative weight to encourage less variance in the accuracy distribution. In addition, we propose a lightweight and scalable distributed method, q-`FedAvg`, to efficiently solve q-`FFL`, which carefully accounts for important characteristics of the federated setting such as communication-efficiency and low participation of devices (Bonawitz et al., 2019; McMahan et al., 2017). We empirically demonstrate the fairness, efficiency, and flexibility of q-`FFL` and q-`FedAvg` compared with existing baselines. On average, q-`FFL` is able to reduce the variance of accuracies across devices by 45% while maintaining the same overall average accuracy.

2. Related Work

Fairness in Machine Learning. There are several widespread approaches in the machine learning community to address *fairness*, which is typically defined as the protection of some specific attribute(s) (e.g., (Hardt et al., 2016)). In addition to preprocess the data (Feldman et al., 2015) and post-process the model (Feldman, 2015; Hardt et al., 2016), another set of works optimize an objective under some explicit fairness constraints during training time (Agarwal et al., 2018; Cotter et al., 2018; Hashimoto et al., 2018; Woodworth et al., 2017; Zafar et al., 2017; 2015). Our work also enforces fairness during training, though we define fairness as the accuracy distribution across devices in federated learning, as opposed to the protection of a specific attribute (Section 3). Cotter et al.

(2018) use a notion of 'minimum accuracy' as one special case of the rate constraints, which is conceptually similar to our goal. However, it requires each device to have one constraint, which is not practical in the federated setting. In federated settings, Mohri et al. (2019) proposes a minimax optimization scheme, Agnostic Federated Learning (AFL), which optimizes for the performance of the single worst device. This method has only been applied at small scales (for a handful of groups). In addition, our objective is more flexible because q may be tuned based on the amount of fairness desired.

Fairness in Resource Allocation. Fair resource allocation has been extensively studied in fields such as network management (Ee & Bajcsy, 2004; Hahne, 1991; Kelly et al., 1998; Neely et al., 2008) and wireless communications (Eryilmaz & Srikant, 2006; Nandagopal et al., 2000; Sanjabi et al., 2014; Shi et al., 2014). In these contexts, the problem is defined as allocating a scarce shared resource, e.g. communication time or power, among many users. In these cases directly maximizing utilities such as total throughput usually leads to unfair allocations where some users receive poor service. Several measurements have been proposed to balance between fairness and total throughput. Among them, a unified framework is captured through α-fairness (Lan et al., 2010; Mo & Walrand, 2000), in which the emphasis on fairness can be tuned by changing a single parameter, α. If we think of the global model as a resource to serve the users (or devices), it is natural to ask similar questions about the fairness of the service that devices receive and use similar tools to promote fairness. Despite this, we are unaware of any work that uses fairness criteria from resource allocation to modify training objectives in machine learning. Inspired by the α-fairness metric, we propose a similarly modified objective function, q-Fair Federated Learning (q-FFL), to encourage a more fair accuracy distribution across devices in the context of federated training. We empirically demonstrate its benefits in Section 4.

Federated and Distributed Optimization. Federated learning faces fundamental challenges such as expensive communication, variability in hardware, network connection, and power of devices, and heterogeneous local data distribution amongst devices, making it distinct from classical distributed optimization (Recht et al., 2011; Shalev-Shwartz & Zhang, 2013; Smith et al., 2018). In order to reduce communication, as well as to tolerate heterogeneity, methods that allow for local updating and low participation among devices have become de facto solvers for this setting (Li et al., 2018; McMahan et al., 2017; Smith et al., 2017). We incorporate recent advancements in this field when designing methods to solve the q-FFL objective, which we describe in Section 3.3.

3. Fair Federated Learning

We first formally define the classical federated learning objective and methods, and introduce our proposed notion of fairness in Section 3.1. We then introduce q-FFL, a novel objective that encourages a more fair accuracy distribution across all devices (Section 3.2). Finally, in Section 3.3, we describe q-FedAvg, an efficient distributed method we develop to solve the objective in federated settings.

3.1. Preliminaries: Classical Federated Learning

Federated learning involves fitting a global model on distributed data generated on hundreds to millions of remote devices. In particular, the goal is to minimize:

$$\min_w F(w) = \sum_{k=1}^{m} p_k F_k(w), \tag{1}$$

where m is the total number of devices, $p_k \geq 0$, and $\sum_k p_k = 1$. The local objective F_k's can be defined by empirical risks over local data, i.e., $F_k(w) = \frac{1}{n_k} \sum_{j_k=1}^{n_k} f_{j_k}(w)$, where n_k is the number of samples available locally. We can set p_k to be $\frac{n_k}{n}$, where $n = \sum_k n_k$ is the total number of samples in the entire dataset.

Most prior work solves (1) by first subsampling devices with probabilities proportional to n_k at each round, and then applying an optimizer such as Stochastic Gradient Descent (SGD) locally to perform updates. These *local updating methods* enable flexible and efficient communication by running the optimizer for a variable number of iterations locally on each device, e.g., compared to traditional distributed (stochastic) gradient descent, which would simply calculate a subset of the gradients (Stich, 2019; Wang & Joshi, 2018; Woodworth et al., 2018; Yu et al., 2019). FedAvg (Algorithm 2, Appendix A) (McMahan et al., 2017) is one of the leading methods to solve (1).

However, solving (1) in this manner can implicitly introduce unfairness among different devices. For instance, the learned model may be biased towards the devices with higher number of data points. Formally, we define our desired fairness criteria for federated learning below.

Definition 1 (Fairness of distribution)**.** For trained models w and $\tilde{w}$, we say that model w provides a more *fair* solution to Objective (1) than model $\tilde{w}$ if the variance of the performance of model w on the m devices, $\{a_1, \ldots a_m\}$, is smaller than the variance of the performance of model $\tilde{w}$ on the m devices, i.e., $\mathbf{Var}(a_1, \ldots, a_m) \leq \mathbf{Var}(\tilde{a}_1, \ldots, \tilde{a}_m)$.

