Client Selection in Federated Learning: Principles, Challenges, and Opportunities

Lei Fu, Huanle Zhang, Ge Gao, Mi Zhang *Senior Member, IEEE*, and Xin Liu *Fellow, IEEE*

***Abstract*—As a privacy-preserving paradigm for training Machine Learning (ML) models, Federated Learning (FL) has received tremendous attention from both industry and academia. In a typical FL scenario, clients exhibit significant heterogeneity in terms of data distribution and hardware configurations. Thus, randomly sampling clients in each training round may not fully exploit the local updates from heterogeneous clients, resulting in lower model accuracy, slower convergence rate, degraded fairness, etc. To tackle the FL client heterogeneity problem, various client selection algorithms have been developed, showing promising performance improvement. In this paper, we systematically present recent advances in the emerging field of FL client selection and its challenges and research opportunities. We hope to facilitate practitioners in choosing the most suitable client selection mechanisms for their applications, as well as inspire researchers and newcomers to better understand this exciting research topic.**

***Index Terms*—Federated learning, Client selection, System heterogeneity, Data heterogeneity**

I. Introduction

FEDERATED Learning (FL) has gained great momentum in both academia and industry [1], [2]. As a decentralized paradigm that preserves data privacy while enabling Machine Learning (ML) model training, FL has been applied to various fields, such as next-word prediction [3], financial fraud detection [4], and healthcare data analysis [5]. According to the POLARIS report, the global FL market was valued at USD 110.8 million in 2021 and is expected to grow at a compound annual growth rate of 10.7%, reaching USD 266.8 million by 2030 [6].

FL models generally require a preliminary training process which, typically, includes four main steps, as illustrated in Figure 1. The training steps repeat until the global model on the server converges or a predefined number of epochs is reached. A plethora of work aims to solve different aspects of FL training, such as optimized aggregation methods [7], [8], enhanced privacy protection [9], [10], and improved robustness [11], [12].

Based on the scale and approach of training, FL can be roughly categorized into cross-silo FL and cross-device FL [28]. Cross-silo FL targets collaborative learning among several organizations, while cross-device FL targets machine learning across large populations, e.g., mobile devices [28], [29]. Currently, cross-device FL is more widely used in various application domains such as mobile phones, Internet-of-Things (IoT) [30], and mobile edge computing [31]. In cross-device scenarios, FL clients (e.g., all sorts of mobile or IoT devices) exhibit significant heterogeneity in terms of data statistics and system configurations, which, if not handled appropriately, can degrade the FL performance [32].

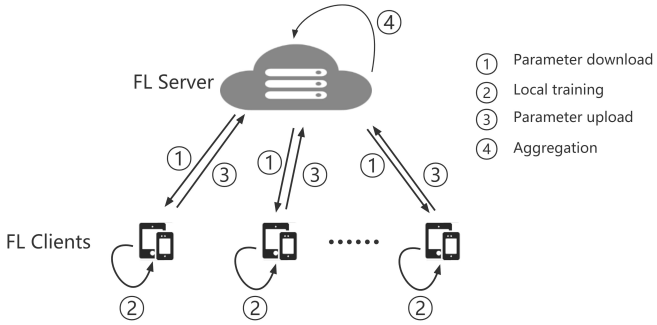


Fig. 1. An illustration of a standard FL training process that incorporates four steps: model parameter download from the server to the clients, local training on the clients, model parameter upload from the clients to the server, and model aggregation on the server.

Thus, to solve client heterogeneity problems, FL client selection (a.k.a. participant selection or device sampling) is an emerging topic. FL client selection decides which client devices are chosen in each training round. An effective FL client selection scheme can significantly improve model accuracy [16], enhance fairness [24], strengthen robustness [18], and reduce training overheads [22]. Therefore, the research community is witnessing a rapid development of FL client selection research in recent years [33].

However, there lacks a high-quality overview paper on FL client selection that can help newcomers quickly acquaint themselves with this research topic. To fill this gap, we provide an overview of FL client selection, covering the most representative work. Instead of simply listing existing work, we adopt a more systematic way: we organize existing papers based on their criteria for prioritizing FL clients. In addition, we discuss challenges and research opportunities for FL client selection. To the best of our knowledge, this is the first overview paper that provides deep insights into FL client selection.

This paper is organized as follows. Section II explains our literature review process. Section III clarifies the client heterogeneity in terms of hardware configurations and data dis-

Lei Fu is with the Bank of Jiangsu and Fudan University, China. E-mail: leileifu@163.sufe.edu.cn; Huanle Zhang and Ge Gao are with the School of Computer Science and Technology, Shandong University, China. E-mail: {dtczhang, vivi_gaoge}@sdu.edu.cn; Mi Zhang is with the Department of Computer Science and Engineering, Ohio State University, USA. E-mail: mizhang.1@osu.edu; Xin Liu is with the Department of Computer Science, University of California, Davis, USA. E-mail: xinliu@ucdavis.edu.

Huanle Zhang is the corresponding author.

