FEDERATED LEARNING WITH TASKONOMY

Hadi Jamali-Rad[1, 3*] **Mohammad Abdizadeh**[2*] **Attila Szabó**[1]
[1]Shell Global Solutions International B.V., Amsterdam, The Netherlands
{hadi.jamali-rad,attila.szabo}@shell.com
[2]Myant Inc., Toronto, ON, Canada
mohammad.abdizadeh@myant.ca
[3]Delft University of Technology (TU Delft), Delft, The Netherlands
h.jamalirad@tudelft.nl

ABSTRACT

Classical federated learning approaches incur significant performance degradation in the presence of non-IID client data. A possible direction to address this issue is forming clusters of clients with roughly IID data. We introduce federated learning with taskonomy (FLT) that generalizes this direction by learning the task-relatedness between clients for more efficient federated aggregation of heterogeneous data. In a one-off process, the server provides the clients with a pretrained encoder to compress their data into a latent representation, and transmit the signature of their data back to the server. The server then learns the task-relatedness among clients via manifold learning, and performs a generalization of federated averaging. We demonstrate that FLT not only outperforms the existing state-of-the-art baselines but also offers improved fairness across clients.[1]

1 INTRODUCTION & RELATED WORK

Federated learning is a new paradigm that offers significant potential in elevating edge-computing capabilities in modern massive distributed networks. While presenting great potential, federated learning also comes with its own unique challenges in practical settings (Konečný et al., 2015; 2016). Recent studies focus on systemic heterogeneity (Kairouz et al., 2019; Li et al., 2020a), communication efficiency (McMahan et al., 2017; Konečný et al., 2016; Sattler et al., 2019), privacy concerns (Geyer et al., 2017; Bagdasaryan et al., 2020) and more recently on fairness (Mohri et al., 2019; Li et al., 2019) and robustness across the network of clients (Sun et al., 2019; Wang et al., 2020). A defining characteristic of massive decentralized networks is stochastic heterogeneity of client data; i.e., clients possess non-independent and identically distributed (non-IID) data. (Li et al., 2020b) identifies statistical heterogeneity as the root cause for tension between fairness and robustness constraints in federated optimization. (McMahan et al., 2017; Li et al., 2018) investigate the impact of heterogeneous data distributions on the performance of federated averaging algorithm, FedAvg, and demonstrate significant performance degradation in non-IID settings. Personalized federated learning tackles data heterogeneity by forming personalized models for clients via meta-learning or multi-task learning (Smith et al., 2017; Jiang et al., 2019; Fallah et al., 2020; Li et al., 2020b). Clustered federated learning addresses this problem by iterative (or recursive) assignment of clients to separate clusters based on model or model update comparisons at the server side (Ghosh et al., 2020; Mansour et al., 2020; Sattler et al., 2020; Briggs et al., 2020; Xie et al., 2020). The effectiveness of clustering approaches hinges upon the quality of cluster formation through this iterative assignment process. Besides, clustered federated learning approaches are sensitive to initialization.

Inspired by the idea of "taskonomy" (Zamir et al., 2018), we introduce a client relatedness exploration approach based on contractive encoding of client data followed by manifold learning at the server. Our main contributions can be summarized as follows: i) we propose federated learning with taskonomy (FLT), which learns the task-relatedness among clients and uses it in the server-side aggregation for federated averaging of non-IID data, without requiring any prior knowledge about the task-relatedness among clients; ii) we empirically show that FLT offers faster convergence compared to existing state-of-the-art baselines; iii) we demonstrate that FLT reaches a higher test accuracy

[*] Equal contributions.

[1] The code is available at https://github.com/hjraad/FLT.

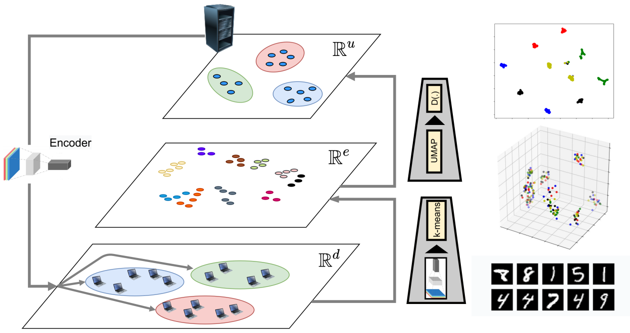

Figure 1: High-level architecture of FLT.