In this work, we take 'performance' for device k, a_k, to be the *testing accuracy* of applying the trained model w on the test data for device k. Our goal is to reduce the variance while maintaining the same (or similar) average accuracy.

3.2. The objective: q-Fair Federated Learning (q-**FFL**)

A natural idea to achieve fairness as defined in (1) would be to *reweight* the objective—assigning higher weight to devices with poor performance, so that the distribution of accuracies in the network reduces in variance. Note that this re-weighting must be done dynamically, as the performance of the devices depends on the model being trained, which cannot be evaluated a priori. Drawing inspiration from α-fairness, a utility function used in fair resource allocation in wireless networks, we propose the following objective q-FFL. For given local non-negative cost functions F_k and parameter $q > 0$, we define the overall q-Fair Federated Learning (q-FFL) objective as

$$\min_{w} \; F_q(w) = \sum_{k=1}^{m} \frac{p_k}{q+1} F_k^{q+1}(w) \tag{2}$$

Intuitively, the larger we set q, the larger relative price we pay for devices k with high local empirical loss, $F_k(w)$. Here, q is a tunable parameter that depends on the amount of fairness we wish to impose in the network. Setting $q = 0$ does not encourage any fairness beyond the classical federated learning objective (1). A larger q means that we emphasize devices with higher losses (lower accuracies), thus reducing the variance between the accuracy distribution and potentially inducing more fairness in accordance with Definition 1.

3.3. The solver: FedAvg-style q-Fair Federated Learning (q-**FedAvg**)

We first propose a *fair* but *less efficient* method q-FedSGD, to illustrate the main techniques we use to solve the q-FFL objective (2). We then provide a more efficient counterpart q-FedAvg, by considering key properties of federated algorithms such as local updating schemes.

Hyperparameter tuning: q and step-sizes. In devising a method to solve q-FFL (2), we begin by noting that it is important to first determine how to set q. In practice, q can be tuned based on the desired amount of fairness. It is therefore common to train a *family of objectives* for different q values so that a practitioner can explore the tradeoff between accuracy and fairness for the application at hand. Nevertheless, to optimize q-FFL in a scalable fashion, we rely on gradient-based methods, where the step-size inversely depends on the Lipchitz constant of the function's gradient, which is often unknown and selected via grid search (Ghadimi & Lan, 2013; Nesterov, 2013). As we intend to optimize q-FFL for various values of q, the Lipchitz constant will change as we change q—requiring step-size tuning for all values of q. This can quickly cause the search space to explode. To overcome this issue, we propose estimating the local Lipchitz constant of the gradient for the family of q-FFL by using the Lipchitz constant we infer on $q = 0$. This allows us to dynamically adjust the step-size for the q-FFL objective, avoiding the manual tuning for each q. In Lemma 2 we formalize the relation between the Lipschitz constant, L, for $q = 0$ and $q > 0$.

Lemma 2. *If the non-negative function $f(\cdot)$ has a Lipchitz gradient with constant L, then for any $q \geq 0$ and at any point w,*

$$L_q(w) = Lf(w)^q + qf(w)^{q-1}\|\nabla f(w)\|^2 \tag{3}$$

is an upper-bound for the local Lipchitz constant of the gradient of $\frac{1}{q+1}f^{q+1}(\cdot)$ at point w. Furthermore, the gradient of $\frac{1}{q+1}f^{q+1}(\cdot)$ at point w is $f^q(w)\nabla f(w)$.

See proof in Appendix B.

A first approach: q-FedSGD. In our first fair algorithm q-FedSGD, we solve Objective (2) using mini-batch SGD on a subset of devices at each round, and apply the above result to each selected device to obtain *local* Lipchitz constants for gradients of local functions F_k. By averaging those estimates, we obtain an estimate for the Lipchitz constant for the gradient of q-FFL. Then, the step-size (inverse of this estimate) is applied, like other gradient based algorithms; see Algorithm 3 in Appendix A for more details.

Algorithm 1 q-FedAvg (proposed method)

```
Input: $K$, $T$, $q$, $1/L$, $\eta$, $w^0$, $p_k$, $k = 1, \cdots, m$
for $t = 0, \cdots, T-1$ do
    Server chooses a subset $S_t$ of $K$ devices at random (each device $k$ is chosen with prob. $p_k$)
    Server sends $w^t$ to all chosen devices
    Each device $k$ updating $w_t$ for $E$ epochs of SGD with step size $\eta$ to obtain $\bar{w}_k^{t+1}$
    Each device computes:
        $\Delta w_k^t = w^t - \bar{w}_k^{t+1}, \Delta_k^t = F_k^q(w^t)\Delta w_k^t$
        $h_k^t = qF_k^{q-1}(w^t)\|\Delta w_k^t\|^2 + LF_k^q(w^t)$
    Each chosen device $k$ sends $\Delta_k^t$ and $h_k^t$ to the server
    Server aggregates the computes $w^{t+1}$ as:
        $w^{t+1} = w^t - \frac{\sum_{k \in S_t} \Delta_k^t}{\sum_{k \in S_t} h_k^t}$
end for
```

Improving communication efficiency: q-FedAvg. In federated settings, communication-efficient schemes that allow for local updating have become de facto solvers. We incorporate this technique by allowing each selected device to run some number of local updates and then apply the updates in the gradient computation of q-FFL. The details of our method (q-FedAvg) are given in Algorithm 1. Note that when q=0, q-FFL corresponds to the normal objective in federated learning (Equation (1)), and q-FedAvg is also reduced to FedAvg (McMahan et al., 2017) where no fairness modification is introduced.

4. Evaluation

We first describe our experimental setup, then demonstrate the improved fairness of the q-FFL objective by comparing q-FFL with several baselines, and finally show the efficiency of q-FedAvg compared with q-FedSGD.

