AUCTION: Automated and Quality-Aware Client Selection Framework for Efficient Federated Learning

Yongheng Deng, Feng Lyu, *Member, IEEE*, Ju Ren, *Senior Member, IEEE*, Huaqing Wu, *Member, IEEE*, Yuezhi Zhou, *Senior Member, IEEE*, Yaoxue Zhang, *Senior Member, IEEE*, and Xuemin Shen, *Fellow, IEEE*

Abstract—The emergency of federated learning (FL) enables distributed data owners to collaboratively build a global model without sharing their raw data, which creates a new business chance for building data market. However, in practical FL scenarios, the hardware conditions and data resources of the participant clients can vary significantly, leading to different positive/negative effects on the FL performance, where the client selection problem becomes crucial. To this end, we propose *AUCTION*, an Automated and qUality-aware Client selecTION framework for efficient FL, which can evaluate the learning quality of clients and select them automatically with quality-awareness for a given FL task within a limited budget. To design *AUCTION*, multiple factors such as data size, data quality, and learning budget that can affect the learning performance should be properly balanced. It is nontrivial since their impacts on the FL model are intricate and unquantifiable. Therefore, *AUCTION* is designed to encode the client selection policy into a neural network and employ reinforcement learning to automatically learn client selection policies based on the observed client status and feedback rewards quantified by the federated learning performance. In particular, the policy network is built upon an encoder-decoder deep neural network with an attention mechanism, which can adapt to dynamic changes of the number of candidate clients and make sequential client selection actions to reduce the learning space significantly. Extensive experiments are carried out based on real-world datasets and well-known learning models to demonstrate the efficiency, robustness, and scalability of *AUCTION*.

Index Terms—Federated learning, distributed system, client selection, data quality, reinforcement learning

1 Introduction

The fast proliferation of mobile edge devices results in the rapid growth of data generated at the edge, which promotes the advancement of modern artificial intelligent applications [1], [2], [3], [4]. However, due to the privacy issues [5] and prohibitive data transmission costs [6], the traditional mechanism that gathers extensive data at the cloud for centralized model training is often inaccessible. To fully exploit the data without leaking privacy, a new learning paradigm has emerged, namely federated learning (FL) [7], which enables mobile edge devices to collaboratively learn a global model without sharing their raw data. In FL, distributed devices train the global model locally using their own data, and then commit the model updates to the server for model aggregation. The aggregated model updates are used to update the global model, which is then returned to each device for the next round of iteration. In this way, the global model can be learned iteratively in a distributed and privacy-preserving manner.

Despite the tremendous potential in privacy preservation, FL still faces technical challenges in achieving satisfying learning quality. Particularly, unlike the training at the data center with unconstrained resources and sufficient data [8], [9], distributed devices that participate in FL are typically resource-constrained and heterogeneous in terms of both hardware conditions and data resources, which can affect the learning performance dramatically. For instance, due to the sensor defects and environmental constraints, the mislabeled and non-independent and identically distributed (non-IID) data are often collected at mobile devices, resulting in diverse local learning qualities. However, inclusively aggregating low-quality model updates can deteriorate the global model quality reversely, which has been verified by our field experiments. Therefore, *client selection*, i.e., selecting appropriate mobile devices from candidate clients to participate in

- *Yongheng Deng, Ju Ren, Yuezhi Zhou, and Yaoxue Zhang are with the Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing 100084, P.R. China. E-mail: dyh19@mails.tsinghua.edu.cn, {renju, zhangyx}@tsinghua.edu.cn, zhouyz@mail.tsinghua.edu.cn.*
- *Feng Lyu is with the School of Computer Science and Engineering, Central South University, Changsha 410083, P.R. China. E-mail: fenglyu@csu.edu.cn.*
- *Huaqing Wu and Xuemin Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada. E-mail: {h272wu, sshen}@uwaterloo.ca.*

Manuscript received 15 Mar. 2021; revised 27 Nov. 2021; accepted 1 Dec. 2021. Date of publication 13 Dec. 2021; date of current version 24 Dec. 2021.
This work was supported in part by the National Natural Science Foundation of China under Grants 62002389, 62122095, 62072472 and U19A2067, in part by the Key-Area Research and Development Program of Guangdong Province under Grant 2019B010137005, the National Key R&D Program of China under Grant 2019YFA0706403, the Natural Science Foundation of Hunan Province, China under Grants 2020JJ2050 and 2021JJ20079, 111 Project under Grant B18059, in part by the Young Elite Scientists Sponsorship Program by CAST under Grant YESS20200238 and in part by the Young Talents Plan of Hunan Province of China under Grants 2019RS2001 and 2021RC3004.
(Corresponding author: Feng Lyu.)
Recommended for acceptance by Y. Yang.
Digital Object Identifier no. 10.1109/TPDS.2021.3134647

1045-9219 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

distributed learning, becomes crucial for high-quality federated learning.

Recently, a few existing works have proposed client selection methods for federated learning [10], [11], [12], [13]. Specifically, the scheme in [10] selects clients based on their computational capacities and channel conditions to reduce the model training time of FL. In work [11], Wang *et al.* proposed to exclude clients with irrelevant model updates to reduce the communication overhead. Wang *et al.* proposed a client selection strategy for FL to speed up the convergence of non-IID data training [12]. However, existing client selection schemes mainly focus on clients' computation/communication capacities or local data size/distribution, but the data quality of individual clients and how they affect the learning performance of the global model are rarely considered, which however is of paramount importance for FL performance. Differently, we comprehensively investigate the individual quality of clients including data size, data distribution, and mislabeling issues, and study the client selection problem to optimize the FL performance, which can bridge the gap to tackle the challenges in achieving satisfying distributed learning quality.

In this paper, we consider an FL platform on the data market, where the machine learning model for different intelligent services is submitted to the platform with a budget. There are a set of mobile devices with labeled data samples willing to collaboratively train the model if a certain price can be paid back. The functionality of the FL platform is to select a subset of clients to participate in distributed model training without exceeding the budget, where the learning quality and the claimed price of the individual client are considered simultaneously to optimize the FL performance. However, designing a highly efficient client selection scheme is technically challenging due to the following reasons. First, due to data privacy concerns, the raw data of each client is inaccessible, posing challenges to the data quality characterization of each client. Second, even if the data quality can be achieved via a privacy-preserving manner, the learning quality of individual clients can be affected by multiple factors, such as data size, data quality. It is difficult to quantify these factors by a numerical individual learning quality value due to the complicated training process. In addition, considering the uninterpretable property of model training and model update aggregation, it is also intractable to comprehend how these quality-influencing factors of each client will affect the aggregated global model. Third, there exist a large set of client selection combinations, especially under the large-scale FL scenarios. Consequently, finding the optimal client selection scheme involves a huge searching space, the time complexity of which is non-polynomial and cannot satisfy the real-time requirement.

To this end, we propose *AUCTION*, an Automated and qUality-aware Client selecTION framework to optimize the FL performance with addressing the above challenges. Functionally, *AUCTION* can utilize the current monitoring information of clients and historical model training records to automatically learn client selection policies, based on which the selection decisions can be made in real time under a specific scenario. In particular, after disclosing the significance of data size and data quality on individual learning accuracy by pilot FL experiments, we adopt the data size and data quality of clients as the driving factors to characterize their importance to the global model aggregation. Empowered by the reinforcement learning (RL) technique, *AUCTION* encodes the client selection policy into a neural network as the agent, which takes the data size, data quality, and claimed price of each client as inputs and outputs a subset of clients selected within the budget. Then, the policy network observes the FL performance of the selected clients and gradually optimizes its client selection policy using the policy gradient algorithm. The policy network is based on the encoder-decoder architecture, in which the encoder network employs the attention mechanism to transform the client information inputs into embedding vectors, and the decoder network can make sequential client selection decisions based on the embeddings from the encoder. The merits of the encoder-decoder network design are twofold. First, caused by the dynamics of the FL environment, the number of participating clients can vary with time while the encoder-decoder network can ably adapt to the number variation. Second, sequential decisions are made in the network which can significantly reduce the RL searching space and facilitate the training process of the policy network.

We evaluate *AUCTION* with real-world datasets including MNIST, Fashion-MNIST (FMNIST), CIFAR-10, and CIFAR-100, as well as widely known learning models including Multi-layer Perceptron (MLP), LeNet-5, MobileNet, and ResNet-18, respectively. Extensive experiments are carried out under various FL settings and the results demonstrate the efficacy of *AUCTION*. Particularly, compared to competitive benchmarks, significant performance gains can be achieved by the proposed *AUCTION* when there exist clients with low-quality data samples. Moreover, with our comprehensive evaluations, the advantages of *AUCTION* in terms of robustness adapting to different learning tasks and scalability fitting to various FL settings, are corroborated.

We highlight our major contributions as follows.

- With pilot experiments, we disclose the importance of individual data quality on FL model accuracy, motivated by which we investigate the quality-aware client selection problem for FL services to deal with the defective data issues at distributed clients. The problem is crucial in practical FL scenarios, but to our best knowledge, is rarely seen in the literature.
- We propose *AUCTION*, an RL-based client selection framework, to automatically learn high-efficient and quality-aware client selection policies. In *AUCTION*, we devise a policy network based on the encoder-decoder architecture, which can adapt to dynamic changes in the number of FL clients, and make sequential client selection decisions to reduce the RL searching space significantly.
- We implement *AUCTION* and conduct extensive data-driven experiments to evaluate its performance. Compared with state-of-the-art client selection methods, *AUCTION* can significantly enhance the FL performance for different learning tasks under various FL settings.

