ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240773

FilFL: Client Filtering for Optimized Client Participation in Federated Learning

Fares Fourati[a,1], **Salma Kharrat**[a,1],
Vaneet Aggarwal[a,b], **Mohamed-Slim Alouini**[a] **and Marco Canini**[a]

[a]KAUST
[b]Purdue University

Abstract. Federated learning, an emerging machine learning paradigm, enables clients to collaboratively train a model without exchanging local data. Clients participating in the training process significantly impact the convergence rate, learning efficiency, and model generalization. We propose a novel approach, client filtering, to improve model generalization and optimize client participation and training. The proposed method periodically filters available clients to identify a subset that maximizes a combinatorial objective function with an efficient greedy filtering algorithm. Thus, the clients are assessed as a combination rather than individually. We theoretically analyze the convergence of federated learning with *client filtering* in heterogeneous settings and evaluate its performance across diverse vision and language tasks, including realistic scenarios with time-varying client availability. Our empirical results demonstrate several benefits of our approach, including improved learning efficiency, faster convergence, and up to 10% higher test accuracy than training without client filtering.

1 Introduction

Federated learning (FL) is an emerging machine learning paradigm that enables collaborative training across multiple clients while preserving their local data privacy [27, 39, 28, 26, 31]. The most commonly used approach in this setting, federated averaging (FedAvg) [35], alternates between local training and server aggregation and broadcasts the latest version of the global model. However, FL faces various challenges,[2] such as training with many clients and data heterogeneity, where the clients' data are non-IID, i.e., different clients have different data distributions [4, 20, 22, 15, 41].

Recent works have analyzed the effect of data heterogeneity on the convergence of local-update stochastic gradient descent (SGD) [38, 16, 24, 40, 42, 25, 23, 45, 37, 34, 31, 1]. Such heterogeneity leads to unstable and slow convergence [31], resulting in suboptimal or even detrimental model performance [46]. This occurs because the data distributions on the clients may differ significantly from the global distribution, causing clients to converge towards their local optima rather than the global optimum, refer to Appendix B [13] for more details. Furthermore, given communication constraints, training with all clients may not be possible; previous works have considered client selection schemes that select a subset $\mathcal{A}_t$ of K clients from a total of N clients to participate at each training round t. Although client selection methods address communication constraints and make the training more practical, they also increase the challenge of managing heterogeneity. Refer to Appendix B [13] for an extended related work.

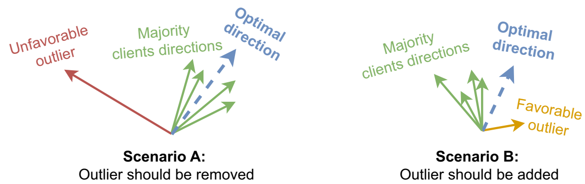

Figure 1: Visualization of two scenarios with different suggested descent directions from different clients. Arrows are color-coded to indicate the quality of direction: blue for optimal, orange for favorable outlier, green for majority consensus, and red for unfavorable outlier.

To address the aforementioned FL challenges, various client selection schemes have been proposed in earlier studies. Some aim to provide unbiased estimations of the gradients that would result from full participation, such as sampling based on the number of local data points [32] or sampling uniformly at random with weighted updates (RS) [31]. While these approaches approximate full participation, they are not explicitly designed to accelerate the training process. Other schemes select subsets of clients that carry representative gradient information for full participation by encouraging diverse gradient selections (DivFL) [3]. However, promoting diversity may also include unfavorable outlier gradients. Additional strategies explicitly aimed at accelerating training include selecting clients with higher update norms more frequently [7] or employing a power-of-choice (PoC) method that biases selection towards clients with higher local losses [9]. However, these approaches consider clients separately rather than as part of a collaborative unit, i.e., they make decisions based on individual performances without considering their collaborative performance at the current stage of the training process.

Assessing clients based on their collaborative performance is essential to optimize client participation beyond mere element-wise selection. Considering gradients from collaboratively-unfavorable clients or excluding collaboratively-favorable ones can lead to degraded collaborative performance. To illustrate this, consider the simplified example depicted in Fig. 1, where we illustrate two possible client combination scenarios. In these scenarios, the blue arrow represents

---

[1] **Equal contribution.** Corresponding authors.
Emails: salma.kharrat@kaust.edu.sa and fares.fourati@kaust.edu.sa.

[2] Although privacy is not the primary concern of this work, it remains a significant challenge in FL. However, conventional techniques like differential privacy and secure multiparty computation could be used in conjunction with our proposed method.

an oracle for the optimal descent direction. In scenario A (Fig. 1), the red arrow, having a significantly different direction and larger norm than most other directions, might be selected by methods that prioritize directions with larger norms or that encourage diversity. However, excluding the red arrow and keeping the green arrows can lead to a better approximation of the optimal descent direction. In scenario B (Fig.1), while the orange direction differs from the majority of directions (green) and may be overlooked by methods that rely on similarity metrics between gradients or due to its small norm, its inclusion—based on its contribution to the subset of green directions—leads to a better approximation of the optimal direction. These examples highlight the importance of assessing collaboration when choosing clients.

In this work, we include combinatorial optimization in the standard FL training to optimize client participation further. We introduce FilFL, which includes a *client filtering* procedure that looks for the best combination of clients within the available ones, which can be conducted as a periodic prepossessing step to any off-the-shelf client selection scheme. To achieve this, we formulate a combinatorial optimization problem to periodically identify the clients most compatible for collaboration. Namely, our objective is to identify the optimal subset of available clients whose averaged performance yields the lowest loss. Solving this combinatorial optimization problem would necessitate an exponential number of tests, rendering it computationally infeasible. As a result, we employ an efficient greedy approach to approximate its solution. To this end, we present two greedy filtering algorithms: a deterministic one and a randomized variant, both relying on marginal gains from adding and removing clients from subsets of clients. Using different vision and language tasks and realistic federated scenarios with time-varying client availability, we evaluate the performance of combining our *client filtering* methods with different FL algorithms, such as FedAvg and FedProx [31], and with various client selection schemes, such as RS, PoC, and DivFL.

Contributions. We propose FilFL, a novel approach that includes combinatorial optimization through *client filtering* in FL to optimize client participation, accelerate the training process, and improve the overall global model performance. To the best of our knowledge, we are the first to define a non-monotone combinatorial optimization problem in the context of FL, aiming to identify the subset of clients from the available clients whose averaged performance yields the lowest loss. We propose a greedy filtering algorithm (χGF) with deterministic (DGF) and randomized (RGF) versions to approximate its solution. We provide a theoretical analysis showing that FilFL achieves a convergence rate of $\mathcal{O}(\frac{1}{t}) + \mathcal{O}(\varphi)$ for t time steps, where φ represents a time constant, under certain assumptions. Empirical evaluations on various vision and language tasks under realistic scenarios of time-varying available clients show that FilFL outperforms FL methods, achieving faster training and up to a 10 percentage point increase in test accuracy. Furthermore, ablation studies and filtering performance analysis have been conducted.