Work	Publication	System	Data	Description
FedCS [13]	ICC'19	✓	✗	Select as many clients as possible within a specified deadline. It is based on a greedy algorithm with a knapsack constraint.
Elsa et al. [14]	ICASSP'21	✗	✓	Two-level importance sampling for clients and data. It first selects clients and then samples data of the selected clients.
OCEAN [15]	TWC'21	✓	✗	Bandwidth allocation under client energy constraints. It utilizes wireless channel information to achieve a better client selection pattern.
Oort [16]	OSDI'21	✓	✓	Exploit data and system heterogeneity in clients. It employs an exploration-exploitation strategy to select participants for robustness to outliers.
Mohammad et al. [17]	TWC'21	✓	✓	Consider shared block fading wireless channels and local gradient norm. It designs a resource allocation policy to schedule low-profile client devices.
FOLB [18]	JSAC'21	✓	✓	Based on the correlation between local update and global update. It estimates clients' capabilities of contribution to the updates.
Wenlin et al. [19]	TMLR'22	✗	✓	Norm-based client selection to tackle the communication bottleneck. It approximates the optimal formula for client selection with a simple algorithm.
FedPNS [20]	TNSE'22	✗	✓	Removing adverse local updates by comparing the gradients of the local and the global. It preferentially selects clients that propel faster model convergency.
POWER-OF-CHOICE [21]	AISTATS'22	✓	✓	Local loss based client selection tradeoff between convergence speed and solution bias. It shows that biasing client selection can speed up the convergence.
PyramidFL [22]	MobiCom'22	✓	✓	Exploit data and system heterogeneity within selected clients. It determines the utility-based client selection and then optimizes utility profiling locally.
FCFL [23]	IMWUT'22	✓	✓	Wearable devices in inferior networking conditions. It proposes movement aware FL to aggregate only the model udpates with top contributions.
Eiffel [24]	TPDS'22	✓	✓	Jointly consider factors such as resource demand and the age of update. It adaptively adjust the frequency of local and global model updates.
Bing et al. [25]	INFOCOM'22	✓	✓	Optimization of client sampling probabilities to address system and statistical heterogeneity. It minimizes the wall-clock convergence time.
F3AST [26]	JSTSP'23	✓	✗	Learns an availability-dependent client selection strategy to minimize the impact of client-selection variance on the global model's convergence.
Haoyu et al. [27]	IoT-J'23	✓	✗	Propose a dynamic user and task scheduling scheme with a block-wise incremental gradient aggregation algorithm.

TABLE I

REPRESENTATIVE WORK OF FL CLIENT SELECTION ALGORITHMS, ORDERED BY THE PUBLICATION YEAR. WE ONLY LIST WORK WITH EXPLICIT UTILITY MEASUREMENT AND SCHEDULING DECISIONS FOR CLIENTS. WE TAG WHETHER THE WORK DESIGNED FOR SYSTEM HETEROGENEITY AND DATA HETEROGENEITY.

tribution. Section IV presents criteria for prioritizing FL clients and Section V provides implementation practices. The research challenges and opportunities are highlighted in Section VI and Section VII, respectively. Last, this paper is concluded in Section VIII.

II. LITERATURE REVIEW PROCESS

We adopt a systematic literature review process [34], [35] in order to provide an unbiased and informative overview on FL client selection.

A. Research Questions

As an emerging topic in FL, client selection entails many unique design considerations. Thus, in this paper, we want to answer the following questions:

1) How does FL clients behave that affects the client selection performance? (Section III)
2) What is the principle behind existing FL client selection algorithms to prioritize FL clients? (Section IV)
3) What is the current practice to implement an FL client selection algorithm? (Section V)
4) What are the challenges to realize an effective FL client selection algorithm? (Section VI)
5) What are the research opportunities to boost performance of FL client selection? (Section VII)

B. Literature Searching and Appraisal

Measuring the utilities of FL clients and then scheduling clients based on their utility measures is the core idea behind FL client selection algorithms. Therefore, we search for works that have explicit utility measurements and scheduling decisions. We use the Scopus, Web of Science, and Google Scholar libraries and apply the following search syntax:

"Federated Learning" AND
("selection" OR "sampling" OR "scheduling")

We sort the searching results of each library by relevance and use the first 20 papers of each library results as the starting point. We keep the papers that meet the following two criteria: (1) target the standard FL scenario (i.e., an FL server learns a model by collaborating with a bunch of FL clients) and (2) have explicit utility measurements and scheduling decisions. In other words, we do not consider papers that are designed for FL variants (e.g., hierarchical FL, serverless FL) or implicit machine learning scheduling (e.g., reinforcement learning-based client selection). Therefore, our identified representative works are general in principle and can be easily adapted to new designs. Our initial literature searching results in 8 high-quality papers. Afterward, we read their citations and references, which includes 7 more high-quality papers. In the end, we identify 15 papers as tabulated in Table I.

C. Literature Synthesis and Analysis

For the selected papers, we carefully analyze the FL client behavior (Section III), categorize their methods to determine a

Refs	Focus Point
[36]	Enabling software and hardware platforms, protocols, real-life applications and use-cases.
[37]	Communication costs, resource allocation, and privacy and security in the implementation of FL at scale.
[38]	Sparsification, robustness, quantization, scalability, security, and privacy of FL-powered IoT applications.
[39]	Data partitioning, FL architectures, aggregation techniques, and personalization techniques.
[40]	Data partitioning, privacy mechanism, machine learning models, communication architecture and systems heterogeneity.
[41]	Core system models and designs, application areas, privacy and security, and resource management.
[42]	IoT data sharing, data offloading and caching, attack detection, localization, mobile crowdsensing, and IoT privacy and security.
[43]	Resource constrained IoT devices, distributed implementation, challenges and issues when applying FL to an IoT environment.
[44]	Data distribution, privacy mechanism, communication architecture, scale of federation and motivation of federation

TABLE II
RELATED SURVEYS ON FL, ORDERED BY THE PUBLICATION YEAR.

client's priority (Section IV), summarize their implementation platforms (Section V), identify the unsolved challenges (Section VI), and pinpoint the research opportunities (Section VII). In the writing of this overview paper, we significantly integrate relevant background and works to provide a deep insight of FL client selection.

D. Existing Surveys/Reviews

Due to the rapid development of federated learning, there are many published surveys on FL. Table II tabulates several relevant surveys, which cover general aspects of FL from data distribution/partitioning to real-world implementations. However, these surveys only touch the topic of FL client selection without providing deep insights as this paper. Specifically, we adopt a literature review process to explain the most important aspects of FL client selection (see Section II-A) and thus readers can quickly obtain a comprehensive view of this research topic.

