Federated Analytics Informed Distributed Industrial IoT Learning with Non-IID Data

Zibo Wang, Yifei Zhu, *Member, IEEE,* Dan Wang, *Senior Member, IEEE,* and Zhu Han, *Fellow, IEEE*..

***Abstract*—The increasing concerns of communication overheads and data privacy greatly challenge the gather-and-analyze paradigm of data-driven tasks currently adopted by the industrial IoT deployments. The federated paradigm resolves this challenge by performing tasks collaboratively without uploading the raw data. However, the inherent data heterogeneity (skewness) of diverse industrial IoT data holders significantly degrades the performances of all kinds of federated industrial IoT learning tasks. Quantifying this skewness is non-trivial and cannot be solved by the existing federated learning techniques. In this paper, we propose a Federated skewness Analytics and Client Selection mechanism (FedACS) to quantify the data skewness in a privacy preserving way and use this information to help downstream federated learning tasks. FedACS provably estimates the skewness of the clients using the Hoeffding's inequality based on the distilled insights of edge data in the form of gradient. It then gracefully handles the drifting estimation and robustly selects clients with milder skewness using a novel dueling bandit approach. FedACS gains advantages in privacy preservation, infrastructure reuse, and optimized overheads. Extensive experiments on open datasets demonstrate that FedACS reduces the accuracy degradation by ~78.2%, and accelerates the FL convergence for ~ 2.4×.**

***Index Terms*—federated analytics, data heterogeneity, federated learning, dueling bandit**

I. Introduction

THE deployment of industrial IoT (IIoT) devices grows exponentially in recent years. As is reported by S&P Global, 86.7 million IIoT devices were deployed in the world in 2020, with 244 exabytes (million terabytes) data generated, and the values are expected to be 152 million and 725 exabytes in 2025 [2]. This huge volume of data after being exploited by artificial intelligence and data science algorithms reveals valuable insights and helps a broad range of verticals in the field of IIoT, such as smart manufacturing, smart city, etc [3]–[5]. Traditionally, the raw edge data from these IIoT devices are firstly gathered to a central server, where complex training and analytic algorithms are further applied. However, the traditional paradigm introduces significant communication overheads as the amount of data exponentially increases at the edge. In addition, laws and regulations are established in recent years to restrict the gathering and usage of raw data, such as GDPR [6] in Europe and CPPA [7] in California, in response to the increasing awareness of privacy.

Supported by edge computing techniques and the increasing awareness of data privacy, the federated paradigm is proposed to conduct the data-oriented tasks collaboratively without uploading the raw data. In this novel paradigm, raw data are no longer gathered by any central server. Instead, selected clients[1] receive calculation models from the server, perform data-oriented host tasks based on its local data, and upload the abstracted results back to the server. After that, the server aggregates the results and pushes forward the host task. The host task eventually achieves a comparable performance like those conducted in the traditional centralized paradigm after multiple iterations.

Unlike traditional distributed computation scenarios, where the data and hardware can be carefully configured, heterogeneity, especially data heterogeneity, naturally arises in the IIoT settings. For example, we consider anomaly detection of industrial sensors. Anomaly events can be recorded by a few sensors in some plants, while other sensors in other plants only record normal events. Collaborative model training in such a data heterogeneous environment can have lower accuracy and slower convergence rate [8], [9]. Due to the prohibition of accessing raw data, the severity of the heterogeneity in the clients is unknown to the central server. *If we can design a federated mechanism to measure the data heterogeneity of the clients, this derived knowledge can help improve the efficiency of the host federated task without uploading the raw data.*

The earliest and the most studied instance of the federated paradigm is federated learning (FL), where clients collaborate on neural network training without sharing the raw data [10]. Due to the wide interest of deep learning, FL has demonstrated its capability in *predictive* tasks like object detection [11] and word prediction [12]. The FL solutions are also widely applied in the field of IIoT [13]. However, including the data heterogeneity quantification task we are interested to solve, there remain many *descriptive* data science problems like heavy hitter discovery, data sketching, distribution estimation, *etc.*, which are not suitable to be solved by neural networks. In these problems, the studied questions are no longer simply "how to collaboratively train a model to do the predictive task", but rather "what is the most frequent word used by users?", "how data in a client are beneficial for the host federated task?", *etc*. With the introduction of stricter privacy

This work is funded by the SJTU Explore-X grant. This work is partially supported by NSF CNS-2107216 and CNS-2128368.

