Information Processing and Management 60 (2023) 103211

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/ipm

Leveraging heuristic client selection for enhanced secure federated submodel learning

Panyu Liu [a], Tongqing Zhou [a,*], Zhiping Cai [a,*], Fang Liu [b], Yeting Guo [a]

[a] *College of Computer, National University of Defense Technology, Changsha, Hunan, 410073, PR China*
[b] *School of Design, Hunan University, Changsha, hunan, 410082, PR China*

ARTICLE INFO

Keywords:
Federated learning
Billion-scale deep learning
Recommendation systems
Private set union
Maximum set cover

ABSTRACT

As the number of clients for federated learning (FL) has expanded to the billion level, a new research branch named secure federated submodel learning (SFSL) has emerged. In SFSL, mobile clients only download a tiny ratio of the global model from the coordinator's global. However, SFSL provides little guarantees on the convergence and accuracy performance as the covered items may be highly biased. In this work, we formulate the problem of client selection through optimizing unbiased coverage of item index set for enhancing SFSL performance. We analyze the NP-hardness of this problem and propose a novel heuristic multi-group client selection framework by jointly optimizing index diversity and similarity. Specifically, heuristic exploration on some random client groups are performed progressively for an empirical approximate solution. Meanwhile, private set operations are used to preserve the privacy of participated clients. We implement the proposal by simulating large-scale SFSL application in a lab environment and conduct evaluations on two real-world data-sets. The results demonstrate the performance (w.r.t., accuracy and convergence speed) superiority of our selection algorithm than SFSL. The proposal is also shown to yield significant computation advantage with similar communication performance as SFSL.

1. Introduction

Recent years have witnessed the widespread and broad adoption of the federated learning (FL) setting for collaboratively building data-driven machine learning systems (Lim, Luong, Hoang, et al., 2020; Yang, Liu, Chen, et al., 2019). As shown in the left branch of Fig. 1, FL allows distributed users to train the downloaded full model on their sensitive data and to upload their local updates to the cloud for global aggregation, achieving model learning and privacy-preserving simultaneously (Hao, Li, Luo, et al., 2019; Wei, Li, Ding, et al., 2020; Yin, Zhu, & Hu, 2021). Such a paradigm has successfully supported many applications, in particular the recommendation systems for shopping (Lai, Dai, Zhu et al., 2021; Tran, Bao, Zomaya, et al., 2019), traveling (Guo, Liu, Cai, et al., 2021), news (Qi, Wu, Wu, et al., 2020), etc.

However, the traditional FL design is ill-suited for large-scale distributed training scenarios where there may be billions of users and millions of goods (a.k.a. products or items). For example, as reported in the recommendation practice of Taobao (owned by Alibaba), the global model can be 134 GB when the embedding matrix dimension is 18 for 2 billion goods. Performing local training and iteratively local–global model transmission on such a scale of data is unacceptable. Secure federated submodel learning (SFSL) is thus proposed to relieve this tension (Niu, Wu, Tang, et al., 2020). The basic idea is that, for a specific user (client in FL), only the model parameters (e.g., for inferring sports news preference) related to its local data (e.g., clicking a series of news on sports

* Corresponding authors.
E-mail addresses: zhoutongqing@nudt.edu.cn (T. Zhou), zpcai@nudt.edu.cn (Z. Cai).

https://doi.org/10.1016/j.ipm.2022.103211
Received 30 June 2022; Received in revised form 21 November 2022; Accepted 24 November 2022
Available online 27 December 2022

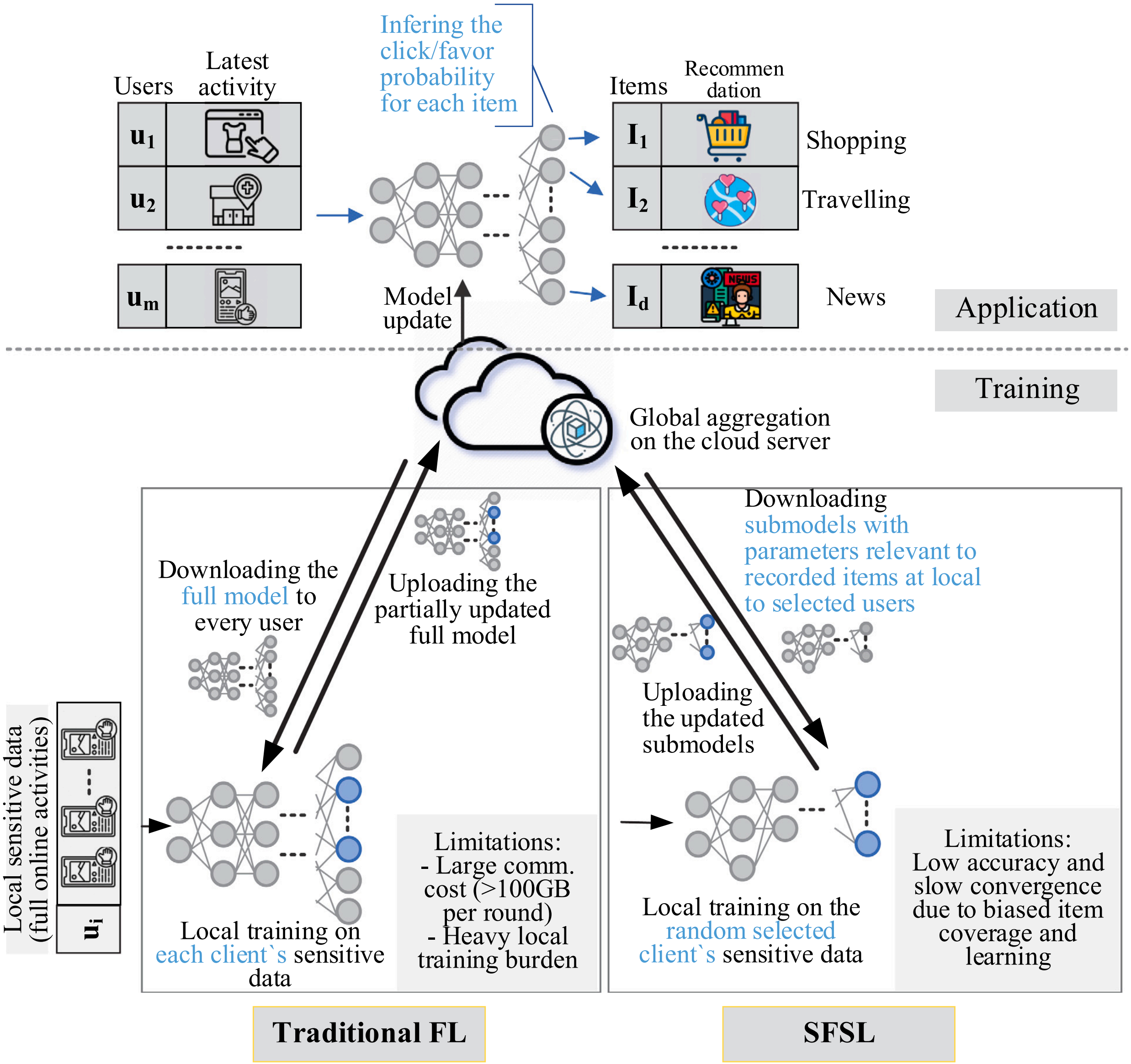

Users
Latest activity
u1
u2
um
Infering the click/favor probability for each item
Model update
Items
Recommendation
I1
I2
Id
Shopping
Travelling
News
Application
Training
Global aggregation on the cloud server
Downloading the full model to every user
Uploading the partially updated full model
Local training on each client`s sensitive data
Limitations:
- Large comm. cost (>100GB per round)
- Heavy local training burden
Local sensitive data (full online activities)
ui
Downloading submodels with parameters relevant to recorded items at local to selected users
Uploading the updated submodels
Local training on the random selected client`s sensitive data
Limitations:
Low accuracy and slow convergence due to biased item coverage and learning
Traditional FL
SFSL

Fig. 1. Frameworks of FL and SFSL with typical supported applications. Briefly, SFSL is proposed to mitigate FL's large communication and training costs on billion-scale training tasks. Yet, randomly selecting users for training generally yields incomplete item coverage, thus biased training, which limits model performance and convergence speed.

star) are tuned during local training on the full model. Hence, one can simply extract a submodel from the full model for each client given its local data characteristics and only transmit small submodels (1.99% of the full model's size Niu et al., 2020) for efficiency, as depicted in the right branch of Fig. 1.

As we all know, communication capacity is the main bottleneck of collaborative training in billion-level FL applications. Wireless and end-user Internet connections generally operate at a lower rate than links within or between data centers and can be expensive and unreliable. The main overhead in communication is the transmission of the intermediate parameters. Regarding communication optimization, many works have given enlightening strategies and schemes (Ji, Jiang, Walid, & Li, 2021; Konečnỳ, McMahan, Yu, et al., 2016; Wu, Wu, Lyu et al., 2022). On achieving the same performance, the communication load has a positive linear relationship with the number of communication rounds. Reducing the number of communication rounds by improving the sampling efficiency can significantly reduce the communication load. In addition, in the federal learning scenario involving hundreds of millions of clients, the aggregation capacity of each round is limited, and the number of clients allowed to participate in the communication round in each round is also limited. These considerations have led to great interest in heuristic sampling. In addition, the clients sampling problem caused by super large scale FL is more challenging than the common FL, and the privacy protection requirement in clients is also more prominent. The new challenges cannot be solved simply by increasing computing resources or introducing optimization methods. It needs a new perspective to analyze and design more sound solutions.