A companion report of this paper with complete technical details is available at [13]. The code can be accessed at https://github.com/salmakh1/FilFL.

2 Problem Formulation

Unlike standard FL training algorithms, where all the available clients are considered for selection and participation, we formulate a bi-level optimization problem that combines the standard continuous training objective with a discrete filtering objective.

2.1 Training Objective

We consider the canonical objective of fitting a global model to the non-IID data $\mathcal{D}$ held across clients [35]. Thus, we consider the following distributed optimization problem:

$$\min_{\mathbf{w}} \left\{ F(\mathbf{w}) \triangleq \sum_{k=1}^{N} p_k F_k(\mathbf{w}) \right\}, \tag{1}$$

where N is the number of clients, and p_k is the weight of the k-th client such that $p_k \geq 0$ and $\sum_{k=1}^{N} p_k = 1$. Suppose the k-th client holds the m_k training data: $x_{k,1}, x_{k,2}, \cdots, x_{k,m_k}$. The local objective $F_k(\cdot)$ is defined as: $F_k(\mathbf{w}) \triangleq \frac{1}{m_k}\sum_{j=1}^{m_k} \ell(\mathbf{w}; x_{k,j})$ where $\ell(\cdot;\cdot)$ is some training loss function. While the training objective seeks the best client weights, the filtering objective finds the best combination of clients to optimize these weights. Although the former is continuous and the latter is discrete, both are interconnected and combined, which have led to remarkable improvements.

2.2 Filtering Objective

Our filtering objective is to find a subset of clients $\mathcal{S}^f$ that approximates a solution to the following combinatorial optimization problem:

$$\max_{\mathcal{S}\in\mathcal{S}_t} \quad -F\left(\frac{\mathbf{1}}{|\mathcal{S}|}\sum_{\mathbf{k}\in\mathcal{S}} \mathbf{w_t^k}\right), \tag{2}$$

such that $\mathbf{w}_t^k$ is the weight of the k^{th} client in round t. Thus, the combinatorial problem aims at finding a subset $\mathcal{S}^f \in \mathcal{S}_t$ where the average of the weights of the clients in the subset $\mathcal{S}^f$ minimizes the weighted average of the local losses, i.e., maximizes the function $-F$. Following the literature on combinatorial optimization, we define the problem as a maximization problem.

Unfortunately, solving the problem defined in Eq. (2) is both communication and computationally expensive. Even the evaluation of one possible set of clients $\mathcal{S}$ requires all clients to evaluate the combination of that set, i.e., each client k needs to compute, $F_k(\frac{\mathbf{1}}{|\mathcal{S}|}\sum_{\mathbf{k}\in\mathcal{S}} \mathbf{w_t^k})$ on their local datasets. Finding or even approximating a solution requires several evaluations, which introduces additional communication and computational overhead on the participating clients.

To make this approach more practical, we propose reformulating the problem into a centrally solvable form, thereby minimizing communication overhead. Therefore, we suggest using a central filtering dataset, denoted by $\mathcal{V}$, without requiring the clients to share any datasets. This can be done in several ways, by leveraging a subset of the server's validation data for filtering, using samples from a public dataset,[3] or randomly choosing a client to perform filtering on a subset of their validation dataset, in each filtering round. We later show that these approaches, solving on a server dataset or a variable filtering dataset, depending on the chosen client (see Section 5.3.3 for details about the stochastic dataset), are possible and show that the filtering dataset can be stochastic, and does not need to adhere to any prohibitive requirements, for example, can be as small as 8 samples, as discussed in detail in Section 5.3.

Unless mentioned otherwise, in the following, we consider a server-held filtering dataset $\mathcal{V}$ with m samples: $x_1, x_2, \cdots, x_m$. Thus, our filtering objective can be defined as follows:

[3] Previous works in FL have used public datasets for various purposes [21, 44, 33, 8, 31].

Algorithm 1 FilFL

Require: T, E, η, $\mathbf{w}_1$, K, $\mathcal{S}_0$, h, n, χ
1: **Initialize** $\mathcal{S}^f \leftarrow \mathcal{S}_0$
2: **for** t = $1, \cdots, T$ **do**
3: **if** ($t \mod h == 0$) OR ($\mathcal{S}_t \neq \mathcal{S}_{t-1}$) **then**
4: Server broadcasts $\mathbf{w}_t$ to all clients in $\mathcal{S}_t$
5: **for** client $k \in \mathcal{S}_t$ **in parallel do**
6: Update $\mathbf{w}^k$ for E local SGD steps
7: Send $\mathbf{w}^k$ back to the server
8: **end for**
9: $\mathcal{S}^f$, $\mathcal{A}_t$ = *client filtering*(Shuffle($\mathcal{S}_t$), n, χ)
10: **else**
11: Server selects $\mathcal{A}_t$ including at most K clients from $\mathcal{S}^f$
12: Server broadcasts $\mathbf{w}_t$ to all clients in $\mathcal{A}_t$
13: **for** client $k \in \mathcal{A}_t$ **in parallel do**
14: Update $\mathbf{w}^k$ for E local SGD steps
15: Send $\mathbf{w}^k$ back to the server
16: **end for**
17: **end if**
18: Server aggregates:
19: $\mathbf{w}_{t+1} \leftarrow \frac{1}{|\mathcal{A}_t|}\sum_{k \in \mathcal{A}_t} \mathbf{w}^k$
20: **end for**

$$\max_{\mathcal{S} \in \mathcal{S}_t} \left\{ \mathcal{R}(\mathcal{S}) \triangleq -F_{\mathcal{V}}\left(\frac{\mathbf{1}}{|\mathcal{S}|}\sum_{\mathbf{k} \in \mathcal{S}} \mathbf{w}^{\mathbf{k}}_{\mathbf{t}}\right)\right\}, \tag{3}$$

where $F_{\mathcal{V}}(\mathbf{w}) \triangleq \frac{1}{m}\sum_{j=1}^{m} \ell(\mathbf{w}; x_j)$ as the loss on dataset $\mathcal{V}$.

While the reformulation proposed in Eq. (3) of the objective in Eq. (2) offers improved tractability, saving communication and computation when evaluated centrally, achieving an exact solution remains non-trivial. Finding an exact solution to the problem in Eq. (3) would typically still necessitate an exponential number of queries, rendering it computationally infeasible. Furthermore, notice that the function in Eq. (3) is not necessarily monotone[4]. Suppose we have a set of clients A and a new client c. If the new client c has a high loss, adding c to the set of clients A may increase the overall loss, thereby decreasing the objective value $\mathcal{R}(A \cup \{c\})$ compared to $\mathcal{R}(A)$, thus violating monotonicity of the function. Thus, we seek to devise a non-monotone approximation algorithm to solve this problem efficiently.