III. FL Client Heterogeneity

In a typical FL training scenario, FL clients exhibit system and statistical heterogeneity. Random client selection does not consider this heterogeneity and thus results in performance degradation. For example, a large-scale study on real-world data from 136k smartphones shows that the heterogeneity reduces the FL model accuracy by up to 9.2% and increases the convergence time by 2.64X [32].

A. System Heterogeneity

Mobile devices are usually equipped with different hardware that have diverse capabilities of computation, communication, energy, etc.

Computation Capability. With the gaining popularity of gaming and AI applications, mobile devices often have AI accelerators such as GPU, NPU, or CUDA cores. However, measurements of mainstream mobile devices show that they can spend more than tens of times difference in running AI models [45]. The time difference is even large if AI models cannot fit into the memory of AI accelerators or AI model operators are not supported on mobile devices [46].

Communication Capability. Transmission speed is essential as FL training involves many rounds of model parameter transmissions between the server and the clients. However, clients can have significantly different transmission speeds because of their transmission standards (e.g., LTE vs. WiFi), locations (indoor vs. outdoor), and wireless channel conditions (clean vs. congested). An analysis of hundreds of mobile phones in a real-world FL deployment [47] shows that the network bandwidth exhibits an order-of-magnitude difference [16].

Other Factors. In addition to the computation and communication capabilities, many other factors also affect the availability and capability of clients. For example, a smartphone with a low battery level would automatically reduce the computation and communication capabilities to save energy; a mobile device with heavy applications running in the background greatly limits the available computing resources.

B. Statistical Heterogeneity

Compared to other model training paradigms (e.g., centralized machine learning [48], conventional distributed machine learning [49]), FL has unique data properties with regards to massively distributed data, unbalanced data, and non-Independent and Identically Distributed (IID) data [50].

Massively Distributed Data: the number of FL clients is much larger than the clients' average number of data points. For example, a million of smartphones are involved in the Google keyboard query suggestion project [47], but a user usually only makes up to dozens of queries a day.

Unbalanced Data: clients have a different amount of data points. This is because the various use patterns result in a highly different local data size. For example, the Reddit comment dataset [51] reveals that 70% users constitute the first quarter of the normalized number of comments while 10% users make three-times more comments [16].

Non-IID Data: each client's data does not represent the overall distribution as data are not IID. The non-IID data property has been widely observed in real-world applications [52]. Both the attribute skew and the label skew greatly affect the FL model training [53].

IV. Prioritizing FL Clients

In each training round, each client is measured for its utility/priority and the clients with the best utility measurements are selected for model training and aggregation. There are various methods for formulating clients' utility. Table I summarizes the representative client selection algorithms that have explicit utility measurement and scheduling decisions. We tag the work designed specially for the system heterogeneity and the data heterogeneity. In this section, we elaborate on the utility functions. Table III tabulates the notations that are used in this paper for quick reference.

Notation	Meaning
N	Total number of FL clients
M	Number of clients selected in a training round
i	Index of FL clients
D_i	Data samples of client i
B_i	A batch of data samples in client i
f	Mapping from input to output
w	Model weights
j	Index of model weights
w_{ij}, $\bar{w}_j$	j-th weight in client i and server, respectively
T	Deadline for an FL round
t_i	Round time of client i
α	Exponent controlling the punishment for stragglers

TABLE III
NOTATIONS USED IN THIS PAPER.

An overall utility function for a client i are often represented by its statistical utility and system utility as follows [16], [22]:

$$Util(i) = Util_{stat}(i) \quad Util_{sys}(i); \tag{1}$$

where $Util_{stat}(i)$ and $Util_{sys}(i)$ represents the statistical and the system utility for client i, respectively. Note that other factors, such as fairness, robustness, and privacy, can be formulated similarly.

A. Statistical Utility

Statistical utility represents the usefulness of a client's local update to the global model. We categorize statistical utilities into data sample-based and model-based.

1) Data Sample-Based Utility Measurement: Data sample-based utility exploits a client's local data to quantify the statistical utility.

A simple way to represent the statistical utility of client i is by the number of data samples on client i, i.e.,

$$jD_ij; \tag{2}$$

where D_i denotes the data samples of client i. This approach is valid when each data sample has the same quality, e.g., IID data.

Importance sampling of data samples has been widely studied in the general ML literature [54], [55] and has been recently applied in the federated setting [56]. The idea is to assign a high importance score to a data sample that is divergent far from the model. Eq. (3) shows one optimal solution:

$$jB_ij\sqrt{\frac{1}{jB_ij}\sum_{k2B_i} jj\,r\,f(k)jj_2^2}; \tag{3}$$

where $jj\,r\,f(k)jj_2$ is the L_2-norm of the data sample k's gradient in bin B_i of FL client i. Albeit its optimality, the computation overhead is overwhelming as it needs to calculate the gradient of each data sample in all possible combinations.

A data sample with a large loss generally has a large gradient norm [16], [57]. Therefore, an alternative to Eq. (3) is replacing the gradient norm with the loss, which results in

$$jB_ij\sqrt{\frac{1}{jB_ij}\sum_{k2B_i} Loss(k)^2}: \tag{4}$$

Since each data sample's loss is available during the local training, the computation overhead is greatly reduced.

The above expression can be further simplified by calculating the cumulative loss of client i's data samples:

$$\sum_{k2B_i} Loss(k): \tag{5}$$

It is adopted by [21] and also shows promising results.

2) Model-Based Utility Measurement: Another approach to determining the priority of a client is to compare its model weights/gradients. Different methods are developed to quantify the potential contribution of a local model based on the model.

The normalized model divergence is defined as the average difference between the model weights in client i and the global model, i.e.,

$$\frac{1}{jwj}\sum_{j=1}^{jwj} j\frac{w_{ij} \quad w_j}{w_j}j: \tag{6}$$

w represents the weights of a model and w represents the weights of the global model; w_{ij} and w_j are the j-th weights of client i and the global model, respectively. If the model divergence is small, then the local update from the client is insignificant and can be ignored [58].