Our method is based on the observation that the quality of participating clients is evaluated under the scenario of partial client participation. Selecting clients with high data diversity to participate in aggregation can lead to better training quality, thus speeding up the training process. We claim that some clients will have "more information" updates in any communication round than others. The training program will benefit from taking advantage of this fact and ignoring some worthless updates. In particular, we propose a pre-client sampling scheme to identify the client with the best local information coverage before any given communication training cycle. Our scheme works by maximizing the coverage of sets and then translates into reducing the number of communication wheels. As far as we know, this method has not been considered before.

Since generating a dedicated submodel a user requires its sensitive access histories (i.e., an index set for its related items in the full item set to specify the position of its submodel), SFSL utilizes a **'randomly asking more'** selection strategy to avoid ruining the privacy tenet of FL (Niu et al., 2020). Basically, it allows a user to generate a randomized index set to replace and protect the real index set when requesting a submodel for local training. Furthermore, it employs random sampling when selecting clients in each FL training round without profiling them for merit-aware selection (Zeng, Zhou, Guo, et al., 2021). Such a random selection strategy means that the index sets of the selected clients may have large redundancy. As a result, although designed for model interaction efficiency, **SFSL provides little guarantees on the convergence and accuracy performance as the covered items may be highly biased**. For example, it may involve a large portion of clients who are fond of sports but with little local training on food data, limiting the recommendation accuracy in the latter category.

The client selection/sampling techniques in traditional FL, either contribution-driven (Chen, Horvath, & Richtarik, 2020; Cho, Wang, & Joshi, 2020; Ribero & Vikalo, 2020) or capacity-sensitive (Chai, Ali, Zawad, et al., 2020; Lai, Zhu, Madhyastha et al., 2021; Nishio & Yonetani, 2019), are ineffective in the SFSL context as relying on independent full model evaluation for the merit of each client. For example, the contribution is measured by the parameter differences between the local model and the global model, and then the updates from clients with higher contributions are selected; Clients with faster computation responses are favored during training in Chai et al. (2020) to attain fast updates in each round. However, sending the large full model to each client for evaluation is unaffordable in billion-scale learning scenarios of SFSL. Meanwhile, high contribution and capacity users may yield redundant training on index sets learned multiple times by other clients submodels.

This work is then devoted to studying the client selection challenges and performance gains for SFSL. For this, we first formulate the client selection problem as optimizing the *unbiased coverage* of the selected clients' index sets. That is, we aim to cover as much heterogeneous item knowledge for training during selection. This optimization problem is NP-hard as it proved to be the boolean combination of two traditional NP-hard problems. The privacy of local index sets makes it more complex as we cannot directly assess the merit of a pair of index sets by requiring their actual values.

To relieve this pitfall, we propose a novel framework based on heuristic multi-group selection. Here a group denotes a sampling of a certain number of clients during each training round. Basically, considering the no-privacy situation, we cast the best client subset selection in multiple training rounds as a variant of the secretary problem (Ferguson, 1989). An empirical approximate solution can then be found by performing multiple group sampling and group merit estimations. Further, to attain the estimations privately, we divide a group of clients into small teams and use private set operations to measure inter-team similarity and diversity. We evaluate the proposal with large-scale simulations on real-world datasets (over 50 thousand samples) and compare it with SFSL on model performance and training overheads.

Our contributions are summarized in the following three folds.

- We identify the **limitations of SFSL** on convergence and **formulate the problem** of client selection through optimizing unbiased coverage of the item index set. We analyze the NP-hardness of this problem and elaborate on the complexity incurred by inherent privacy requirements.
- We propose a novel heuristic **multi-group client selection** framework for enhancing SFSL performance. It leverages the private set intersection and union techniques to securely measure a (group) selection's set coverage capacity from the aspects of **diversity and similarity**. By jointly favoring both aspects in each selection round, we present an approximate empirical solution for the problem.
- We implement the proposal by simulating large-scale SFSL application in lab environment and conduct evaluations on three real-world datasets. The results demonstrate the performance (w.r.t., accuracy and convergence speed) superiority of our framework over SFSL. It is also shown to yield significant computation reduction with similar communication overhead as SFSL.

The rest of this paper is organized as follows. The related work is summarized in Section 2. In Section 3, we describe the preliminaries and formally state the client selection problem of SFSL and its challenges. The overview of our system is in Section 4 and the design details of three heuristic sampling algorithms are introduced in Section 5. Section 6 gives the experiments settings and evaluation results. We conclude this work in Section 7.

2. Related work

There are several data/client selection methods proposed for FL (Alain, Lamb, Sankar, et al., 2015; Katharopoulos & Fleuret, 2018; Li, Zhang, Qian, et al., 2019; Zhang, Li, Xiao, et al., 2018). In recent research, new schemes have also been proposed to solve the adoption problems in FL (Fraboni, Vidal, Kameni, & Lorenzi, 2022; Ji et al., 2021; Khodadadian, Sharma, Joshi, & Maguluri, 2022; Luo, Xiao, Wang, Huang, & Tassiulas, 2022). Banabilah, Aloqaily, Alsayed, et al. (2022) give an in-depth overview of FL applications and trends in different application scenarios. It also discuss the future open directions and challenges in FL. Wu, Deng and Li (2022) propose an anomaly detection classification model that incorporates federated learning and mixed Gaussian variational selfencoding networks which can effectively address network attack and sample dissimilarity. In contrast, few works deal with the data selection problem for billion scales FL (Pu, Chen, Yun, et al., 2020; Tuor, Wang, Ko, et al., 2020). As far as we know, the current research work on FL sampling is mainly divided into two fields: the first branch focuses on the intermediate training results (e.g., during every round of training). The second branch concentrates on the non-training intermediate results of clients (e.g., before every round of training).

(1) The first branch sampling methods have been studied extensively in the last few years. Luo et al. (2022) design an adaptive client sampling algorithm that tackles both system and statistical heterogeneity to minimize the wall-clock convergence time. Allen-Zhu, Qu, Richtárik, et al. (2016) propose a novel non-uniform sampling that selects each coordinate with a probability proportional to the square root of its smoothness parameter. Stich, Raj, and Jaggi (2017) propose an efficient approximation of gradient-based sampling, based on safe bounds on the gradient, which is generic and can easily be integrated into existing algorithms. Ribero and Vikalo (2020) model the progression of the global model's weights and then derive an optimal sampling strategy for selecting a subset of clients with significant weight updates. Cho et al. (2020) quantitatively analyze how biased clients sampling affects the convergence speed, and selects the clients with higher local loss as the training clients to achieve faster convergence of the global model. Chen et al. (2020) construct important client selection by limiting the number of clients whose updates are sent back to the cloud server. Only those clients with important updates are allowed to communicate with the server. Although the above work proposes an efficient sampling mechanism, they not only do not consider the problem of client privacy leakage but also do not consider the problem of incompatibility in large-scale scenarios. Their most fatal deficiency is that they are all based on the intermediate results of training, which cannot be satisfied in our application.

(2) The second branch mainly exploits clients statistics, such as label distribution (Zeng et al., 2021), computing capacity (Nishio & Yonetani, 2019), training overhead (Chai et al., 2020) et al. Nishio and Yonetani (2019) actively manage clients according to their computing resource conditions and solve the problem via a deadline-based approach that filters out slowly responding clients. It allows the server to aggregate as many client updates as possible and accelerates the performance improvement of the global model. Chai et al. (2020) propose a layer-based FL clients sampling system, which divides clients into different layers according to their training time and selects clients from the same layer in each round of training. Lai, Zhu, Madhyastha et al. (2021) propose the design of Oort, which gives priority to those clients who have the data that can provide the most significant utility in improving the accuracy of the model and have strong training ability. However, these efforts do not consider maximizing the training efficiency when only part of the client-side data evaluation can be obtained. In our scenario, only a few clients can participate in training every round. The above methods cannot be used to evaluate the quality of all clients.

Limitations. On the one hand, the above two branches' methods are based on evaluating the results of local training, which further needs to send the full global model to each client. Hence, in SFSL, the schemes mentioned above cannot be directly used as distributing the global model goes against its design tenet for efficiency (as depicted in Fig. 1). In addition, evaluating client contributions one by one would violate local privacy laws since each client's data is sensitive. On the other hand, In conventional FL, the existing similarity-based mechanisms are typically divided into two categories: (1) evaluating data contents and (2) evaluating data embedding feature vectors. There are no FL similarity-based sampling schemes that can be directly implemented in SFSL because of higher privacy and security requirements in SFSL. SFSL protocol constraints differ from FL's base setup for similarity calculation methods. On the one hand, clients are not allowed to directly transmit data or verify each other's data under SFSL. On the other hand, all intermediary results exchanges must take place through the honest-but-curious cloud coordinator, which means they will not diverge from the prescribed protocol while being curious about others' secrets and wondering to know everything they can about them.