3 Client Filtering

We introduce our approach, FilFL, which incorporates *client filtering* into standard FL algorithms such as FedAvg and FedProx, alongside with different client selection algorithms, such as RS, PoC, and DivFL. FilFL filters the available clients, considering only the filtered-in clients $\mathcal{S}^f$ as potential participants in the training process. This ensures that the chosen client selection method is only applied to the chosen subset $\mathcal{S}^f$, rather than the entire pool of available clients $\mathcal{S}_t$. To implement *client filtering*, we define a combinatorial objective function on the discrete and large space of client combinations in Eq. (3) and introduce a periodic greedy algorithm denoted as χGF, which approximates a solution for this objective, optimizing client combinations for better client participation in FL.

3.1 Client Filtering in FL (FilFL)

FilFL is a FL approach that incorporates *client filtering*. Algorithm 1 presents its pseudocode. FilFL applies *client filtering* (line 4) whenever the current set of available clients differs from the previous round. Furthermore, to improve computational efficiency, FilFL applies *client filtering* periodically every h rounds. We empirically observe similar results when running χGF every round or running it every few rounds; a sensitivity analysis to h is given in Section 5.3.1. The *client filtering* procedure (cf. Algorithm 2) determines $\mathcal{S}^f$ by approximating a solution for the problem defined in Eq. (3). To determine the set of active clients $\mathcal{A}_t$, FilFL uses any client selection method to select K clients from $\mathcal{S}^f$ (line 6). In case $\mathcal{S}^f$ only contains K or fewer clients, FilFL uses $\mathcal{S}^f$ as $\mathcal{A}_t$ (line 6). FilFL then runs local steps of SGD for each active client in $\mathcal{A}_t$ (lines 8-11). Finally, the server aggregates the weights returned from the active clients and moves to the next round.

Remark 1. *FilFL generalizes standard FL. FilFL adds an extra layer in FL, which is client filtering. Using an identity filtering algorithm that accepts all the available clients, i.e., $\mathcal{S}^f = \mathcal{S}_t$, FilFL reduces to standard FL training schemes. Thus, FilFL can be considered as a generalization of those. In this paper, we propose χGF for filtering. However, future work might consider other filtering methods.*

Remark 2. *Client filtering and client selection are distinct yet complementary methods with key differences. First, client filtering does not produce a subset with a fixed cardinality, K; therefore, client selection is subsequently applied to the filtered-in group. Second, client filtering can be implemented periodically, whereas client selection occurs in every communication round. Finally, we opted to separate the two for the sake of generality, allowing the flexibility to combine any filtering algorithm with any off-the-shelf selection method.*

Remark 3. *FilFL reduces the complexity of client selection schemes. Firstly, FilFL skips client selection whenever $|\mathcal{S}^f| \leq K$ (line 6). Furthermore, client filtering often leads to the rejection of multiple clients. As a result, when FilFL applies client selection on the filtered-in set $\mathcal{S}^f$ instead of the full set of available clients $\mathcal{S}_t$, the search space for client selection becomes smaller. For instance, the DivFL selection method complexity is $\mathcal{O}(N\mathcal{G}(N)K)$, where N represents the number of all the clients, K is the cardinality constraint, and $\mathcal{G}(N)$ represents the cost of calling their oracle function, which is a linearly increasing function of N. Consequently, the complexity of DivFL is $\mathcal{O}(N^2K)$. However, by incorporating χGF with DivFL, the selection complexity is reduced to $\mathcal{O}(|\mathcal{S}^f|^2K)$, with $|\mathcal{S}^f|$ the number of filtered-in clients typically being smaller than n, smaller than N.*

3.2 Greedy Filtering (χGF)

Motivated by the successful application of greedy algorithms in combinatorial optimization [11, 5, 12, 14], we introduce a greedy *client filtering* algorithm, called χGF. While monotone approximation algorithms, greedily adds elements based on their adding marginal gains [14], non-monotone algorithms considers both the marginal gain of adding and the marginal gain of removing the same entity [11, 5, 12]. Adapting the non-monotone algorithm in [12], which has been demonstrated to be robust to small errors in function evaluations, as shown in Corollary 2 in [12], we propose two versions for filtering: randomized (RGF) and deterministic (DGF). Algorithm 2 lists their pseudocode. The algorithm iterates over each available client and decides whether to add it to the set of clients X (initially empty) or remove it from the set of clients Y (initially containing all available clients). The server determines X and Y in a greedy fashion using measures of marginal gains of adding and removing until a decision is made for all individual clients. The algorithm returns the chosen (filtered-in) set of clients. Specifically, let X_i and Y_i be two sets of clients. Initially, $X_0 = \emptyset$ and $Y_0 = \mathcal{S}_t$. The algorithm has at most n

[4] A function f is monotone, if any set A is a subset of B ($A \subseteq B$), then $f(A) \leq f(B)$ [14].

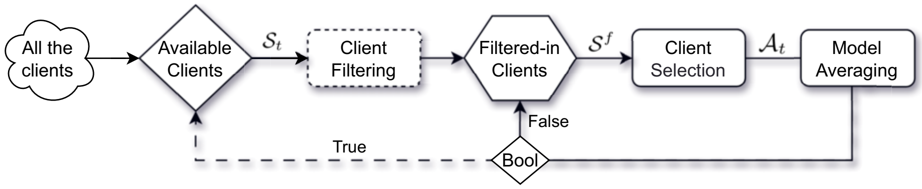

Figure 2: FilFL incorporates *client filtering* in FL, which is activated when the boolean condition 'Bool' becomes true, either when new clients become available or when h rounds have elapsed since the last filtering call. Otherwise, the condition remains false. In both scenarios, clients are selected from the filtered-in subset of clients, denoted as $\mathcal{S}^f$.

Algorithm 2 χGF ($\chi \in \{D, R\}$)

Require: $\mathcal{S}_t, n, \chi$
Initialize $X_0 \leftarrow \emptyset$, $Y_0 \leftarrow \mathcal{S}_t$
for index $i \in \{1, ..., n\}$ **do**
  $u_i \leftarrow$ client of index i in $\mathcal{S}_t$
  $a_i \leftarrow \mathcal{R}(X_{i-1} \cup \{u_i\}) - \mathcal{R}(X_{i-1})$
  $b_i \leftarrow \mathcal{R}(Y_{i-1} \setminus \{u_i\}) - \mathcal{R}(Y_{i-1})$
  $a_i' \leftarrow \max(a_i, 0)$ and $b_i' \leftarrow \max(b_i, 0)$
  if $\chi = D$ **then**
    $p_i = \mathbf{1}\{a_i > b_i\}$
  else if $\chi = R$ **then**
    $p_i = \frac{a_i'}{a_i'+b_i'}$ ($p_i = 1$ **if** $a_i' = b_i' = 0$)
  end if
  with probability p_i **do**
    $X_i \leftarrow X_{i-1} \cup \{u_i\}$ and $Y_i \leftarrow Y_{i-1}$
  else
    $Y_i \leftarrow Y_{i-1} \setminus \{u_i\}$ and $X_i \leftarrow X_{i-1}$
end for
Select Z including at most K clients from X_n
Return X_n, Z

steps, where n is the maximum number of considerable clients. In step i, χGF computes two variables: a_i and b_i, defined as follows:

$$\begin{aligned} a_i &\triangleq \mathcal{R}(X_{i-1} \cup \{u_i\}) - \mathcal{R}(X_{i-1}), \\ b_i &\triangleq \mathcal{R}(Y_{i-1} \setminus \{u_i\}) - \mathcal{R}(Y_{i-1}). \end{aligned} \tag{4}$$