The remainder of this paper is organized as follows. We describe the system scenario and give the problem definition

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

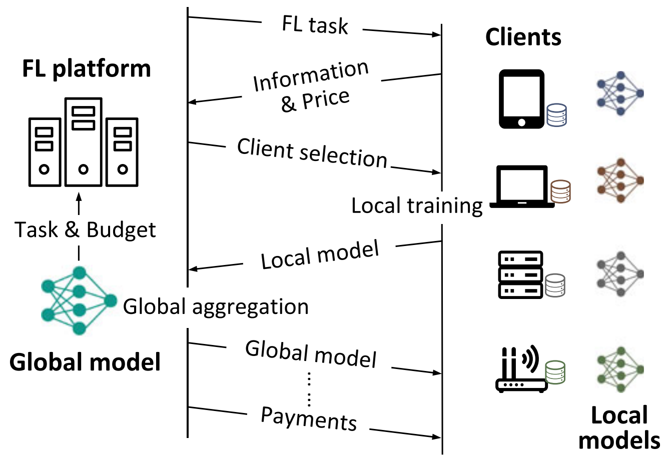


Fig. 1. A typical federated learning services market.

in Section 2. To achieve the solution direction, we conduct data-driven analysis in Section 3. In Section 4, we present the design overview of *AUCTION* with highlighting the design challenges and introducing the RL model for client selection in FL. We elaborate on the design of the policy network in *AUCTION* in Section 5, and carry out extensive experiments to evaluate the performance of *AUCTION* in Section 6. Section 7 surveys the related studies. Finally, Section 8 concludes this paper and directs the future work.

2 System Description and Problem Definition

In this section, we first describe the FL system by detailing the architecture and workflow, and then we define the client selection problem in the system.

2.1 System Description

The mechanism of FL can enable the fusion of distributed data without sacrificing privacy, where the establishment of an FL platform is important to guarantee learning performance. As shown in Fig. 1, in this paper, we focus on a typical FL services market, where there are one FL platform and a set of clients in the alliance. Users can submit FL tasks to the FL platform with a certain budget for recruiting clients to accomplish them. For a given FL task, there exist N clients denoted by $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$, willing to participate in the distributed learning with a claimed price $\{b_1, b_2, \ldots, b_n\}$, and each client C_i has a set of private local data samples $\mathcal{D}_i$ relevant to the FL task. It is worth noting that, the training samples of some clients may be mislabeled or non-IID, which is very common in practice but can affect the learning performance significantly. Therefore, to achieve satisfying FL performance, the platform needs to select an optimal subset of clients from $\mathcal{C}$ within the budget B for the given FL task. The selected clients can collaboratively train the FL model using their local data samples and then receive their declared payments. The system works as follows:

1) *Task initialization*. A learning task is submitted to the FL platform, and there is a limited budget B which can be used to recruit clients to update the parameters w of the global learning model based on their local training results.
2) *Client initialization*. The set of clients $\mathcal{C}$ that are willing to participate in the task, i.e., candidate clients, report their client-side information and prices, which will be used for the client selection in the next step.
3) *Client selection*. The FL platform conducts client selection to choose a subset of participants from the candidate clients, and then delivers the initial global model $\boldsymbol{w}^0$ to the selected participating clients.
4) *Local training*. In each round r, based on the global model $\boldsymbol{w}^r$, each participating client conducts model training individually by using the local data set $\mathcal{D}_i$, the training results of which can be used to update the local model parameters $\boldsymbol{w}_i^r$. Specifically, for each data sample $j \in \mathcal{D}_i$ at the client, we can define a loss function of the global model, denoted by $f_j(\boldsymbol{w}_i^r)$, and the local training process is to minimize the local loss function $F_i(\boldsymbol{w}_i^r)$ on the training data set $\mathcal{D}_i$, which is defined as

$$F_i(\boldsymbol{w}_i^r) = \frac{1}{|\mathcal{D}_i|} \sum_{j \in \mathcal{D}_i} f_j(\boldsymbol{w}_i^r). \tag{1}$$

   At first, the local parameters $\boldsymbol{w}_i^r$ are initialized to the value of $\boldsymbol{w}^r$, and then are gradually updated using the gradient-descent update rule on the local loss function $F_i(\boldsymbol{w}_i^r)$. After local training, the updated local model parameters $\boldsymbol{w}_i^r$ of each participant are uploaded to the FL platform for the global model aggregation.
5) *Global aggregation*. The FL platform aggregates the received local model parameters from participating clients using the classical Federated Averaging algorithm [7]

$$\boldsymbol{w}^{r+1} = \frac{\sum_{i=1}^{N} d_i \boldsymbol{w}_i^r}{\sum_{i=1}^{N} d_i}, \tag{2}$$

   where $d_i = |\mathcal{D}_i|$ is the number of data samples used by participating client C_i for local training. Then, the updated global model parameters $\boldsymbol{w}^{r+1}$ are sent back to all participants for the next round of iteration (proceeding to step 4)).
6) *Learning finishing*. The FL process ends when the test accuracy of the global model reaches a target learning accuracy.

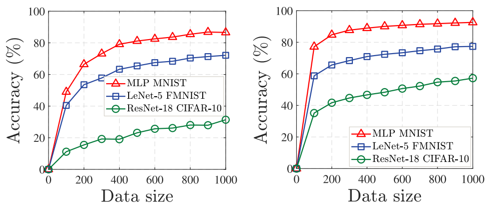
2.2 Problem Definition

In this paper, we focus on the client selection process, in which the FL platform aims to select a subset of clients within the limited budget, such that they can collaboratively train the global model using the local data with maximizing the global model accuracy. Formally, our client selection problem in the FL services market can be cast as follows.

Definition 1 (The Client Selection Problem). *For a given learning task with a set of candidate clients $\mathcal{C}$ and learning budget B, how to select a subset of clients $\mathcal{C}_s \subseteq \mathcal{C}$ within the budget B, such that the accuracy of the global model achieved via their collaborative training, can be maximized?*

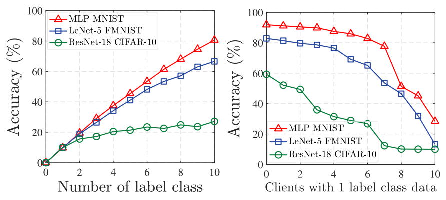
To address the above *Client Selection Problem*, we need to characterize the impact of each candidate client on the global model. However, due to the intricate process of individual

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.



(a) Individual accuracy (b) Global accuracy

Fig. 2. Impact of data size on FL.



(a) Individual accuracy (b) Global accuracy

Fig. 4. Impact of skewed data on FL.

model training and model aggregation, it is infeasible to model this impact mathematically. Therefore, in the next section, we conduct data-driven analysis to explore the factors at each client that can affect the global model quality, to inspire our client selection solution.

3 Data Driven Analysis and Observations

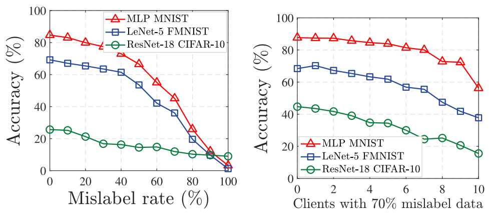
In this section, to explore the influencing factors at participating clients, we build an FL platform with 10 clients and adopt the practical learning models of MLP, LeNet-5 [14], and ResNet-18 [15] which are respectively trained with the well-known image datasets MNIST [16], FMNIST [17], and CIFAR-10 [18], to carry out data-driven experiments.

3.1 Impact of Data Size

In the real FL services market, the participating clients are usually heterogeneous in terms of hardware conditions, which results in different amounts of data that can be used for training the local model. To explore the impact of data size on the global model, we vary the amount of data used for training at each client, and evaluate the test accuracy of the individual client model and the global model. Particularly, Fig. 2a shows the average test accuracy of individual models of 10 clients after 5 epochs, and Fig. 2b shows the test accuracy of the global model after 30 rounds of federated learning. From both figures, we can observe that the data size of participating clients can affect the learning accuracy of the individual model and the global model significantly since both figures have a clear increasing trend with data size. For instance, given the date size of 200 and 600, the individual model accuracy can be increased from 66% to 83%, 54% to 68%, and 15% to 26%, for the learning models of MLP, LeNet-5, and ResNet-18, respectively, and the global model accuracy can be increased from 85% to 91%, 66% to 73%, and 42% to 51%, respectively.

3.2 Impact of Data Quality

The variations in data quality among clients are also common in practical FL systems, where typical cases are the existence of mislabeled and non-IID data samples in client



(a) Individual accuracy (b) Global accuracy

Fig. 3. Impact of mislabeled data on FL.

datasets. To examine the impact of data quality on the global model, we vary the mislabel rate and non-IID level of client data samples, and evaluate the test accuracy of the individual model and global model. In specific, Fig. 3a shows the average test accuracy of individual client models under different mislabel rates after 5 epochs, and we can observe that mislabeled data samples can deteriorate the individual learning performance significantly, especially when the mislabel rate becomes large. For instance, for the learning task of MLP and LeNet-5, when the mislabel rate increases from 40% to 80%, the learning accuracy can drop dramatically from 73% to 26%, and 61% to 20%, respectively. Likewise, Fig. 3b shows the test accuracy of the global model after 30 rounds of federated learning, where we vary the number of clients with 70% of mislabeled data among 10 participating clients. It can be easily seen that the global learning performance also degrades with the mislabeled data samples. Fig. 4a shows the average test accuracy of individual client models under different non-IID levels, where we fix the training data size as 600 and vary the number of data label classes in client datasets. We can observe that the skewed data distributions can degrade the learning performance significantly. For example, for the learning task of MLP MNIST, when the dataset has only one class of label, the accuracy can only reach 10% after 5 epochs, while the accuracy can reach 80% when the data distribution is balanced. The same observation can also be achieved from Fig. 4b, where we vary the number of clients with one label class data among 10 participating clients. Therefore, we can conclude that the data quality of participating clients has a significant impact on the FL performance, and it is of paramount importance to select clients with high-quality (correctly labeled and evenly distributed) data samples to reduce the side impact of poor datasets.

3.3 Unquantifiable Factor Impacts

Given the importance of data size and data quality at clients, it is usually beneficial to model the impacts of these factors on the FL performance, which can be the basis for optimization algorithm design to select appropriate clients. With different data sizes and mislabel rates, Figs. 5 and 6 show the average individual learning accuracy and the global learning accuracy, respectively, and we can have two important observations. First, for each learning model, the relationship between the data size/quality and the final achieved model accuracy seems to be unquantifiable. For instance, for each column or row, the variation is uneven, and the variation pattern is quite different from columns/rows. It means that the created model can hardly capture the impacts of these factors precisely. Second, for different learning models, the impacts of these factors also differentiate from each other.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

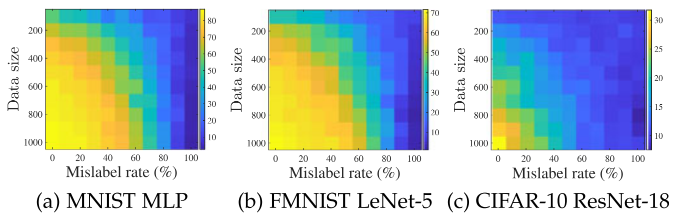


Fig. 5. Individual learning accuracy (%) versus data size and data quality.