3. Problem statement

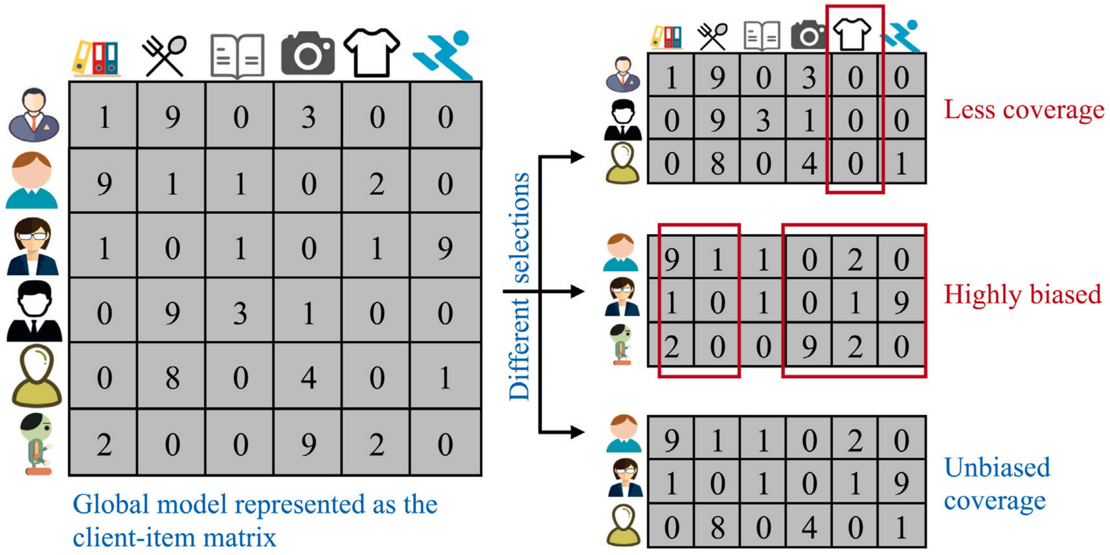
We present some preliminary knowledge and the problem formulation in this section. Some challenges are extracted from the formulation to motivate dedicated designs.

3.1. Preliminaries

This work considers billion-scale FL applications, e.g., recommendation service (Boratto, Fenu, & Marras, 2021; Gómez, Boratto, & Salamó, 2022; Hu, Li, Shi, et al., 2020) in e-commerce platforms like Taobao, Amazon, etc. As shown in Fig. 1, the general goal is to maintain a model that can recommend users items (e.g., products, points of interest, news) according to their latest activities. The underlying privacy requirement is that the comprehensive and true user access histories on these platforms are sensitive and should be kept to users themselves at local (Guo et al., 2021; Lai, Dai, Zhu et al., 2021; Tran et al., 2019), making local training and global aggregation a natural choice. Formally, the full global model is expressed as a two-dimensional matrix $\mathbf{W}_{m\times d}$ (m is the number of users/clients and d is the number of items/products). Let $\mathcal{P} = \{1, 2, \dots, d\}$ denote the full index set of items in $\mathbf{W}$ and $\mathcal{P}^{(i)}$ denote the real index set (i.e., the items that s/he has clicked or accessed) of the ith client. We use C_t to represent clients selected by the cloud server to take part in the tth training round, and use $\mathcal{P}^{(C_t)}$ to denote the union of the clients' index sets. As a convenience, some key notations are presented in Table 1.

Table 1
Frequently used notations.

Notation	Description
T	The total communication rounds
n	The number of clients in each round
K	The number of sampling groups in each round
r	The number of clients in every team
$\mathbf{W}, \mathbf{W}^*$	The global model, the optimal global model
$\mathbf{W}_{\mathcal{P}}$	The submodel w.r.t., index set $\mathcal{P}$
$C_t^k(q)$	The clients set of the qth team in the kth sampling of the tth training round
C_t^*	The optimal client selection in the tth round
$\mathcal{P}^{(i)}$	The real index set of the ith client
$\widehat{\mathcal{P}}^{(i)}$	The randomized index set of the ith client
$\mathcal{P}^{(C)}$	The union index set of client set C
sim_k^j	The similarity of the jth pair of teams in the kth group sampling.
n_w	The size of sliding window



1	9	0	3	0	0
9	1	1	0	2	0
1	0	1	0	1	9
0	9	3	1	0	0
0	8	0	4	0	1
2	0	0	9	2	0

Global model represented as the client-item matrix

Different selections

1	9	0	3	0	0
0	9	3	1	0	0
0	8	0	4	0	1

Less coverage

9	1	1	0	2	0
1	0	1	0	1	9
2	0	0	9	2	0

Highly biased

9	1	1	0	2	0
1	0	1	0	1	9
0	8	0	4	0	1

Unbiased coverage

Fig. 2. A toy example on unbiased coverage selection of clients. Wherein each row indicates the items' access histories of a client, and the corresponding indices of the items construct the index set.

3.2. Problem formulation

With carefully client selection, we essentially try to optimize the performance gain of the global model during each round of local training, which can be represented as:

$$argmax_{C_t^k}(Accuracy(\mathbf{W}_{t+1}|C_t^k) - Accuracy(\mathbf{W}_t|C_{t-1}^*)). \tag{1}$$

In the tth round, each client in C_t performs local training on dedicated submodel of $\mathbf{W}_t$ and the server aggregates their submitted gradients to attain an updated global model $\mathbf{W}_{t+1}$. However, as we have already identified in Section 2, assessing the accuracy gain in Eq. (1) requires sending the large $\mathbf{W}_t$ or dedicated submodel to every client. Both would incur tremendous communication overheads, while the latter choice would unavoidably break the privacy of an individual's index set.

As an alternative, we change to ask what are the most beneficial data for training $\mathbf{W}$. Ideally, from the perspective of centralized model training, samples that cover as many labels evenly are favored as they facilitate comprehensive knowledge gain for the inference tasks. Otherwise, if the covered labels are small, the overall performance is bad; If the covered labels have different amounts, the inference performance on labels would be very different. Here, we introduce the concept of *unbiased coverage* to denote such an expected property. From the view of set operations, *unbiased coverage* optimization can be further decomposed into **maximizing the size of the union set (i.e., diversity) and minimizing the size of the intersection (i.e., similarity)**, as illustratively shown in Fig. 2. Hence, the **SFSL client selection problem** in Eq. (1) transforms to:

$$\begin{aligned} &argmax_{C_t^k}(Diversity(C_t^k) - Similarity(C_t^k)) \\ &\quad(\bigcup_{i=1}^{n}\mathcal{P}^{(i)} - \bigcap_{i=1}^{n}\mathcal{P}^{(i)}),\ i \in C_t^k. \end{aligned} \tag{2}$$

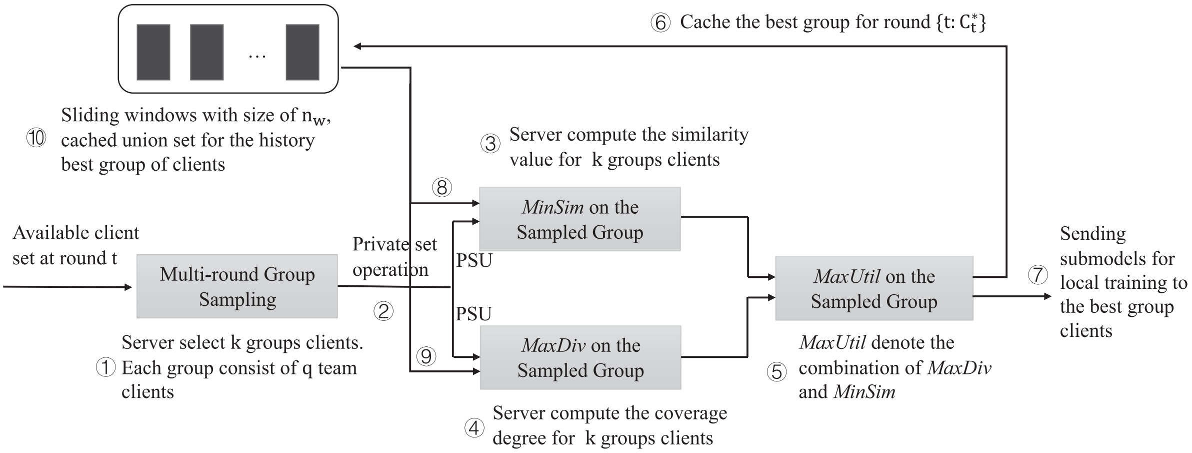

⑥ Cache the best group for round {t: C_t^*}
⑩ Sliding windows with size of n_w, cached union set for the history best group of clients
③ Server compute the similarity value for k groups clients
⑧
MinSim on the Sampled Group
Available client set at round t
Multi-round Group Sampling
Private set operation
PSU
PSU
②
MaxUtil on the Sampled Group
⑦ Sending submodels for local training to the best group clients
① Server select k groups clients. Each group consist of q team clients
⑨
MaxDiv on the Sampled Group
⑤ MaxUtil denote the combination of MaxDiv and MinSim
④ Server compute the coverage degree for k groups clients

Fig. 3. An overview of heuristic multi-group selection framework. For brevity, we use MaxUtil to represent *MaxDiv+MinSim*.

3.3. Challenges analysis

The problem described by Eq. (2) is an NP-hard problem. Specifically, the ($max\ Diversity(C_t^k)$)-problem can be reduced from the classic Maximum Set Coverage (MSC) problem (Chekuri & Kumar, 2004), where a solver tries to find a group of subsets to cover the largest number of distinct elements. Meanwhile, the ($max\ -Similarity(C_t^k)$)-problem can be reduced from the classic Weighted K-clique problem (Vassilevska, 2009), where a subset of nodes in a graph with the minimum inner connections is expected. According to the Boolean Hierarchy theory (Cai, Gundermann, Hartmanis, et al., 1988), their additional combination is still NP-hard.

Another challenge is that measuring the union or intersection of clients' index sets, as requires the real item indices, would implicitly cause privacy leakage to them. How to effectively gauge a selection's properties is important yet non-trivial. For the same reason, the general greedy-based approximate solution is not appropriate to be applied here.