Experiment setups. We explore both convex and non-convex models on four federated datasets curated from prior work in federated learning (Smith et al., 2017; Li et al., 2018; Caldas et al., 2018). Full details of the datasets and models are given in Appendix D. Throughout the experiments, we show results on the Vehicle dataset. Similar results on all datasets are provided in Appendix C.

Fairness of q-FFL. We verify that the proposed objective q-FFL leads to more fair solutions (according to Definition 1) for federated data, compared with FedAvg and two other baselines that are likely to impose fairness in federated networks.

(1) Compare with FedAvg. In Figure 1 (left), we compare the final testing accuracy distributions of two objectives (q=0 and a tuned value of q=5) averaged across 5 random shuffles of Vehicle. We observe that the objective with q=5 results in more centered (i.e., fair) testing accuracy distributions with lower variance. We further report the worst and best 10% testing accuracies and the variances of accuracies in Table 1. We see that the average testing accuracy remains almost unchanged with the proposed objective despite significant reductions in variance. See similar results on other datasets in Figure 2 and Table 2 in Appendix C.

(2) Compare with weighing each device equally. We compare q-FFL with a heuristic that samples devices uniformly and report testing accuracy in Figure 1 (middle). A table with the statistics of accuracy distribution on all datasets is given in the appendix in Table 3. While the 'weighing each device equally' heuristic tends to outperform our method in training accuracy distributions (Figure 5 and Table 7 in Appendix D.3), our method produces more fair solutions in terms of testing accuracies. One explanation for this is that uniform sampling is a static method and can easily overfit to devices with very few data points, whereas q-FFL has better generalization properties due to its dynamic nature.

(3) Compare with weighing each device adversarially. We further compare with AFL (Mohri et al., 2019), which weighs each device adversarially, namely, optimizes for the performance of the device with the highest loss. This is the only work we are aware of that aims to address fairness issues in federated learning. See Appendix D.2 for details of our AFL implementation. We also observe that q-FFL outperforms AFL when q is set appropriately (Table 4, Appendix D). We note that q is also tunable depending on the amount of fairness desired. Interestingly, we observe q-FFL converges faster compared with AFL (see Figure 7 in Appendix D.3) in terms of communication rounds.

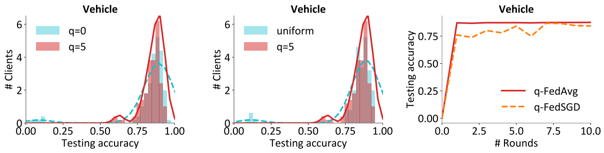

Figure 1. Fairness of q-FFL and efficiency of q-FedAvg. *Left:* Fairness of q-FFL q>0 compared with the original objective (q=0, Equation (1)). *Middle:* Fairness of q-FFL q>0 compared with the uniform sampling baseline. *Right:* q-FedAvg converges faster than q-FedSGD.

Table 1. Statistics of the testing accuracy distribution for q-FFL on **Vehicle**. By setting $q > 0$, the variance of the final accuracy distribution decreases, and the worst 10% accuracy increases, while the overall accuracy remains fairly constant.

Obj.	Avg.	Worst 10%	Best 10%	Var.
$q = 0$	87.3%	43.0%	**95.7%**	291
$q = 5$	87.7%	**69.9%**	94.0%	**48**

Choosing q. A natural question is determine how q should be tuned in the q-FFL objective. The framework is flexible in that it allows one to choose q to tradeoff between reduced variance of the accuracy distribution and a high average accuracy. In particular, a reasonable approach in practice would be to run Algorithm 1 with multiple q's in parallel to obtain multiple final global models, and then let each device select amongst these based on performance on the validation data. We show benefits of this device-specific strategy in Table 8 in Appendix D.3.

Efficiency of q-FedAvg. Finally, we show the efficiency of q-FedAvg by comparing Algorithm 1 with its non-local-updating baseline q-FedSGD (Algorithm 3) with the same objective ($q > 0$). At each communication round, q-FedAvg runs one epoch of local updates on each selected device, while q-FedSGD runs gradient descent using all local training data on that device. In Figure 1 (right), q-FedAvg converges faster than q-FedSGD in terms of communication rounds. We note here that these two methods communicate and compute equivalent amounts at each round. See full (and better) results on all datasets in Appendix C. Our method is also lightweight, and can be easily integrated into existing implementations of federated learning algorithms such as TensorFlow Federated (TFF).

References

Tensorflow federated: Machine learning on decentralized data. URL https://www.tensorflow.org/federated.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M. K., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. Tensorflow: A system for large-scale machine learning. In *Operating Systems Design and Implementation*, pp. 265–283, 2016.

Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., and Wallach, H. A reductions approach to fair classification. *arXiv preprint arXiv:1803.02453*, 2018.

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konecny, J., Mazzocchi, S., McMahan, H. B., Overveldt, T. V., Petrou, D., Ramage, D., and Roselande, J. Towards federated learning at scale: System design. In *Conference on Systems and Machine Learning*, 2019.

Caldas, S., Wu, P., Li, T., Konečnỳ, J., McMahan, H. B., Smith, V., and Talwalkar, A. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Cotter, A., Jiang, H., Wang, S., Narayan, T., Gupta, M., You, S., and Sridharan, K. Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals. *arXiv preprint arXiv:1809.04198*, 2018.

Ee, C. T. and Bajcsy, R. Congestion control and fairness for many-to-one routing in sensor networks. In *International Conference on Embedded Networked Sensor Systems*, pp. 148–161, 2004.

Eryilmaz, A. and Srikant, R. Joint congestion control, routing, and mac for stability and fairness in wireless networks. *IEEE Journal on Selected Areas in Communications*, 24(8):1514–1524, 2006.

Feldman, M. Computational fairness: Preventing machine-learned discrimination. 2015.

Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., and Venkatasubramanian, S. Certifying and removing disparate impact. In *International Conference on Knowledge Discovery and Data Mining*, pp. 259–268, 2015.