For instance, compared to the learning model of MNIST and FMNIST, CIFAR-10 is more sensitive to the data size and data quality since both the individual and global learning accuracy deteriorates dramatically when the data size becomes small or there are mislabeled data samples. Therefore, it is difficult to mathematically create one single model to represent the factor impacts for different learning tasks. This uninterpretable property of impacts poses technical challenges in designing high-quality client selection algorithms, since candidate clients cannot be quantitatively evaluated.

The sophisticated interrelationships among data size, data quality, and acquired model accuracy motivate us to adopt modern machine learning techniques to automatically learn task-specific client selection strategies. RL is an important unsupervised learning method that can automatically make decisions based on the policy learned by the agent interacting with the environment. The agent first observes the environment state and performs an action according to the policy, which is a mapping between states and actions. Performing the action, the agent then receives a reward, based on which the agent optimizes its policy with the aim of maximizing the expected reward. To this end, in what follows, we propose *AUCTION*, which employs the RL framework and a neural network to learn specific characteristics of a given learning task and select optimal participating clients by creating satisfying trade-offs among data size, data quality, and price of each candidate client.

4 Design of *AUCTION*

In this section, we first highlight the technical challenges that *AUCTION* has to deal with. Then, we present the design overview of our proposed *AUCTION*, and finally we detail the RL framework used in the design of *AUCTION*.

4.1 Design Challenges

The design of *AUCTION* faces the following significant challenges:

- *Inaccessible Data Quality:* Due to the data privacy concern, it is usually infeasible to directly access the raw data on each client. Therefore, it is essential to capture the data quality features of each client in a privacy-preserving manner;
- *Variable Number of Candidate Clients:* In the real-world FL services market, the number of clients that are willing to participate in a given FL task is dynamic and unpredictable. Therefore, the offline trained decision model should be adaptive to the dynamic changes in the number of candidate clients, which is challenging since the input of a neural network usually needs to be a fixed size;
- *Exponential Searching Space:* The space of feasible client selection combinations grows exponentially with the number of FL clients. Therefore, there exists a huge searching space for the learning algorithm to make the optimal client selection decision, especially when the number of candidate clients becomes large.

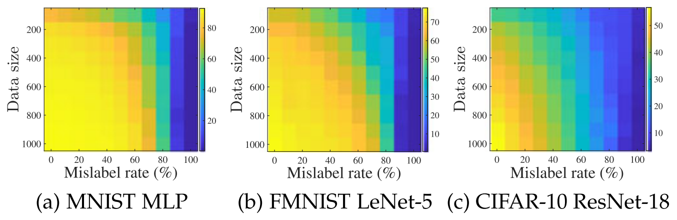


Fig. 6. Global learning accuracy (%) versus data size and data quality.

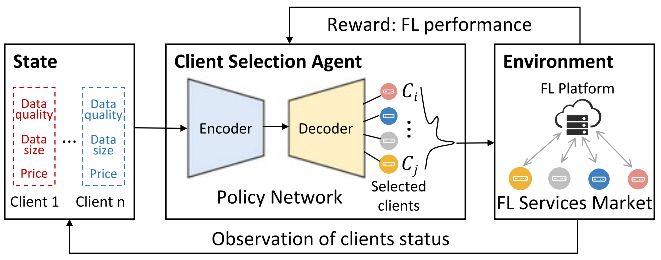


Fig. 7. An overview of the proposed *AUCTION*.

4.2 Design Overview

Inspired by the approaches proposed in [19], [20] that tackle combinatorial optimization problems with RL, we adopt the RL architecture to train the client selection *agent* by interacting with the *environment*. Particularly, as shown in Fig. 7, *AUCTION* first collects the clients' *status* information from the FL services market, and the agent then takes a client selection *action* on the environment. Afterwards, the selected clients train the FL model collaboratively and the training performance is returned to the agent as a *reward*. Finally, *AUCTION* uses this reward to update the agent policy to enhance its performance. The process runs iteratively until the agent can make satisfying client selection decisions in accordance with the dynamic environments.

To improve the intelligence of the client selection agent, we devise a neural network, namely *policy network*, as the client selection agent, which takes the FL clients *state* as inputs and outputs a client selection action. The state captures the status of candidate clients for a given FL task, including the data quality, data size, and the claimed price, while the action determines which clients are selected to participate in the task. To cope with the dynamic changes in the number of candidate clients and reduce the searching space of the RL algorithm, the policy network is designed as an encoder-decoder structure, where the encoder maps the status of each client to vector representations, based on which the decoder then generates an output of selected clients accordingly.

4.3 Reinforcement Learning Model

We devise the following RL model to tackle the client selection problem. The components of the RL model including *state*, *action*, *reward*, and *policy* are defined as follows.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

1) State: The state $\boldsymbol{s} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ consists of the features of all candidate clients for a given learning task. To facilitate high-quality client selection within a limited budget, following the principles in Section 3, we set the features $\boldsymbol{x}_i$ of each client C_i as a 4-dimensional vector represented by $\boldsymbol{x}_i = \{d_i, q_i^l, q_i^d, b_i\}$, where d_i are the number of data samples used for training, q_i^l and q_i^d are the data quality in terms of data labels and data distribution, respectively, and b_i is the required price (i.e., payment) for client C_i to complete the learning task.

It should be noted that, due to privacy issues, it is infeasible to access the raw data of training samples at clients and measure the data quality (e.g., mislabel rate, data distribution) directly. In this paper, we propose a simple but efficient method to characterize the data quality of each client. In particular, the FL platform usually maintains a small set of test data to evaluate the quality of FL models. Therefore, to characterize the data label quality of each client, the FL platform first trains the global model using the test dataset, and then the trained global model is used to evaluate the loss for the data samples in each client dataset. Although the trained model is not sufficient to predict the correct labels of client data samples, we find that there is a strong correlation between the mislabel rate and the test loss. Consequently, we adopt the loss of the global model on the local dataset of each client C_i to represent the data label quality q_i^l. To characterize the skewness of client's data distribution, each candidate client first independently trains the initial global model downloaded from the FL platform using a small subset of its local data, and then commits the local model to the FL platform for evaluation. It is verified that the loss of each local model on the test dataset of the FL platform is closely related to the skewness of client's data distribution, where a more skewed data distribution is reflected by a larger loss value. Therefore, we use the loss of the local model of each client C_i on the test dataset of the FL platform to represent the data distribution quality q_i^d.[1]

2) Action: Given a set of candidate clients $\mathcal{C}$ with size N, the client selection agent needs to select a subset of clients without exceeding the budget B. The full space of client selection actions can be as large as $O(2^N)$, which grows exponentially with the number of candidate clients in the FL services market. The large combinatorial action space can complicate the RL training process dramatically. To deal with it, we employ sequential actions to make client selection decisions one by one. For each sequential action that selects one client from at most N candidate clients, the action space can be reduced to $O(N)$, and a sequence of such actions can finish the client selection process.

3) Reward: The goal of the client selection policy is to make the global model quickly converge to be accurate and reliable via the collaborative training of the selected clients. Therefore, we set the reward r to be

$$r = \sum_{j=0}^{R} \lambda^{acc^j}, \tag{3}$$

1. The additional cost raised by the data quality characterization process is acceptable since only quite a small part of data is used, and this is a one-shot process performed offline before FL.

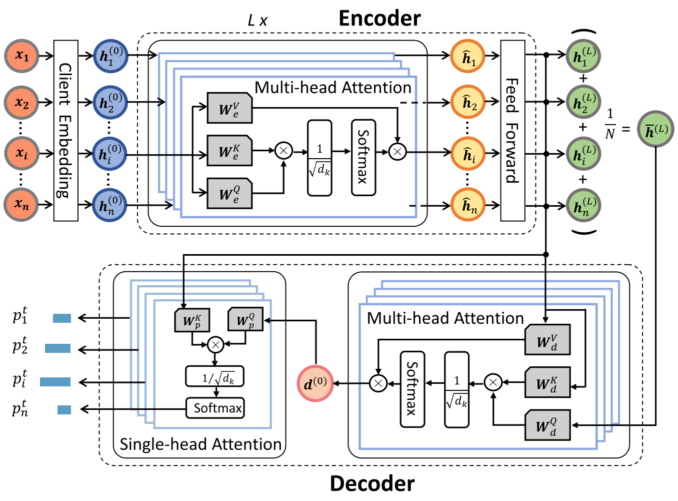


Fig. 8. The architecture of the policy network.

where acc^j is the test accuracy achieved by the global model on the test dataset of the FL platform after round j, λ is a constant that makes reward r grow with the test accuracy ($\lambda = 64$ in our setting), and R is the total number of rounds of the entire FL process.

4) Policy: We define a feasible action of the client selection $\boldsymbol{a} = (a_1, \ldots, a_i, \ldots)$ as a subset of the candidate clients, where $a_i \in \{C_1, C_2, \ldots, C_n\}$ and $\sum_{a_i \in \boldsymbol{a}} b_i \leq B$. The policy network of *AUCTION* defines a stochastic client selection policy $\pi(\boldsymbol{a}|\boldsymbol{s}, B)$ for selecting a feasible action $\boldsymbol{a}$ given a state $\boldsymbol{s}$ and learning budget B. That is, the policy network takes any state and learning budget as inputs and outputs a feasible client selection action.

5 Design of Policy Network

In this section, we elaborate on the design of the policy network in *AUCTION*. Particularly, we first detail the encoder-decoder architecture design of the network, and then describe its training process in the system.

5.1 Encoder-Decoder Policy Network

As shown in Fig. 8, the policy network of *AUCTION* is an attention-based deep neural model that consists of the encoder and decoder networks. In the encoder network, the raw level features of clients are first transformed to vector representations, based on which the decoder network makes sequential decisions on client selections.

