Federated Learning Optimization Methods Review

Jia Zhao
Computer Science
Cornell Tech
New York, NY 10044
`jz538@cornell.edu`

Abstract

Federated learning is a paradigm for distributed training of machine learning models in networks of remote heterogeneous devices. The heterogeneity in these devices include both system heterogeneity and statistical heterogeneity. The inherent heterogeneity combined with communication costs in large federated learning networks pose challenges to solve the optimization objective. In this work, we review the challenges and discuss several novel algorithms for solving the challenging objective.

1 Introduction

Federated learning is a paradigm for distributed training of machine learning models in networks of remote heterogeneous devices, such as mobile phones, wearable devices, and autonomous vehicles. These devices are only a few of the ubiquitously existing distributed networks that are generating a wealth of data each day. Compared to the past paradigm where models are trained in a data center, the growing needs of computational power and concerns about user privacy have pushed models to be trained on local devices with locally stored data. Federated learning has become a rising solution to tackle the problem of local training with local data in practice. The standard federated learning methods is to learn a single global model from heterogeneous data stored locally on millions of remote devices. In the learning process, the single global model will be deployed on each remote device to be trained with the local data. Each device will only return the intermediate updates back to the central server periodically to improve the global model collectively. The goal is typically to solve the following objective function:

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

where m is the total number of devices, $p_k \geq 0$ and $\sum_{k=1}^{m} p_k = 1$, $F(k)$ is the local objective function for device k. The local objective function $F(k)$ is usually defined as the empirical risk over local data $F_k(w) = \frac{1}{n_k} \sum_{j_k=1}^{n_k} l_{j_k}(w)$, where n_k is the number of training samples available on device k. The user-defined term p_k specifies the relative impact of each device and can be set to $p_k = \frac{n_k}{n}$, where $n = \sum_k n_k$ is the total number of training samples over all devices.

The objective function in 1 is challenging to solve due to two main reasons: heterogeneity in both devices and data, and concerns about communication efficiency. These two reasons also mark the distinction between classical distributed learning and federated learning.

System Heterogeneity System heterogeneity is the differences among devices. The computational power (CPU or GPU), storage (memory), network connections (2G, 3G, 4G, 5G, and Wi-Fi) vary from device to device in the network of devices involved in the same training process [1]. It is also common to have only a small fraction of devices to participate in the current training process

[2]. Some active devices may drop out at any time due to various reasons (for example, battery life). This system-level heterogeneity is an inherent feature that must be considered when designing optimization methods in federated learning. A good optimization method must be applicable on heterogeneous hardware, anticipate a low level of participation among devices, and appropriately handle the dropped out devices.

Statistical Heterogeneity Statistical heterogeneity is the differences among training data samples. Training data in federated learning come from daily users with different life style and user habits. For example, users using the same phone model will have totally different texting behaviors due to demographics and living environments. Thus, the next-word prediction is expected to behave accordingly to best serve different users. Moreover, the number of data samples generated on each device also varies significantly. A model generating photo memories will have different results between a user taking lots of photos everyday and a user merely taking any photos. The properties and number of data samples does not follow a i.i.d (independent and identically distributed) distribution that is usually assumed in classical distributed learning and thus adds complexity in federated learning optimization [3].

Communication Efficiency Communication efficiency is considered as a critical bottleneck in the overall efficiency and scalability of the federated learning network [2]. Since federated learning networks contain a massive number of devices, the in-network communication between each device and the central server can be slower than the local by many orders of magnitude due to bandwidth and other resource limitations [1]. The bottlenecks can be solved via local updaing, which sends messages in small batches as part of the training process for in-network communications.

2 Optimization Methods

Solving the optimization objective in 1 is challenging due to inherent heterogeneity (both system level and statistical level) in devices and data, and the bottleneck of communication efficiency for in-network communications. The most commonly used solver for 1 is federated averaging (FedAvg) [4], which allows flexible local updating and low client participation. Later, FedProx was proposed as a generalization and re-parametrization of FedAvg to tackle system heterogeneity and statistical heterogeneity, which are failed to be addressed in FedAvg [5]. The developed global model may not adapt to individual user's needs and a personalized variant of the global model is required for each user. Per-FedAvg is proposed to address the personalization problem [6]. Another approach to address the model performance is q-FedAvg. With a novel definition of fairness in federated learning, q-FedAvg modifies the classical FedAvg and generates a model with more uniform performance over all devices. A novel algorithm FedMGDA+ further discusses the importance of fairness in federated learning and addresses the robustness concerns in q-FedAvg [7].