4. Overview

The framework of our proposal is shown in Fig. 3. It mainly contains three key components: *MinSim*, *MaxDiv*, and *MaxUtil*, which denotes *MaxDiv+MinSim*. First of all, the server starts the sampling module, which mainly completes peer sampling K groups of users. Each group is composed of q teams. Then the server runs private set operation to get the index union of each group of users, mainly to get the index union of q teams first and then to get the index union of each group of users through merging. The server will index the similarity and coverage of each set of users, which is realized by *MinSim* and *MaxDiv*, respectively. The similarity value of each group of users is mainly to calculate the average similarity between q teams in each group to approximately replace the similarity of this group of clients. Unlike the similarity calculation, the coverage calculation is simpler because each group of clients' coverage refers to the set's size. In the *MaxUtil* entity, the data quality of each group of clients is evaluated comprehensively by combining similarity and coverage. Then, according to the request of a selected group of clients, the server sends the corresponding sub-model to the group of clients and caches the group's client ID, and the corresponding index set union in the sliding window.

In particular, we adopt a multi-round group sampling strategy to solve the NP-hard problem, akin to the secretary problem. A typical secretary problem has N rankable applicants (groups of clients) arrive in random order. The interviewer (server) tries to make the decision on the acceptance (selection) of an applicant after interviewing (evaluating the utility of unbiased coverage property) immediately. Fortunately, online solutions can find the best secretary with probability 1/e. The server continuously generates a sampling group, calculates the utility of the first N/e groups, and finds the first group with a utility beyond the maximum utility observed so far. This way, the selected group shall have a utility within a 1/e approximate ratio to the global best utility. However, this naive strategy is problematic in the SFSL settings as the number of candidate groups $N = \binom{m}{n}$ can be huge. Considering this, we propose to use multi-round random sampling to attain an *approximate empirical solution*. Specifically, given the active clients, which shall be much smaller than m, in each round, we first perform K rounds of random group sampling according to the budget of the server. Then we use sampling to find the best utility among the k groups and use this sampled client set as the final selection.

In addition, to ensure the privacy of clients' index sets, the server implements a private set protocol for accessing local data. Briefly, the cloud server uses the Private Set Union (PSU) (De Cristofaro, Gasti, & Tsudik, 2012) and Private Set Intersection (PSI) (De Cristofaro & Tsudik, 2010) techniques to calculate the diversity and similarity of a group of clients. The historical best selection in each round is cached, and we use a n_w-size sliding window to evaluate the inter-round diversity and similarity.

We now present the design overview with heuristic selection in Algorithm 1. There is a brief description of SFSL, showcasing the basic SFSL process and retaining the necessary security and privacy properties, following which the heuristic mechanism is discussed. In each communication round of SFSL, the cloud server first selects K groups clients because it is impractical to evaluate one billion clients in one communication round. There is a remarkable search space for finding the optimal set of clients in an undirected graph of one billion clients. To simplify the problem, we divide it into sub-problems to approximate the optimal solution: partitioning into K groups clients. Next, the cloud server launch *MinSim* (Algorithm 2), *MaxDiv* (Algorithm 3) or *MaxDiv+MinSim* (Algorithm 4) to determine which group has the largest difference value among K groups of clients. These group clients are saved as C_t^*, and the union sets associated with C_t^* are also memorized as $\mathcal{P}_t^*$, also Stored on the cloud server cache. The purpose of saving the selected clients and the corresponding union sets for each round is to ensure that we not only consider the coverage diversity of clients within a round but also calculate the coverage diversity between clients among rounds. This ensures that clients' local data overlap as small as possible. In other words, in multiple rounds of client selection, the union set of user local data can cover as many labels evenly, and the global model indexes to be updated are as many as possible.

Regarding the privacy protection of real indexing, a chosen client determines its real index set based on its local data, specifying the index of its required submodel. For example, if the goods IDs of a Taobao user include 1, 2, 4, then he/she requires the first, second, and fourth rows of the embedding matrix for goods IDs, further implying that the real index set should contain 1, 2, 4. Then, the cloud server launches PSU to obtain the union of the chosen clients' real index sets while keeping each client's real index set secret. The union is delivered to live clients to generate randomized index sets with customized LDP guarantees against the cloud server. Each live client will use its randomized index set, rather than its real index set, to download and securely upload the submodel update. Upon receiving the randomized index set from a client, the cloud server stores it for later usage and returns the corresponding submodel and training hyperparameters to the client.

Depending on the intersection of the real and randomized index sets (i.e., the succinct index set), the client extracts a succinct submodel and prepares involved data as the succinct training set. For example, if a Taobao user's real index set is 1, 2, 4, and his/her randomized index set is 2, 4, 6, 9, he/she receives a submodel with the row indices 2, 4, 6, 9 from the cloud server, but just needs to train the succinct submodel with the row indices 2, 4 over his/her local data involving the goods IDs 2, 4. After training under the preset hyperparameters, the client obtains the update of the succinct submodel. Then, it prepares the submodel update to be uploaded with the randomized index set by adding the update of the succinct submodel to the rows with the succinct indices and padding zero vectors to the other rows.

$$\mathbf{W}_j^{(t+1)} = \mathbf{W}_j^{(t)} + \frac{\sum_{i\in C_t^*} \nabla \mathbf{W}_j^{(t+1)}(i)}{\sum_{i\in C_t^*} x_j^i},\ j \in \mathcal{P}^{C_t^*} \tag{3}$$

$$\mathbf{W}^{(t+1)}_{\widehat{\mathcal{P}}^{(i)}}(i) = \mathbf{W}^{(t)}_{\widehat{\mathcal{P}}^{(i)}}(i) - \sigma \cdot \nabla \mathbf{W}^{(t)}_{\widehat{\mathcal{P}}^{(i)}}(i) \tag{4}$$

Algorithm 1 Client selection with heuristic scheme

```
Input: $K$, $C_t^k$, $T$, $n$
Output: $W_*$
 1: Initializes $W$ randomly
 2: for each communication round t = 1, 2,…,$T$ do
 3:   for k = 1, 2,…,$K$ do
 4:     Randomly select $n$ clients to get $C_t^k$
 5:     Use the PSU technique to attain $\mathcal{P}^{(C_t^k)}$
 6:   end for
 7:   Perform $Algorithm$ 2, $Algorithm$ 3, and $Algorithm$ 4, and obtain $C_t^*$ and the corresponding index set $\mathcal{P}^{(C_t^*)}$, and cached in server respectively.
 8:   Deliver the union $\mathcal{P}^{(C_t^*)}$ to every client in $C_t^*$
 9:   /*Client $i's$ local process (Lines 10–13)*/
10:   Determine its randomized index set $\widehat{\mathcal{P}}^{(i)}$ based on the local data and $\mathcal{P}^{(C_t^*)}$
11:   Uses $\widehat{\mathcal{P}}^{(i)}$ to download the submodel $W^{(t)}_{\widehat{\mathcal{P}}^{(i)}}$ from the server
12:   Perform local training and calculate the update using Eq. (4)
13:   Upload $\nabla W^{(t+1)}_{\widehat{\mathcal{P}}^{(i)}}(i)$ to the cloud server for securely aggregation
14:   for every index $j \in \mathcal{P}^{(C_t^*)}$ do
15:     Count the number of clients that have index $j$ as $\sum_{i\in C_t^*} x_j^i$ ($x_j^i = 1$ if the i-th client has index $j$)
16:     Obtain the sum of weighted updates $\sum_{i\in C_t^*} \nabla \mathbf{W}_j(i)$ and calculate the global update of $\mathbf{W}_j^{(t+1)}$ with Eq. (3)
17:   end for
18: end for
```

Then, the cloud server initiates the PSU algorithm to obtain every group of clients' union of real index sets while keeping client index sets confidential. Upon receiving the union, every client utilizes it to generate random index sets and launch tunable local differential privacy (LDP). Each client will use these random index sets to download submodels and securely upload updates to submodels.

5. Client selection algorithms

5.1. Similarity minimization

We introduce the *MinSim* module. Similarity-based measurement mechanisms have many applications across research fields, including distance measurement between data sets. It is the most commonly used algorithm in the broad field of artificial intelligence.

$$J(c_i, c_j) = \frac{|c_i \cap c_j|}{|c_i \cup c_j|} \tag{5}$$

In terms of similarity-based applications, the most commonly used solution is the Jaccard coefficient metrics (Niwattanakul, Singthongchai, Naenudorn, et al., 2013), which means the clients' data is processed as a set, and the Eq. (5) is used to compute the similarity of set c_i and set c_j. In this procedure, the most significant step is obtaining the intersection of the data of two clients in a confidential manner. In addition, we turn our attention to PSI, a problem within the broader field of secure computation. The goal of PSI is to design a protocol by which client i and client j obtain the intersection $c_i \cap c_j$, under the following privacy restriction: The protocol must not reveal anything about items' index that is not in the intersection. But in SFSL, we need to protect the items' index of the intersection of two clients themselves, not the items other than the intersection. To solve the privacy of intersection $c_i \cap c_j$, we split a group of clients into $n/r(\geq 2)$ small teams and then determine the similarity between all teams, $\mathcal{P}^{C_t^k(q)}$ represents qth team real index union sets in kth group in round t. This segmentation strategy can also satisfy the following two basic privacy principles:

- Principle 1: The cloud server cannot ascertain that an item in the intersection belongs to some client.
- Principle 2: The cloud server cannot ascertain that an item in the intersection does not belong to some client.