A prior version of this article was presented at the IEEE/ACM International Symposium on Quality of Service (IWQoS 2021) [1].

Z. Wang and Y. Zhu are with UM-SJTU Joint Institute, Shanghai Jiao Tong University, Shanghai, China; e-mail: {wangzibo,yifei.zhu}@sjtu.edu.cn.

D. Wang is with Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China; e-mail: csdwang@comp.polyu.edu.hk.

Z. Han is with the Department of Electrical and Computer Engineering at the University of Houston, Houston, USA, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul, South Korea; e-mail: zhan2@uh.edu.

The corresponding author is Y. Zhu.

[1]We use edge device and client interchangeably in this paper

requirements, these equally important tasks call for a federated solution as well.

Considering the equal importance of these tasks, federated analytics (FA) is proposed by Google in May 2020 to fill the vacancy of data science tasks' federation [14]. While FA shares the same federation characteristics with FL, following the patterns of local model computation, central model aggregation, and interactive updates, it intrinsically differs itself from FL in its ultimate goal and the task-varying design details. FL targets training neural networks, while FA targets performing non-training data analytics tasks. Due to the diverse data analytics tasks, the design details also vary. For example, in an FA-based heavy hitters discovery, a prefix tree serves as the model; the local computation is adding a leaf to the tree; the aggregation is updating the tree with high-support leaves [15]. In another FA-based clustering work, the local computation is hashing their local data to form a binary vector; the central aggregation is to assign the clients with identical (or similar) vectors into the same cluster; [16]. Though a variety of data analytic tasks, e.g., calculating histogram [17], frequent pattern mining [18], clustering [19], have been proposed, data skewness analysis have not been studied yet.

In this paper, we present the first work on characterizing the class distribution heterogeneity in federated systems and use this insight to create a desirable data environment via intelligent client selection[2]. Unlike learning an unknown probability distribution from random samples or manually selecting features to heuristically determine a client's data heterogeneity, we use the term *skewness*[3] to describe the severity of the local class distribution of a client skew from the global, virtually centralized, one and aim at gaining a provable estimation on it. Specifically, we propose a joint Federated skewness Analytics and Client Selection mechanism (FedACS). FedACS is a typical instance of FA, where clients generate insensitive insights about their local data, and the server aggregates the insights to infer the skewness of each client. FedACS can either be treated as a stand-alone analytic design or integrate with other federated tasks. We further integrate FedACS with FL to help distributed IIoT training, and fully demonstrate its potential and effectiveness. FedACS has three modules: insight derivation, skewness estimation, and client selection. The insight derivation part is designed to share the computation and communication with the host task, so that the infrastructure is reused and the communication is saved. In the skewness estimation part, the Hoeffding's inequality is employed to bridge the aggregated insights and the client skewness. In the client selection part, facing the drifting estimations of the skewness level, we adopt a dueling bandit solution to robustly select clients. The three modules are executed iteratively, enabling FedACS to accurately lock on the clients with low skewness and to help the host FL task achieve higher accuracy and faster convergence.

In summary, our contributions are:

- To the best of our knowledge, this is the first work on federated data skewness analytics in decentralized data environments.
- Our proposed analytic mechanism measures the skewness of clients in a mathematical provable way with the power of the Hoeffding's inequality, and guarantees the data privacy at the same time.
- Facing the drifting skewness information, we novelly formulate the client selection problem as a dueling bandit problem. Our proposed solutions successfully balance the trade-off between sample size and data quality, greatly improving the FL performance.
- Extensive experiments under different data heterogeneity environments show that, FedACS-assisted FL reduces $\sim$ 78.2% of accuracy degrading, speeds up the convergence for $\sim$ 2.1×, and outperforms existing methods tackling data heterogeneity in FL, with negligible overheads.

The rest of this paper is organized as follows: in Section II, the background knowledge of our work is reviewed. In Section III, we model the data heterogeneity under different situations and demonstrate its harm to model training. Section IV gives an overview of FedACS. Two major components of FedACS, federated skewness estimation and client selection, are elaborated in Sections V and VI, respectively. In Section VII, experiments and results are presented. Related work is surveyed in Section VIII, followed by the conclusion in Section IX.