These two variables are important for the decision-making process. a_i measures the marginal gain of adding client u_i to X_{i-1}, while b_i measures the marginal gain of removing client u_i from Y_{i-1}, which can be positive due to non-monotonicity. While DGF decides by comparing both marginal gains via $p_i = \mathbf{1}\{a_i > b_i\}$, RGF decides based on $p_i = \frac{a_i'}{a_i'+b_i'}$, where $a_i' = \max(a_i, 0)$ and $b_i' = \max(b_i, 0)$. In the special case when $a_i' = b_i' = 0$, we set $p = 1$ for RGF. With probability p, the client u_i is added to the set X_{i-1} and kept in Y_{i-1}; otherwise, the client is removed from Y_{i-1} and X_{i-1} is kept the same. Therefore, $X_i \subseteq Y_i$ for all $i = 1, \ldots, n$. After checking all n clients, it can be easily seen that by the algorithm's construction, both sets X_n and Y_n contain the same clients, i.e., $X_n \equiv Y_n$. Hereafter, at round t, we refer to the final set X_n as the filtered-in set $\mathcal{S}^f$.

Remark 4. *In cases where both a_i and b_i are non-positive, i.e., $a_i' = b_i' = 0$, the RGF algorithm accepts the client with a probability of 1. On the other hand, even when both a_i and b_i are non-positive, the DGF algorithm may reject this client with a probability of 1 if $a_i < b_i$. Hence, by design, DGF can reject more clients than RGF. This observation is empirically validated in Fig. 10. Generally, the clients that are accepted by RGF and rejected by DGF have minimal impact on FilFL performance, as they are the ones with both negative marginal gains of adding them to X_{i-1} or removing them from Y_{i-1}.*

Remark 5. *The computational complexity of using χGF is $\mathcal{O}(n\mathcal{I}(m))$, where n is the number of considerable available clients, fixed by the user, and $\mathcal{I}(m)$ is the cost of inference over the server dataset of size m data points. Therefore, the computational cost of using the χGF algorithm does not scale with the scaling number of clients and increases only linearly with the number of considered available clients n (for reference, DivFL's computational cost scales quadratically with the total number of clients N). Therefore, our method remains practical even as the number of clients increase. Furthermore, the cost of forward passes can be reduced by distributing the computation across multiple graphical processing units, leading to faster and more efficient computations.*

4 FilFL Convergence Analysis

We now provide a theoretical analysis of the convergence properties of our proposed FilFL algorithm (see Algorithm 1). Specifically, we analyze the convergence of the average model weights $\bar{\mathbf{w}}_t$ at round t to the optimal solution $\mathbf{w}^*$, under practical assumptions of non-IID data, partial client participation, and local updates. Our analysis focuses on the impact of incorporating *client filtering* into the FedAvg setting, assuming random sampling as the client selection method. While our results mainly apply to FedAvg with random sampling, they can be easily extended to other methods. In the following, we provide the necessary definitions and assumptions for our analysis and present the theorem statement for convergence. The proofs of the main lemmas are provided in Appendix D [13].

4.1 Assumptions and Definitions

The following assumptions are standard assumptions for the convergence analysis in the literature of FL, such as [3, 32].

Assumption 1. *$F_1, \cdots, F_N$ are all L-smooth*[5].

Assumption 2. *$F_1, \cdots, F_N$ are all μ-strongly convex*[6].

Assumption 3. *Let ψ_t^k be sampled from the k-th client's local data uniformly at random. The variance of stochastic gradients in each client is bounded by σ_k^2, i.e., $\mathbb{E}\left[\left\|\nabla F_k\left(\mathbf{w}_t^k, \psi_t^k\right) - \nabla F_k\left(\mathbf{w}_t^k\right)\right\|^2\right] \le \sigma_k^2$ for all $k = 1, \cdots, N$.*

Assumption 4. *The norms of the stochastic gradients are uniformly bounded by G, i.e., $\left\|\nabla F_k\left(\mathbf{w}_t^k, \psi_t^k\right)\right\|^2 \le G^2$ for all $k = 1, \cdots, N$ and $t = 1, \cdots, T-1$.*

Assumption 5. *Statistical heterogeneity: $F^* - \sum_{k\in[N]} p_k F_k^*$ is bounded, where $F^* := \min_{\mathbf{w}} F(\mathbf{w})$ and $F_k^* := \min_{\mathbf{v}} F_k(\mathbf{v})$.*

Assumption 6. *Assume $\mathcal{A}_t$ contains a subset of K indices randomly selected with replacement according to the sampling probabilities $p_i' = 1/|\mathcal{S}^f|$, with simple averaging for aggregation* [7].

[5] For all k,$\mathbf{v}$ and $\mathbf{w}$, $F_k(\mathbf{v}) \le F_k(\mathbf{w}) + (\mathbf{v}-\mathbf{w})^T \nabla F_k(\mathbf{w}) + \frac{L}{2}\|\mathbf{v}-\mathbf{w}\|_2^2$.
[6] For all k, $\mathbf{v}$ and $\mathbf{w}$, $F_k(\mathbf{v}) \ge F_k(\mathbf{w}) + (\mathbf{v}-\mathbf{w})^T \nabla F_k(\mathbf{w}) + \frac{\mu}{2}\|\mathbf{v}-\mathbf{w}\|_2^2$.
[7] A theoretical analysis of this sampling scheme was provided in [32].

Limited to realistic scenarios (for communication efficiency and low straggler effect), FilFL samples a subset $\mathcal{A}_t$ from the filtred-in set $\mathcal{S}^f$ and then only performs updates on them. This makes the analysis intricate since $\mathcal{A}_t$ varies each E steps. However, we can use an approach similar to the one used in [32] to circumvent this difficulty. We assume that FilFL activates all clients at the beginning of each round and then uses the parameters maintained in only a few sampled clients to produce the next-round parameter. It is clear that this updating scheme is equivalent to the original.