Ghadimi, S. and Lan, G. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Go, A., Bhayani, R., and Huang, L. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12), 2009.

Hahne, E. L. Round-robin scheduling for max-min fairness in data networks. *IEEE Journal on Selected Areas in communications*, 9(7):1024–1039, 1991.

Hardt, M., Price, E., and Srebro, N. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, pp. 3315–3323, 2016.

Hashimoto, T. B., Srivastava, M., Namkoong, H., and Liang, P. Fairness without demographics in repeated loss minimization. *arXiv preprint arXiv:1806.08010*, 2018.

Kelly, F. P., Maulloo, A. K., and Tan, D. K. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research society*, 49(3):237–252, 1998.

Lan, T., Kao, D., Chiang, M., and Sabharwal, A. An axiomatic theory of fairness in network resource allocation. In *Conference on Information Communications*, pp. 1343–1351, 2010.

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

McMahan, H. B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics*, 2017.

Mo, J. and Walrand, J. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on networking*, (5):556–567, 2000.

Mohri, M., Sivek, G., and Suresh, A. T. Agnostic federated learning. *arXiv preprint arXiv:1902.00146*, 2019.

Nandagopal, T., Kim, T.-E., Gao, X., and Bharghavan, V. Achieving mac layer fairness in wireless packet networks. In *International Conference on Mobile Computing and Networking*, pp. 87–98, 2000.

Neely, M. J., Modiano, E., and Li, C.-P. Fairness and optimal stochastic control for heterogeneous networks. *IEEE/ACM Transactions On Networking*, 16(2):396–409, 2008.

Nesterov, Y. *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013.

Pennington, J., Socher, R., and Manning, C. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing*, pp. 1532–1543, 2014.

Recht, B., Re, C., Wright, S., and Niu, F. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pp. 693–701, 2011.

Sanjabi, M., Razaviyayn, M., and Luo, Z.-Q. Optimal joint base station assignment and beamforming for heterogeneous networks. *IEEE Transactions on Signal Processing*, 62(8):1950–1961, 2014.

Shalev-Shwartz, S. and Zhang, T. Accelerated mini-batch stochastic dual coordinate ascent. In *Advances in Neural Information Processing Systems*, pp. 378–385, 2013.

Shamir, O., Srebro, N., and Zhang, T. Communication-efficient distributed optimization using an approximate newton-type method. In *International Conference on Machine Learning*, pp. 1000–1008, 2014.

Shi, H., Prasad, R. V., Onur, E., and Niemegeers, I. Fairness in wireless networks: Issues, measures and challenges. *IEEE Communications Surveys and Tutorials*, 16(1):5–24, 2014.

Smith, V., Chiang, C.-K., Sanjabi, M., and Talwalkar, A. S. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pp. 4424–4434, 2017.

Smith, V., Forte, S., Chenxin, M., Takáč, M., Jordan, M. I., and Jaggi, M. Cocoa: A general framework for communication-efficient distributed optimization. *Journal of Machine Learning Research*, 18:230, 2018.

Stich, S. U. Local sgd converges fast and communicates little. In *International Conference on Learning Representations*, 2019.

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

William Shakespeare. The Complete Works of William Shakespeare. Publicly available at //www.gutenberg.org/ebooks/100.

Woodworth, B., Gunasekar, S., Ohannessian, M. I., and Srebro, N. Learning non-discriminatory predictors. *arXiv preprint arXiv:1702.06081*, 2017.

Woodworth, B. E., Wang, J., Smith, A., McMahan, B., and Srebro, N. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. In *Advances in Neural Information Processing Systems*, pp. 8496–8506, 2018.

Yu, H., Yang, S., and Zhu, S. Parallel restarted sgd for non-convex optimization with faster convergence and less communication. In *AAAI Conference on Artificial Intelligence*, 2019.

Zafar, M. B., Valera, I., Rodriguez, M. G., and Gummadi, K. P. Fairness constraints: Mechanisms for fair classification. *arXiv preprint arXiv:1507.05259*, 2015.

Zafar, M. B., Valera, I., Gomez Rodriguez, M., and Gummadi, K. P. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *International Conference on World Wide Web*, pp. 1171–1180, 2017.

A. Algorithms

We summarize the FedAvg algorithm proposed in (McMahan et al., 2017) below.

Algorithm 2 Federated Averaging (McMahan et al., 2017) (FedAvg)

```
Input: $K$, $T$, $\eta$, $E$, $w^0$, $N$, $p_k$, $k = 1, \cdots, N$
for $t = 0, \cdots, T-1$ do
    Server chooses a subset $S_t$ of $K$ devices at random (each device $k$ is chosen with probability $p_k$)
    Server sends $w^t$ to all chosen devices
    Each device $k$ updates $w^t$ for $E$ epochs of SGD on $F_k$ with step-size $\eta$ to obtain $w_k^{t+1}$
    Each chosen device $k$ sends $w_k^{t+1}$ back to the server
    Server aggregates the $w$'s as $w^{t+1} = \frac{1}{K}\sum_{k \in S_t} w_k^{t+1}$
end for
```

We summarize our proposed method q-FedSGD below.

Algorithm 3 q-FedSGD

```
Input: $K$, $T$, $q$, $1/L$, $w^0$, $p_k$, $k = 1, \cdots, m$
for $t = 0, \cdots, T-1$ do
    Server chooses a subset $S_t$ of $K$ devices at random (each device $k$ is chosen with prob. $p_k$)
    Server sends $w^t$ to all chosen devices
    Each device computes:
        $\Delta_k^t = F_k^q(w^t)\nabla F_k(w^t)$
        $h_k^t = qF_k^{q-1}(w^t)\|\nabla F_k(w^t)\|^2 + LF_k^q(w^t)$
    Each chosen device $k$ sends $\Delta_k^t$ and $h_k^t$ to the server
    Server aggregates the computes $w^{t+1}$ as:
        $w^{t+1} = w^t - \frac{\sum_{k \in S_t} \Delta_k^t}{\sum_{k \in S_t} h_k^t}$
end for
```

B. Proof for Lemma 2

Proof. At any point w, we can compute $\nabla^2 f(w)$

$$\nabla^2 f(w) = qf^{q-1}(w) \underbrace{\nabla f(w) \nabla^T f(w)}_{\preceq \|\nabla f(w)\|^2 \times I} + f^q(w) \underbrace{\nabla^2 f(w)}_{\preceq L \times I}. \tag{4}$$

As a result, $\|\nabla^2 f(w)\|_2 \le L_q(w) = Lf(w)^q + qf(w)^{q-1}\|\nabla f(w)\|^2$. □

C. Full Results

Fairness of q-FFL. We demonstrate the improved fairness of q-FFL on all the four datasets in Figure 2 and Table 2.