II. Preliminaries and Background

In this section, we briefly introduce several important concepts and tools that we use in our work.

A. Federated learning and analytics

Federated learning and analytics refer to the process of multiple clients collaborating to solve data-oriented problems without sharing their raw data [20]. FL is a distributed learning framework to collaboratively train a neural network [10]. An FL process works in an iterative way. Each iteration has four phases: model distribution, local training, model upload, and model aggregation. In each iteration, the selected clients train the global neural network with its local data, and then let the centralized server aggregate the neural networks. In FA, distributed stored data is exploited for non-training data science services. It retains advantages of the federated paradigm, such as privacy preservation and communication efficiency.

B. Hoeffding's inequality

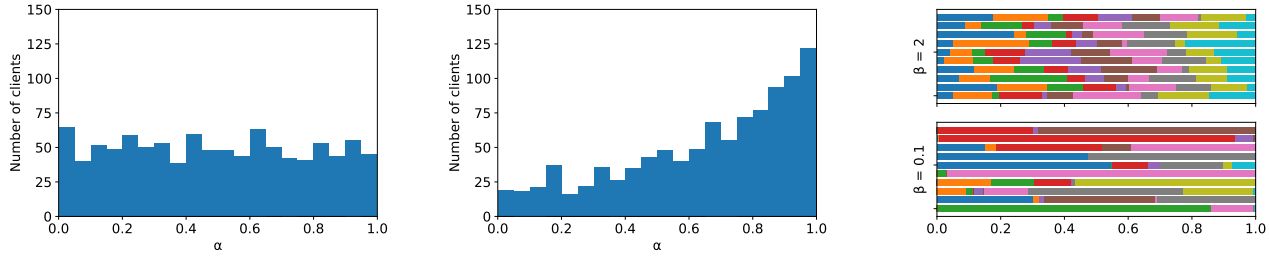
The Hoeffding's inequality is a statistical tool first introduced in [21]. It estimates the deviation of the average of independent random variables from its exception and provides a probabilistic bound for it.

Hoeffing's Inequality: Supposed $X_1, ..., X_n$ are independent variables, $X_i \in [a_i, b_i]$, $\forall i \in [1, n]$, $\overline{X}$ is the average of X_i, there's

$$\mathbb{P}(|\overline{X} - \mathbb{E}(\overline{X})|) \geq \epsilon) \leq 2exp\bigg(-\frac{2\epsilon^2 n^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\bigg). \quad (1)$$

[2]We will make the implementation open source.

[3]Note that skewness has a different definition in statistics. In this paper, we follow a similar definition to describe the label distribution skew as in [10] and [8].



(a) Uniform skewness environment (b) Inverse Pareto skewness environment (c) Dirichlet skewness environment

Fig. 1. Skewness of clients under different non-IID data environments.

C. Dueling bandit

Dueling bandit is firstly introduced in [22]. It considers a different scenario than conventional stochastic multi-armed bandits (MAB). In the dueling bandit, a bandit receives information about noisy comparison (dueling) of arms. For each pair of arms a_i and a_j, the probability for a_i to be stronger than a_j:

$$\mathcal{P}(a_i \succ a_j) = \phi(a_i, a_j) + \frac{1}{2}, \tag{2}$$

where $\phi(a_i, a_j)$ denotes the stochastic preference between a_i and a_j. In conventional dueling bandit [22], only two arms are selected to perform one duel in each round. The goal of the bandit is to find the *Condorcet winner*, denoted as a^*, which can beat all other arms with a probability higher than 0.5. Namely, dueling bandit aims to minimize the cumulative regret after T plays:

$$R_T = \sum_{t=1}^{T} \max\left(\phi(a^*, a_i^{(t)}), \phi(a^*, a_j^{(t)})\right). \tag{3}$$

Multi-dueling bandit is an extension of conventional dueling bandit, where simultaneous dueling of multiple arms are allowed [23].

III. Skewness hurts: a Case Study

In this section, we first present a set of models to formalize and quantify distribution skewness in the real world in Section III-A, and then demonstrate the degradation effect of data heterogeneity on FL based on our designed skewness description models in Section III-B.