Another work calculates the percentage of same-sign weights between a client model and the global model which can be regarded as a direction relative to the global model [59]:

$$\frac{1}{jwj}\sum_{j=1}^{jwj} \mathbb{1}(sign(w_{ij}) = sign(w_j)); \tag{7}$$

where $\mathbb{1}(sign(w_{ij}) = sign(w_j)) = 1$ if w_{ij} and w_j are of the same sign. Although most works believe that a more divergent local model is more important (e.g., [16], [22], [23]), [59] shows that a lower percentage of same-sign weights results in better communication efficiency upon converges.

The similarity to the convergence trend is also used to select clients with weights that are moving the most away from 0. For a L-layer model, the importance score of client i can be expressed as [23]:

$$\frac{1}{L}\sum_{l=1}^{L}\frac{mov(u_{il}) \quad mov(u_l)}{jjmov(u_{il})jj \quad jjmov(u_l)jj}; \tag{8}$$

where u_{il} represents the gradient of the loss with respect to the weights of the l-th layer in client i. Correspondingly, u_l represents the counterpart of the global model. The movement function $mov(\,)$ represents the movement direction of the weights and is defined in [60].

Instead of comparing the local model with the global model, [61] proposes to compare the change of local model before and after the local training. That is,

$$jjw_i^{after} \quad w_i^{before}jj_2; \tag{9}$$

A client i is assumed to have a higher contribution if its local training results in a significant different local model [61].

Equivalently, the L_2-norm of the model's gradients can be used [14], [17], defined below.

$$\|\nabla w_i\|_2 \quad (10)$$

A higher norm indicates a more valuable client. Variants of L_2-norm based client selection are also used, e.g., in [19].

The inner product between the gradients of the local model and the global represents its relative direction, which also indicates the divergence between a local model and the global model.

$$< \nabla w_i, \nabla w > \quad (11)$$

FOLB [18] and FedPNS [20] removes clients that have negative inner products.

B. System Utility

Due to the different hardware configurations, FL clients results in different system overheads (e.g., training and transmission time) in each training round. Slow devices (i.e., stragglers) can deteriorate the overall training performance by prolonging the training round if not carefully considered. There are a few mainstream system utilities to prioritize clients.

A deadline can be set to avoid excessively long server waiting time in each training round. The deadline-based selection has been widely used (e.g., [3], [13]), due to its easy implementation. Mathematically, clients with a deadline longer than T are removed from the FL aggregation, i.e.,

$$\mathbb{1}(t_i < T); \quad (12)$$

where t_i is the total round time of client i that includes the local training, transmission, compression, etc.

A hard deadline, as above, might be too strict for some application scenarios. Thus, a soft deadline is imposed by some work (e.g., Oort [16]) to penalize stragglers in the following manner:

$$(\frac{T}{t_i})^{\mathbb{1}(T<t_i)} \quad ; \quad (13)$$

where is the exponent controlling the penalty for stragglers. Eq. (13) equals 1 (i.e., no punishment) for non-stragglers and increases exponentially for stragglers.

Note that T can represent other metrics other than time. For example, if the system targets the computation speed, then T is in FLOPs; if the system focuses the transmission bandwidth, then T is in Mbps. Many approaches set an empirical deadline, e.g., 2 minutes in the Gboard projects [3], [47] to ignore straggler clients. Manually determining the deadline could result in inferior performance as T significantly affects the clients for aggregation. FedBalancer designs an algorithm to automatically adjust T [62], showing better performance than using the fixed deadline.

C. Scheduling

Once the statistical utility function and the system utility function have been defined, the overall utility for each client can be calculated using Eq. (1). Ideally, in each training round, each client is measured for its utility and the clients of the highest utility measures can be chosen to join the training. However, it is not practical to measure every client's utility in each training round, as a client's utility often can only be determined after it has participated in a training round. Therefore, a mainstream approach is to forecast a client's utility along the training stage and update/rectify its utility measure once it is selected to join the training round [16], [22]. In addition, the scheduling is faced with the exploitation-exploration dilemma, which is explained in Section VI.

D. Discussion

In this paper, the overall utility of each client is a multiplication of the statistical utility and the system utility (Eq. (1)), the form of which has been adopted by recent works such as Oort [16] and PyramidFL [22]. The multiplication form can be extended to include other utility aspects such as fairness [63] and robustness [18] by multiplying corresponding utility functions. It can also ignore unwanted aspects by simply assigning 1 to the corresponding utility functions. Therefore, the multiplication form is expressive. Other forms are also applicable. For example, Eiffel [24] expresses a client's overall utility by adding the loss value of its local model, the local data size, the computation power, the resource demand, and the age of update, with adjustable weights for different utility aspects. The utility functions covered in this section can be applied to other FL client selection designs.

V. Current Implementation

In this section, we briefly explain data simulation and frameworks for FL client selection research.

A. Data Simulation

In research, FL clients' data are often assigned in one of the following manners. (1) Synthetic data partitions. Researchers can use conventional ML datasets (e.g., MNIST [64], Shakespeare, CIFAR-10 [65]) and partition the dataset into different clients. This approach allows great flexibility as different degrees of data heterogeneity can be simulated [50]. (2) Realistic data partitions. The other approach is to adopt datasets with the client ID and partition the dataset using the unique client ID. A variety of realistically partitioned dataset are available (refer to [16], [52], [66] for more information), such as OpenImage [67], StackOverflow [68], and Reddit [51]. This data approach can more accurately capture the FL performance in real-world scenarios.

B. Federated Learning Frameworks

In addition to implementing FL from scratch using e.g, PyTorch and TensorFlow, we can also use frameworks that are designed specifically for FL, which could facilitate FL

research and deployment. Below are some representative FL frameworks used for FL client selection research.