2.1 FedAvg

As the first widely accepted optimization solver for (1), FedAvg emphasizes non-i.i.d properties of the optimization and the communication constraints in federated learning networks [4]. FedAvg is based on stochastic gradient descent (SGD). In the training process, FedAvg performs E epochs of stochastic gradient descent (SGD) locally on k devices, where E is a constant and k is the number of participating devices (a small fraction of the total devices in the network). The devices then send back their model updates to a central server to be averaged. The algorithm is shown as Fig.1.

```
Algorithm 1 FederatedAveraging. The $K$ clients are indexed by $k$; $B$ is the local minibatch size, $E$ is the number of local epochs, and $\eta$ is the learning rate.

Server executes:
  initialize $w_0$
  for each round $t = 1, 2, \dots$ do
    $m \leftarrow \max(C \cdot K, 1)$
    $S_t \leftarrow$ (random set of $m$ clients)
    for each client $k \in S_t$ in parallel do
      $w_{t+1}^k \leftarrow$ ClientUpdate$(k, w_t)$
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$

ClientUpdate$(k, w)$:  // Run on client k
  $\mathcal{B} \leftarrow$ (split $\mathcal{P}_k$ into batches of size $B$)
  for each local epoch $i$ from 1 to $E$ do
    for batch $b \in \mathcal{B}$ do
      $w \leftarrow w - \eta \nabla \ell(w; b)$
  return $w$ to server
```

Figure 1: FedAvg Algorithm [4]

FedAvg is evaluated on both image classification and language modeling tasks with five model structures: a multi-layer perceptron, two different convolutional NNs, a two-layer character LSTM, and a large-scale word-level LSTM [4]. The results show that FedAvg can be applied in practice to train high-quality models using relatively few rounds of communication.

2.2 FedProx

While FedAvg has emphasized on heterogeneous networks, it does not fully address the challenges posed by both system heterogeneity and statistical heterogeneity. For system heterogeneity, FedAvg simply drops the devices that fail to complete E epochs within a specified time window due to resource limitations of the devices [4], which implicitly increases system heterogeneity. For statistical heterogeneity, FedAvg is proposed as a solution that takes into account the non-i.i.d data. However, FedAvg is difficult to analyze theoretically for its convergence behavior in a more realistic scenario [5].

FedProx is proposed to address the above two problems in FedAvg [5]. Instead of simply dropping the local devices that fail to complete E epochs, FedProx allows different devices with different hardware to perform variable tasks. Particularly, devices with resource limitations can perform only a fraction of the required tasks and still participate in the network. The partial participation is safely incorporated using the proximal term. This can be viewed as a re-parameterization of FedAvg because tuning the proximal term is "effectively equivalent" to tuning the number of local epochs E in FedAvg. By introducing the proximal term, FedProx encourages more local updates and safely incorporates the partial updates using the proximal term. The algorithm is shown as Fig.1.

```
Algorithm 2 FedProx (Proposed Framework)

  Input: $K, T, \mu, \gamma, w^0, N, p_k, k = 1, \cdots, N$
  for $t = 0, \cdots, T-1$ do
    Server selects 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 chosen device $k \in S_t$ finds a $w_k^{t+1}$ which is a $\gamma_k^t$-inexact minimizer of: $w_k^{t+1} \approx \arg\min_w \ h_k(w; \ w^t) = F_k(w) + \frac{\mu}{2}\|w - w^t\|^2$
    Each device $k \in S_t$ 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
```

Figure 2: FedProx Algorithm [5]

The convergence analysis is quantitatively analyzed between FedAvg and FedProx as Fig.3. FedProx provides convergence guarantees for both convex and non-convex functions. FedProx also significantly improves the convergence in heterogeneous networks compared to FedAvg. The experiements in Fig.3 contains three scenarios of straggles: 0%, 50%, and 90%, which is the fraction of devices dropped by FedAvg. As we can see, FedProx always converges faster than FedAvg in the same scenario.