A. Non-IID environments modelling

To evaluate the performance of federated learning in non-IID environments, previous works usually synthesize the dataset to let each client possess a small number of classes (one class or two classes only) [8], [24]. As an improvement of this simple approach, researchers in [25], [26] let one or two classes fill the majority (*e.g.* 80%) of raw data. Since the Dirichlet distribution generates a random vector with an invariant sum, which can be transformed into class distribution, another branch of works also employ the Dirichlet distribution to generate non-IID environments recently [27], [28].

However, all these approaches fail to capture the heterogeneous skewness situation in the real world. If all clients have one or two classes fill up the majority (or even all) of their data, their skewness is exactly similar. Though the lately adopted Dirichlet distribution approach slightly diverges the skewness across clients, the skewness of clients is still closed to each other when the concentration parameter is fixed. In practice, the skewness of clients is not only different, but also heavily diverges. For example, the monitoring result of an ordinary factory is likely to cover a wide range of events. Meanwhile, the result of an unmanned factory tends to be much more skewed.

To fully capture the heterogeneity at the client side, we model the non-IID environment in two steps. We first model the client-level class distribution heterogeneity similar to other's work. We then diverge the key parameters that determine the skewness of each individual client, so that clients have different skewness at the population level.

Specifically, at the client level, we first generalize the existing heterogeneity settings and introduce the definition of the *α-dominance* client and *β-Dirichlet client*.

Definition 1 (α-dominance client)**.** *An α-dominance client has the portion of α of the data that belongs to a dominant class, while the rest is evenly partitioned in all classes (including the dominant class).*

The definition of the *α-dominance* client is a generalization of those in [25], [26]. Though *α-dominance* still has limited expressibility, it allows us to quantify and compare the skewness of clients, which is beneficial for our future evaluation part. For example, if we assume that there are ten classes in total, one 0.5-dominance client with 100 data means that it has 55 data belonging to the dominant class, and 5 data for each remaining class. As can be seen, assuming that all classes are evenly distributed in the global data, a 0-dominance client is also an IID client. With this definition, the skewness of clients can be easily quantified, that low α indicates low skewness, and vice versa.

Following the approaches used in [27], [28], we also introduce another client-level class distribution definition called the *β-Dirichlet client*.

Definition 2 (β-Dirichlet client)**.** *An β-Dirichlet client has the class distribution following the Dirichlet distribution, with concentration parameter β.*

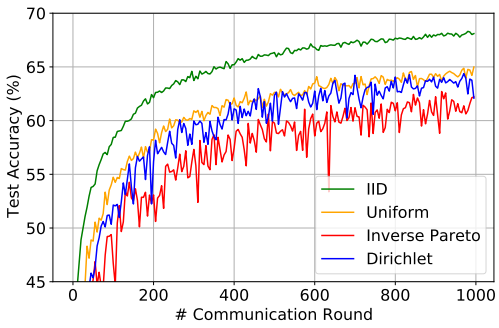


Fig. 2. Test accuracy v.s. communication rounds on different skewness environments.

The Direchlet distribution can be seen as a multivariate generalization of a Beta distribution. Denote the class distribution of a client as a vector x with dimension d (also the number of classes), the probability of x can be measured as

$$p(x) \propto \prod_{i=1}^{d} x_i^{\beta-1}. \tag{4}$$

A β-Dirichlet client is provided more freedom on the class distribution, and is therefore more difficult to quantify and compare its skewness. However, there also exists a weak relationship that a client with a higher β is likely to be less skewed.

Based on these two client-level skewness definitions, we then define three population-level skewness environments as follows, which model the diverse skewness situation across different clients.

Definition 3 (Uniform skewness environment)**.** *In an uniform skewness environment, all clients are α-dominance clients, and the value of α of each client is a random variable following continuous uniform distribution in the range of* $[0, 1]$.

Uniform skewness environment defines a comparatively mild skewness environment with the client skewness parameter α uniformly distributed in its domain.

Definition 4 (Inverse Pareto skewness environment)**.** *In an inverse Pareto skewness environment, all clients are α-dominance clients. For each client, we first sample x from truncated Pareto distribution with shape parameter s and domain* $[1, 2]$. *Then, we assign the value of α to be $2 - x$. The resulted PDF of α is*

$$\mathbb{P}(\alpha) = \frac{s(2-\alpha)^{-s}}{1-2^{s}}, \quad \alpha \in [0, 1]. \tag{5}$$

The inverse Pareto skewness environment is based on the Pareto distribution, a powerful tool in describing scientific and social observable phenomena [29]. Particularly, we use the truncated Pareto distribution, which is with a more practical form, and inverse it to force the values of α biased to 1 to simulate a more heavily skewed scenario.