TensorFlow Federated (TFF) [69] is developed by Google and is an open-source framework for experimenting with FL. It supports mobile devices and has been used for commercial projects such as mobile keyboard prediction [3].
FedScale [66] is initialized by the University of Michigan. It provides many useful features, such as mobile device profiles of computation and communications speeds. FedScale enables on-device FL evaluation on smartphones and in-cluster simulations. Recent work such as PyramidFL [22] and Oort [16] use FedScale for their experiments.
Leaf [70] is developed by a research group at Carnegie Mellon University. It includes a suite of federated datasets, an evaluation framework, and a set of reference implementations. Leaf has been used by FLASH [32] to study the heterogeneity impacts on FL client selection algorithms.
FedML [52] is an open-source platform for an end-to-end machine learning ecosystem. It supports a lightweight and cross-platform design for secure edge training. Therefore, FedML is useful for designing IoT-based FL systems [71].

These representative FL frameworks have advantages and disadvantages. FedML focuses on FL ecosystems for diverse application domains, but it is still in its early stage with moderate community support. TFF is actively maintained by a large community. Building an industrial-level FL client selection solution with TFF is convenient as TFF can be easily integrated with other Google products/services. However, TFF does not provide dataset partitions designed specifically for FL research. To this end, Leaf is proposed to facilitate FL research by providing a unified API for several popular datasets but with no support for device profiles. As a newcomer to this field, FedScale is developed with both the statistical and system heterogeneity in mind from the beginning. However, its community support is still early and requires further testing.

VI. Challenges

In this section, we highlight several challenges that hinder the development of high-performance FL client selections.

Existing work mostly assumes that devices are always available for FL training, which is not true in practice. Very often, devices are only available when they are idle, charged, and connected to WiFi, in order to protect the user experience [3]. For example, Google observes lower training accuracy during the day, as few devices satisfy this requirement, which generally represents a skewed population [47]. Datasets for device availability are scarce. FLASH [32] provides an input method App dataset containing smartphone status traces that can be used to emulate the device availability. Its data analysis reveals that some active devices dominate the global model, which leads to participant bias.

Always selecting the prioritized clients tends to result in sub-optimal performance as underrepresented clients may never have the chance to be selected [23]. Therefore, there is a tradeoff between exploitation (selecting prioritized clients) and exploration (selecting more diverse clients). This exploitation-exploration dilemma is common in many research fields, such as data annotation in active learning [72] and space search in reinforcement learning [73]. The exploitation-exploration dilemma is challenging, especially for FL, which needs more comprehensive studies of this problem. Current FL client selection work adopts simple methods to trade off the exploitation and exploration [16], [22].

Designing an effective and general client selection algorithm remains challenging. Worse, heterogeneity is different across different regions and application scenarios. For example, mobile users in the US generally have stable network conditions, contrary to the widely assumed that mobile users always suffer from transmission interruptions [23].

VII. Research Opportunities

FL client selection is a new research topic with many unsolved problems and challenges. It also offers many research opportunities that are worth exploring. In addition to designing high-performance FL client selection algorithms for different application scenarios, the below aspects are also critical for FL client selection.

A. Optimal Number of Selected Clients

The current mainstream practice is to determine the number empirically. For example, the Google Gboard project uses 100 clients for training keyboard query suggestions. Many pieces of work have shown that the FL convergence rate can be improved by selecting more clients, with diminishing improvement gains as the number increases [50]. Besides, more clients are preferred in each training round when data follow a more heterogeneous distribution [17]. However, selecting more clients is not always possible or optimal when clients are subject to constraints such as energy or bandwidth [15], [17]. Furthermore, different FL training stages may prefer different numbers of selected clients. A "later-is-better" phenomenon is observed in which an ascending number of client patterns is generally desired [15]. However, all these observations are made in hindsight, and thus, research is needed to identify the optimal number of selected clients for diverse applications. Another promising line of research is automatically tuning the number during FL training. For example, FedTune [74] proposes a simple tuning algorithm that increases or decreases the number by one in each decision step. Overall, we need high-quality approaches designed specifically for tuning the number of selected clients in each training round.

B. Theoretical Performance Guarantee

Existing work on FL client selection mostly rely on experiments to demonstrate their effectiveness; thus, the results given in these work may be susceptible to experiment bias. For example, some work demonstrates that clients with more divergent models from the global are preferred (e.g., [16], [23]), while some work shows the opposite (e.g., [20], [59]). The contradictory observations may stem from different application scenarios. Due to the heuristic properties of most FL

client selection algorithms, it is challenging, if possible, to provide theoretical guarantee for their algorithms with regard to model accuracy, convergence rate, robustness, fairness, etc. Without performance guarantee of client selection algorithms, FL practitioners tend to adopt the random client selection method in their projects, resulting in sub-optimal performance. Therefore, more research is needed to provide theoretical analysis frameworks for FL client selection algorithms.

C. Benchmark and Evaluation Metrics

Existing work adopts various metrics to evaluate their performance. Final model accuracy [22], time-to-accuracy [16], round-to-accuracy [7], transmission load [75], etc, are well-known metrics. However, different metrics are not comparable. For example, selecting more clients in each training round results in a better round-to-accuracy performance. But it does not necessarily mean a better time-to-accuracy as the time length of each training round increases with the number of selected clients, nor a better transmission efficiency as more clients need to transmit model parameters in each training round. In addition, different experiment settings are adopted by existing work, whose results may not apply to other applications. Besides, the local data sampling of the selected clients can also affect the FL performance [76]. Therefore, the community needs well-established benchmarks and evaluation metrics to fairly and objectively compare different FL client selection algorithms.

D. Extension to Other Federated Learning Scenarios

In addition to the classical FL scenario, where a global model is trained using a bunch of clients (e.g., FedAvg settings), other variants of FL scenarios are gaining increasing attention from academia. For example, (1) Hierarchical FL [77], which often includes cloud space, edge space, and client space compared to the standard server-client paradigm; (2) Cluster-based FL [78], which groups clients based on the data distribution or system capability into clusters and then schedules clusters for better performance; (3) Online FL [79], which requires lightweight FL updates during the user applications instead of the idle-charging-WiFi condition for client selection; (4) Serverless FL [80], where there is no fixed and permanent server for FL training coordination but in a peer-to-peer approach. Each variant poses unique challenges and entails specialized solutions. Therefore, more research efforts for client selection solutions are required to tackle different scenarios.