5.1.1 *Encoder Network*

In the encoder network, the *Client Embedding layer* first transforms the 4-dimensional input features $\boldsymbol{x}_i$ into initial d_h-dimensional (we set $d_h = 128$ in our implementation) embeddings $\boldsymbol{h}_i^{(0)}$ via a linear projection: $\boldsymbol{h}_i^{(0)} = \boldsymbol{W}^x \boldsymbol{x}_i + \boldsymbol{b}^x$, where $\boldsymbol{W}^x$ and $\boldsymbol{b}^x$ are learnable parameters. Next, the embeddings are updated through L attention layers, each of which produces embeddings $\boldsymbol{h}_i^{(l)}$ for $l \in \{1, \ldots, L\}$.

Following the encoder architecture of Transformer [21], each attention layer consists of a multi-head attention (MHA) layer and a fully connected feed-forward (FF) layer, each of which adds a skip-connection and batch normalization (BN) [20]

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

$$\widehat{\boldsymbol{h}}_i = \mathrm{BN}^l(\boldsymbol{h}_i^{(l-1)} + \mathrm{MHA}_i^l(\boldsymbol{h}_1^{(l-1)}, \ldots, \boldsymbol{h}_n^{(l-1)})), \tag{4}$$

$$\boldsymbol{h}_i^{(l)} = \mathrm{BN}^l(\widehat{\boldsymbol{h}}_i + \mathrm{FF}^l(\widehat{\boldsymbol{h}}_i)). \tag{5}$$

The MHA layer consists of M attention heads running in parallel, and the MHA value for each client C_i is calculated based on the output of each head $\boldsymbol{h}_{im}^a$

$$\mathrm{MHA}_i(\boldsymbol{h}_1, \ldots, \boldsymbol{h}_n) = \sum_{m=1}^{M} \boldsymbol{W}_m^O \boldsymbol{h}_{im}^a, \tag{6}$$

where $\boldsymbol{W}_m^O$ is a learnable parameter matrix. Specifically, given a client embedding $\boldsymbol{h}_i$, the value of $\boldsymbol{h}_{im}^a$ is computed by the self-attention mechanism (we omit the m for readability)

$$\boldsymbol{h}_i^a = \mathrm{Attention}(\boldsymbol{W}_e^Q \boldsymbol{h}_i, \boldsymbol{W}_e^K \boldsymbol{h}_i, \boldsymbol{W}_e^V \boldsymbol{h}_i), \tag{7}$$

$$\mathrm{Attention}(\boldsymbol{q}_i, \boldsymbol{k}_i, \boldsymbol{v}_i) = \sum_{j=1}^{N} \boldsymbol{v}_j \mathrm{Softmax}\left(\frac{\boldsymbol{q}_i^T \boldsymbol{k}_j}{\sqrt{d_k}}\right), \tag{8}$$

$$\mathrm{Softmax}(a_{ij}) = \frac{\exp(a_{ij})}{\sum_{j'=1}^{N} \exp(a_{ij'})}, \tag{9}$$

where $\boldsymbol{W}_e^Q$, $\boldsymbol{W}_e^K$, and $\boldsymbol{W}_e^V$ are learnable parameter matrices, the query $\boldsymbol{q}_i$, key $\boldsymbol{k}_i$, and value v_i for each client C_i are computed by projecting the same embedding $\boldsymbol{h}_i$, and d_k is the dimension of the query/key vector.

The feed-forward (FF) value is calculated by two linear transformations with ReLu activation

$$\mathrm{FF}(\widehat{\boldsymbol{h}}_i) = \mathrm{ReLu}(\widehat{\boldsymbol{h}}_i \boldsymbol{W}_0^F + \boldsymbol{b}_0^F)\boldsymbol{W}_1^F + \boldsymbol{b}_1^F, \tag{10}$$

where $\boldsymbol{W}_0^F$, $\boldsymbol{b}_0^F$, $\boldsymbol{W}_1^F$, and $\boldsymbol{b}_1^F$ are learnable parameters.

5.1.2 Decoder Network

Based on the embeddings from the encoder, the decoder outputs one selected client a_t at the time t, and the sequential client selection results can be achieved until the learning budget is used up. For the decoder architecture, it consists of a multi-head attention layer and a single-head attention layer [20]. The value of the multi-head attention layer $\boldsymbol{d}^{(0)}$ is calculated by the multi-head attention mechanism. Specifically, taking the outputs of the encoder, i.e., the final client embeddings $\boldsymbol{h}_i^{(L)}$ as inputs, the decoder first computes an aggregated embedding $\bar{\boldsymbol{h}}^{(L)} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{h}_i^{(L)}$. To improve the efficiency, we only calculate a single query $\boldsymbol{q}_s$ for each head from the aggregated embedding $\bar{\boldsymbol{h}}^{(L)}$, while computing the key $\boldsymbol{k}_i$ and value $\boldsymbol{v}_i$ from the client embeddings $\boldsymbol{h}_i^{(L)}$

$$\boldsymbol{q}_s = \boldsymbol{W}_d^Q \bar{\boldsymbol{h}}^{(L)}, \quad \boldsymbol{k}_i = \boldsymbol{W}_d^K \boldsymbol{h}_i^{(L)}, \quad \boldsymbol{v}_i = \boldsymbol{W}_d^V \boldsymbol{h}_i^{(L)}, \tag{11}$$

where $\boldsymbol{W}_d^Q$, $\boldsymbol{W}_d^K$, and $\boldsymbol{W}_d^V$ are learnable parameter matrices. To make sure the selected clients are not repeated and the required payment does not exceed the learning budget, we define an attention mask $\eta_i^t \in \{0, 1\}$ at time t for each client $C_i \in \mathcal{C}$. Let $\boldsymbol{a}_{t-1} = (a_1, a_2, \ldots, a_{t-1})$ represents the clients that have been selected at time $t-1$, and B_{t-1} denote the remaining budget, i.e., $B_{t-1} = B - \sum_{C_i \in \boldsymbol{a}_{t-1}} b_i$. We define

$$\eta_i^t = \begin{cases} 1, & \text{if } \; C_i \notin \boldsymbol{a}_{t-1} \; \text{ and } \; b_i \leq B_{t-1}, \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

Then, we compute the weight μ_{sj}, and mask clients that cannot be selected at time t, i.e.,

$$\mu_{sj} = \begin{cases} \frac{\boldsymbol{q}_s^T \boldsymbol{k}_j}{\sqrt{d_k}}, & \text{if } \; \eta_j^t = 1, \\ -\infty, & \text{otherwise.} \end{cases} \tag{13}$$

Finally, the multi-head attention value $\boldsymbol{d}^{(0)}$ can be calculated using Equation (6) based on the output value of each head $\boldsymbol{d}_m^{(0)}$, where

$$\boldsymbol{d}_m^{(0)} = \sum_{j=1}^{N} \boldsymbol{v}_j \mathrm{Softmax}(\mu_{sj}). \tag{14}$$

To compute the probability p_i^t of selecting client C_i at time t, the multi-head attention layer is followed by an attention layer with a single attention head. Specifically, we compute the query $\boldsymbol{q}$ and key $\boldsymbol{k}_i$ from the multi-head attention value $\boldsymbol{d}^{(0)}$ and the client embedding $\boldsymbol{h}_i^{(L)}$, respectively. That is

$$\boldsymbol{q} = \boldsymbol{W}_p^Q \boldsymbol{d}^{(0)}, \quad \boldsymbol{k}_i = \boldsymbol{W}_p^K \boldsymbol{h}_i^{(L)}, \tag{15}$$

where $\boldsymbol{W}_p^Q$ and $\boldsymbol{W}_p^K$ are learnable parameter matrices. Then we compute weight μ_i for each client and clip the result within $[-C, C]$ using $\tanh$ (we set $C = 10$ in our implementation)

$$\mu_i = \begin{cases} C \cdot \tanh(\frac{\boldsymbol{q}^T \boldsymbol{k}_i}{\sqrt{d_k}}), & \text{if } \; \eta_i^t = 1, \\ -\infty, & \text{otherwise.} \end{cases} \tag{16}$$

The probability p_i^t of selecting client C_i at time t can be calculated using a softmax

$$p_i^t = \pi(a_t = C_i | \boldsymbol{s}, \boldsymbol{a}_{1:t-1}, B_{t-1}) = \frac{\exp(\mu_i)}{\sum_{j=1}^{N} \exp(\mu_j)}. \tag{17}$$

Finally, the decoder selects one participating client a_t at time t according to the probabilities $(p_1^t, p_2^t, \ldots, p_n^t)$.

5.2 Training the Policy Network

The parameters of the policy network $\boldsymbol{\theta}$ are the concatenation of the encoder and decoder learnable parameters. The goal of training is to optimize the parameters $\boldsymbol{\theta}$ of the stochastic policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{s}, B)$ with a given input set of clients with state $\boldsymbol{s}$, i.e., assigning larger probabilities to the client selection strategies with high learning performance (i.e., with high reward). To this end, we use the policy-gradient method to optimize the parameters of the policy network as follows. For a given learning task, the client selection agent first observes the state $\boldsymbol{s}$ of the FL services market environment, i.e., the features $\boldsymbol{x}_i = \{d_i, q_i^l, q_i^d, b_i\}$ of each candidate client C_i. Specifically, the data quality features q_i^l and q_i^d can be acquired according to the approach described in Section 4.3 in a privacy-preserving way. The data size feature d_i is available during distributed model training, and the price feature b_i for each client is also public for the FL platform. Then, the client selection agent chooses an action sequence $\boldsymbol{a}$ based on the policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{s}, B)$. For the action conduction, the FL services market will select the clients in $\boldsymbol{a}$ to participate in the model training. Specifically, in each round, each selected client C_i trains the global model with

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

d_i local data samples and commits the model updates to the FL platform for aggregation. As a result, the global model is iteratively updated with the aggregated model updates and the training process terminates after a certain number of rounds. Subsequently, the agent evaluates the global model and acquires a reward r.