The previous two skewness environments are both based on Definition 1. The next skewness environment, named Dirichlet skewness environment, is based on Definition 2.

Definition 5 (Dirichlet skewness environment)**.** *In a Dirichlet skewness environment, all clients are β-Dirichlet clients. Half of the client has β values following continuous uniform distribution in range $(0, x_{med}]$, and those of the rest clients are in range $(x_{med}, x_{max}]$.*

In the Dirichlet skewness environment, the values of β follow a layered uniform distribution with median and maximum predefined. The rationale of our setting is that the change of skewness is not "linear" with the change of β, *e.g.* if we change β from 0.2 to 0.1, the change of client skewness, is much more violent than when changing β from 2.1 to 2.0. By layering β, we prevent the case that the majority of the clients still have similar skewness.

We visualize three previously defined environments for better understanding as shown in Fig. 1. For the uniform skewness environment and the inverse Pareto skewness environment, since the clients are quantifiable via α, we sample 1,000 clients and create a histogram of their skewness with 20 bins. For the Dirichlet skewness environment, where the control on skewness is indirect, we show the class distribution of 20 clients, with two values of β, as a demonstration. Compared with the uniform skewness environment in Fig. 1(a), for the inverse Pareto skewness environment in Fig.1(b), the distribution of α is biased to 1.0, and therefore it is generally more heavily skewed. In Fig. 1(c), colors represent classes of raw data, and each row describes the class distribution of classes in one client. We can see that a β-Dirichlet client with a higher value of β is less skewed.

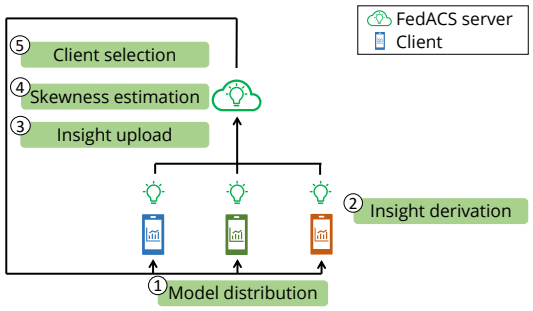
B. Harm of data heterogenity on FL

To study the degrading effect of the skewed clients, the performance of FL on these skewness environments is measured. CIFAR-10 dataset [30] are used in our experiments. We adopt the same structure of CNN as suggested in [31]. 50,000 training samples are evenly distributed to $N = 200$ clients, with the class distributions controlled by the definitions above. In each round, the portion of clients being selected $C = 0.05$, the number of local epoch $E = 5$, and local batch size $B = 50$. Learning rate $\gamma = 0.1$, and learning rate decay of each local epoch $\gamma_d = 0.9993$.

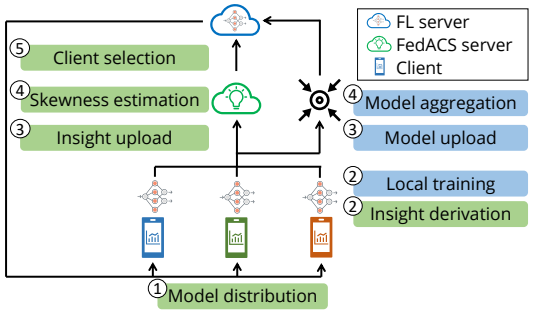
As is shown in Fig. 2, all three skewness environments degrade the performance of FL, compared to the IID baseline. The terminal test accuracy of the uniform skewness environment, the inverse Pareto skewness environment, and the Dirichlet skewness environment are decreased for 3.6%, 6.7%, and 4.9%, respectively. The degrading effect of the inverse Pareto environment is more severe than that of the uniform environment, since it is more heavily skewed. The Dirichlet environment has a moderate degrading effect.