E. Large-Scale Open FL Testbeds

Although high-quality FL libraries are available for research and deployment, existing work mostly relies on either simulation or a small-scale implementation setting (e.g., a few devices). As a result, the observations made in existing work do not totally match the actual FL performance, especially for the FL client selection research, whose application scenarios often require a large number of clients. In addition, the current practice of building own experiment environments not only has difficulty in reproducing results but also hinders a fair comparison among different approaches. We envision large-scale, open testbeds for FL research, with a similar role to FlockLab testbed [81] for the wireless sensor network and IoT research.

VIII. Conclusion

This paper is by no means an exhaustive survey, as FL client selection has many variants for different scenarios (e.g., trust-driven FL [82], hybrid FL [83]). Also, we do not emphasize work that does not has explicit utility function and/or decision, e.g., fuzzy logic-based client selection [84], reinforcement learning-based client selection [85]. Instead, we cover the most general selection criteria that are widely applicable to new FL system designs. This paper provides insights into statistical and system heterogeneity, their utility functions, implementation suggestions, challenges, and research opportunities. We hope this paper could inspire more research efforts in FL client selection.

Acknowledgment

This work is partially supported by the Jiangsu Funding Program for Excellent Postdoctoral Talent (No. 2022ZB804). This work is also supported by the National Science Foundation of the United States through grants USDA/NIFA 2020-67021-32855, IIS-1838207, CNS 1901218, and OIA-2134901.

References

[1] T. Zhang, L. Gao, C. He, et al. Federated learning for the internet of things: Applications, challenges, and opportunities. *IEEE Internet of Things Magazine*, 5(1):24–29, 2022.

[2] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[3] A. Hard, K. Rao, R. Mathews, et al. Federated learning for mobile keyboard prediction. *arXiv:1811.03604*, 2019.

[4] T. Suzumura, Y. Zhou, R. Kawahara, et al. Federated learning for collaborative financial crimes detection. *Federated Learning*, pages 455–466, 2022.

[5] R. S. Antunes, C. A. da Costa, A. Kuderle, et al. Federated learning for healthcare: Systematic review and architecture proposal. *ACM Transactions on Intelligent Systems and Technology*, 13(4):1–23, 2022.

[6] POLARIS Market Research. Federated learning market share, size, trends, industry analysis report. https://www.polarismarketresearch.com/industry-analysis/federated-learning-market/, 2021.

[7] J. Wang, Q. Liu, H. Liang, et al. Tackling the objective inconsistency problem in heterogeneous federated optimization. In *NeurIPS*, 2020.

[8] S. J. Reddi, Z. Charles, M. Zaheer, et al. Adaptive federated optimization. In *International Conference on Learning Representations (ICLR)*, 2021.

[9] A. E. Ouadrhiri and A. Abdelhadi. Differential privacy for deep and federated learning: A survey. *IEEE Access*, 10:22359–22380, 2022.

[10] B. Jia, X. Zhang, J. Liu, et al. Blockchain-enabled federated learning data protection aggregation scheme with differential privacy and homomorphic encryption in iiot. *IEEE Transactions on Industrial Infomatics*, 18(6):4049–4058, 2022.

[11] J. Sun, A. Li, L. DiValentin, et al. Fl-wbc: Enhancing robustness against model poisoning attacks in federated learning from a client perspective. In *NeurIPS*, 2021.

[12] T. Li, S. Hu, A. Beirami, and V. Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning (ICML)*, 2021.

[13] T. Nishio and R. Yonetani. Client selection for federated learning with heterogeneous resources in mobile edge. In *IEEE International Conference on Communications (ICC)*, 2019.

[14] E. Rizk, S. Vlaski, and A. H. Sayed. Optimal importance sampling for federated learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[15] J. Xu and H. Wang. Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective. *IEEE Transactions on Wireless Communications*, 20(2):1188–1200, 2021.

[16] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury. Oort: Efficient federated learning via guided participant selection. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2021.

[17] M. M. Amiri, D. Gunduz, S. R. Kulkarni, and H. V. Poor. Convergence of update aware device scheduling for federated learning at the wireless edge. *IEEE Transactions on Wireless Communications*, 20(6):3643–3658, 2021.

[18] H. T. Nguyen, V. Sehwag, S. Hosseinalipour, et al. Fast-convergent federated learning. *IEEE Journal on Selected Areas in Communications*, pages 201–218, 2021.

[19] W. Chen, S. Horvath, and P. Richtarik. Optimal client sampling for federated learning. *Transactions on Machine Learning Research*, pages 1–32, 2022.

[20] H. Wu and P. Wang. Node selection toward faster convergence for federated learning on non-iid data. *IEEE Transactions on Network Science and Engineering*, 9(5):3099–3111, 2022.

[21] Y. J. Cho, J. Wang, and G. Joshi. Towards understanding biased client selection in federated learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022.

[22] C. Li, X. Zeng, M. Zhang, and Z. Cao. Pyramidfl: A fine-grained client selection framework for efficient federated learning. In *Conference on Mobile Computing and Networking (MobiCom)*, 2022.

[23] P. Zhou, H. Xu, L. H. Lee, et al. Are you left out? an efficient and fair federated learning for personalized profiles on wearable devices of inferior networking conditions. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 6:1–25, 2022.

[24] A. Sultana, M. M. Haque, L. Chen, et al. Eiffel: Efficient and fair scheduling in adaptive federated learning. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 33(12):1–13, 2022.

[25] B. Luo, W. Xiao, S. Wang, et al. Tackling system and statistical heterogeneity for federated learning with adaptive client sampling. In *IEEE Conference on Computer Communications (INFOCOM)*, 2022.