Afterwards, the policy network can be updated based on the $(state, action, reward)$ conditions, and the training objective is to maximize the expected reward

$$J(\boldsymbol{\theta}|\boldsymbol{s}) = \mathbb{E}_{\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{s},B)} r(\boldsymbol{a}|\boldsymbol{s}), \tag{18}$$

where $r(\boldsymbol{a}|\boldsymbol{s})$ is the reward after performing action $\boldsymbol{a}$ at state $\boldsymbol{s}$. We adopt the REINFORCE algorithm [22] to optimize J, whose parameters $\boldsymbol{\theta}$ are gradually optimized by gradient descent:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}|\boldsymbol{s}) = \mathbb{E}_{\boldsymbol{a}\sim\pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{s},B)}[(r(\boldsymbol{a}|\boldsymbol{s}) - b(\boldsymbol{s}))\nabla_{\boldsymbol{\theta}}\log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}|\boldsymbol{s},B)], \tag{19}$$

where $b(\boldsymbol{s})$ denotes a baseline function that is independent of $\boldsymbol{a}$ to reduce the gradient variance and hence accelerate the training process. In this paper, we define $b(\boldsymbol{s})$ as the reward of an action from a greedy rollout [20] of the policy defined by the best model so far. That is, the value of $b(\boldsymbol{s})$ is obtained by selecting the action with maximum probability greedily. In this way, the policy model is trained to improve progressively, since $r(\boldsymbol{a}|\boldsymbol{s}) - b(\boldsymbol{s})$ is positive if the reward of the client selection action $\boldsymbol{a}$ is better than the greedy rollout, making actions to be reinforced.

Algorithm 1. REINFORCE-Based Training Algorithm

```
Input: (1) training set $\mathcal{S}$; (2) number of training epochs $E$; (3)
       batch size $B_s$; (4) learning budget $B$.
Output: the parameters $\boldsymbol{\theta}$ of the policy network.
Initialize policy network parameters $\boldsymbol{\theta}$;
for $epoch = 1, \ldots, E$ do
    for each $j \in \{1, \ldots, B_s\}$ do
        $\boldsymbol{s}_j \leftarrow SampleStatus(\mathcal{S})$;
        $\boldsymbol{a}_j \leftarrow SampleRollout(\boldsymbol{s}_j, \pi_{\boldsymbol{\theta}}(\boldsymbol{s}_j, B))$;
        $\boldsymbol{a}_j^b \leftarrow GreedyRollout(\boldsymbol{s}_j, \pi_{\boldsymbol{\theta}}(\boldsymbol{s}_j, B))$;
        Conduct $\boldsymbol{a}_j$ and $\boldsymbol{a}_j^b$, and compute $r(\boldsymbol{a}_j|\boldsymbol{s}_j)$ and $r(\boldsymbol{a}_j^b|\boldsymbol{s}_j)$,
        respectively;
    end
    Compute $\nabla J \leftarrow \frac{1}{B_s}\sum_{j=1}^{B_s}(r(\boldsymbol{a}_j|\boldsymbol{s}_j) - r(\boldsymbol{a}_j^b|\boldsymbol{s}_j))\nabla_{\boldsymbol{\theta}}\log \pi_{\boldsymbol{\theta}}(\boldsymbol{a}_j|\boldsymbol{s}_j, B)$;
    $\boldsymbol{\theta} \leftarrow Adam(\boldsymbol{\theta}, \nabla J)$;
end
return $\boldsymbol{\theta}$
```

The details of our training algorithm are described in Algorithm 1. First, we randomly generate a training set $\mathcal{S}$, where each sample $\boldsymbol{s}_j \in \mathcal{S}$ represents a state of the FL services market, and the features of the candidate clients in $\boldsymbol{s}_j$ are randomly generated from a uniform distribution (details are shown in Section 6). Taking the training set $\mathcal{S}$, the number of training epochs E, the batch size B_s, and the learning budget B as inputs, the algorithm outputs the updated parameters $\boldsymbol{\theta}$ of the policy network after E epochs. In each epoch, the algorithm takes a batch of B_s samples from $\mathcal{S}$, and for each sample $\boldsymbol{s}_j$, the agent first takes samples from policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{s}_j, B)$ to obtain a feasible action $\boldsymbol{a}_j$, and then greedily selects the action $\boldsymbol{a}_j^b$. Afterwards, the FL services market conducts action $\boldsymbol{a}_j$ and $\boldsymbol{a}_j^b$, and computes reward $r(\boldsymbol{a}_j|\boldsymbol{s}_j)$ and $r(\boldsymbol{a}_j^b|\boldsymbol{s}_j)$, respectively. Finally, the algorithm computes the gradient ∇J and updates $\boldsymbol{\theta}$ using the Adam optimizer [23].

6 Performance Evaluation

In this section, we conduct extensive experiments to evaluate the performance of *AUCTION*. Particularly, we first describe the evaluation methodology with experiment setup, learning task setup, hyperparameter settings, and benchmark design. Then, we demonstrate the training process of the client selection agent and carry out the overall performance comparison. Finally, we present the client selection process of *AUCTION*, investigate the impact of the learning budget and examine the scalability of *AUCTION*.

6.1 Evaluation Methodology

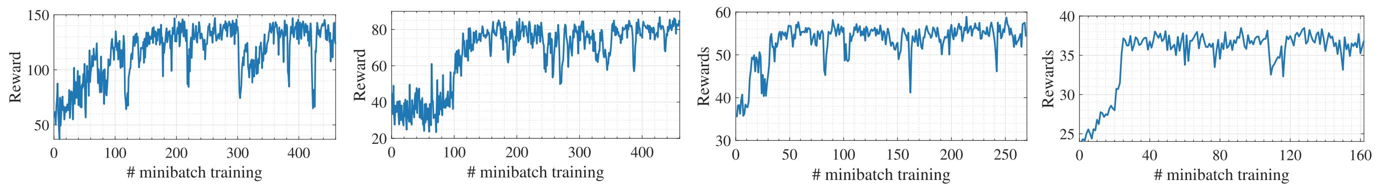
Experiment Setup. We build an experimental FL services market with real-world learning tasks to emulate mislabeled and non-IID scenarios and implement *AUCTION* as the client selection agent for the FL platform. We first emulate the entire FL process on the server under various scenarios where the emulated candidate clients have different training data size, mislabel rate, and data distribution, and train the client selection model of *AUCTION* offline through interacting with the experimental FL services market. Then with the trained client selection model, we evaluate the performance of *AUCTION* with varying the quality-affecting factors of candidate clients.

Learning Tasks. We evaluate the performance of *AUCTION* on four FL learning tasks: 1) *MLP MNIST*. The Multilayer Perceptron (MLP) model[2] is trained with the MNIST dataset [16], which is a dataset of handwritten digits with 10 classes, 60 thousand training data samples, and 10 thousand testing data samples. 2) *LeNet-5 FMNIST*. The LeNet-5 model [14] is trained with the Fashion-MNIST (FMNIST) dataset [17], which is a dataset of Zalando's fashion article images with 10 classes, 60 thousand training data samples, and 10 thousand testing data samples. 3) *MobileNet CIFAR-10*. The MobileNet model [24] is trained with the CIFAR-10 dataset [18], which consists of 50 thousand training images and 10 thousand testing images in 10 classes. 4) *ResNet-18 CIFAR-100*. The ResNet-18 model [15] is trained with the CIFAR-100 dataset, which is similar to the CIFAR-10 dataset, except that it has 100 classes each containing 500 training images and 100 testing images.

Hyperparameters. In the policy network, we use $L = 3$ attention layers in the encoder, each of which consists of a multi-head attention layer with $M = 8$ attention heads and a fully connected feed-forward layer with one 512-dimensional hidden sublayer. Likewise, the multi-head attention layer in the decoder network has $M = 8$ attention heads. In Algorithm 1, we set the batch size $B_s = 8$, and use a learning rate of 10^{-3} for the Adam optimizer. During the training process of the policy network, we randomly generate a training set $\mathcal{S}$ with 3200 training data samples for each learning task. For each sample, the number of candidate clients is fixed,

2. It has two hidden layers with 50 neurons each using ReLu activations, and the output layer uses a softmax function.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.



(a) MLP MNIST (b) LeNet-5 FMNIST (c) MobileNet CIFAR-10 (d) ResNet-18 CIFAR-100

Fig. 9. The training process of the client selection agent.

and the features of clients are randomly generated. Specifically, among the candidate clients, a random 50% of them have mislabeled training data samples, and a random 50% of them have randomly generated non-IID data distribution. Besides, for the *MLP MNIST* and *LeNet-5 FMNIST* tasks, the data size of clients is generated uniformly within the range [100,1000], while for the *MobileNet CIFAR-10* and *ResNet-18 CIFAR-100* tasks, the range is [300,3000]. The price of clients is generated uniformly within the range [1,3] for all tasks. Moreover, to achieve the data quality features of each client in a privacy-preserving manner, we set the FL platform to train the global model with the test dataset for 5 epochs to evaluate client local data label quality, and set each client to randomly select 100 data samples from its local data set to train the global model for 5 epochs (30 epochs for the *ResNet-18 CIFAR-100* task) to characterize the skewness of client's data distribution. Note that, in the performance evaluation of *AUCTION*, the features of candidate clients, including training data size, price, mislabel rate, and data distribution, are generated with the same settings, but the testing features combinations are not appeared in the training data set.

Benchmarks. The performance of *AUCTION* is compared with the following four benchmark client selection approaches:

- *Greedy:* It greedily selects clients with high quality and low price based on the value of q_i/b_i, where q_i and b_i are the quality and price of client C_i, respectively. Here, we utilize a heuristic method to characterize the quality of each client, i.e., $q_i = d_i \cdot acc_i$, where d_i is the training data size and acc_i is the test accuracy of the pre-trained local model of each client C_i (each client uses a small subset of local data to train the global model which is committed to the FL platform for evaluation). The greedy-based approach is widely used for quality-aware budgeted incentive mechanisms in recent researches, e.g., [25], [26].
- *POW-D:* It prefers the clients with larger local loss, which is a biased client selection scheme proposed in [27]. In each round, the FL platform sends the current global model to candidate clients which compute and send back their local loss, and then the FL platform selects clients in the order of the local loss values within the learning budget.
- *Random:* It randomly selects a subset of clients within the learning budget B, which is widely used in state-of-the-art researches (e.g., [7], [28], [29]).
- *Price first:* It prioritizes the clients with low prices for a given learning task, the aim of which is to select as many clients as possible within the limited budget.

6.2 The Agent Training Process

Fig. 9 showsthe training process of the client selection agent on four FL tasks, where the number of candidate clients is fixed to 50 and the learning budget B of each task is set to 10. The experiment settings follow the description in Section 6.1, and the reward refers to the mean reward within a minibatch. In addition, for the *MLP MNIST* task and the *LeNet-5 FMNIST* task, the total number of rounds of the entire emulated FL process R is set to 5, while for the *MobileNet CIFAR-10* task and the *ResNet-18 CIFAR-100* task, we set $R = 20$. We can observe that, for each learning task, the training reward can converge fast to a stable and high value after training a few hundreds of mini-batches. It demonstrates that with the design of *AUCTION*, the agent can learn how to make the optimal client selection strategy intelligently.