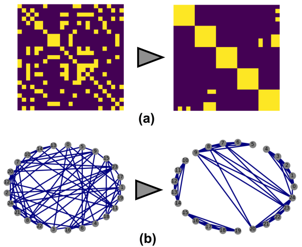

Figure 2: Forming $C = 5$ clusters for a network of $M = 25$ clients. a) $\tilde{A}$ in the form of an adjacency matrix (re-ordered on the right), b) corresponding client relatedness graph (re-ordered on the right).

compared to the state-of-the-art baselines across two federated learning settings (2% and 40%, on the commonly used FEMNIST (Caldas et al., 2018) and a newly-introduced "Structured Non-IID FEMNIST" datasets); iv) finally, we show that FLT offers improved fairness (measured in terms of variance of test accuracies across clients), besides the improved accuracy.

2 Federated Learning with Taskonomy

Majority of the clustered federated learning approaches (Ghosh et al., 2020; Mansour et al., 2020; Sattler et al., 2020; Briggs et al., 2020; Duan et al., 2020) enforce a hard membership constraint on the clients to form disjoint clusters where every client can belong to only one cluster. In contrast, we allow for an arbitrary symmetric task-relatedness matrix. To form clusters, these approaches mostly compare the clients based on their model parameters using a distance metric (such as L_2 norm, or cosine similarity), which in practice does not capture the underlying manifold of data in the model parameter space or any other representation space. We instead propose a one-shot method for learning the task-relatedness matrix (coined as FLT) that benefits from manifold approximation (metric learning with UMAP (McInnes et al., 2018)) at the server side before applying a distance metric. A high-level sketch of the proposed approach is depicted in Fig. 1. As can be seen, we consider three abstraction levels: i) data level, where data samples live in $\mathbb{R}^d$; ii) encoder level, where a contractive latent space representation of client data is extracted in an unsupervised fashion (samples are nonlinearly projected to $\mathbb{R}^e$); iii) manifold approximation level with UMAP, where samples live in $\mathbb{R}^u$. The encoder is provided by the server to the clients. This allows them to apply one-shot contractive encoding on their local data, followed by k-means on the outcome and return the results to the server. At server side, UMAP is applied to approximate the arriving clients embeddings. This is followed by applying a distance threshold to determine client dependencies and form an adjacency matrix or a client task-relatedness graph (see Fig. 2).

Learning client task-relatedness. The proposed approach, $\texttt{FCR}(\cdot)$, is described in Algorithm 1. The server broadcasts an encoder $G(\cdot)$ to all the M clients in $\mathcal{S}$. This broadcast is one-off and this downlink communication will not be repeated. We considered a convolutional autoencoder (ConvAE) with its frozen encoder section employed for extracting latent embeddings. ConvAE not only helps compressing the information that has to be sent to the server, but also creates a less noisy representation of the client data. Upon receiving $G(\cdot)$ clients compute $\mathcal{E}_m := G(\mathcal{D}_m)$, where $\mathcal{D}_m$ denotes the dataset of client m ($\in [M]$) and $\mathcal{E}_m$ denotes its embedding set of size $|\mathcal{D}_m|$. The elements of $\mathcal{E}_m$ live in $\mathbb{R}^e$ with e referring to the latent embedding dimension. Even though $\mathcal{E}_m$ is compressed as compared to $\mathcal{D}_m$, it turns out that it can still be further distilled and yet capture enough information for our downstream federated learning purpose. Therefore, each client applies $\texttt{kMEANS}(\cdot)$ on $\mathcal{E}_m$ and sends the outcome $\mathcal{M}_m := \{\mu_1, \cdots, \mu_k\}$ (a set of size k) back to the server. The *fine-tune* mode is provisioned to accommodate encoders pretrained on a totally different dataset. In such a case, clients will be asked to run F epochs of SGD from their latest state on their most recent dataset. The server constructs $\mathcal{M} := \{\mathcal{M}_1, \cdots, \mathcal{M}_M\}$ and applies UMAP (McInnes et al., 2018) to $\mathcal{M}$ and constructs $\mathcal{Z} := \{\mathcal{Z}_1, \cdots, \mathcal{Z}_M\}$ with $\mathcal{Z}_m := \{u_{m,1}, \cdots, u_{m,k}\}$. $\mathcal{Z}$ contains $k \times M$ elements each living in $\mathbb{R}^u$, with u being typically 2 or 3. In most prior work, a distance metric (L_2 or cosine) is directly applied, which could be a limiting factor for non-convex risk functions and incongruent non-IID settings (Sattler et al., 2020). After UMAP, the server applies a distance metric to construct an adjacency matrix $A_{i,j} := \min \|\mathcal{Z}_i - \mathcal{Z}_j\|_2$ where the minimum pairwise distance among the elements of $\mathcal{Z}_i$