Comparison with uniform sampling. We compare q-FFL with uniform sampling schemes and report testing accuracy on all datasets in Figure 3. A table with the final accuracies and variances is given in Table 3. While the 'weighing each device equally' heuristic tends to outperform our method in training accuracy distributions (Figure 5 and Table 7), our

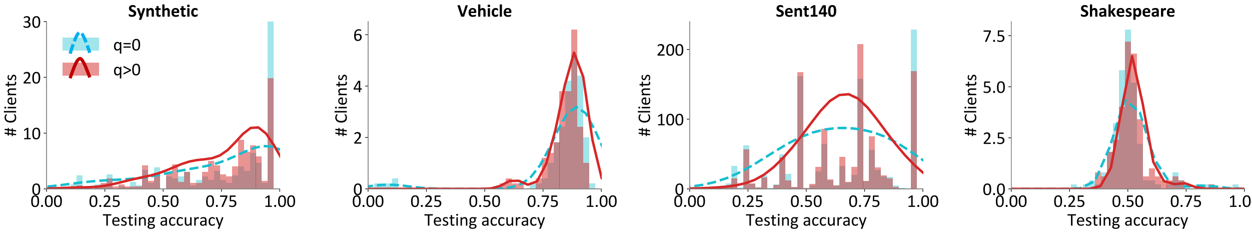

Figure 2. q-FFL leads to fairer test accuracy distributions. With $q > 0$, the distributions shift towards the center as low accuracies increase at the cost of decreasing high accuracies on some devices. Setting q=0 corresponds to the original objective (Equation (1)). The selected q values for $q > 0$ on the four datasets, as well as distribution statistics, are shown in Table 2.

Table 2. Statistics of the testing accuracy distribution for q-FFL. By setting $q > 0$, the accuracy of the worst 10% devices is increased at the cost of possibly decreasing the accuracy of the best 10% devices. While the average accuracy remains similar, the variance of the final accuracy distribution decreases.

Dataset	**Objective**	**Average**	**Worst 10%**	**Best 10%**	**Variance**
Synthetic	$q = 0$	80.8% ± .9%	18.8% ± 5.0%	100.0% ± 0.0%	724 ± 72
	$q = 1$	79.0% ± 1.2%	**31.1%** ± 1.8%	100.0% ± 0.0%	**472** ± 14
Vehicle	$q = 0$	87.3% ± .5%	43.0% ± 1.0%	**95.7%** ± 1.0%	291 ± 18
	$q = 5$	87.7% ± .7%	**69.9%** ± .6%	94.0% ± .9%	**48** ± 5
Sent140	$q = 0$	65.1% ± 4.8%	15.9% ± 4.9%	100.0% ± 0.0%	697 ± 132
	$q = 1$	66.5% ± .2%	**23.0%** ± 1.4%	100.0% ± 0.0%	**509** ± 30
Shakespeare	$q = 0$	51.1% ± .3%	39.7% ± 2.8%	**72.9%** ± 6.7%	82 ± 41
	$q = .001$	52.1% ± .3%	**42.1%** ± 2.1%	69.0% ± 4.4%	**54** ± 27

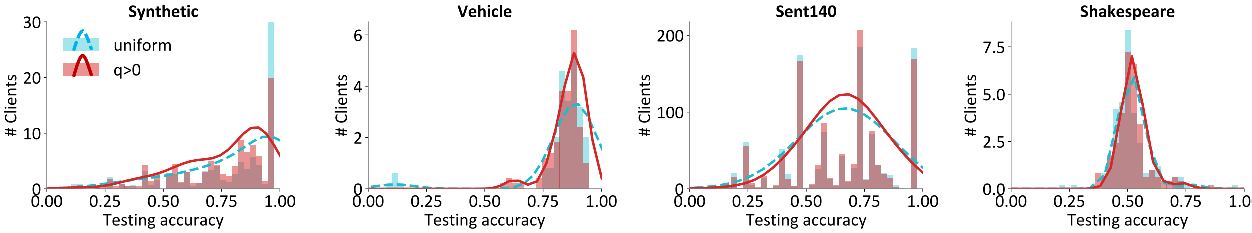

Figure 3. q-FFL ($q > 0$) compared with uniform sampling. In terms of testing accuracy, our objective produces more fair solutions than uniform sampling. Distribution statistics are provided in Table 3.

Table 3. More statistics indicating the resulting fairness of q-FFL compared with the uniform sampling baseline. Again, we observe that the testing accuracy of the worst 10% devices tends to increase, and the variance of the final testing accuracies is smaller.