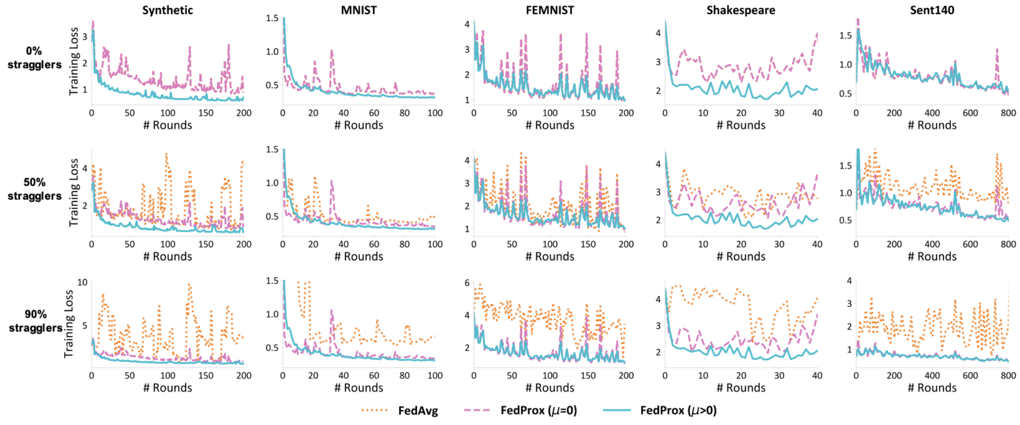

Figure 3: Convergence Analysis Between FedAvg and FedProx [5]

2.3 Per-FedAvg

FedProx improves the model convergence under a more realistic scenario with system heterogeneity and statistical heterogeneity. However, the output of federated learning only develops a global model for all the users. This developed common model cannot capture each user's need given the heterogeneity of the underlying data distribution for millions of users. Personalized FedAvg (Per-FedAvg) is proposed to address the missing feature from FedAvg and FedProx [6]. Per-FedAvg generates a personalized variant of the global model that can best fit each device by performing one or a few steps of gradient descent only on the local data [6]. This approach follows the structure of the orignal FedAvg and generates a more personalized model for each use. The algorithm is showin in Fig.4

```
Algorithm 1: The proposed Personalized FedAvg (Per-FedAvg) Algorithm
Input: Initial iterate $w_0$, fraction of active users $r$.
for $k$ : 0 to $K-1$ do
    Server chooses a subset of users $\mathcal{A}_k$ uniformly at random and with size $rn$;
    Server sends $w_k$ to all users in $\mathcal{A}_k$;
    for all $i \in \mathcal{A}_k$ do
        Set $w^i_{k+1,0} = w_k$;
        for $t$ : 1 to $\tau$ do
            Compute the stochastic gradient $\tilde{\nabla} f_i(w^i_{k+1,t-1}, \mathcal{D}^i_t)$ using dataset $\mathcal{D}^i_t$;
            Set $\tilde{w}^i_{k+1,t} = w^i_{k+1,t-1} - \alpha \tilde{\nabla} f_i(w^i_{k+1,t-1}, \mathcal{D}^i_t)$;
            Set $w^i_{k+1,t} = w^i_{k+1,t-1} - \beta(I - \alpha \tilde{\nabla}^2 f_i(w^i_{k+1,t-1}, \mathcal{D}''^i_t)) \tilde{\nabla} f_i(\tilde{w}^i_{k+1,t}, \mathcal{D}'^i_t)$;
        end for
        Agent $i$ sends $w^i_{k+1,\tau}$ back to server;
    end for
    Server updates its model by averaging over received models: $w_{k+1} = \frac{1}{rn} \sum_{i \in \mathcal{A}_k} w^i_{k+1,\tau}$;
end for
```

Figure 4: Per-FedAvg Algorithm [6]

The algorithm is inspired by the Model-Agnostic Meta-Learning (MAML) framework in [8]. MAML framework assumes a sequence of tasks, each with a limited computational budget to update the model for the current task before the arrival of the next task. Thus, the goal of MAML is not to find the model that fits best overall, but to find an initialization that can perform better within one or a few steps of gradient descent with respect to each task [8]. The optimization problem in federated learning is thus transformed as treating each user as an initialization of a task and updating its local model using one step of gradient descent with respect to its own loss function. The original optimization objective in 1 is updated to:

$$\min_{w} F(w) = \frac{1}{m} \sum_{k=1}^{m} f_k(w - \alpha \nabla f_k(w)), \tag{2}$$

where $\alpha \geq 0$ is the stepsize. $f_k(w) = \sum_{x,y \in S_i} p_i(x, y) l_k(w; x, y)$, where S_i is the dataset of device k, $p_k(x, y)$ is the probability that device k is assigned to a specific task. $l_k(w; x, y)$ is the local loss function.

This new optimization objective maintains the structure and advantages of federated learning, but also allows the differences among devices. Every device can take an initialization and update it based on its own data. One step of local gradient descent update can lead to a better model for each device. The model performance is evaluated in terms of gradient norm for non-convex loss functions [6].