Let $\mathbf{w}_t^k$ be the model parameter maintained in the k-th client at the t-th step. Let $\mathcal{I}_E$ be the set of global synchronization steps, i.e., $\mathcal{I}_E = \{iE \mid i = 1, 2, \cdots\}$. If $t+1 \in \mathcal{I}_E$, i.e., the time step to communication, FilFL activates all clients. Then, the update of our algorithm can be described as: for all $k \in [N]$,

$$\mathbf{v}_{t+1}^k = \mathbf{w}_t^k - \eta_t \nabla F_k\left(\mathbf{w}_t^k, \psi_t^k\right),$$

$$\mathbf{w}_{t+1}^k = \begin{cases} \mathbf{v}_{t+1}^k & \text{if } t+1 \notin \mathcal{I}_E, \\ \text{sample } \mathcal{A}_{t+1} \text{ from } \mathcal{S}_{t+1}^f & \\ \text{and average } \left\{\mathbf{v}_{t+1}^k\right\}_{k \in \mathcal{A}_{t+1}} & \text{if } t+1 \in \mathcal{I}_E. \end{cases}$$

Let $\mathbf{w}^* \in \arg\min_{\mathbf{w}} F(\mathbf{w})$ and $\mathbf{v}_k^* \in \arg\min_{\mathbf{v}} F_k(\mathbf{v})$ for $k \in [N]$. Let $\bar{\mathbf{v}}_t \triangleq \sum_{k\in[N]} p_k \mathbf{v}_t^k$, and $\bar{\mathbf{w}}_t \triangleq \sum_{k\in[N]} p_k \mathbf{w}_t^k$, where $p_k \geq 0$ is the given weight of the k^{th} client and w.l.o.g., we assume $\sum_k p_k = 1$.

Filtering the clients before selection, using biased greedy filtering algorithms, made the theoretical analysis more challenging. Compared to previous theoretical federated convergence analysis, such as [32] and [3], that introduce $\bar{\mathbf{v}}_t$ and $\bar{\mathbf{w}}_t$, to proceed with our analysis we introduce an extra variable $\bar{\mathbf{z}}_t$, defined as follows $\bar{\mathbf{z}}_t \triangleq \frac{1}{|\mathcal{S}_t^f|} \sum_{k \in \mathcal{S}_t^f} \mathbf{v}_t^k$. Furthermore, we define a filtering gap as follows:

$$\delta_t \triangleq F(\bar{\mathbf{v}}_t) - F(\bar{\mathbf{z}}_t). \tag{5}$$

An optimal filtering method leads to the highest δ_t possible at every round t. In FilFL, using χGF as a filtering method, we expect the δ_t to be optimized over the rounds. In Lemma 1, in Appendix D, we show that $\mathbb{E}[\delta_t]$ is lower bounded by a constant δ.

4.2 FilFL Theoretical Convergence Results

We present our convergence result as follows.

Theorem 1. *Let assumptions 1, 2, 3, 4, 5, and 6 hold, then we have*

$$\mathbb{E}[\|\overline{\mathbf{w}}_{t+1} - \mathbf{w}^*\|^2] \leq \mathcal{O}(\frac{1}{t}) + \mathcal{O}(\varphi) \tag{6}$$

for some time constant φ that depends on the filtering.

Proof. Note that

$$\mathbb{E}\left[\|\overline{\mathbf{w}}_{t+1} - \mathbf{w}^*\|^2\right] = \mathbb{E}\left[\|\overline{\mathbf{w}}_{t+1} - \overline{\mathbf{v}}_{t+1}\|^2\right] + \mathbb{E}\left[\|\overline{\mathbf{v}}_{t+1} - \mathbf{w}^*\|^2\right] + 2\mathbb{E}\left[\langle \overline{\mathbf{w}}_{t+1} - \overline{\mathbf{v}}_{t+1}, \overline{\mathbf{v}}_{t+1} - \mathbf{w}^* \rangle\right]. \tag{7}$$

We bound the three terms in Eq. (7). Using Lemma 4 result, shown in Appendix D, we have $\mathcal{T}_1 \triangleq \mathbb{E}\left[\|\overline{\mathbf{w}}_{t+1} - \overline{\mathbf{v}}_{t+1}\|^2\right] \leq \xi$, for some constant ζ and $\xi = \zeta - \frac{2\delta}{\mu}$. Moreover, using Lemma 1, 2, and 3 in [32], define $\mathcal{T}_2 \triangleq \mathbb{E}\left[\|\overline{\mathbf{v}}_{t+1} - \mathbf{w}^*\|^2\right]$, we have $\mathcal{T}_2 \leq (1 - \eta_t \mu)\, \mathbb{E}\left[\|\bar{\mathbf{w}}_t - \mathbf{w}^*\|^2\right] + \eta_t^2 B$, for a stepsize η_t and some constant B. Furthermore, using Corollary 1, in Appendix D, we have $\mathcal{T}_3 \triangleq \mathbb{E}\left[\langle \overline{\mathbf{w}}_t - \overline{\mathbf{v}}_t, \overline{\mathbf{v}}_t - \mathbf{w}^* \rangle\right] \leq \rho\sqrt{\xi}$, for some constant ρ.

Define $\Delta_t \triangleq \mathbb{E}\left[\|\overline{\mathbf{w}}_t - \mathbf{w}^*\|^2\right]$, and $\varphi = \xi + 2\rho\sqrt{\xi}$, thus $\Delta_{t+1} \leq (1 - \eta_t \mu)\, \Delta_t + \eta_t^2 B + \varphi$. With a stepsize, $\eta_t = \frac{\beta}{t}$, for $\beta \geq \frac{1}{\mu}$, the final convergence result follows from Lemma 3 in [36]. ☐

The above result provides a convergence rate guarantee of $\mathcal{O}(\frac{1}{t})$ for FilFL up to a certain neighborhood of size $\mathcal{O}(\varphi)$, which depends on the *client filtering*. While our approach differs from that of DivFL, we obtain similar theoretical guarantees (albeit with different constants) and better empirical results. Furthermore, our experiments show that FilFL enhances different FL algorithms; see Experiments Section, which includes FedAvg and FedProx. It is worth noting that a good filtering algorithm implies larger values of δ_t for all t, as defined in Eq. (5). This, in turn, leads to a larger value of δ, thus smaller ξ, hence a smaller value of φ. Our greedy filtering algorithms are designed to maximize δ_t, thereby minimizing φ. Empirical results demonstrate that both χGF accelerate the training and lead to better test accuracy. As discussed in the Experiments section, both versions of χGF enjoy significantly large approximation ratios of the optimal solution OPT, specifically, $\mathcal{R}(\mathcal{S}^f) \geq 0.96\mathcal{R}(OPT)$, indicating that greedy filtering identifies near optimal combinations of clients over the rounds.

5 Experiments

As we are the first to propose *client filtering* in FL, we evaluate the performance of combining χGF with different FL algorithms, such as FedAvg [35] and FedProx [31] with different client selection schemes, namely, random selection (RS) [32], power-of-choice (PoC) [9], and diverse selection (DivFL) [3]. Moreover, we conduct ablation studies, analyzing the sensitivity of FilFL to different filtering periodicity values and for various filtering dataset scenarios, including different sizes and distributions, and we examine the behavior of χGF.