For example, in Taobao e-commerce scenario, if client A's real index set is $\{1, 2, 4\}$, client B's real index set is $\{1, 5, 6\}$, client C's real index set is $\{3, 4, 7\}$, client D's real index set is $\{4, 7, 8\}$. Assuming in tth training round, the total number of groups K is 1 and the total number of teams r is 2, then we can consider A and B as $team_1$ and C and D as $team_2$. Consequently, $team_1 = A \cup B = \{1, 2, 4, 5, 6\}$ and $team_2 = C \cup D = \{3, 4, 7, 8\}$. Therefore, the intersection between $team_1$ and $team_2$ is $\{4\}$. It is easy for the cloud server to know that $\{4\}$ exist in both $team_1$ and $team_2$, but according to principle 1, the cloud server cannot ascertains that $\{4\}$ belongs to client A or client B (resp., client C or client D), and according to principle 2, the cloud server cannot ascertain that $\{4\}$ does not belong to client A or client B (resp., client C or client D).

Furthermore, a relevant scheme in the research field of PSI can be employed to protect the privacy of clients' intersections (Takuma & Yanagisawa, 2013). This scheme ensures only the intersection cardinality is obtained without knowing the detailed items of the intersection. However, due to the complexity of this algorithm and the high communication costs, especially in federated networks, it is impractical to utilize this method in our problem that needs to be solved.

We next show the details of *MinSim* in SFSL, as shown in Algorithm 2. In the following text, the meaning of these notations remains the same. At the initial stage, the cloud server randomly initializes the global model (Line 1). In each communication round, the cloud server selects K group clients, and the cloud server samples the clients of r teams in turn (Line 2) to participate and maintains an up-to-date set of the chosen clients. They are alive throughout the whole round. Next, the cloud server initiates PSU algorithm (as implemented in SFSL) for each team's clients, which determines each team's real index union sets (Line 4). The teams in each group combine according to the 2-combinations approach, e.g., $\binom{n/r}{2}$ (Line 7). By averaging the similarity values of all the combinations of n/r teams, the similarity of one group can be determined (Line 10). It is inappropriate to select the group clients with the smallest similarity as the training clients in each communication round. The smallest group of clients is only a measure of similarity within a round. Without considering the similarity of clients from round to round. In another round, it may appear that most of the clients are the same. Repetitive client selection has a severe negative impact on training effectiveness and efficiency.

To eliminate the problem of repeated selection of some clients, we propose a sliding window mechanism to minimize clients' similarity values in different rounds. n_w represents the size of the sliding window. Obtain the union of the real indexes of clients within the sliding window, then calculate the similarity of all groups of clients in the current round with $[\mathcal{P}^{C^*_{t-w}} : \mathcal{P}^{C^*_t}]$ respectively (Line 12). The clients with the smallest similarity value are selected as participants (Line 13). The *MinSim* algorithm output consists of two parts. One is C_t^* represents one group of clients ID with the least similarity among k groups clients. The other is $\mathcal{P}^{(C_t^*)}$, which is the union set of clients indexes corresponding to C_t^*.

5.2. Diversity maximization

We introduce the last *MaxDiv* module. In the e-commerce recommendation model, a row index of the global model corresponds to a client's item in local data. In order to improve the training efficiency of the global model W, it is essential to ensure that the local data of the selected clients in each round cover as many global model indexes as possible. We design two targets of the *MaxDiv* algorithm as follows:

- Target 1: Local data of clients sampled within and between rounds can cover the maximum number of data items.
- Target 2: An index of the client data cannot be leaked during one or multiple rounds of sampling.

Algorithm 2 The MinSim Algorithm

```
Input: $W, K, C_t^k, T, n$
Output: $C_t^*$ and corresponding $\mathcal{P}^{(C_t^*)}$
1: for k = 1, 2, ..., $K$ do
2:     for each team q = 1, 2, ..., $n/r$ do
3:         Randomly select $r$ clients to form $C_t^k(q)$
4:         Launch the PSU algorithm, obtain the $\mathcal{P}^{(C_t^k(q))}$
5:     end for
6:     $C_t^k \leftarrow \bigcup_{q=1}^{n/r} \mathcal{P}^{(C_t^k(q))}$
7:     for each pair of team $j \in \binom{n/r}{2}$ do
8:         $sim_k^j = J(\mathcal{P}^{C_t^k(q1)}, \mathcal{P}^{C_t^k(q2)}$
9:     end for
10:    $sim_k \leftarrow \sum_{j=1}^{n/r} sim_k^j$
11: end for
12: $sim_w = J([\mathcal{P}^{C^*_{t-w}} : \mathcal{P}^{C^*_t}], \mathcal{P}^{C_t^k})$
13: return $C_t^* = \min\{sim_k + sim_w\}$ and $\mathcal{P}^{(C_t^*)}$
```

In SFSL, the PSU algorithm mainly prepares data for the index set randomization of each client. Before getting into *MaxDiv*, we first briefly review the definition of PSU. The PSU is a special case of secure two-party computation. It allows two parties holding sets X and Y respectively, to compute the union $X \cup Y$ without revealing anything else. Its most important characteristic is that it can obtain the union of multiple users while protecting their privacy. This is consistent with the **Target 2** designing the *MaxDiv* algorithm.

Due to the feasibility of the PSU for SFSL and the objective for client selection, we propose *MaxDiv* as shown in Algorithm 3. Every group of clients in one round is first randomly sampled by the cloud server (Line 2). Then, the cloud server starts the PSU algorithm to obtain the union of each group client's real index of local data (Line 3). Afterward, the cloud server can get the cardinality of every group of clients. Note that, All cardinality values used in this paper are calculated by Min–max normalization. Here, it is not necessary to calculate the 2-combination for **Q** because the greater the cardinality of the union, the more global indexes are trained at one round. Get the union of the real indexes of clients in the sliding window by searching the historical rounds (e.g., $[\mathcal{P}^{C^*_{t-w}} : \mathcal{P}^{C^*_t}]$), then calculate the similarity between $\mathcal{P}^{C_t^k}$ with the history round clients respectively (Line 5), and the group with the smallest similarity value are selected as participants for round t (Line 6). The *MaxDiv* algorithm output also consists of two parts. One is C_t^* represents one group of clients ID with the largest coverage degree among k groups clients. The other is $\mathcal{P}^{(C_t^*)}$, which is the union set of clients indexes corresponding to C_t^*.

Algorithm 3 The MaxDiv Algorithm

```
Input: $W, K, C_t^k, T, n$
Output: $C_t^*$ and corresponding $\mathcal{P}^{(C_t^*)}$
1: for each group client k = 1, 2, ..., $K$ do
2:     Randomly select $n$ clients to form $C_t^k(q)$
3:     Launch the PSU algorithm, obtain the union of real index sets of $\mathcal{P}^{C_t^k}$
4:     Calculate the cardinality of $\mathcal{P}^{C_t^k}$, denoted as $Card_k$
5: end for
6: $sim_w = J([\mathcal{P}^{C^*_{t-w}} : \mathcal{P}^{C^*_t}], \mathcal{P}^{C_t^k}))$
7: return $C_t^* = \max\{Card_k - sim_w\}$ and $\mathcal{P}^{(C_t^*)}$
```

Algorithm 4 The MaxDiv+MinSim Algorithm

```
1: for each group client k = 1, 2, ..., $K$ do
2:     Randomly select $n$ clients to form $C_t^k(q)$
3:     Launch the PSU algorithm, obtain the union of real index sets in $C_t^k$
4:     Calculate $sim_k$ (Refer to lines 7–10 in Algorithm 2) and corresponding coverage cardinality of $\mathcal{P}^{(C_t^k)}$, denoted as $Card_k$
5: end for
6: $sim_w = J([\mathcal{P}^{C^*_{t-w}} : \mathcal{P}^{(C_t^*)}], \mathcal{P}^{(C_t^k)})$
7: $Util_i = Card_k/(sim_k + sim_w)$ $(i \in K, w \in W)$
8: return $C_t^* = \max\{Util_i\}$ $(i \in K)$ and $\mathcal{P}^{(C_t^*)}$
```

Table 2
Statistics of three datasets.

Dataset	# Clients	# Samples	Samples per client
MovieLen	1,971	563,706	285
Taobao	3,349	524,718	157
Amazon	2,806	392,841	140

5.3. *Joint similarity–diversity optimization*

During our experiments, using a similarity minimization-based sampling scheme, we found that it easily tends to select clients with rarer data and corresponding features. On the one hand, these characteristics significantly improve the model's accuracy. On the other hand, clients with rare data have a generally small amount of local data, which means fewer rows of the global model are updated in this round. Alternatively, a coverage maximization-based client sampling scheme can maximize the rows of the global model updating. However, many local data of sampling clients are overlapping, and the features associated with these data are also overlapping. Namely, simply maximizing diversity also fails to achieve the maximum sampling efficiency. It can also slow down the training process. Considering the above factors, it is imperative that we consider the combination of two schemes to maximize the efficiency of global model updates, e.g., striking a balance between maximizing the number of global model update rows and maximizing the inclusion of new local data features. Consequently, we developed a joint scheme named *MaxDiv+MinSim* to ensure high sampling efficiency while achieving the optimal balance. Therefore, we introduce a new index to measure the quality of sampling. It can be expressed as:

$$Util_i = \frac{Card_k}{sim_k + sim_w} \tag{6}$$

sim_k measures the similarity diversity of clients in a group, and sim_w evaluates the inter-round clients' similarity. We use the arithmetic sum of sim_k and sim_w to represent the total similarity diversity of a group of sampled clients. According to the Eq. (6), The group with the largest $Util_i$ can achieve local optimal unbiased coverage. In other words, it maximizes the efficiency of local data feature selection.