Dataset	**Objective**	**Average**	**Worst 10%**	**Best 10%**	**Variance**
Synthetic	uniform	82.2% ± 1.1%	30.0% ± .4%	100.0% ± 0.0%	525 ± 47
	$q = 1$	79.0% ± 1.2%	**31.1%** ± 1.8%	100.0% ± 0.0%	**472** ± 14
Vehicle	uniform	86.8% ± .3%	45.4% ± .3%	**95.4%** ± .7%	267 ± 7
	$q = 5$	87.7% ± 0.7%	**69.9%** ± .6%	94.0% ± .9%	**48** ± 5
Sent140	uniform	66.6% ± 2.6%	21.1% ± 1.9%	100.0% ± 0.0%	560 ± 19
	$q = 1$	66.5% ± .2%	**23.0 %** ± 1.4%	100.0% ± 0.0%	**509** ± 30
Shakespeare	uniform	50.9% ± .4%	41.0% ± 3.7%	**70.6%** ± 5.4%	71 ± 38
	$q = .001$	52.1% ± .3%	**42.1%** ± 2.1%	69.0% ± 4.4%	**54** ± 27

Table 4. Our objective compared with baseline of weighing devices adversarially. q-FFL ($q > 0$) outperforms AFL on devices with lowest testing accuracy. The tunable parameter q controls how much fairness we would like to achieve. Each accuracy is averaged across 5 runs with different random initializations.

	Adult			**Fashion MNIST**			
Objectives	average	Dr.	non-Dr.	average	shirt	pullover	T-shirt
q-FFL, q=0	83.2% ± .1%	69.9% ± .4%	**83.3%** ± .1%	78.8% ± .2%	66.0% ± .7%	84.5% ± .8%	**85.9%** ± .7%
AFL	82.5% ± .5%	73.0% ± 2.2%	82.6% ± .5%	77.8% ± 1.2%	71.4% ± 4.2%	81.0% ± 3.6%	82.1% ± 3.9%
q-FFL, q_1>0	82.6% ± .1%	**74.1%** ± .6%	82.7% ± .1%	77.8% ± .2%	**74.2%** ± .3%	78.9% ± .4%	80.4% ± .6%
q-FFL, q_2>q_1	82.3% ± .1%	**74.4%** ± .9%	82.4% ± .1%	77.1% ± .4%	**74.7%** ± .9%	77.9% ± .4%	78.7% ± .6%

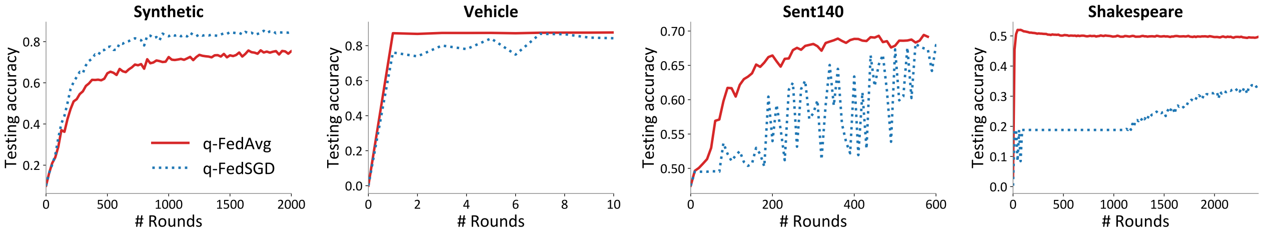

Figure 4. Fix an objective (i.e., using the same q) for each dataset, q-FedAvg (Algorithm 1) compared with q-FedSGD (Algorithm 3). We can see that our method adopting local updating schemes converges faster in terms of communication rounds on most datasets.

method produces more fair solutions in terms of testing accuracies. One explanation for this is that uniform sampling is a static method and can easily overfit to devices with very few data points, whereas q-FFL has better generalization properties due to its dynamic nature.

Comparison with weighing each device adversarially. We show the results of comparing q-FFL with AFL on the two datasets in Table 4. q-FFL outperforms AFL in terms of increasing the lowest accuracies. In addition, q-FFL is more flexible as the parameter q enables the trade-off between increasing the worst accuracies and decreasing the best accuracies.

Efficiency of q-FedAvg. In Figure 4, we show that on most datasets, q-FedAvg converges faster than q-FedSGD in terms of communication rounds due to its local updating scheme. We note here that number of rounds is a reasonable metric for comparison between these methods as they process the same amount of data and perform equivalent amount of communication at each round. Our method is also lightweight, and can be easily integrated into existing implementations of federated learning algorithms such as TensorFlow Federated (TFF).

D. Experimental Details

D.1. Datasets and Models

We provide full details on the datasets and models used in our experiments. The statistics of four federated datasets are summarized in Table 5. We report total number of devices, total number of samples, and mean and deviation in the sizes of total data points on each device. Additional details on the datasets and models are described below.

- **Synthetic:** We follow a similar set up as that in (Shamir et al., 2014) and impose additional heterogeneity. The model is $y = argmax(\text{softmax}(Wx + b))$, $x \in \mathbb{R}^{60}, W \in \mathbb{R}^{10\times 60}, b \in \mathbb{R}^{10}$, and the goal is to learn a global W and b. Samples (X_k, Y_k) and local models on each device k satisfies $W_k \sim \mathcal{N}(u_k, 1)$, $b_k \sim \mathcal{N}(u_k, 1)$, $u_k \sim \mathcal{N}(0, 1)$; $x_k \sim \mathcal{N}(v_k, \Sigma)$, where the covariance matrix Σ is diagonal with $\Sigma_{j,j} = j^{-1.2}$. Each element in v_k is drawn from

$\mathcal{N}(B_k, 1), B_k \sim N(0,1)$. There are 100 devices in total and the number of samples on each devices follows a power law.

- **Vehicle**[1]**:** We use the same Vehicle Sensor (Vehicle) dataset as (Smith et al., 2017), modelling each sensor as a device. Each sample has a 100-dimension feature and a binary label indicating whether this sample is on an AAV-type or DW-type vehicle. We train a linear SVM. We tune the hyperparameters in SVM and report the best configuration.
- **Sent140:** It is a collection of tweets from Sentiment140 (Go et al., 2009) (Sent140). The task is text sentiment analysis which we model as a binary classification problem. The model takes as input a 25-word sequence, embeds each word into a 300-dimensional space using pretrained Glove (Pennington et al., 2014), and outputs a binary label after two LSTM layers and one densely-connected layer.
- **Shakespeare:** This dataset is built from *The Complete Works of William Shakespeare* (McMahan et al., 2017; William Shakespeare. The Complete Works of William Shakespeare). Each speaking role in the plays is associated with a device. We subsample 31 speaking roles to train a deep model for next character prediction. The model takes as input an 80-character sequence, embeds each character into a learnt 8-dimensional space, and outputs one character after two LSTM layers and one densely-connected layer.