[26] M. Ribero, H. Vikalo, and G. de Veciana. Federated learning under intermittent client availability and time-varying communication constraints. *IEEE Journal of Selected Topics in Signal Processing (PSTSP)*, 17(1):98–111, 2023.

[27] H. Ma, H. Guo, and V. K. N. Lau. Communication-efficient federated multitask learning over wireless networks. *IEEE Internet of Things Journal (IoT-J)*, 10(1):609–624, 2023.

[28] J. Wang, Z. Charles, Z. Xu, et al. A field guide to federated optimization. *arXiv: 2107.06917*, 2021.

[29] K. Bonawitz, H. Eichner, W. Grieskamp, et al. Towards federated learning at scale: System design. In *Conference on Machine Learning and Systems (MLSys)*, 2019.

[30] R. Kontar, N. Shi, X. Yue, et al. The internet of federated things (ioft). *IEEE Access*, 9:156071–156113, 2021.

[31] S. Wang, T. Tuor, T. Salonidis, et al. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 37(6):1205–1221, 2019.

[32] C. Yang, M. Xu, Q. Wang, et al. Flash: Heterogeneity-aware federated learning at scale. *IEEE Transactions on Mobile Computing*, pages 1–18, 2022.

[33] B. Soltani, V. Haghighi, A. Mahmood, et al. A survey on participant selection for federated learning in mobile networks. In *ACM Workshop on Mobility in the Evolving Internet Architecture (MobiArch)*, 2022.

[34] W. Mengist, T. Soromessa, and G. Legese. Method for conducting systematic literature review and meta-analysis for environmental science research. *MethodsX*, 7:1–11, 2020.

[35] Y. Xiao and M. Watson. Guidance on conducting a systematic literature review. *Journal of Planning Education and Research*, 39:93–112, 2017.

[36] M. Aledhari, R. Razzak, R. M. Parizi, and F. Saeed. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access*, 8:140699–140725, 2020.

[37] W. Y. B. Lim, N. C. Luong, D. T. Hoang, et al. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22(3):2031–2063, 2020.

[38] L. U. Khan, W. Saad, Z. Han, et al. Federated learning for internet of things: Recent advances, taxonomy, and open challenges. *IEEE Communications Surveys & Tutorials*, 23(3):1759–1799, 2021.

[39] K. M. J. Rahman, F. Ahmed, N. Akhter, et al. Challenges, applications and design aspects of federated learning: A survey. *IEEE Access*, 9:124682–124700, 2021.

[40] C. Zhang, Y. Xie, H. Bai, et al. A survey on federated learning. *Knowledge-Based Systems*, 216:1–11, 2021.

[41] S. AbdulRahman, H. Tout, H. Ould-Slimane, et al. A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. *IEEE Internet of Things Journal (IoT-J)*, 8(7):5476–5497, 2021.

[42] D. C. Nguyen, M. Ding, P. N. Pathirana, et al. Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658, 2021.

[43] A. Imteaj, U. Thakker, S. Wang, et al. A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal (IoT-J)*, 9(1):1–24, 2022.

[44] Q. Li, Z. Wen, Z. Wu, et al. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3347–3366, 2023.

[45] A. Ignatov, R. Timofte, A. Kulik, et al. Ai benchmark: All about deep learning on smartphones in 2019. In *IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019.

[46] H. Zhang, B. Han, and P. Mohapatra. Toward mobile 3d vision. In *IEEE International Conference on Computer Communications and Networks (ICCCN)*, 2020.

[47] T. Yang, G. Andrew, H. Eichner, et al. Applied federated learning: Improving google keyboard query suggestions. *arXiv:1812.02903*, 2018.

[48] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[49] J. Verbraeken, M. Wolting, J. Katzy, et al. A survey on distributed machine learning. *ACM Computing Surveys*, 53(2):1–33, 2020.

[50] H. B. McMahan, E. Moore, D. Ramage, et al. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[51] Reddit comment dataset. https://files.pushshift.io/reddit/comments/. accessed in October 2022.

[52] C. He, S. Li, J. So, et al. Fedml: A research library and benchmark for federated machine learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[53] H. Zhu, J. Xu, S. Liu, and Y. Jin. Federated learning on non-iid data: A survey. *Neurocomputing*, 465:371–390, 2021.

[54] A. Katharopoulos and F. Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *International Conference on Machine Learning (ICML)*, 2018.

[55] P. Zhao and T. Zhang. Stochastic optimization with importance sampling for regularized loss minimization. In *International Conference on Machine Learning (ICML)*, 2015.

[56] X. Zeng, M. Yan, and M. Zhang. Mercury: Efficient on-device distributed dnn training via stochastic importance sampling. In *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2021.

[57] H. Cao, Q. Pan, Y. Zhu, and J. Liu. Birds of a feather help: Context-aware client selection for federated learning. In *International Workshop on Trustable, Verifiable and Auditable Federated Learning in Conjunction with AAAI (FL-AAAI)*, 2022.

[58] K. Hsieh, A. Harlap, N. Vijaykumar, et al. Gaia: Geo-distributed machine learning approaching lan speeds. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2017.

[59] L. Wang, W. Wang, and B. Li. Cmfl: Mitigating communication overhead for federated learning. In *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2019.

[60] V. Sanh, T. Wolf, and A. M. Rush. Movement pruning: Adaptive sparsity by fine-tuning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[61] J. Zhao, X. Chang, Y. Feng, et al. Participant selection for federated learning with heterogeneous data in intelligent transport system. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–10, 2022.

[62] J. Shin, Y. Li, Y. Liu, and S.-J. Lee. Fedbalancer: Data and pace control for efficient federated learning on heterogeneous clients. In *International Conference on Mobile Systems, Applications and Services (MobiSys)*, 2022.

[63] T. Li, M. Sanjabi, A. Beirami, and V. Smith. Fair resource allocation in federated learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[64] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 1998.