6.3 Performance Comparison

With adopting the trained model, we then compare the performance of *AUCTION* with the benchmarks. Particularly, we emulate an FL services market with 20 candidate clients for each learning task, where the features of the candidate clients follow the description in Section 6.1. With a learning budget of 10, Fig. 10 shows the accuracy of each learning task by adopting different client selection strategies. We can observe that for all learning tasks, our proposed *AUCTION* can outperform other benchmarks significantly. Taking the *MobileNet CIFAR-10* task as an example, after 30 learning rounds, the Greedy, POW-D, Random, and Price first mechanisms can only achieve an accuracy score of 37%, 30%, 23%, and 35%, respectively, while *AUCTION* can achieve a score of 49%, improving the performance by 32%, 63%, 113%, and 40%, respectively. Additionally, to demonstrate the robustness of our proposed *AUCTION*, we evaluate its performance in the FL services market with large-scale candidate clients. Fig. 11 shows the learning performance of each task with 50 candidate clients. Likewise, the features of the candidate clients follow the description in Section 6.1, and the learning budget of each task is set to 10. We can observe that *AUCTION* still works well in large-scale client scenarios, outperforming other benchmarks significantly. Moreover, it can be seen that for almost all learning tasks, the Greedy mechanism performs better than other mechanisms, since the effects of data size, data quality, and price are simultaneously considered during client selection. It demonstrates that the factors of data size, data quality, and price are all paramount to the learning performance, and *AUCTION* can ably achieve a satisfying trade-off among them with superior performance.

6.4 Client Selection Process

To better understand how *AUCTION* selects clients to outperform other schemes, taking the *LeNet-5 FMNIST* task as

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

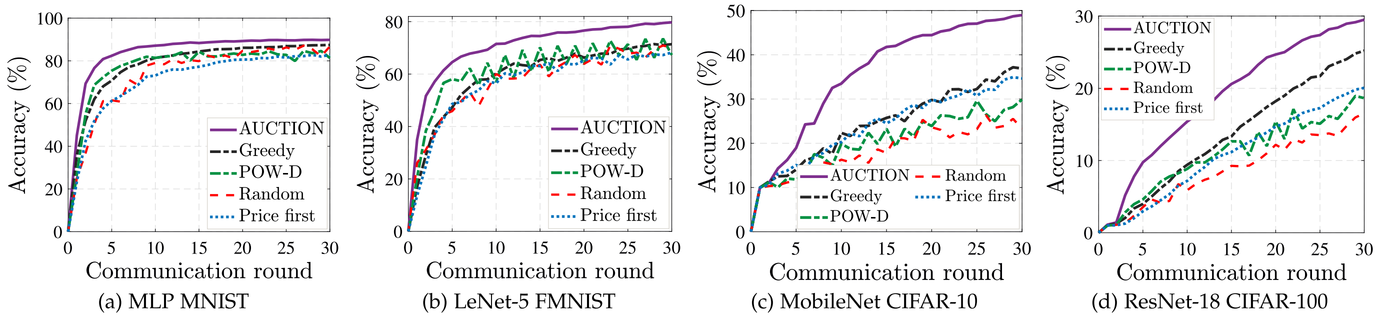


Fig. 10. Performance comparison with 20 candidate clients.

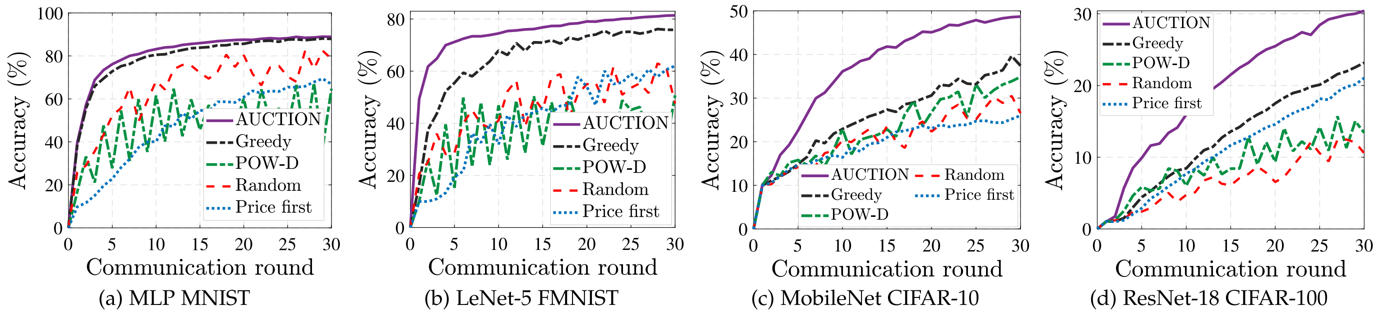


Fig. 11. Performance comparison with 50 candidate clients.

an example, we plot the client selection results of *AUCTION* and Greedy in Fig. 12. In this experiment, there are 50 candidate clients with a learning budget of 20, and the local data samples of each client are correctly labeled but half of the clients have randomly generated non-IID data distributions. The features of the clients selected by *AUCTION* and Greedy are summarized in Table 1. We can observe that, *AUCTION* selects 9 clients to participate in the learning task with achieving a 77.1% accuracy[3] score, while Greedy selects 11 participating clients but inversely achieves a score of 71.4%. Furthermore, the average data size of the clients selected by Greedy is larger than *AUCTION* and the average price is lower, but the average KLD value[4] of the selected clients is relatively larger. It means that the Greedy mechanism prioritizes clients with high data size and low price, while *AUCTION* emphasizes more importance on the data quality, which can achieve better performance. Therefore, as shown in Fig. 12b, the clients with large data size, low price, but relatively skewed data distribution can be improperly selected by the Greedy mechanism. However, the non-IID data can deteriorate the FL performance significantly, resulting in the suboptimal performance of the Greedy mechanism. In contrast, *AUCTION* can intelligently learn how to make tradeoffs among data size, data quality, and price, with selecting appropriate clients through RL training.

Then, to fairly compare the client selection scheme of *AUCTION* and POW-D, we set the price of each client to be the same as 2, and then plot their client selection results in Fig. 13 and summarize the features of their selected clients in Table 2. Likewise, there are 50 candidate clients, half of which have mislabeled data samples and half of which have non-IID data distributions. We can observe that *AUCTION* gives more preference to the clients with large training data size, less mislabeled data, and small KLD values, while POW-D priorities the clients with large training data size and high mislabel rate. The reason lies in that larger training data size and higher mislabel rate can result in larger local loss, while POW-D prefers clients with larger local loss. However, mislabeled and skewed data can degrade the FL performance significantly, and that is why *AUCTION* is able to achieve 71.4% accuracy while POW-D can only achieve an accuracy of 43.7%.

6.5 Impact of Budget

In this subsection, we further explore the impact of the learning budget. With 40 available candidate clients, Fig. 14 shows the performance of the LeNet-5 FMNIST task when there are different learning budgets. We can make the following two major statements. First, our proposed *AUCTION* can outperform other benchmarks in all settings. For

3. The average accuracy of 30 FL rounds.
4. We use the Kullback–Leibler divergence (KLD) between client's local data distribution and the uniform distribution to reflect the non-IID degree of each client, where the KLD values are larger for more skewed data distributions.

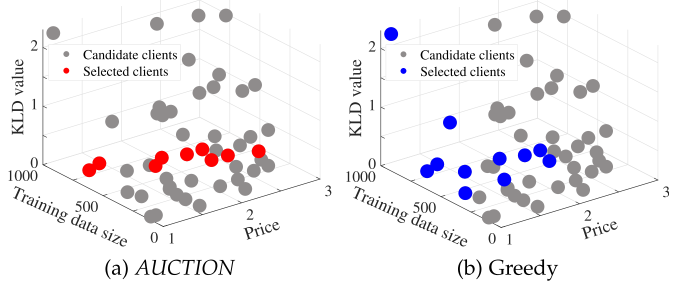


Fig. 12. The client selection results of *AUCTION* and *Greedy* without mislabeled samples.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

TABLE 1
Features of Selected Clients: *AUCTION* and *Greedy*

Selected Clients	*AUCTION*	Greedy
Number	9	11
Avg. data size	649	688
Avg. KLD value	0	0.27
Avg. price	2.17	1.76
Avg. accuracy	77.1%	71.4%

TABLE 2
Features of Selected Clients: *AUCTION* and *POW-D*

Selected Clients	*AUCTION*	POW-D
Number	10	10
Avg. data size	608	796
Avg. mislabel rate	4.4%	40.4%
Avg. KLD value	0.09	0.99
Avg. accuracy	71.4%	43.7%

instance, when the learning budget is 5, after 30 rounds, *AUCTION* can achieve an accuracy of 81%, but the achieved values are less than 70% for other benchmarks. Besides, *AUCTION* converges faster than other benchmarks, since it takes fewer rounds to reach the same target model accuracy, being able to save much training cost and the learning budget. Second, the performance gap between *AUCTION* and other benchmarks becomes more significant when the learning budget is smaller. It happens since the performance of the benchmarks can improve as the learning budget increases, while *AUCTION* can consistently maintain relatively high performance even with a relatively small budget. For example, when the learning budget increases from 5 to 20, after 30 rounds, the performance of the Price first mechanism improves from 57% to 73%, while the performance of *AUCTION* can stabilize at 83%. It means that *AUCTION* can complete federated learning tasks with less budget, which is beneficial for its practical usage.