and $\mathcal{Z}_j$ are taken into account. We refer to the (i, j)-th element of matrix X as $X_{i,j}$ and its i-th row with X_i. Here, for the sake of simplicity, we consider a *hard-thresholding* operator Γ applied on A leading to $\tilde{A}$, where $\tilde{A}_{i,j} = \Gamma(A_{i,j}) = \texttt{Sign}(A_{i,j} - \gamma)$ with γ a threshold value. Two different demonstrations of $\tilde{A}$ for a toy setup with $M = 25$ clients is depicted in Fig 2: a) adjacency matrix, b) client relatedness graph. On the left, the clients are ordered based on their ID's and on the right they are re-ordered according to their adjacency weights resulting in the formation of $C = 5$ clusters.

Algorithm 1: Form Client Relatedness (FCR)

```
Require: MODE, $\mathcal{S}$, $G(\cdot)$, $\Gamma(\cdot)$, $k$
Server broadcasts $G(\cdot)$ to all clients in $\mathcal{S}$
for each client $m$ in $\mathcal{S}$ do
    if MODE = fine-tune then
        Client runs $F$ epochs to fine-tune;
    end
    Client computes its own embedding:
      $\mathcal{E}_m \leftarrow G(\mathcal{D}_m)$;
    Client applies $k$-means clustering to $\mathcal{E}_m$'s:
    $\mathcal{M}_m := \{\mu_1, \cdots, \mu_k\} \leftarrow \texttt{kMEANS}(\mathcal{E}_m)$;
    Client sends $\mathcal{M}_m$ to the server.
end
Server updates $\mathcal{M} := \{\mathcal{M}_1, \cdots, \mathcal{M}_M\}$
Server (re)computes
    $\mathcal{Z} := \{\mathcal{Z}_1, \cdots, \mathcal{Z}_M\} \leftarrow \texttt{UMAP}(\mathcal{M})$
Server constructs the adjacency matrix:
$A_{i,j} := \min_{r,s} \|u_{i,r} - u_{j,s}\|_2$,
                    $\forall i, j \in [M]$ & $\forall r, s \in [k]$ .
Return: $\tilde{A} := \Gamma(A)$
```

Algorithm 2: FLT

```
Require: $\mathcal{S}$, $M$, $T$, $W^0$, $p_m$
Initialize Clustering:
    $\tilde{A} \leftarrow \texttt{FCR}(\text{normal}, \mathcal{S}, G(\cdot), \Gamma(\cdot), k)$
for $t = 0, \cdots, T - 1$ do
    $w_1^t, \cdots, w_m^t \leftarrow W^t$
    Server selects a subset $\mathcal{S}^t$ of clients;
    Server sends $w^t$ to all clients in $\mathcal{S}^t$;
    for each client $m$ in $\mathcal{S}^t$ do
        for epoch $e = 1, \cdots, E$ do
            $w_m \leftarrow w_m - \eta \nabla F_m(w_m)$
        end
        $w_m^{t+1} \leftarrow w_m$
    end
    Each client sends $w_m^{t+1}$ to the server.
    Server updates $W^t = [w_1^t, \cdots, w_M^t]$;
    Server updates the model weights:
      $W^{t+1} \leftarrow W^t \tilde{A}\, \texttt{diag}(p_m / \|\tilde{A}_m\|_0)$
end
```