5.1 Setup

We experiment with different vision and language tasks in a range of scenarios. We use Shakespeare dataset [6], built from "The Complete Works of William Shakespeare," where each speaking role in every play is considered a different client. The task is a next-character prediction with 80 classes of characters in total. We use a small filtering dataset from a different distribution, specifically consisting of parts of this paper's introduction, as shown in Table 3 in the Appendix [13]. We use a two-layer LSTM [19] classifier containing 256 hidden units with an 8-dimension embedding layer. Moreover, we use CIFAR-10 [29] in a non-IID setting with ResNet18 [18]. We split CIFAR-10 train dataset into private and filtering datasets, where the filtering partition fraction is 0.01. Similar to existing works [2, 17, 43], to simulate the non-IID data distribution among clients, we use the Dirichlet distribution Dir(α), with $\alpha = 0.5$. We use the existing CIFAR-10 test sets as global test sets. Furthermore, we use Federated Extended MNIST (FEMNIST) [6], which is built by partitioning the data in Extended MNIST [10, 30] based on the writer of the digit/character. We use the test set as a global test set. Similar to [6], we use a model with two convolutional layers followed by pooling and ReLU and a final dense layer with 2048 units.

In the following experiments, we consider N clients, with only n considerable available ones, with K selected clients, periodicity h, and filtering data size m. FilFL samples $\mathcal{A}_t$ from the filtered-in set of clients $\mathcal{S}^f$, while other FL algorithms sample $\mathcal{A}_t$ from the full set of available clients $\mathcal{S}_t$. We experiment with three different seeds and present the averaged results together with the standard deviation. Appendix C reports further details about the setup.

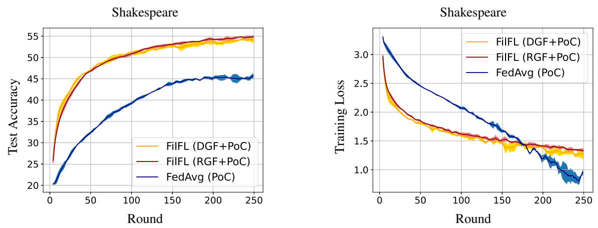

Figure 3: FilFL (FedAvg with χGF) vs FedAvg (w/o filtering) both with PoC on Shakespeare dataset with $N = 143$, $n = 100$, $K = 10$, $m = 34$, and $h = 5$.

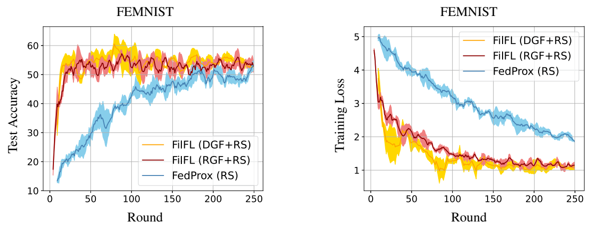

Figure 4: FilFL (FedProx with χGF) vs FedProx (w/o filtering) both with RS on FEMNIST dataset with $N = 190$, $n = 50$, $K = 5$, $m = 2000$, and $h = 5$.

5.2 *FilFL Outperforms Standard FL Algorithms*

FilFL, for any given FL algorithm and any applied client selection algorithm, includes an extra layer of *client filtering* using χGF. In the following sections, we demonstrate the advantages of adding this extra layer to various combinations of FL algorithms and client selection methods. For the same FL algorithm and client selection, we assess the marginal gain of adding such a filtering step.

5.2.1 *FilFL (FedAvg with χGF and PoC) vs FedAvg (PoC)*

We compare the performance of FilFL (FedAvg with χGF) against FedAvg, both using PoC for client selection on different datasets. Fig. 3 illustrates the results of the Shakespeare dataset, with a small filtering dataset from a different distribution; specifically consisting of parts of this paper's introduction (see the filtering dataset in Appendix C.3). Our results demonstrate that FilFL using DGF or RGF performs significantly better than FedAvg. In particular, as depicted in the left plot, FilFL with both filtering methods accomplishes accelerated training and attains around 10 percentage points higher test accuracy than FedAvg. Furthermore, we conducted the t-test, and the resulting two-tailed p-value was 0.0001, considered extremely statistically significant. After 200 rounds, the right plot displays a lower training loss for FedAvg. In Appendix E.1, we present the results on CIFAR-10 and FEMNIST, which exhibit improved training and better test accuracy by 5 and 7 percentage points, respectively.

5.2.2 *FilFL (FedProx with χGF and RS) vs FedProx (RS)*

We compare the performance of FilFL (FedProx with χGF) against FedProx, both using RS for selection. Fig. 4 demonstrates that FilFL using χGF achieves significantly superior performance compared to FedProx on the FEMNIST dataset. Specifically, the left plot illustrates that FilFL with DGF and RGF achieves approximately 7 and 4 percentage points higher test accuracy, respectively than FedProx. The right plot reveals lower training loss for FilFL than FedProx. Moreover, Fig.5, shows the results on the Shakespeare dataset, where FilFL with DGF and RGF attains around 3 and 6 percentage points higher test accuracy, respectively than FedProx.

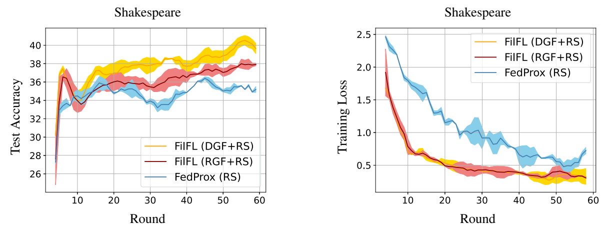

Figure 5: FilFL (FedProx + χGF + RS) vs FedProx (RS) without filtering on Shakespeare dataset.

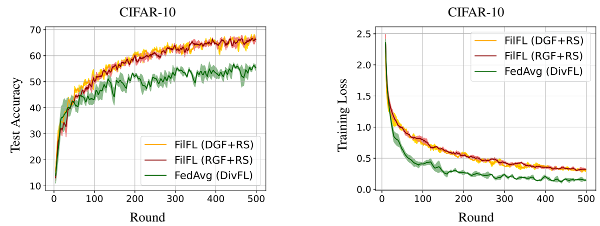

Figure 6: FilFL (FedAvg with χGF with RS) vs FedAvg (DivFL w/o filtering) on CIFAR-10 dataset with $N = 200$, $n = 30$, $K = 3$, $m = 500$, and $h = 5$.

5.2.3 *FilFL (FedAvg with χGF and RS) vs FedAvg (DivFL).*

As shown in [3], FedAvg with DivFL performs better than FedAvg with RS or PoC. However, it remains computationally more expensive than both selection methods. To investigate whether a simple client selection method like RS combined with χGF can outperform a sophisticated selection method like DivFL, we compare FilFL using RS against FedAvg (DivFL). Fig.6 shows that on the CIFAR-10 dataset, χGF achieves 10 percentage points higher accuracy than FedAvg (DivFL) (left plot). While FedAvg (DivFL) exhibits lower training loss than FilFL (right plot), it suffers from significantly larger test loss (see the Appendix), which can be due to the overfitting of FedAvg (DivFL) and the better generalization capabilities of FilFL. Moreover, our results indicate that although FilFL with RS and FedAvg (DivFL) have similar convergence theoretical results, our approach empirically outperforms FedAvg (DivFL). The two-tailed p-value from the t-test is 0.0018, considered as very statistically significant. In the Appendix, we show that FilFL with DivFL surpasses FedAvg (DivFL).