The result is shown in Fig.5. Besides the baseline FedAvg, Per-FedAvg (FO) and Per-FedAvg (HF) are proposed as two efficient approximations of Per-FedAvg. Per-FedAvg (FO) replaces the gradient estimate by its first-order approximation. Per-FedAvg (HF) replaces the Hessian-vector product in the MAML update by difference of gradients using the following approximation [6]. The two approximations follow the analysis in [9].

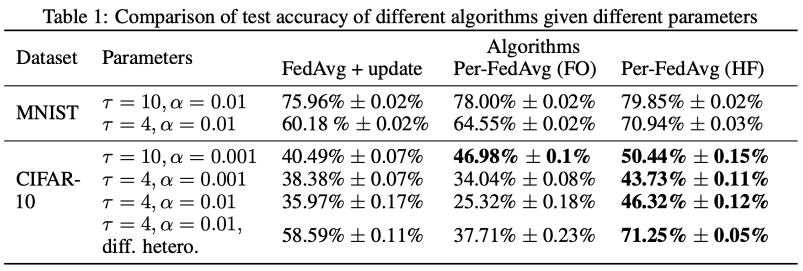

Table 1: Comparison of test accuracy of different algorithms given different parameters

Dataset	Parameters	Algorithms		
		FedAvg + update	Per-FedAvg (FO)	Per-FedAvg (HF)
MNIST	$\tau = 10, \alpha = 0.01$	75.96% $\pm$ 0.02%	78.00% $\pm$ 0.02%	79.85% $\pm$ 0.02%
	$\tau = 4, \alpha = 0.01$	60.18 % $\pm$ 0.02%	64.55% $\pm$ 0.02%	70.94% $\pm$ 0.03%
CIFAR-10	$\tau = 10, \alpha = 0.001$	40.49% $\pm$ 0.07%	**46.98%** $\pm$ **0.1%**	**50.44%** $\pm$ **0.15%**
	$\tau = 4, \alpha = 0.001$	38.38% $\pm$ 0.07%	34.04% $\pm$ 0.08%	**43.73%** $\pm$ **0.11%**
	$\tau = 4, \alpha = 0.01$	35.97% $\pm$ 0.17%	25.32% $\pm$ 0.18%	**46.32%** $\pm$ **0.12%**
	$\tau = 4, \alpha = 0.01$, diff. hetero.	58.59% $\pm$ 0.11%	37.71% $\pm$ 0.23%	**71.25%** $\pm$ **0.05%**

Figure 5: Per-FedAvg Evaluation [6]

Overall, the two approximations both perform better than the baseline FedAvg under all experiments. The effect of data heterogeneity is examined by changing the data distribution of half of the users [6]. As the last line in Fig.5 shows, Per-FedAvg (HF) performs significantly better that FedAvg under the new distributions while Per-FedAvg (FO) suffers from worse performance due to its less accurate approximation of Per-FedAvg. Thus, Per-FedAvg, the more accurate approximation of Per-FedAvg, performs better than the baseline FedAvg in all cases and leads to a more personalized result. The only concern of the the heterogeneity examination is that the experiment is limited to only one change of data distribution.

2.4 q-FedAvg

Most prior work solves the objective in (1) by sampling a subset of devices with probabilities p_k at each round, and then running SGD iteratively for E epochs on each local device. However, this approach will implicitly introduce highly variable performance among different devices because the output model can be biased towards the device with more training data. The goal of q-FedAvg is to generate a more uniform distribution of testing accuracy among all devices in a network of federated learning [10]. The idea of fairness is inspired by fair resource allocation in [11][12]. More fairness means more uniform distribution of testing accuracy. The modified objection function with respect to the defined fairness is

$$\min_{w} F_q(w) = \sum_{k=1}^{m} \frac{p_k}{q+1} F_k^{q+1}(w), \tag{3}$$

where q is the parameter that tunes the amount of fairness. The objective will be the same in (1) by setting $q = 0$. Intuitively, the larger q is, the larger weight is for devices with higher local empirical loss $F_k(w)$, and the more fairness it imposes on the generated model.

Since q is a parameter to tune, a natural approach is to train the same model with different q's and select the best one. This approach is computationally expensive and not scalable in large networks. The proposed solvers for the objective q-Fed in (6) provides dynamical adjust of the step-size to avoid manual tuning for each q.