Federated averaging with taskonomy. FLT in Algorithm 2 starts with an initialization stage by calling the *normal* mode of $\texttt{FCR}(\cdot)$. Note that this initial round with $\texttt{FCR}(\cdot)$ can happen in a few stages and excluding a few clients does not impact FLT. Next, the typical T rounds of communication akin to FedAvg will be run, where F_m is the empirical risk over local data $F_m = \frac{1}{n_m} \sum_{j=1}^{n_m} l_j(w)$, with n_m denoting the sample size of client m. The server will construct and optimize a set of local models in $W^t = [w_1^t, \cdots, w_M^t]$. For notation simplicity, each model parameter set w_m^t is assumed to be reshaped into a vector. Following that, the sever updates the local models weights using $W^{t+1} = W^t.\, \tilde{A}.\, \texttt{diag}(p_m / \|\tilde{A}_m\|_0)$. The adjacency matrix $\tilde{A}$ is re-ordered using hierarchical clustering (Müllner, 2011) into a soft clustering matrix and determines which client models are associated and should be aggregated and updated together. Notably, when distinct cluster formations are discovered, all the clients in one cluster (blocks in Fig. 2) will only have a single model. In that case, $W^t = [w_1^t, \cdots, w_C^t]$, with C denoting the total number of extracted clusters with be updated.

3 Evaluation

We opt for image classification as our downstream application of federated learning. For performance evaluation, we employ two different datasets: i) Federated EMNIST (FEMNIST) in LEAF (Caldas et al., 2018) which is made out of Extended MNIST (EMNIST) (Cohen et al., 2017), ii) a newly designed structured non-IID FEMNIST. FEMNIST is a standard dataset with $805,263$ samples that can accommodate up to 3550 clients. Based upon EMNIST and similar to FEMNIST, we build a new dataset with more extreme structured non-IID conditions and call it *Structured Non-IID FEMNIST*. To this aim, we consider the "balanced" dataset of EMNIST, containing $131,600$ samples on 47 classes. We use $112,800$ for training (2400 samples per class) and the remainder $18,800$ for testing. The encoder provided to clients is pretrained on a totally different dataset, CIFAR100 (Krizhevsky et al., 2009), and thus, an initial fine-tuning per client would be required. This is to demonstrate that lacking a holistic encoder is not a bottleneck for FLT.

Scenario-1 [FEMNIST]: We import the standard FEMNIST dataset and construct a network of 200 clients according to train and test data distributions defined in (Caldas et al., 2018). There are no predefined clusters in FEMNIST and it is up to the federated learning method to form clusters.

Table 1: Test accuracies (%) for Scenario-1.

	MLP		CNN	
Method	acc.	var.	acc.	var.
FedAvg	72.76±0.76	202.61	81.64±0.66	147.03
IFCA	61.24±0.84	176.38	81.47±0.66	118.71
FedSEM	72.45±0.76	185.96	79.99±0.68	156.27
FLT	74.11±0.74	171.31	82.14±0.65	145.03

Table 2: Test accuracies (%) for Scenario-2.

Method	acc.	var.
FedAvg	46.50±0.36	100.27
FedSEM	43.53±0.36	406.60
FLT	86.51±0.24	207.60

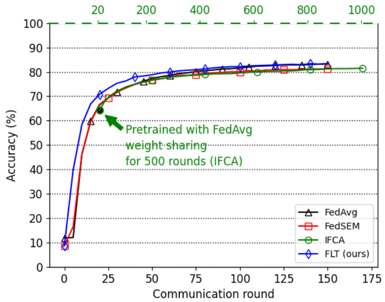

Figure 3: Average accuracy of Scenario 1 for CNN.

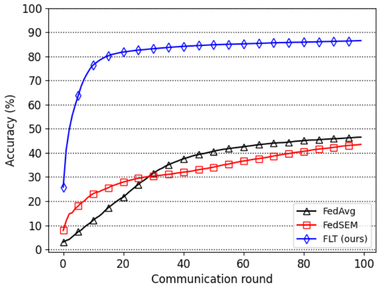

Figure 4: Acc. Scenario 2, $C = 10$, $M = 2400$.