Table 5. Statistics of Real Federated Datasets

Dataset	**Devices**	**Samples**	**Samples/device**	
			mean	stdev
Synthetic	100	12,697	127	73
Vehicle	23	43,695	1,899	349
Sent140	1,101	58,170	53	32
Shakespeare	31	116,214	3,749	6,912

D.2. Implementation Details

D.2.1. Machines

We simulate the federated setting (1 server and K devices) on a server with 2 Intel® Xeon® E5-2650 v4 CPUs and 8 NVidia® 1080Ti GPUs.

D.2.2. Softwares

We implement all code in TensorFlow (Abadi et al., 2016) Version 1.10.1.

D.2.3. Implementation and comparison with AFL.

We implement a non-stochastic version of AFL where all devices are selected and updated each round and do a grid search on the AFL hyperparameters, γ_w and γ_λ. In order to draw a fair comparison, we modify Algorithm 1 by sampling all devices and letting each of them run gradient descent at each round. We use the same public datasets (Adult and Fashion MNIST) as in (Mohri et al., 2019).

D.2.4. Hyperparameters

We randomly split data on each local device into 80% training set, 10% testing set, and 10% validation set. We tune q from $\{0.001, 0.01, 0.1, 1, 2, 5, 10, 15\}$ on the validation set and report accuracy distributions on the testing set. For each dataset,

[1] http://www.ecs.umass.edu/~mduarte/Software.html

Table 6. Average testing accuracy under q-FFL objectives. We show that the resulting solutions of q=0 and q¿0 objectives have approximately the same accuracies both with respect to all data points and with respect to all devices.

Dataset	Objective	Accuracy w.r.t. Data Points	Accuracy w.r.t. Devices
Synthetic	$q = 0$	80.8% ± .9%	77.3% ± .6%
	$q = 1$	79.0% ± 1.2%	76.3% ± 1.7%
Vehicle	$q = 0$	87.3% ± .5%	85.6% ± .4%
	$q = 5$	87.7% ± .7%	86.5% ± .7%
Sent140	$q = 0$	65.1% ± 4.8%	64.6% ± 4.5%
	$q = 1$	66.5% ± .2%	66.2% ± .2%
Shakespeare	$q = 0$	51.1% ± .3%	61.4% ± 2.7%
	$q = .001$	52.1% ± .3%	60.0% ± .5%

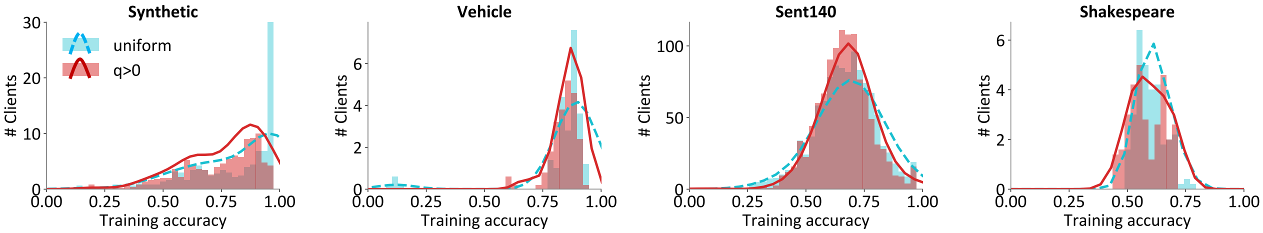

Figure 5. q-FFL ($q > 0$) compared with uniform sampling in training accuracy. We see that in most cases uniform sampling has higher (and more fair) training accuracies due to the fact that it is overfitting to devices with few samples.

we repeat this process for five randomly selected train/test/validation splits, and report the mean and standard deviation across these five runs where applicable. For Synthetic, Vehicle, Sent140, and Shakespeare, optimal[2] q values are 1, 5, 1, and 0.001 respectively. We randomly sample 10 devices each round. We tune the learning rate on FedAvg and use the same learning rate for all experiments of that dataset. The learning rates for Synthetic, Vehicle, Sent140, and Shakespeare are 0.1, 0.01, 0.03, and 0.8 respectively. When running AFL methods, we search for a best γ_w and γ_λ such that AFL achieves the highest testing accuracy on the device with the highest loss within a fixed number of rounds. For Adult, we use $\gamma_w = 0.1$ and $\gamma_\lambda = 0.1$; for Fashion MNIST, we use $\gamma_w = 0.001$ and $\gamma_\lambda = 0.01$. We use the same γ_w as step sizes for q-FedAvg on Adult and Fashion MNIST. In Table 4, $q_1 = 0.01, q_2 = 2$ for q-FFL on Adult and $q_1 = 5, q_2 = 15$ for q-FFL on Fashion MNIST. The number of local epochs is fixed to 1 whenever we do local updates.

D.3. Additional Experiments

Average testing accuracy with respect to devices. We have shown that q-FFL leads to more fair accuracy distributions while maintaining approximately the same testing accuracies in Section 4. Note that we report average testing accuracy with respect to *all data points* in Table 1 and 2. We observe similar results on average accuracy with respect to *all devices* between $q = 0$ and $q > 0$ objectives, as shown in Table 6.

Efficiency of q-FFL compared with AFL. One added benefit of q-FFL is that it leads to faster convergence than AFL even when we use *non-local-updating* methods for both objectives. In Figure 7, we show that when fixing the final testing accuracy for the single worst device, q-FFL converges faster than AFL. As the number of devices increases (from Fashion MNIST to Vehicle), the performance gap between AFL and q-FFL becomes larger because AFL introduces larger variance.