6. Experiments

In this section, we conduct extensive experiments to evaluate the effectiveness of our heuristic client selection design on two real-world datasets.

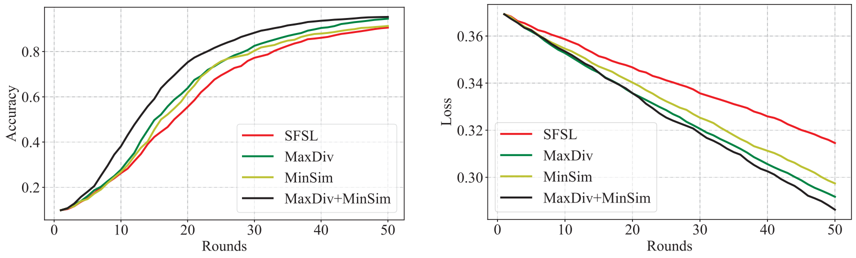
6.1. *Datasets*

We constructed three different types of training datasets for different tasks. One is MovieLens-10M dataset (Harper & Konstan, 2015), the other is public Taobao datasets (Alimama, 2018) from Tianchi Big Data Crowd Intelligence Platform, and the last is Amazon Electronics datasets. We use the public industrial datasets in Alibaba, which were built from the 8 day impressions and clicked logs of Taobao users during May 6–13, 2017. The Taobao dataset contains 26 million records for 1.14 million users. For a certain Taobao user, we leverage his/her click behaviors in the previous 7 days as historical data to predict his/her click and nonclick behaviors in the following 1 day. Amazon datasets are also processed in the same way.

We perform a rating classification task over the MovieLens 10M dataset, a data set commonly used in international publicity. It contains 10 million ratings and 95 thousand tags applied to 10 thousand movies by 71 thousand users of the online movie recommendation service. All selected users had rated at least 20 movies. We randomly selected a certain proportion of data from the original data set as the experimental object. Statistics of the three datasets are as shown in Table 2.

6.2. *Model, parameters and metric*

Our recommendation model is based on the Deep Interest Network (DIN) model (Zhou, Zhu, Song, et al., 2018) proposed by Alibaba, where the embedding dimension is set to 18. For each client's local training, we apply mini-batch stochastic gradient descent as the optimizer initially set the learning rate to 0.01 and further applies exponential decay with the decay rate of 0.999 per communication round. The batch size and the local epoch number are 2 and 5, respectively. For the cloud server's global testing, we set the test batch size to 1,024 and adopt *loss* and *accuracy* as metrics. We focus on the *round-to-loss* performance and *round-to-accuracy* of model training tasks on the testing set. For each experiment, we report the mean value over 5 runs. For *MaxDiv*, *MinSim* and *MaxDiv+MinSim*, parameter K is set to 20. Specifically, the parameter r on *MinSim* is randomly set to 4, i.e., each group has 4 teams, and each team has 5 clients.



(a) Test accuracy on MovieLens

(b) Train losses on MovieLens

Fig. 4. Training loss and accuracy of SFSL, *MinSim*, *MaxDiv* and *MaxDiv+MinSim* on the MovieLens.

6.3. Experimental setup

SFSL is designed to operate in large machine-learning recommendation task deployments with billions of mobile devices. It is infeasible to deploy such a billion-scale system considering it prohibitively expensive. Thus, we conduct an evaluation on SFSL where we extend the SFSL implementation of the random sampling module to fit our requirements. We resort to a Linux workstation with one NVIDIA GeForce RTX 2080 Ti graphics card. The workstation is operated on Ubuntu 20.04 64-bit and equipped with 10 Intel(R) Core(TM) i9-10900X@3.3 GHz and 32 GB memory. We simulate 1917 clients for MovieLens-10M datasets and 3349 clients for the Alibaba dataset.

We implemented the prototypes of our sampling methods and the baseline SFSL in Python 2.7.16. We followed the work of synchronous architecture on top and implemented a communication module between the cloud server and each client with standard socket programming in Niu et al. (2020). We also used TensorFlow 1.12.0 to implement DIN. We mainly used PyCryptodome 3.7.3 to implement the secure aggregation protocol in Bonawitz, Ivanov, Kreuter, et al. (2017). To support modular addition underlying secure aggregation, we performed float-to-integer conversion for each client's submodel or full model update with stochastic quantization (Suresh, Felix, Kumar, et al., 2017), where the quantization level is set to 2^{15}.

6.4. Evaluation analysis

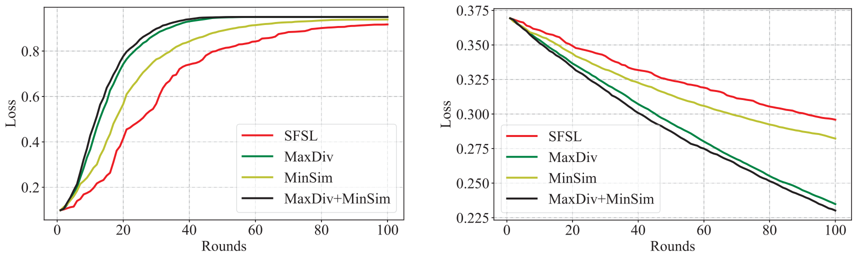
We next analyze our experiment results from the following four aspects.

Model accuracy and convergence. We compare SFSL with *MinSim*, *MaxDiv* and *MaxDiv+MinSim*. For the rating classification on the MovieLens dataset, we plot the test accuracy and training loss in Fig. 4(a) and Fig. 4(b) respectively. As shown in Fig. 8, we observe that both *MaxDiv* and *MinSim* are always superior to random sampling in training loss and test accuracy. In each round of sampling, more information is contained in the local data of clients chosen by *MaxDiv* and *MinSim*, which contributes to the accuracy and convergence speed of the global model. We also noticed that *MaxDiv+MinSim* always outperforms *MinSim* and *MaxDiv*. This is mainly due to the fact that *MaxDiv+MinSim* considers both information redundancy (similarity) and the information volume (coverage). Thus, the quality of clients selected by *MaxDiv+MinSim* is relatively higher. As a result, *MaxDiv+MinSim* can train more data in one round, and the global update rate is faster than random sampling. For the CTR prediction on the Alibaba Taobao dataset, as shown in Fig. 5, *MaxDiv*, *MinaSim* and *MaxDiv+MinSim* still perform better than the random sampling scheme in loss and accuracy. More importantly, our optimal sampling strategy is significantly better than random sampling in large-scale SFSL scenarios. In Fig. 6, Amazon data shows similar results. According to Tables 3–5, our heuristic sampling method can improve model performance in the same number of rounds. That is, the convergence rate of the global model is much faster than that of the random algorithm.

Quantitative results are shown in Figs. 7, 8 and 9, wherein significance markers are computed by Student's t-test. Totally different markers indicate there is a significant difference between two methods on the dataset with confidence level 95%. For example, "a", "ab", "bc" means that there is no significant difference between "a" and "ab", but "a" and "bc" are significantly different. (See Figs. 5 and 6).

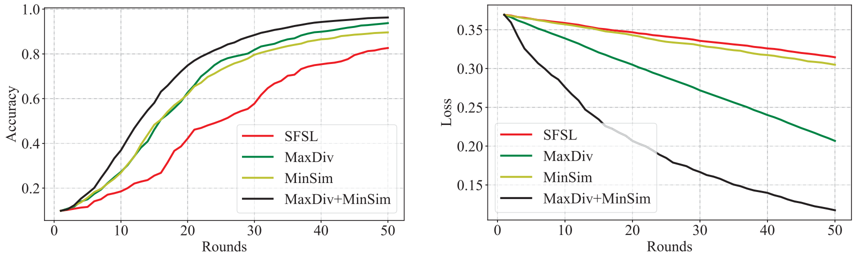
Computation overhead. We first demonstrate the computation overhead of our method and select SFSL as a baseline. The more union data clients are involved in within each group, the more computation resources they consume. Owing to the powerful computing power of the cloud server, it can completely cover all computing expenses. One of the most critical aspects of our method is the separation of samples into multiple groups, which also incurs the cost of repeated calculations for each client. As a consequence, we need to overcome the disadvantage of the high computing cost of the clients to optimize the computing time, e.g., repetition of computation caused by being selected multiple times. On the one hand, although mobile devices are highly parallel in practice, parallel in SFSL refers to a round of random sampling clients will execute in parallel.

We can, however, perform finer-grained parallelism among small teams in our scheme because of the nature of the small teams. On the other hand, in order to further accelerate the speed of local computations on the client side, we allowed each client to perform the computation task only once when it was selected for the first time, and the intermediate results will be stored locally. Then, it is optional to rerun the compute procedure again in the following sampling for clients, but rather to upload the results to the cloud directly.



(a) Test accuracy on Alibaba Taobao (b) Train losses on Alibaba Taobao

Fig. 5. Training loss and accuracy of SFSL, *MinSim*, *MaxDiv* and *MaxDiv+MinSim* on the Alibaba Taobao dataset.



(a) Test accuracy on Amazon (b) Train losses on Amazon

Fig. 6. Training loss and accuracy of SFSL, *MinSim*, *MaxDiv* and *MaxDiv+MinSim* on the Amazon dataset.