6.6 Scalability

In this subsection, we demonstrate the scalability of our proposed *AUCTION* by evaluating its online client selection performance when adopting different agents which are trained offline with a various number of emulated candidate clients. Fig. 15 shows the average accuracy of the LeNet-5 FMNIST task for 30 rounds of training when using different client selection models including our proposed *AUCTION* and other benchmarks, where 'ours-10' represents the *AUCTION* model which is trained offline with 10 emulated candidate clients. Specifically, we train the *AUCTION* agents offline using [10,50] emulated candidate clients respectively, and then evaluate their client selection performance online with a various number of candidate clients ranging from 10 to 50, where the learning budget is set to 10. We can observe that the trained *AUCTION* models can perform well with good scalability when the number of online candidate clients varies. For example, the trained model 'ours-50' can achieve an average

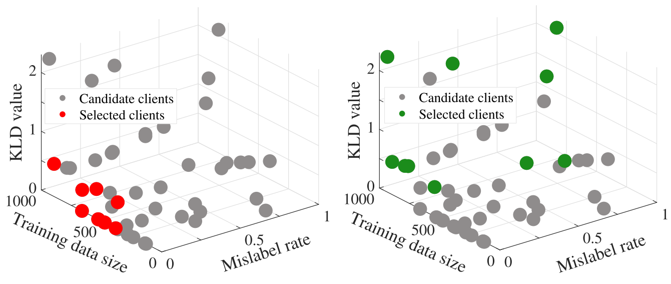


Fig. 13. The client selection results of *AUCTION* and *POW-D* with identical price.

accuracy score of 68.8%, 71.5%, 76.9%, 77.7%, and 75.3%, respectively, when there are 10, 20, 30, 40, and 50 online candidate clients, outperforming other benchmarks significantly. Similar observations can also be obtained for other *AUCTION* models. It demonstrates that *AUCTION* can be excellent in scalability to keep pace with the dynamics of the available candidate clients, and this feature is important for *AUCTION* to be extensively adopted in the practical FL services market.

7 Related Work

Recently, FL has attracted significant research attention, including communication efficiency enhancement [30], [31], [32], privacy protection [33], [34], incentive mechanism [35], [36], usage in applications [37], [38], etc. In this section, we focus on two aspects, i.e., FL performance enhancement and client selection, which are highly related to our study.

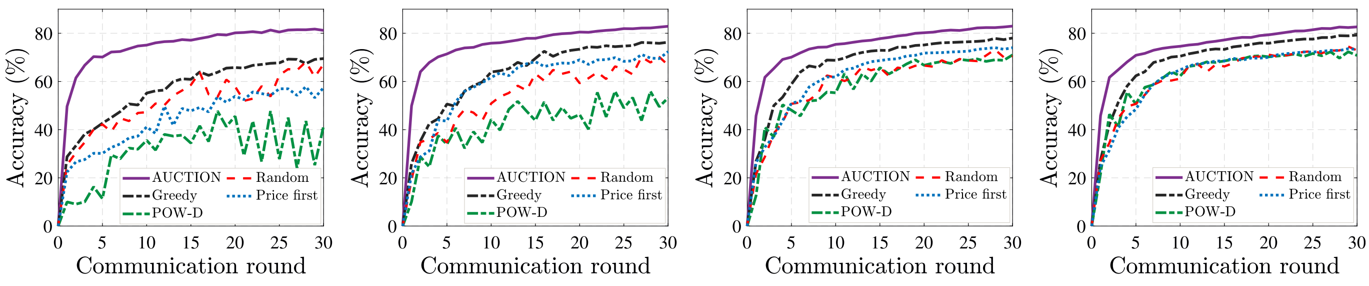
7.1 FL Performance Enhancement

In order to enhance the FL performance, various optimization methods have been developed in the fields of algorithm optimization [28], [39], [40], [41], attack defense [42], [43], [44], [45], etc. For example, Liu *et al.* proposed Momentum Federated Learning that uses Momentum Gradient Descent in the local update step to accelerate the convergence of FL [39]. A new stochastic algorithm SCAFFOLD proposed in [28] employs variance reduction to overcome the client gradient dissimilarity problem caused by the heterogeneous data, which could yield faster convergence. In [40], Li *et al.* proposed FedProx, a federated optimization algorithm, which enables better heterogeneity and learning performance by allowing variable learning workloads across clients. Additionally, Reddi *et al.* proposed a variety of optimization methods, which demonstrated that adaptive optimizers can be powerful tools in improving the performance of FL [41]. In addition to the algorithm optimization for FL, other researches are dedicated to robust FL via defending attacks from malicious clients. For example, Li *et al.* proposed a spectral anomaly detection-based framework for FL, which can detect and remove malicious model updates from adversarial clients to eliminate their negative impact [42]. Moreover, other works such as [46] proposed a control algorithm for FL to balance the trade-off between local update and global aggregation in order to minimize the learning loss.

7.2 FL Client Selection

Client selection is also an effective way to improve the performance of FL. In this regard, various client selection policies [10], [11], [12], [13], [47], [48] have been proposed to optimize the selection number, training time, or communication overhead of FL. For example, Nishio *et al.* proposed a

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.



(a) Budget=5 (b) Budget=10 (c) Budget=15 (d) Budget=20

Fig. 14. Impact of budget.

resource-aware selection scheme, which selects clients based on their computation/communication resources to maximize the number of participants within resource constraints [10]. Xu *et al.* proposed a long-term client selection and bandwidth allocation scheme depending on client's wireless channel condition and remaining battery to maximize the long-term learning performance [49]. To minimize the training latency, Xia *et al.* proposed a multi-armed bandit-based framework for online client scheduling, which can learn to schedule clients online in FL training without wireless channel state information and client statistical characteristics [47]. To reduce the communication overhead of FL, Wang *et al.* proposed a scheme to identify the clients with irrelevant model updates trained over client-specific or biased data, and preclude those clients from uploading their updates [11]. Considering the possible unavailability of clients caused by the computation/communication outages, Mohammed *et al.* devised an online selection algorithm that selects a fixed of R clients from dynamic candidate clients to cope with the stochastic process of incoming candidate clients [13]. Considering the selection fairness of clients, Huang *et al.* designed a fairness-guaranteed algorithm to select the clients within the limited bandwidth which focused on the tradeoff between training efficiency and fairness [48]. However, these algorithms pay less attention to the data quality of clients (e.g., the mislabeling or non-IID issue for distributed data), which is rather important for FL performance.

Empowered by modern machine learning techniques, a few state-of-the-art researches have leveraged RL techniques for FL client selection, which learn to select candidate clients via interacting with the FL environment. For example, Wang *et al.* designed a DRL-based agent that applies the double Deep Q-learning Network (DDQN) algorithm to select participating clients for FL, the goal of which is to counterbalance the bias from different non-IID data while minimizing the communication rounds of FL training [12]. Furthermore, a few incentive mechanisms also proposed to use RL to incentivize and select clients to participate in FL. For instance, Jiao *et al.* proposed a deep RL-based auction mechanism to encourage and select data owners to participate in FL, with considering the data size, data distribution, and wireless channel demand of each client [50].

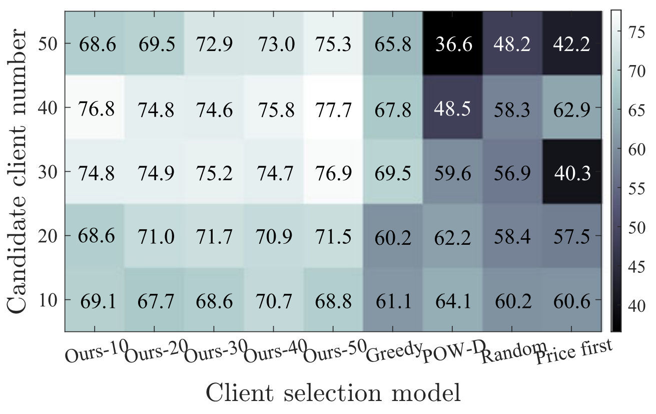


Fig. 15. Accuracy (%) of LeNet-5 FMNIST achieved by adopting different client selection models.

Nevertheless, there is no existing research that investigates the data quality (including the issues of mislabeled and non-IID local data) of individual clients and focuses on the client selection problem to optimize the learning performance. Besides, the existing RL-based approaches cannot cope with the dynamic changes in the number of FL clients, restraining their usage in real-world FL systems. In contrast, we disclose the importance of data quality on FL performance and propose a quality-aware client selection scheme. It takes into account the data size, the issues of data mislabeling and non-IID distribution, as well as the learning budget, simultaneously, which can bridge the gap. Moreover, our proposed *AUCTION* provides a scalable client selection agent, which can automatically learn client selection policies applying to variable client scales and make more efficient and flexible client selection decisions in practical FL systems.

8 Conclusion

In this paper, we have proposed *AUCTION*, an automated and quality-aware client selection framework that employs machine learning techniques to select participants for efficient FL. The proposed *AUCTION* has encoded the client selection policy into an attention-based neural network, and used RL to gradually improve the policy performance via interacting with the FL platform. The client selection policy has taken multiple features of clients into consideration, including the training data size, data quality, and learning price, which are significant to the learning performance. Extensive experiments have been carried out and the results have demonstrated that the proposed AUCTION can be adaptive to different learning tasks with efficacy, robustness, and scalability, which paves the path for its wide application in practical FL systems. For our future work, we will consider the communication cost and computing latency of each client, and integrate these features into *AUCTION* to further expand its client selection functionality.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] N. Cheng *et al.*, "Big data driven vehicular networks," *IEEE Netw.*, vol. 32, no. 6, pp. 160–167, Nov./Dec. 2018.

[2] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Netw.*, vol. 33, no. 5, pp. 156–165, Sep./Oct. 2019.

[3] W. Zhuang, Q. Ye, F. Lyu, N. Cheng, and J. Ren, "SDN/NFV-empowered future IoV with enhanced communication, computing, and caching," *Proc. IEEE*, vol. 108, no. 2, pp. 274–291, Feb. 2020.

[4] F. Lyu *et al.*"LeaD: Large-scale edge cache deployment based on spatio-temporal WiFi traffic statistics," *IEEE Trans. Mobile Comput.*, vol. 20, no. 8, pp. 2607–2623, Aug. 2021.

[5] J. Ren, D. J. Dubois, D. Choffnes, A. M. Mandalari, R. Kolcun, and H. Haddadi, "Information exposure from consumer IoT devices: A multidimensional, network-informed measurement approach," in *Proc. ACM Internet Meas. Conf.*, 2019, pp. 267–279.

[6] J. Ren, D. Zhang, S. He, Y. Zhang, and T. Li, "A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet," *ACM Comput. Surv.*, vol. 52, no. 6, pp. 1–36, 2019.

[7] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[8] M. Li *et al.*"Scaling distributed machine learning with the parameter server," in *Proc. 11th USENIX Conf. Oper. Syst. Des. Implementation*, 2014, pp. 583–598.