[2] By optimal we mean the setting where the variance of accuracy decreases the most, while keeping the overall average accuracy unchanged.

Table 7. More statistics showing that uniform sampling outperforms q-FFL in terms of training accuracies. We observe that uniform sampling could result in more fair training accuracy distributions with smaller variance in most cases.

Dataset	Objective	Average	Worst 10%	Best 10%	Variance
Synthetic	uniform	83.5% ± .2%	**42.6%** ± 1.4%	**100.0%** ± 0.0%	366 ± 17
	$q = 1$	78.9% ± .2%	41.8% ± 1.0%	96.8% ± .5%	**292** ± 11
Vehicle	uniform	87.3% ± .3%	46.6% ± .8%	**94.8%** ± .5%	261 ± 10
	$q = 5$	87.8% ± .5%	**71.3%** ± 2.2%	93.1% ± 1.4%	**122** ± 12
Sent140	uniform	69.1% ± .5%	42.2% ± 1.1%	**91.0%** ± 1.3%	188 ± 19
	$q = 1$	68.2% ± .6%	**46.0** % ± .3%	88.8% ± .8%	**143** ± 4
Shakespeare	uniform	57.7% ± 1.5%	**54.1%** ± 1.7%	**72.4%** ± 3.2%	**32** ± 7
	$q = .001$	66.7% ± 1.2%	48.0% ± .4%	71.2% ± 1.9%	56 ± 9

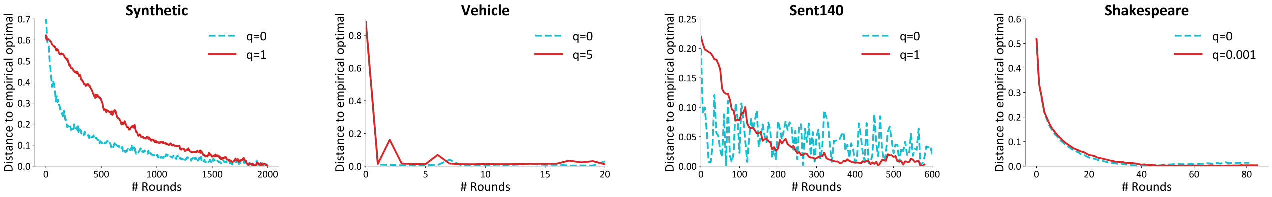

Figure 6. The convergence speed of q-FFL compared with FedAvg. We plot the distance to the highest accuracy achieved versus communication rounds. Although q-FFL with $q > 0$ is a more difficult optimization problem, for the q values we choose that could lead to more fair results, the convergence speed is comparable to that of $q = 0$.

Choosing q. We solve q-FFL with $q \in \{0, 0.001, 0.01, 0.1, 1, 2, 5, 10\}$ in parallel. After training, each device selects the best resulting model based on the validation data and tests the performance of the model using testing set. We report the results in terms of testing accuracy in Table 8. Using this strategy, accuracy variance is reduced and average accuracy is increased. However, this will induce more local computation and additional communication load in each round. But this does not increase the number of communication rounds.

Convergence speed of q-FFL. In Section 4, we show that our solver q-FedAvg using local updating schemes converges significantly faster than q-FedSGD. A natural question one might ask is: will the q-FFL ($q > 0$) objective slows the convergence compared with FedAvg? We empirically investigate this on real datasets. We use q-FedAvg to solve q-FFL, and compare it with FedAvg. As demonstrated in Figure 6, the q values we are choosing that result in more fair solutions do not significantly slowdown convergence.

Table 8. Effects of running q-FFL with several q's in parallel. Multiple global models (corresponding to different q's) are maintained independently during the training process. While this adds additional local computation and more communication load per round, the device-specific strategy has the added benefit of increasing the accuracies of devices with worst 10% accuracies and devices with best 10% accuracies simultaneously.

Dataset	Objective	Average	Worst 10%	Best 10%	Variance
Vehicle	q=0	87.3% ± .5%	43.0% ± 1.0%	95.7% ± 1.0%	291 ± 18
	q=5	87.7% ± .7%	69.9% ± .6%	94.0% ± .9%	48 ± 5
	multiple q's	88.5% ± .3%	70.0% ± 2.0%	95.8% ± .6%	52 ± 7
Shakespeare	q=0	51.1% ± .3%	39.7% ± 2.8%	72.9% ± 6.7%	82 ± 41
	q=.001	52.1% ± .3%	42.1% ± 2.1%	69.0% ± 4.4%	54 ± 27
	multiple q's	52.0 ± 1.5% %	41.0% ± 4.3%	72.0% ± 4.8%	72 ± 32

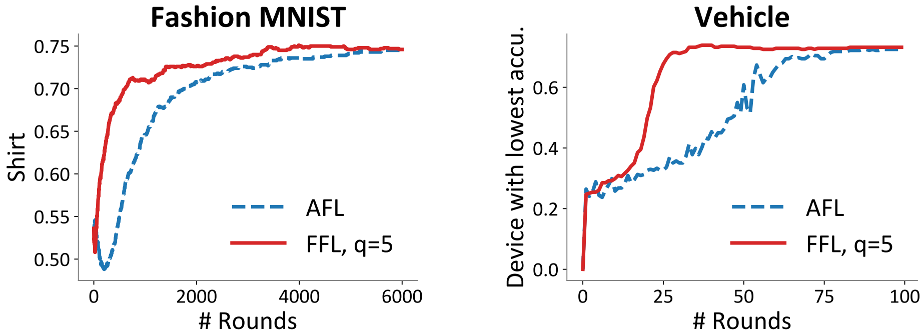

Figure 7. q-`FFL` is more efficient than `AFL`. With the worst device achieving the same final testing accuracy, q-`FFL` converges faster than `AFL`. From Fashion MNIST to Vehicle, as the number of devices increases, the performance gap is larger. We run full gradient descent at each round for both methods.