Table 3
Taobao, 90% accuracy.

n	20	40	60	80	100
MaxDiv+MinSim	**30**	**32**	**31**	**31**	**33**
MaxDiv	38	37	36	39	35
MinSim	30	48	48	46	47
SFSL	38	42	43	41	42

Table 4
MovieLens, 90% accuracy.

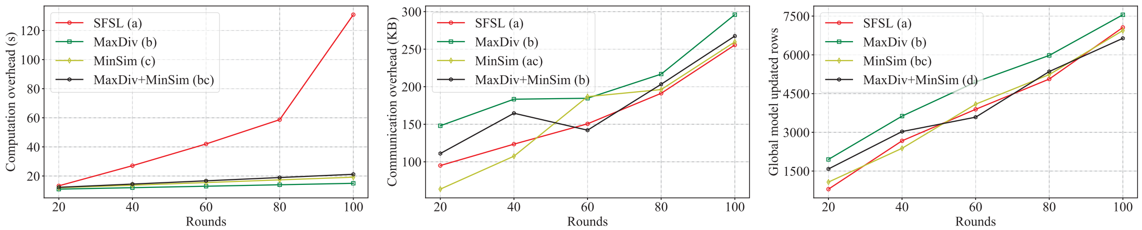
n	20	40	60	80	100
MaxDiv+MinSim	32	**30**	**31**	**32**	**31**
MaxDiv	**31**	37	36	39	35
MinSim	45	48	50	51	49
SFSL	42	45	44	46	45

Table 5
Amazon, 90% accuracy.

n	20	40	60	80	100
MaxDiv+MinSim	31	**30**	**31**	**32**	**30**
MaxDiv	**30**	35	33	32	34
MinSim	43	45	47	50	46
SFSL	40	41	43	42	44

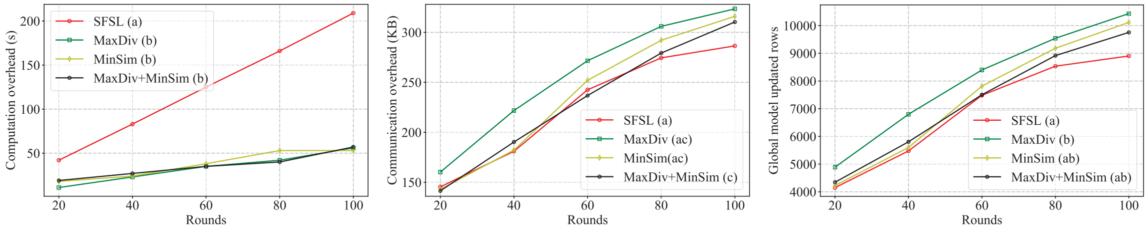
Accordingly, the total computation overload of our sampling in one sampling round comprises the computing time of one team multiplied by the number of groups K, and then the time spent on local training. As the client's other computing time is tiny and does not significantly contribute to the overall computing overhead, it is not included in our computation evaluation. We measured the computational cost based on Taobao and MovieLens datasets, respectively.

We set the number of groups K to 20 and each group consisting of 40 clients, set the size of sliding window W to 3, and set the number of teams r to 8, each team consisting of 5. Training hyper-parameters are set in the same way as previously described. As shown in Fig. 7(a), Fig. 8(a) and Fig. 9(a), we can see that compared with SFSL random sampling, our sampling methods can sharply reduce the computation overhead. The reason can be explained by the team sampling mechanism used in each group selection. A



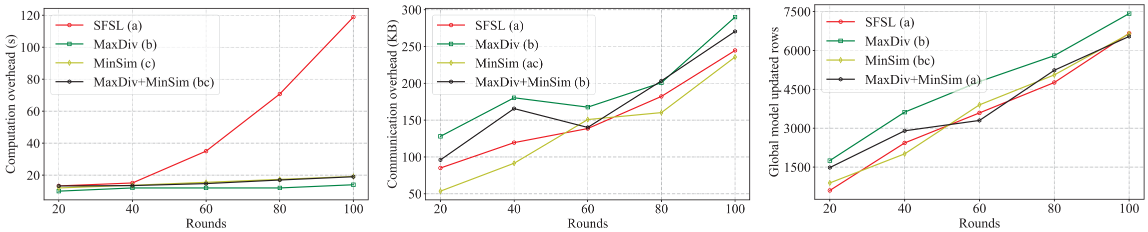
(a) Computation overhead (b) Communication overhead (c) Global updated rows

Fig. 7. Computation overhead, communication overhead and global updated rows per round in our methods and the baseline SFSL for Alibaba Taobao dataset.



(a) Computation overhead (b) Communication overhead (c) Global updated rows

Fig. 8. Computation overhead, communication overhead and global updated rows per round in our methods and the baseline SFSL for MovieLens dataset.



(a) Computation overhead (b) Communication overhead (c) Global updated rows

Fig. 9. Computation overhead, communication overhead and global updated rows per round in our methods and the baseline SFSL for Amazon dataset.

sampling process is performed in parallel for all small teams of each group. For example, when the group consists of 100 clients, our method reduces an average 73.2% of the computation time than SFSL.

Communication overhead. The communication overhead of our method is demonstrated as follows, and SFSL is introduced as a baseline. We observe that there are two aspects that contribute to the greatest cost of client communication overheads: (1) Multiple transmissions of PSU protocol data, (2) Transmission of submodel gradient updated data. The reason is that the clients sampled by *MaxDiv* each round are those client groups whose local data cover the most data. The more union data involved by clients in each group, the larger the transmission space occupied. Compared to the random sampling of SFSL, multiple group sampling result in a substantial increase in transmission overhead for each client. In order to alleviate the transmission pressure of the client, we have optimized the procedure of PSU protocol. When each client executes the PSU protocol for the first time, it is necessary to save the intermediate results locally. The client will be able to transmit the local intermediate results directly if one client is selected again later.

The Parameters to be set here are the same as those for computation overhead evaluation. We tested the overall average clients' transmission overload per training round sampling in Taobao and MovieLens datasets, respectively. The transmission overload at one time is the transmission cost of the PUS for a small team multiplied by the number of groups and plus the transmission cost of gradient aggregation and updates.

As shown in Fig. 7(b), Fig. 8(b) and Fig. 9(b), the main observation from the figures is that the communication overhead per client in all sampling methods increases with the number of chosen clients. On the one hand, the communication complexity grows linearly with the number of chosen clients. On the other hand, it is increasing with union data covered by each group of clients. Additionally, we can see that the overall communication overhead of our method is larger than random sampling in SFSL. As a result, we can conclude that the clients selected by heuristic algorithms are more effective than random sampling on the whole. By sacrificing a small part of transmission space, our proposed algorithm can achieve higher sampling quality, faster convergence, and higher accuracy at the same round compared to SFSL.

Global model updating evaluation. As a final step, we estimate the efficiency of the global model updating based on the chosen clients local data union in each round. To determine the degree of update coverage of the central model after a round of aggregation, we measure the number of rows of the gradient of the updated global model. Accordingly, we counted the total number of rows with a gradient update of nonzero after aggregation, as shown in Fig. 7(c), Fig. 8(c) and Fig. 9(c). The maximum number of updated rows is achieved by *MaxDiv* in three datasets. This is mainly because *MaxDiv* is mainly based on the size of the union of clients index sets. In the judgment process, the group with the largest set size is usually selected as the training client. That is to say, the larger the set size, the more local data to participate in the training, so the more global model updates rows. However, it does not reach the highest accuracy. The reason for the least accuracy for *MaxDiv* is the result of more overlapping feature training points among the clients. In this context, it should be noted that the training effectiveness of the global model is not only affected by the number of updated rows but is also influenced by the characteristics of data sampling.

7. Conclusion and discussions

To the best of our knowledge, we propose the first effective and privacy-preserving heuristic selection solution for SFSL to obtain models with high accuracy and fast convergence speed when there are billion-scale clients. We first formulate the client selection problem using the unbiased coverage concept. By jointly optimizing metric diversity and similarity, we propose a new heuristic framework for multi-group customer selection that is NP-hard. Participating clients' privacy is protected through private set operations (PSI and PSU). Through experiments on two real-world datasets, the results indicate that our algorithm outperforms SFSL in terms of accuracy, and convergence speed. It also offers significant computing benefits while providing comparable communication performance to SFSL.

We finally discuss the limitation of our schemes. As presented by our methods, the performance of *MaxDiv* with our proposed optimal client sampling strategy is in between that with *MaxDiv+MinSim* and *MinSim*. For all datasets, the *MinSim* sampling strategy performs slightly better than but is still competitive with the *MaxDiv* strategy regarding the number of communication rounds. During the experiment, we found that in the sampling process, the number of *MaxDiv* samples is often larger than that of *MinSim*. Although the diversity of data increases, the number is smaller, which also leads to the reduction of training quality, which is also why *MaxDiv+MinSim* has the highest quality. We now discuss the issues as the size of the full index set (which controls the size of the full model) and the total number of clients scale further to billions in practice. First, the baseline SFSL, depending on the random selection method, will be prohibitively inefficient to be applicable. Second, we analyze our heuristic strategies, which relives the dependence on a large number of communication rounds, and involve only a constant number of diverse clients in each round. Thus, no additional overhead will be incurred in our strategies due to the disadvantages caused by random. Another problem is that mobile phones and communication resources are scarce in the actual scenario. It is usually assumed that the mobile device is asleep at night before the above process is executed. This will limit the update frequency of the model. The follow-up optimization strategy of this scheme needs to be further studied, including how to train asynchronously and how to update quickly. We will also further study how to migrate our scheme to the scene of pictures, which supports many applications of computer vision.