[65] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[66] F. Lai, Y. Dai, S. S. Singapuram, et al. Fedscale: Benchmarking model and system performance of federated learning at scale. In *International Conference on Machine Learning (ICML)*, 2022.

[67] Google open images dataset. https://storage.googleapis.com/openimages/web/index.html. accessed in October 2022.

[68] Stack Overflow dataset. https://cloud.google.com/bigquery/public-data. accessed in October 2022.

[69] Google Inc. Tensorflow federated. https://www.tensorflow.org/federated. accessed in October 2022.

[70] S. Caldas, S. M. K. Duddu, P. Wu, et al. Leaf: A benchmark for federated settings. In *Workshop on Federated Learning for Data Privacy and Confidentiality*, 2019.

[71] T. Zhang, C. He, T. Ma, et al. Federated learning for internet of things. In *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2021.

[72] X. Wang, N. Rai, B. M. P. Pereira, et al. Accelerating knowledge discovery from omics data by optimal experimental design. *Nature Communications*, 11(5026):1–9, 2020.

[73] Y. Liu, A. Halev, and X. Liu. Policy learning with constraints in model-free reinforcement learning: A survey. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[74] H. Zhang, M. Zhang, X. Liu, et al. Fedtune: Automatic tuning of federated learning hyper-parameters from system perspective. In *IEEE Military Communications Conference (MILCOM)*, 2022.

[75] X. Wang, N. Rai, B. M. P. Pereira, et al. Performance-aware client and quantization level selection algorithm for fast federated learning. In *IEEE Wireless Communications and Networking Conference (WCNC)*, 2022.

[76] L. Cai, D. Lin, J. Zhang, and S. Yu. Dynamic sample selection for federated learning with heterogeneous data in fog computing. In *IEEE International Conference on Communications (ICC)*, 2020.

[77] Z. Qu, R. Duan, L. Chen, et al. Context-aware online client selection for hierarchical federated learning. *IEEE Transactions on Parallel and Distributed Systems*, 33(12):1–15, 2022.

[78] J. Wolfrath, N. Sreekumar, D. Kumar, et al. Haccs: Heterogeneity-aware clustered client selection for accelerated federated learning. In *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2022.

[79] G. Damaskinos, R. Guerraoui, A.-M. Kermarrec, et al. Fleet: Online federated learning via staleness awareness and performance prediction. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(5):1–30, 2022.

[80] C. Che, X. Li, C. Chen, et al. A decentralized federated learning framework via committee mechanism with convergence guarantee. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 33(12):4783–4800, 2022.

[81] R. Trub, R. D. Forno, L. Sigrist, et al. Flocklab 2: Multi-modal testing and validation for wireless iot. In *Workshop on Benchmarking Cyber-Physical Systems and Internet of Things (CPS-IoTBench)*, 2020.

[82] G. Rjoub, O. A. Wahab, J. Bentahar, and A. Bataineh. Trust-driven reinforcement selection for federated learning on iot devices. *Computing*, 2022.

[83] X. Liu, Y. Deng, and T. Mahmoodi. Energy efficient user scheduling for hybrid split and federated learning in wireless uav networks. In *IEEE International Conference on Communications (ICC)*, 2022.

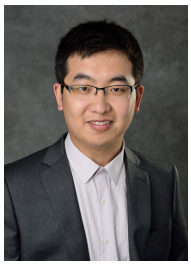
[84] N. Cha, Z. Du, C. Wu, et al. Fuzzy logic based client selection for federated learning in vehicular networks. *IEEE Open Journal of the Computer Society*, 3:39–50, 2022.

[85] W. Xia, T. Q. S. Quek, K. Guo, et al. Multi-armed bandit-based client scheduling for federated learning. *IEEE Transactions on Wireless Communications*, 19(11):7108–7123, 2020.

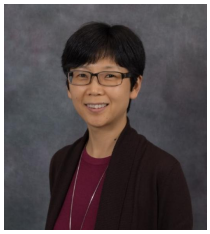
Lei Fu received her Ph.D. degree in Financial Information Engineering, Shanghai University of Finance and Economics in 2020. She is a postdoc in Fudan University and a quantitative investment analyst in Bank of Jiangsu, China. She received the Excellent Postdoc Award of Jiangsu Province in 2022. Her research interests include quantitative investment, macroeconomic computing, and financial technology.

Huanle Zhang is an associate professor in the School of Computer Science and Technology at Shandong University, China. He received his Ph.D. degree in Computer Science from the University of California, Davis (UC Davis), in 2020. He was employed as a postdoc at UC Davis 2020-2022 and a project officer at Nanyang Technological University 2014-2016. His research interests include data-centric AI, data privacy, IoT, and mobile systems.

Ge Gao received her M.S. degree in Computer Science from Illinois Institute of Technology. Since 2020, she works at the School of Computer Science and Technology, Shandong University, China. Her current research interests include wearables activity data analysis, machine learning in healthcare, data privacy, and mobile computing.



Mi Zhang is an associate professor in the Department of Computer Science and Engineering at The Ohio State University. He received his Ph.D. in Computer Engineering from the University of Southern California (USC) in 2013. His research lies at the intersection of mobile/edge/IoT systems and machine learning. He is the recipient of seven best paper awards and nominations. He is also the recipient of the National Science Foundation CRII Award, Facebook Faculty Research Award, Amazon Machine Learning Research Award, and MSU Innovation of the Year Award. He is a senior member of the IEEE.



Xin Liu received her Ph.D. degree in electrical engineering from Purdue University in 2002. She is currently a Professor in Computer Science at the University of California, Davis. She received the Computer Networks Journal Best Paper of Year award in 2003 and NSF CAREER award in 2005. She became a Chancellor's Fellow in 2011. She is a co-PI and AI cluster co-lead for the USD 20M AI Institute for Next Generation Food Systems. She is a fellow of the IEEE.