The proposed solvers for the objective q-Fed in (3) include q-FedSGD (baseline) and q-FedAvg (communication efficient). Algorithms for both q-FedSGD and q-FedAvg are shown in Fig.6

Algorithm 1 q-FedSGD

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

Algorithm 2 q-FedAvg

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

Figure 6: q-FedSGD and q-FedAvg Algorithms [10]

The fairness is evaluated on four datasets between FedAvg and q-FedAvg as shown in Fig.7. q-FedAvg achieves smoother and more uniform distribution of testing accuracy compared to FedAvg. The overall accuracy is maintained while improving the worst performing devices.

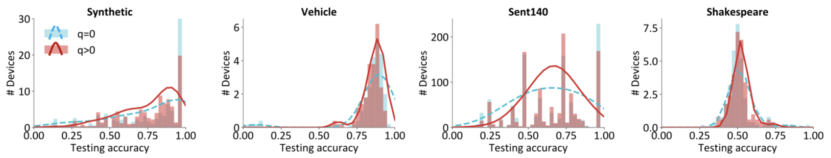

Figure 7: Fairness Evaluation of q-FedAvg [10]

The communication efficiency is evaluated on the same datasets between q-FedSGD and q-FedAvg as shown in Fig.8. q-FedAvg outperforms the baseline q-FedSGD on all datasets but the Synthetic dataset. The exception on the Synthetic dataset may be due to the fact that local updating may allow local models to move too far away from the initial global model and thus takes longer to converge.

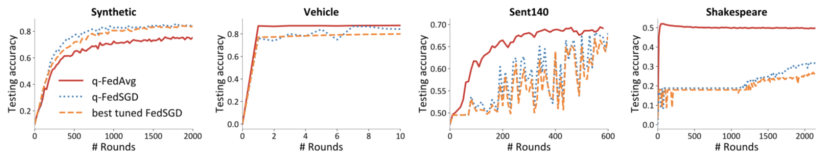

Figure 8: Communication Efficiency Evaluation of q-FedSGD and q-FedAvg [10]

In summary, q-FedAvg successfully addresses the problem of fairness in federated learning and significantly improves the worst performing devices while maintaining the overall performance. However, the slower convergence on the synthetic dataset may be a sign of its potential failure addressing the underlying data heterogeneity problem as discussed in Per-FedAvg.

2.5 FedMGDA+

On top of the idea and definition of fairness in federated learning, another novel algorithm FedMGDA+ is proposed in [7] to address fairness among all participating users. FedMGDA+ is designed as an easy to implement, more robust algorithm to encourage user participation and protect the model from multiplicative attacks. FedMGDA+ is inspired by multi-objective optimization (MoM), where multiple scalar objective functions need to be minimized simultaneously. In the federated learning setting, each local empirical risk F_k is effectively equivalent to a scalar objective funtion in MoM. The FedMGDA+ algorithm is shown in Fig.9.

```
Algorithm 1: FedMGDA+
for $t = 1, 2, \dots$ do
    choose a subset $I_t$ of $\lceil pm \rceil$ clients/users
    for $i \in I_t$ do
        $\mathbf{g}_i \leftarrow$ ClientUpdate($i, \mathbf{w}_t$)
        $\bar{\mathbf{g}}_i := \mathbf{g}_i / \|\mathbf{g}_i\|$                // normalize
    $\boldsymbol{\lambda}^* \leftarrow \mathrm{argmin}_{\boldsymbol{\lambda} \in \Delta, \|\boldsymbol{\lambda} - \boldsymbol{\lambda}_0\|_\infty \le \epsilon} \|\sum_i \lambda_i \bar{\mathbf{g}}_i\|^2$
    $\mathbf{d}_t \leftarrow \sum_i \lambda_i^* \bar{\mathbf{g}}_i$              // common direction
    choose (global) step size $\eta_t$
    $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \eta_t \mathbf{d}_t$
Function ClientUpdate($i, \mathbf{w}$):
    $\mathbf{w}^0 \leftarrow \mathbf{w}$
    repeat $k$ epochs
        // split local data into $r$ batches
        $\mathcal{D}_i \to \mathcal{D}_{i,1} \cup \cdots \cup \mathcal{D}_{i,r}$
        for $j \in \{1, \dots, r\}$ do
            $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla f_i(\mathbf{w}; \mathcal{D}_{i,j})$
    return $\mathbf{g} := \mathbf{w}^0 - \mathbf{w}$ to server
```

Figure 9: FedMGDA+ Algorithm [7]

The fairness is evaluated among the most widely applied optimization solvers including FedAvg, q-FedAvg and the novel FedMGDA+ in Fig.10. FedMGDA+ has the highest overall accuracy and lowest standard deviation of the testing accuracy, which outperforms the above q-FedAvg we discussed.