Scenario-2 [Structured Non-IID FEMNIST]: We employ our newly introduced "Structured Non-IID FEMNIST". As mentioned earlier, this is built with the purpose of introducing more extreme and structured non-IIDness in FEMNIST. To this aim, we impose label distribution skew (across clusters) as well as quantity skew following a power law for the number of samples per client in each cluster, akin to (Li et al., 2018). We consider $C = 10$ clusters, each containing 5 distinct character classes (total of $12,000$ data samples per cluster), except the last one containing 2 classes ($4,800$ samples). We also consider a larger network with $M = 2400$ clients and 240 clients per cluster.

Network parameters. For the experiments on Structured Non-IID FEMNIST, we use a multi-layer perceptron (MLP) with ReLU activation, and a single hidden layer of size 200. For experiments on FEMNIST, we use both the MLP mentioned earlier, as well as the standard CNN proposed in (Caldas et al., 2018). See supplementary material for more details. We set the number of local epochs to $E = 5$, and the total communication rounds to $T = 100$, unless otherwise mentioned. The local training is a mini-batch SGD with batch size of 10 and learning rate $\eta = 0.01$. For FLT, the size of the latent embedding is $e = 128$ and k in kMEANS is set to 2. Even though on the client side k can be adjusted according to the number of client classes. The number of fine-tuning epochs is set to $F = 5$, and $\gamma = 1$. The client participation fraction is set to 20%.

Baselines, competitors, and fairness. We consider FedAvg (McMahan et al., 2017) as baseline where a *single global model* is trained for the whole network; We also compare our performance with two of the most recent state-of-the-art clustered federated learning approaches called IFCA (Ghosh et al., 2020) and FedSEM (Xie et al., 2020). Note that FedSEM already outperforms other recent baselines such as FedProx (Li et al., 2018) and CFL (Sattler et al., 2019). Among several interesting approaches to fairness in federated learning, following (Li et al., 2019), we report the variance of model accuracies across clients as a measure of fairness.

Evaluation results for Scenarios-1. The results in (Ghosh et al., 2020), IFCA, rely on an initialization with FedAvg for "weight sharing". Here, all the MLP experiments are run for $T = 1000$ communication rounds, and those for CNN are run for only $T = 100$ rounds, except for IFCA which is run for 1500 rounds for both MLP and CNN (including 500 rounds for initialization). The *test* accuracies and corresponding variances are summarized in Table 1. We marginally outperform FedSEM and FedAvg (by about 2%) for both MLP and CNN. We also significantly outperform IFCA in the MLP setting and marginally outperform in the case of CNN. However, it takes IFCA 1500 communication rounds to reach the performance regime the other methods converged to in about 100 rounds. For this reason, we omit IFCA in our next experiments. The convergence graphs of average *test* accuracies is illustrated in Fig 3 where FLT is the fastest in terms of convergence. We argue that FEMNIST may not have a clear cluster structure and thus a cluster-based methods might not offer a significant gain. This is the main motivation behind designing Scenario-2.

Table 3: Communication complexity analysis.

FLT	FedSEM (Xie et al., 2020)	IFCA (Ghosh et al., 2020)
$\underbrace{M * W_{\text{enc}} + k\,M\,e}_{\text{one-off}} + 2\rho M W_{\text{local}} T$	$2\rho M W_{\text{local}} T$	$\rho M W_{\text{local}} T(C + 1)$

Evaluation results for Scenarios-2. Convergence graphs of average *test* accuracies are shown in Fig. 4. The *test* accuracies (and their standard error) together with the fairness measure (variance across clients) for at the last communication round $T = 100$ are summarized in Table 2. Interestingly, FedSEM suffers in this scenario and roughly performs as good as FedAvg. This is due to tremendous heterogeneity in model space and thus considerable increase in complexity of pairwise model comparisons. This is exacerbated because of limited number of samples per clients resulting in lower-quality models training. As a result, FedSEM falls almost back to FedAvg in performance.