5.3 *Ablation Studies*

We conduct an ablation study of the proposed approach, testing the filtering approach with various periodicity, using filtering datasets of different sizes and distributions, and using variable filtering datasets.

5.3.1 *Sensitivity to Filtering Periodicity*

The set of available clients may remain the same over several rounds; however, their model weights change due to local training and weight aggregation. This means that *client filtering* in each round may not necessarily exclude the same clients. The optimal set of clients changes significantly as the model weights change over rounds. However, *client filtering* may filter in similar sets of clients for a few rounds when the weights do not change much. That is why we suggest running *client filtering* periodically and applying client selection on the filtered-in set for a few rounds to exploit the set it has already found. We experiment with different periodicities $h \in \{1, 3, 5, 10, 20\}$, as shown in Fig.7, and find that FilFL's performance is similar for these values of h. However, from a computational perspective, our approach is more efficient for larger periodicity h.

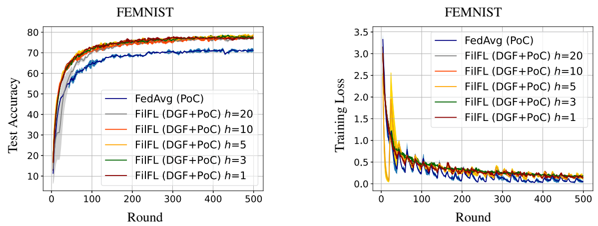

Figure 7: FilFL (FedAvg + χGF + PoC) sensitivity to periodicity h on FEMNIST dataset.

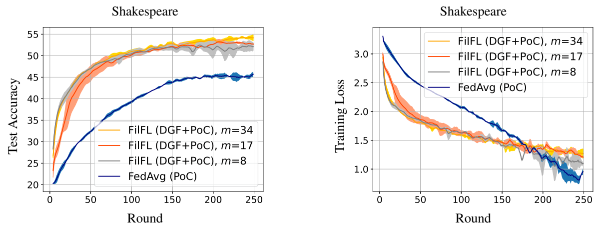

Figure 8: FilFL (FedAvg with DGF) sensitivity to filtering dataset size m on Shakespeare dataset with PoC for client selection, $N = 143$, $n = 100$, $K = 10$, and $h = 5$.

5.3.2 *Sensitivity to Filtering Dataset Size & Distribution*

We evaluate the effectiveness of FilFL under different filtering datasets scenarios, showing its robustness across various sizes and distributions. In the Shakespeare experiment, we use small datasets consisting of parts of this paper's introduction, containing only 34, 17, and 8 samples. Fig. 8, shows that FilFL remains effective even with tiny filtering datasets with different distributions than the clients' datasets. The left plot shows higher test accuracy for FilFL than FedAvg, with a slight advantage for larger values of m. The middle and right plots also reveal lower training loss for smaller m and lower test loss for larger m, indicating that larger m leads to better generalization. Similar results concerning the effect of dataset size on the FEMNIST dataset are presented in Appendix, with datasets of 2000, 1000, and 500 samples. Hence, FilFL shows insensitivity to the number of data points, performing well even with smaller datasets and under distribution shifts, thereby proving its versatility and robustness.

5.3.3 *Sensitivity to Variable Filtering Datasets*

We evaluate the use of a variable dataset for client filtering. Instead of solving the filtering objective on a central dataset, possibly on a subset of the server validation dataset or one single client throughout the training, we consider the case of randomly selecting a client from the available clients to perform the *client filtering* task. The chosen client performs *client filtering* on its own validation dataset. Therefore, the filtering dataset becomes variable depending on the chosen client in that round. Our results demonstrate that FilFL, using RGF, even in such a stochastic scenario, achieves significantly better performance than FedAvg. In particular, as depicted in Fig. 9, FilFL accomplishes accelerated training and attains approximately 10 percentage points higher test accuracy than FedAvg.

5.4 *χGF Behavior*

We examine the filtering rates and approximation ratios of the RGF and DGF algorithms compared to brute force search results.

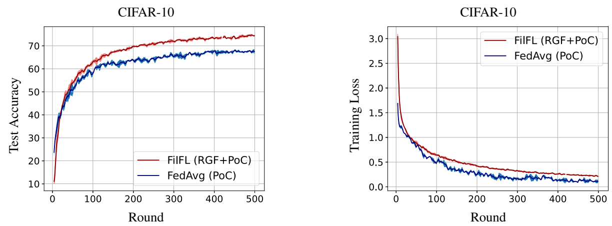

Figure 9: FilFL (FedAvg + RGF + PoC) vs FedAvg (PoC) without filtering on CIFAR-10 dataset.

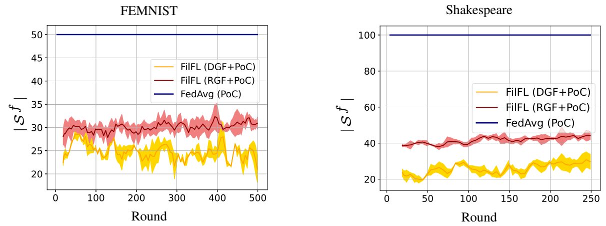

Figure 10: The number of filtered-in clients, denoted as $|\mathcal{S}^f|$, for FilFL (FedAvg with χGF), over the rounds in different settings of CIFAR-10, FEMNIST, and Shakespeare datasets, with considerable available clients n being 30, 50, and 100, respectively. For FedAvg without filtering, we consider $\mathcal{S}^f$ to be equal to $\mathcal{S}_t$.

5.4.1 *Filtering Rates*

χGF rejects multiple clients, with the average rejection rate varying depending on the task and the version (randomized or deterministic). As mentioned in Remark 4, DGF rejects more clients than RGF, roughly half the number of clients (cf. Fig. 10). Therefore, DGF is more efficient in reducing the complexity of client selection by significantly reducing the sampling space.

5.4.2 *Approximation Ratios*

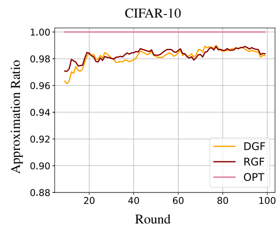

Figure 11: Approximation ratios of the filtering objective solution.

Fig. 11 shows the approximation ratios of both χGF versions compared to the optimal filtering (OPT) on CIFAR-10 with $N = 200$ and $n = 10$, which we find by evaluating $2^n - 1$ combinations. We find that both χGF versions achieve approximation ratios higher than 0.96, i.e., $\mathcal{R}(\mathcal{S}^f) \geq 0.96\mathcal{R}(OPT)$ over the multiple rounds. This indicates that greedy filtering identifies near-optimal combinations of clients. Finally, the filtering performance can be measured by the improved FL performance and the higher approximation ratios. Since both versions of χGF show similarly high ratios and improved FL performance, both can be considered effective for filtering.