[9] Y. Bao, Y. Peng, C. Wu, and Z. Li, "Online job scheduling in distributed machine learning clusters," in *Proc. IEEE Conf. Comput. Commun.*, 2018, pp. 495–503.

[10] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–7.

[11] L. Wang, W. Wang, and B. Li, "CMFL: Mitigating communication overhead for federated learning," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 954–964.

[12] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-IID data with reinforcement learning," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 1698–1707.

[13] I. Mohammed *et al.* "Budgeted online selection of candidate IoT clients to participate in federated learning," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5938–5952, Apr. 2021.

[14] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[16] Y. Lecun, "The MNIST database of handwritten digits," 1998. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[17] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[18] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, Canada, Tech. Rep. TR-2009, 2009.

[19] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural combinatorial optimization with reinforcement learning," 2016, *arXiv:1611.09940*.

[20] W. Kool, H. Van Hoof , and M. Welling, "Attention, learn to solve routing problems!" 2018, *arXiv:1803.08475*.

[21] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[22] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3/4, pp. 229–256, 1992.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4510–4520.

[25] H. Wang, S. Guo, J. Cao, and M. Guo, "MeLoDy: A long-term dynamic quality-aware incentive mechanism for crowdsourcing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 4, pp. 901–914, Apr. 2018.

[26] B. Song, H. Shah-Mansouri , and V. W. S. Wong, "Quality of sensing aware budget feasible mechanism for mobile crowdsensing," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3619–3631, Jun. 2017.

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

[28] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5132–5143.

[29] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," 2019, *arXiv:1907.02189*.

[30] Y. Deng *et al.*, "SHARE: Shaping data distribution at edge for communication-efficient hierarchical federated learning," in *Proc. IEEE 41st Int. Conf. Distrib. Comput. Syst.*, 2021, pp. 24–34.

[31] M. Duan, D. Liu, X. Chen, R. Liu, Y. Tan, and L. Liang, "Self-balancing federated learning with global imbalanced data in mobile systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 1, pp. 59–71, Jan. 2021.

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

[33] C. Niu *et al.*, "Billion-scale federated learning on mobile clients: A submodel design with tunable privacy," in *Proc. 26th Annu. Int. Conf. Mobile Comput. Netw.*, 2020, pp. 1–14.

[34] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *Proc. IEEE Conf. Comput. Commun.*, 2019, pp. 2512–2520.

[35] W. Y. B. Lim *et al.*, "When information freshness meets service latency in federated learning: A task-aware incentive scheme for smart industries," *IEEE Trans. Ind. Informat.*, vol. 18, no. 1, pp. 457–466, Jan. 2022.

[36] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10 700–10 714, Dec. 2019.

[37] J. Feng, C. Rong, F. Sun, D. Guo, and Y. Li, "PMF: A privacy-preserving human mobility prediction framework via federated learning," *Proc. ACM Interactive Mobile Wearable Ubiquitous Technol.*, vol. 4, no. 1, pp. 1–21, 2020.

[38] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for IoT," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 756–767.

[39] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating federated learning via momentum gradient descent," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 8, pp. 1754–1766, Aug. 2020.

[40] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*.

[41] S. Reddi *et al.*, "Adaptive federated optimization," 2020, *arXiv: 2003.00295*.

[42] S. Li, Y. Cheng, W. Wang, Y. Liu, and T. Chen, "Learning to detect malicious clients for robust federated learning," 2020, *arXiv:2002.00211*.

[43] C. Fung, C. J. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," 2018, *arXiv:1808.04866*.

[44] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2938–2948.

[45] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 118–128.

[46] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

[47] W. Xia, T. Q. S. Quek, K. Guo, W. Wen, H. H. Yang, and H. Zhu, "Multi-armed bandit-based client scheduling for federated learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 11, pp. 7108–7123, Nov. 2020.

[48] T. Huang, W. Lin, W. Wu, L. He, K. Li, and A. Y. Zomaya, "An efficiency-boosting client selection scheme for federated learning with fairness guarantee," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1552–1564, Jul. 2021.

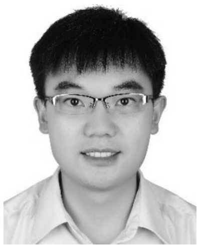
[49] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1188–1200, Feb. 2021.

[50] Y. Jiao, P. Wang, D. Niyato, B. Lin, and D. I. Kim, "Toward an automated auction framework for wireless federated learning services market," *IEEE Trans. Mobile Comput.*, vol. 20, no. 10, pp. 3034–3048, Oct. 2021.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.

Yongheng Deng received the BS degree from Nankai University, Tianjin, China, in 2019, and is currently working toward the PhD degree with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. Her research interests include federated learning, reinforcement learning, edge intelligence, distributed system, and mobile/edge computing.

Feng Lyu (Member, IEEE) received the BS degree in software engineering from Central South University, Changsha, China, in 2013, and the PhD degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2018. During September 2018–December 2019 and October 2016–October 2017, he worked as a postdoctoral fellow and was a visiting PhD student with BBCR Group, Department of Electrical and Computer Engineering, University of Waterloo, Canada. He is currently a professor with the School of Computer Science and Engineering, Central South University, Changsha, China. His research interests include vehicular networks, beyond 5G networks, big data measurement and application design, and edge computing. He is the recipient of the Best Paper Award of IEEE ICC 2019. He currently serves as associate editor of the *IEEE Systems Journal* and leading guest editor of the *Peer-to-Peer Networking and Applications*, and served as TPC members for many international conferences. He is a member of the IEEE Computer Society, Communication Society, and Vehicular Technology Society.



Ju Ren (Senior Member, IEEE) received the BSc, MSc, and PhD degrees in computer science from Central South University, Changsha, China, in 2009, 2012, and 2016, respectively. Currently, he is an associate professor with the Department of Computer Science and Technology, Tsinghua University, China. His research interests include Internet of Things, edge computing, operating system, as well as security and privacy. He currently serves as an associate editor of the *IEEE Transactions on Vehicular Technology* and *Peer-to-Peer Networking and Applications*. He also served as the general co-chair for IEEE BigDataSE'20, TPC co-chair for IEEE BigDataSE'19, a poster co-chair for IEEE MASS'18, a track co-chair for IEEE/CIC ICCC'19, IEEE I-SPAN'18, and VTC'17 Fall, and an active reviewer for more than 20 international journals. He received many best paper awards from IEEE flagship conferences, including IEEE ICC'19 and IEEE HPCC'19, etc., the IEEE TCSC Early Career Researcher Award (2019), and the IEEE ComSoc Asia-Pacific Best Young Researcher Award (2021). He is recognized as a highly cited researcher by Clarivate (2020 & 2021).

Huaqing Wu (Member, IEEE) received the BE and ME degrees from the Beijing University of Posts and Telecommunications, Beijing, China, in 2014 and 2017, respectively, and the PhD degree from the University of Waterloo, Waterloo, Ontario, Canada, in 2021. She received the prestigious Natural Sciences and Engineering Research Council of Canada (NSERC) Postdoctoral Fellowship Award in 2021. She is currently a postdoctoral research fellow with the McMaster University, Ontario, Canada. Her current research interests include vehicular networks with emphasis on edge caching, wireless resource management, space-air-ground integrated networks, and application of artificial intelligence (AI) for wireless networks. She received the Best Paper Award at IEEE GLOBECOM 2018 and *Chinese Journal on Internet of Things* 2020.

Yuezhi Zhou (Senior Member, IEEE) received the PhD degree in computer science and technology from Tsinghua University, Beijing, China, in 2004. He worked as a visiting scientist with the Computer Science Department, Carnegie Mellon University in 2005. He is currently working as a full professor with the Department of Computer Science and Technology, Tsinghua University. He has authored or coauthored more than 100 technical papers in international journals or conferences. His research interests include pervasive computing, distributed systems, and mobile networks. He received the Best Paper Award at IEEE International Conference on Advanced Information Networking and Applications (AINA) 2007 and IEEE International Conference on High Performance Computing and Communications (HPCC) 2014.

Yaoxue Zhang (Senior Member, IEEE) received the BS degree from the Northwest Institute of Telecommunication Engineering, Xi'an, China, in 1982, and the PhD degree in computer networking from Tohoku University, Sendai, Japan, in 1989. Currently, he is a professor with the School of Computer Science and Engineering, Central South University, China, and a professor with the Department of Computer Science and Technology, Tsinghua University, China. His research interests include computer networking, operating systems, ubiquitous/pervasive computing, transparent computing, and big data. He has published more than 200 technical papers in international journals and conferences, as well as nine monographs and textbooks. He is a fellow of the Chinese Academy of Engineering, China, and the editor-in-chief of the *Chinese Journal of Electronics*.

Xuemin Shen (Fellow, IEEE) received the PhD degree in electrical engineering from Rutgers University, New Brunswick, New Jersey, in 1990. He is currently a University professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research interests include network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular ad hoc and sensor networks. He is a registered professional engineer of Ontario, Canada, an Engineering Institute of Canada fellow, a Canadian Academy of Engineering fellow, a Royal Society of Canada fellow, a Chinese Academy of Engineering foreign member, and a distinguished lecturer of the IEEE Vehicular Technology Society and Communications Society. He received the Canadian Award for Telecommunications Research from the Canadian Society of Information Theory (CSIT) in 2021, R. A. Fessenden Award in 2019 from IEEE, Canada, Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) in 2019, James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society (ComSoc), and Technical Recognition Award from Wireless Communications Technical Committee (2019) and AHSN Technical Committee (2013). He has also received the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee chair/co-chair for IEEE Globecom'16, IEEE Infocom'14, IEEE VTC'10 Fall, IEEE Globecom'07, and the chair for the IEEE ComSoc Technical Committee on Wireless Communications. He is the president elect. of the IEEE ComSoc. He was the vice president for technical and educational activities, vice president for publications, member-at-large on the Board of Governors, chair of the Distinguished Lecturer Selection Committee, and member of IEEE Fellow Selection Committee of the ComSoc. He served as the editor-in-chief of *IEEE Internet of Things Journal*, *IEEE Network*, and *IET Communications*.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.

Authorized licensed use limited to: University of Waterloo. Downloaded on February 26,2022 at 19:53:12 UTC from IEEE Xplore. Restrictions apply.