CRediT authorship contribution statement

Panyu Liu: Conceptualization, Methodology, Original draft preparation. **Tongqing Zhou:** Conceptualization, Methodology, Original draft preparation, Reviewing and editing. **Zhiping Cai:** Reviewing and editing. **Fang Liu:** Conceptualization, Validation. **Yeting Guo:** Investigation, Writing, Data curation.

Data availability

Data will be made available on request.

Acknowledgment

This work is supported by the National Key Research and Development Program of China (2020YFC2003404), the National Natural Science Foundation of China (Nos. 62072465, 62172155, 62102425, 62102429, U22B2005), the Science and Technology Innovation Program of Hunan Province (Nos. 2022RC3061, 2021RC2071), and the Natural Science Foundation of Hunan Province (No. 2022JJ40564).

References

Alain, G., Lamb, A., Sankar, C., et al. (2015). Variance reduction in SGD by distributed importance sampling. arXiv preprint arXiv:1511.06481.

Alimama (2018). Ad display/click data on taobao.com. URL https://tianchi.aliyun.com/dataset/dataDetail?dataId=56.

Allen-Zhu, Z., Qu, Z., Richtárik, P., et al. (2016). Even faster accelerated coordinate descent using non-uniform sampling. In *International conference on machine learning* (pp. 1110–1119). PMLR.

Banabilah, S., Aloqaily, M., Alsayed, E., et al. (2022). Federated learning review: Fundamentals, enabling technologies, and future applications. *Information Processing & Management*, *59*(6), Article 103061.

Bonawitz, K., Ivanov, V., Kreuter, B., et al. (2017). Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security* (pp. 1175–1191).

Boratto, L., Fenu, G., & Marras, M. (2021). Connecting user and item perspectives in popularity debiasing for collaborative recommendation. *Information Processing & Management*, *58*(1), Article 102387.

Cai, J. r., Gundermann, T., Hartmanis, J., et al. (1988). The boolean hierarchy I: Structural properties. *SIAM Journal on Computing*, *17*(6), 1232–1252.

Chai, Z., Ali, A., Zawad, S., et al. (2020). TiFL: A tier-based federated learning system. In *Proceedings of the 29th international symposium on high-performance parallel and distributed computing* (pp. 125–136).

Chekuri, C., & Kumar, A. (2004). Maximum coverage problem with group budget constraints and applications. In *Approximation, randomization, and combinatorial optimization. Algorithms and techniques*.

Chen, W., Horvath, S., & Richtarik, P. (2020). Optimal client sampling for federated learning. arXiv preprint arXiv:2010.13723.

Cho, Y. J., Wang, J., & Joshi, G. (2020). Client selection in federated learning: Convergence analysis and power-of-choice selection strategies. arXiv preprint arXiv:2010.01243.

De Cristofaro, E., Gasti, P., & Tsudik, G. (2012). Fast and private computation of cardinality of set intersection and union. In *Proc. of ACNS* (pp. 218–231). Springer.

De Cristofaro, E., & Tsudik, G. (2010). Practical private set intersection protocols with linear complexity. In *Proc. of FC* (pp. 143–159). Springer.

Ferguson, T. S. (1989). Who solved the secretary problem? *Statistical Science*, *4*(3), 282–289.

Fraboni, Y., Vidal, R., Kameni, L., & Lorenzi, M. (2022). A general theory for client sampling in federated learning. In *IJCAI 2022-31st international joint conférence on artificial intelligence*.

Gómez, E., Boratto, L., & Salamó, M. (2022). Provider fairness across continents in collaborative recommender systems. *Information Processing & Management*, *59*(1), Article 102719.

Guo, Y., Liu, F., Cai, Z., et al. (2021). PREFER: Point-of-interest recommendation with efficiency and privacy-preservation via federated edge learning. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, *5*(1), 1–25.

Hao, M., Li, H., Luo, X., et al. (2019). Efficient and privacy-enhanced federated learning for industrial artificial intelligence. *IEEE Transactions on Industrial Informatics*, *16*(10), 6532–6542.

Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, *5*(4), 1–19.

Hu, L., Li, C., Shi, C., et al. (2020). Graph neural news recommendation with long-term and short-term interest modeling. *Information Processing & Management*, *57*(2), Article 102142.

Ji, S., Jiang, W., Walid, A., & Li, X. (2021). Dynamic sampling and selective masking for communication-efficient federated learning. *IEEE Intelligent Systems*.

Katharopoulos, A., & Fleuret, F. (2018). Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning* (pp. 2525–2534). PMLR.

Khodadadian, S., Sharma, P., Joshi, G., & Maguluri, S. T. (2022). Federated reinforcement learning: Linear speedup under Markovian sampling. In *International conference on machine learning* (pp. 10997–11057). PMLR.

Konečnỳ, J., McMahan, H. B., Yu, F. X., et al. (2016). Federated learning: Strategies for improving communication efficiency. arXiv preprint arXiv:1610.05492.

Lai, F., Dai, Y., Zhu, X., et al. (2021). FedScale: Benchmarking model and system performance of federated learning. In *Proceedings of the first workshop on systems challenges in reliable and secure federated learning* (pp. 1–3).

Lai, F., Zhu, X., Madhyastha, H. V., et al. (2021). Oort: Efficient federated learning via guided participant selection. In *15th {USENIX} symposium on operating systems design and implementation ({OSDI} 21)* (pp. 19–35).

Li, A., Zhang, L., Qian, J., et al. (2019). TODQA: Efficient task-oriented data quality assessment. In *2019 15th International conference on mobile ad-hoc and sensor networks* (pp. 81–88). IEEE.

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

Luo, B., Xiao, W., Wang, S., Huang, J., & Tassiulas, L. (2022). Tackling system and statistical heterogeneity for federated learning with adaptive client sampling. In *IEEE INFOCOM 2022-IEEE conference on computer communications* (pp. 1739–1748). IEEE.

Nishio, T., & Yonetani, R. (2019). Client selection for federated learning with heterogeneous resources in mobile edge. In *ICC 2019-2019 IEEE international conference on communications* (pp. 1–7). IEEE.

Niu, C., Wu, F., Tang, S., et al. (2020). Billion-scale federated learning on mobile clients: A submodel design with tunable privacy. In *Proceedings of the 26th annual international conference on mobile computing and networking* (pp. 1–14).

Niwattanakul, S., Singthongchai, J., Naenudorn, E., et al. (2013). Using of Jaccard coefficient for keywords similarity. *1*, In *Proc. of multiconference of engineers and computer scientists* (6), (pp. 380–384).

Pu, L., Chen, X., Yun, R., et al. (2020). Cocktail: Cost-efficient and data skew-aware online in-network distributed machine learning for intelligent 5G and beyond. arXiv preprint arXiv:2004.00799.

Qi, T., Wu, F., Wu, C., et al. (2020). Privacy-preserving news recommendation model learning. arXiv preprint arXiv:2003.09592.

Ribero, M., & Vikalo, H. (2020). Communication-efficient federated learning via optimal client sampling. arXiv preprint arXiv:2007.15197.

Stich, S. U., Raj, A., & Jaggi, M. (2017). Safe adaptive importance sampling. *Advances in Neural Information Processing Systems*, *30*.

Suresh, A. T., Felix, X. Y., Kumar, S., et al. (2017). Distributed mean estimation with limited communication. In *International conference on machine learning* (pp. 3329–3337). PMLR.

Takuma, D., & Yanagisawa, H. (2013). Faster upper bounding of intersection sizes. In *Proceedings of the 36th international ACM SIGIR conference on research and development in information retrieval* (pp. 703–712).

Tran, N., Bao, W., Zomaya, A., et al. (2019). Federated learning over wireless networks: Optimization model design and analysis. In *IEEE INFOCOM 2019-IEEE conference on computer communications* (pp. 1387–1395). IEEE.

Tuor, T., Wang, S., Ko, B. J., et al. (2020). Data selection for federated learning with relevant and irrelevant data at clients. arXiv preprint arXiv:2001.08300.

Vassilevska, V. (2009). Efficient algorithms for clique problems. *Information Processing Letters*, *109*(4), 254–257.

Wei, K., Li, J., Ding, M., et al. (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, *15*, 3454–3469.

Wu, D., Deng, Y., & Li, M. (2022). FL-MGVN: Federated learning for anomaly detection using mixed gaussian variational self-encoding network. *Information Processing & Management*, *59*(2), Article 102839.

Wu, C., Wu, F., Lyu, L., et al. (2022). Communication-efficient federated learning via knowledge distillation. *Nature Communications*, *13*(1), 1–8.

Yang, Q., Liu, Y., Chen, T., et al. (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, *10*(2), 1–19.

Yin, X., Zhu, Y., & Hu, J. (2021). A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. *ACM Computing Surveys*, *54*(6), 1–36.

Zeng, H., Zhou, T., Guo, Y., et al. (2021). FedCav: Contribution-aware model aggregation on distributed heterogeneous data in federated learning. In *50th international conference on parallel processing* (pp. 1–10).

Zhang, L., Li, Y., Xiao, X., et al. (2018). Crowdbuy: Privacy-friendly image dataset purchasing via crowdsourcing. In *Conference on Computer Communications* (pp. 2735–2743). IEEE.

Zhou, G., Zhu, X., Song, C., et al. (2018). Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International conference on knowledge discovery & data mining* (pp. 1059–1068).