6 Conclusion

We proposed *client filtering* as a promising technique to optimize client participation and training in FL. Our proposed FL algorithm, FilFL, which incorporates the greedy filtering algorithm χGF, has proven theoretical convergence guarantees and empirically shows better learning efficiency, accelerated convergence, and higher test accuracy across different vision and language tasks.

References

[1] A. M. Abdelmoniem, C.-Y. Ho, P. Papageorgiou, and M. Canini. Empirical Analysis of Federated Learning in Heterogeneous Environments. In *EuroMLSys*, 2022.

[2] D. A. E. Acar, Y. Zhao, R. M. Navarro, M. Mattina, P. N. Whatmough, and V. Saligrama. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*, 2021.

[3] R. Balakrishnan, T. Li, T. Zhou, N. Himayat, V. Smith, and J. Bilmes. Diverse client selection for federated learning via submodular maximization. In *International Conference on Learning Representations*, 2021.

[4] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečn´ y, S. Mazzocchi, B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander. Towards Federated Learning at Scale: System Design. In *MLSys*, 2019.

[5] N. Buchbinder, M. Feldman, J. Seffi, and R. Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM Journal on Computing*, 44(5):1384–1402, 2015.

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

[7] W. Chen, S. Horvath, and P. Richtarik. Optimal client sampling for federated learning. *arXiv preprint arXiv:2010.13723*, 2020.

[8] S. Cheng, J. Wu, Y. Xiao, and Y. Liu. Fedgems: Federated learning of larger server models via selective knowledge fusion. *arXiv preprint arXiv:2110.11027*, 2021.

[9] Y. J. Cho, J. Wang, and G. Joshi. Client selection in federated learning: Convergence analysis and power-of-choice selection strategies. *arXiv preprint arXiv:2010.01243*, 2020.

[10] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

[11] U. Feige, V. S. Mirrokni, and J. Vondrák. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

[12] F. Fourati, V. Aggarwal, C. Quinn, and M.-S. Alouini. Randomized greedy learning for non-monotone stochastic submodular maximization under full-bandit feedback. In *International Conference on Artificial Intelligence and Statistics*, pages 7455–7471. PMLR, 2023.

[13] F. Fourati, S. Kharrat, V. Aggarwal, M.-S. Alouini, and M. Canini. Filfl: Client filtering for optimized client participation in federated learning. *arXiv preprint arXiv:2302.06599*, 2023.

[14] F. Fourati, C. J. Quinn, M.-S. Alouini, and V. Aggarwal. Combinatorial stochastic-greedy bandit. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 12052–12060, 2024.

[15] B. Ganguly, S. Hosseinalipour, K. T. Kim, C. G. Brinton, V. Aggarwal, D. J. Love, and M. Chiang. Multi-edge server-assisted dynamic federated learning with an optimized floating aggregation point. *IEEE/ACM Transactions on Networking*, 2023.

[16] F. Haddadpour and M. Mahdavi. On the convergence of local descent methods in federated learning. *arXiv preprint arXiv:1910.14425*, 2019.

[17] C. He, S. Li, J. So, X. Zeng, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu, et al. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[20] S. Hosseinalipour, C. G. Brinton, V. Aggarwal, H. Dai, and M. Chiang. From federated to fog learning: Distributed machine learning over heterogeneous wireless networks. *IEEE Communications Magazine*, 58 (12):41–47, 2020. doi: 10.1109/MCOM.001.2000410.

[21] W. Huang, M. Ye, and B. Du. Learn from others and be yourself in heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10143–10153, 2022.

[22] D. Huba, J. Nguyen, K. Malik, R. Zhu, M. Rabbat, A. Yousefpour, C.-J. Wu, H. Zhan, P. Ustinov, H. Srinivas, K. Wang, A. Shoumikhin, J. Min, and M. Malek. PAPAYA: Practical, Private, and Scalable Federated Learning. In *MLSys*, 2022.

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

[24] A. Khaled, K. Mishchenko, and P. Richtárik. Tighter theory for local sgd on identical and heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, pages 4519–4529. PMLR, 2020.

[25] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. Stich. A unified theory of decentralized sgd with changing topology and local updates. In *International Conference on Machine Learning*, pages 5381–5393. PMLR, 2020.

[26] J. Konečnỳ. Stochastic, distributed and federated optimization for machine learning. *arXiv preprint arXiv:1707.01155*, 2017.

[27] J. Konečnỳ, B. McMahan, and D. Ramage. Federated optimization: Distributed optimization beyond the datacenter. *arXiv preprint arXiv:1511.03575*, 2015.

[28] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[29] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. *Canadian Institute for Advanced Research, 2009. URL http://www.cs.toronto.edu/ kriz/cifar.html*, 2009.

[30] Y. LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[31] 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*, 2:429–450, 2020.

[32] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.

[33] T. Lin, L. Kong, S. U. Stich, and M. Jaggi. Ensemble distillation for robust model fusion in federated learning. *Advances in Neural Information Processing Systems*, 33:2351–2363, 2020.

[34] G. Malinovskiy, D. Kovalev, E. Gasanov, L. Condat, and P. Richtarik. From local sgd to local fixed-point methods for federated learning. In *International Conference on Machine Learning*, pages 6692–6701. PMLR, 2020.

[35] 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*, pages 1273–1282. PMLR, 2017.

[36] B. Mirzasoleiman, J. Bilmes, and J. Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pages 6950–6960. PMLR, 2020.

[37] R. Pathak and M. J. Wainwright. Fedsplit: An algorithmic framework for fast federated optimization. *Advances in Neural Information Processing Systems*, 33:7057–7066, 2020.

[38] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

[39] R. Shokri and V. Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1310–1321, 2015.

[40] S. U. Stich and S. P. Karimireddy. The error-feedback framework: Better rates for sgd with delayed gradients and compressed communication. *arXiv preprint arXiv:1909.05350*, 2019.

[41] S. Wang, S. Hosseinalipour, V. Aggarwal, C. G. Brinton, D. J. Love, W. Su, and M. Chiang. Towards cooperative federated learning over heterogeneous edge/fog networks. *arXiv preprint arXiv:2303.08361*, 2023.

[42] B. Woodworth, K. K. Patel, S. Stich, Z. Dai, B. Bullins, B. Mcmahan, O. Shamir, and N. Srebro. Is local sgd better than minibatch sgd? In *International Conference on Machine Learning*, pages 10334–10343. PMLR, 2020.

[43] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning*, pages 7252–7261. PMLR, 2019.

[44] J. Zhang, S. Guo, X. Ma, H. Wang, W. Xu, and F. Wu. Parameterized knowledge transfer for personalized federated learning. *Advances in Neural Information Processing Systems*, 34:10092–10104, 2021.

[45] X. Zhang, M. Hong, S. Dhople, W. Yin, and Y. Liu. Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data. *arXiv preprint arXiv:2005.11418*, 2020.

[46] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.