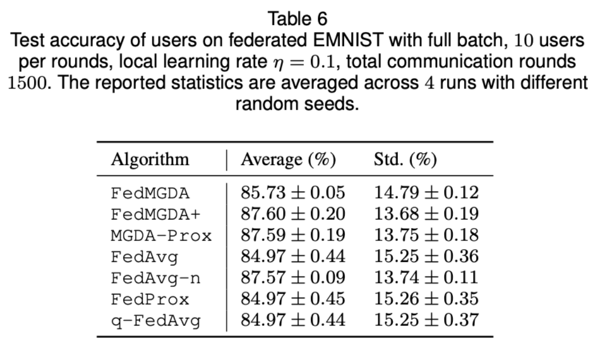

Table 6
Test accuracy of users on federated EMNIST with full batch, 10 users per rounds, local learning rate $\eta = 0.1$, total communication rounds 1500. The reported statistics are averaged across 4 runs with different random seeds.

Algorithm	Average (%)	Std. (%)
`FedMGDA`	85.73 ± 0.05	14.79 ± 0.12
`FedMGDA+`	87.60 ± 0.20	13.68 ± 0.19
`MGDA-Prox`	87.59 ± 0.19	13.75 ± 0.18
`FedAvg`	84.97 ± 0.44	15.25 ± 0.36
`FedAvg-n`	87.57 ± 0.09	13.74 ± 0.11
`FedProx`	84.97 ± 0.45	15.26 ± 0.35
`q-FedAvg`	84.97 ± 0.44	15.25 ± 0.37

Figure 10: Fairness Evaluation of FedMGDA+ [7]

Interestingly, the performance of q-FedAvg and FedAvg is the same under this evaluation, which contradicts with the result that q-FedAvg generates more fairness than FedAvg as discussed above.

3 Future Prospects

System Heterogeneity, statistical heterogeneity and communication efficiency are three major challenges for solving the optimization objective in (1) in federated learning. FedAvg is the most widely applied solver but does not fully address the heterogeneity challenges. FedProx improved the solver by re-parameterize FedAvg to tackle system heterogeneity and statistical heterogeneity. The underlying heterogeneity problem seems to be solved on the network level but may not meet individual user's needs. Per-FedAvg is then proposed to address the personalization problem. q-FedAvg tackles the same problem from a fairness perspective and aims to improve the worst performing devices. A novel algorithm FedMGDA+ further discusses fairness and addresses the robustness concerns in q-FedAvg.

As we can see, even the most novel FedMGDA+ is not capable of solving all challenges at the same time. Methods that address one challenge often fail to address another one. With the emerging needs of training models on a network of massive devices, the trade-off between the model's overall performance and the performance on each device needs to be addressed based on different scenarios. Evaluation of the methods also need to be done on a larger scale with a uniformly agreed standard.

References

[1] C. Van Berkel, "Multi-core for mobile phones," in *2009 Design, Automation & Test in Europe Conference & Exhibition*, pp. 1260–1265, IEEE, 2009.

[2] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan, *et al.*, "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[3] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," *Advances in neural information processing systems*, vol. 30, 2017.

[4] 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.

[5] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.

[6] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3557–3568, 2020.

[7] Z. Hu, K. Shaloudegi, G. Zhang, and Y. Yu, "Federated learning meets multi-objective optimization," *IEEE Transactions on Network Science and Engineering*, 2022.

[8] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*, pp. 1126–1135, PMLR, 2017.

[9] A. Fallah, A. Mokhtari, and A. Ozdaglar, "On the convergence theory of gradient-based model-agnostic meta-learning algorithms," in *International Conference on Artificial Intelligence and Statistics*, pp. 1082–1092, PMLR, 2020.

[10] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," *arXiv preprint arXiv:1905.10497*, 2019.

[11] M. J. Neely, E. Modiano, and C.-P. Li, "Fairness and optimal stochastic control for heterogeneous networks," *IEEE/ACM Transactions On Networking*, vol. 16, no. 2, pp. 396–409, 2008.

[12] E. L. Hahne, "Round-robin scheduling for max-min fairness in data networks," *IEEE Journal on Selected Areas in communications*, vol. 9, no. 7, pp. 1024–1039, 1991.