4 Concluding Remarks

Summary and extensions. We proposed FLT that comes with the following notable advantages. First, it is one-shot and considerably faster in convergence compared to its competitors, especially in structured non-IID scenarios. Second, in contrast to most existing baselines, it does not require prior knowledge about number of clusters to form them. Third, it performs slightly better than the state-of-the-art baselines in standard federated learning settings and significantly outperforms them in structured non-IID scenarios. Fourth, FLT offers improved fairness (least performance disparity among clients) compared to the existing baselines in most presented scenarios. Finally, in our extended work[2], we provide more detailed experimentation and analyses also a convergence proof for FLT under common assumptions required for the convergence of FedAvg. Another interesting extension that we cover therein is when the number of clients grows to tens of thousands (in very large-scale networks). In that case, the extended version of FLT has the flexibility to decompose the client relatedness graph with hierarchical clustering (Müllner, 2011) into disjoint clusters and degenerate to the same computation complexity level its competitors inflict.

Complexity and practical considerations. FLT introduces a *one-off* overhead due to the client relatedness discovery process (FCR, Algorithm 1). However, owing to FCR, it is faster than the existing iterative baselines and less prone to convergence issues. One can argue that this step can be susceptible to security issues during the uplink communication (akin to standard FedAvg communications). A possible solution to address this is adding encryption and client ID verification processes, which are outside the scope of our work. From communication complexity perspective, this overhead requires the server to send an encoder model (W_{enc}) to the clients, and the clients to send an array of size ke (with k in k-means and e denoting the latent embedding dimension of the encoder) to the server. A rough estimate of the communication complexity of the proposed FLT, and two discussed state-of-the-art competitors (FedSEM and IFCA) is summarized Table 3. As can be seen, the communication complexity of FLT and FedSEM are essentially the same except for the first two one-off terms (without T for total communication rounds) and could be neglected. Note that this is an initialization step and it can also happen in multiple steps. Excluding a few clients from this process, due to for instance their unavailability, does not impact the performance of FCR and in turn FLT. On the other hand, IFCA mandates roughly $(C + 1)/2$ times (C being the number of clusters) more communication complexity. This is because in every communication round, C virtual center models will have to be sent to all the participating clients. From compute complexity perspective, possible *fine-tuning* of the encoder is only for a small number of epochs ($F = 5$ in our experiments) on an encoder which is as simple as the local client models; and this is yet another one-off process that can be neglected over long runs.

Acknowledgments

The authors would like to thank Shell Global Solutions International B.V. and Delft University of Technology (TU Delft) for their support and for the permission to publish this work. The authors also thank Ahmad Beirami from Facebook AI for helpful discussions.

[2] https://arxiv.org/abs/2103.15947

References

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 2938–2948. PMLR, 2020.

Christopher Briggs, Zhong Fan, and Peter Andras. Federated learning with hierarchical clustering of local updates to improve training on non-iid data. *arXiv preprint arXiv:2004.11791*, 2020.

Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Moming Duan, Duo Liu, Xinyuan Ji, Renping Liu, Liang Liang, Xianzhang Chen, and Yujuan Tan. Fedgroup: Ternary cosine similarity-based clustered federated learning framework toward high accuracy in heterogeneity data. *arXiv preprint arXiv:2010.06870*, 2020.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *arXiv preprint arXiv:2006.04088*, 2020.

Yihan Jiang, Jakub Konečný, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Jakub Konečný, Brendan McMahan, and Daniel Ramage. Federated optimization: Distributed optimization beyond the datacenter. *arXiv preprint arXiv:1511.03575*, 2015.

Jakub Konečný, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Citeseer*, 2009.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. *arXiv preprint arXiv:1905.10497*, 2019.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Federated multi-task learning for competing constraints. *arXiv preprint arXiv:2012.04221*, 2020a.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020b.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR, 2017.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. *arXiv preprint arXiv:1902.00146*, 2019.

Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 2019.

Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. Clustered federated learning: model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in neural information processing systems*, 30:4424–4434, 2017.

Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Ming Xie, Guodong Long, Tao Shen, Tianyi Zhou, Xianzhi Wang, and Jing Jiang. Multi-center federated learning. *arXiv preprint arXiv:2005.01026*, 2020.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3712–3722, 2018.