IV. FedACS: an Overview

In this section, we introduce the overall structure of FedACS after integrating with FL. In this system, each client is in charge of performing the local training phase in FL and the insight derivation phase in FedACS. The server in the federated systems is free to choose a subset of all available clients to



(a) Workflow of FedACS



(b) Workflow of FedACS assisted FL (The sharing modules are in green shade)

Fig. 3. An overview of FedACS and FedACS assisted FL. Insight derivation and skewness estimation modules are demonstrated in Section V, and client selection module is demonstrated in Section VI.

participate in each round. The clients have different intrinsic data skewness. Therefore, the benefit provided by each client varies. To maximize the overall benefit, the server wants to let clients with low skewness participate more frequently, and reduce the participation of clients with high skewness. This is the underlying idea of FedACS.

As is shown in Fig. 3(a), FedACS has three components: insight derivation, skewness estimation, and client selection. In the insight derivation part, each participating client generates insight, which will be utilized by the server to infer client skewness. The design of insight is of a high degree of freedom, but should not leverage direct information about raw data. In this paper, the insight is the results of a batch gradient descent (BGD) to cater to the host task, FL. Our design reuses the infrastructure of FL, and obtains the same level of privacy protection as FL.

In the skewness estimation part, the server transforms the insights derived by clients into the estimations of clients' skewness. The Hoeffding's inequality is applied to bridge the connection between the uploaded insights and clients' skewness in a non-heuristic way.

In the client selection part, a MAB is formulated, where clients with low skewness are interpreted as the desirable arms. Our tailored dueling bandit solution carries out an exploration-exploitation trade-off under the challenge of the non-stationary reward while meeting the requirement of the host task.

To further demonstrate the usefulness of our extracted skewness information, we enhance FL with our proposed FedACS, which is illustrated in Fig. 3(b). The client performs two gradient descent procedures in one iteration. One is for the host FL task, and the other is for the insight derivation procedure of FedACS. The two procedures are almost identical except for parametric settings. Both results are uploaded to the server, and are used for FL model aggregation and client skewness estimation, respectively. The overall structure of FedACS assisted FL is present in Algo. 1.

Algorithm 1 FedACS: overall structure

```
Initialize the bandit B
Initialize the neural network θ
for Round 1, 2, ..., T do
    B selects participating clients S                ▷ Algo. 2
    for c ∈ S do
        Δw_c ← gradient descent on θ with FL settings
        Δŵ_c ← gradient descent on θ with FA settings
    end for
    Update θ with Δw
    R ← derive skewness estimate with Δŵ            ▷ Eq. (16)
    Update B with R                                  ▷ Algo. 3
end for
```

V. Federated Client Skewness Estimation

In this section, we provide details on the federated client skewness estimation based on the Hoeffding's inequality. We take FL as an example of FedACS application, and the skewness estimation scheme is therefore adapted for FL. In Section V-A, we show how the Hoeffding's inequality is employed in the federated client skewness estimation. In Section V-B, we further build the relationship between the results from the Hoeffding's inequality and the client skewness. In Section V-C, an estimation for the expectation of the uploaded insights is provided. In Section V-D, a practical estimation of the client skewness is finally concluded. In Section V-E, we discuss that FedACS is a modular design and can be easily adapted to all kinds of federated tasks. In addition, for reading convenience, we summarize the important notations in Table I.

A. Usage of the Hoeffding's inequality

Estimating client skewness has two major challenges. First, "skewness" of clients does not depend on class distribution on one client, but on how close it is to the global distribution. Therefore, the skewness must be calculated by considering the situation of all clients. Second, the class distribution is usually considered private. As a result, the insight should be sufficiently "indirect" that the server is unable to infer the class distribution of each client.

In this part, we demonstrate how the Hoeffding's inequality is applied in skewness estimation. Since the insight we used is in the form of a gradient (weight change) from the neural network, we first briefly introduce how the gradient is derived, and then show how the Hoeffding's inequality is linked to it.

Suppose there are N clients in total in the system. Denote $d_{i,m}$ as the m-th datum in the i-th client. M_i is the number of

data in the i-th client. Neural networks calculate gradient via backward propagation. First, the client calculates a loss function $Loss(d)$ for each datum d indicating how the prediction of one datum d is closed to the truth. Then, the client averages the loss function of all data it owns to form a cost function $Cost_i$. Finally, the client calculates the gradient of the neural network. Denote the index (dimension) of weight as k, and the number of weights is K, we have

$$\Delta w_i^{(k)} = \gamma \times \frac{\partial Cost_i}{\partial w^{(k)}}, \tag{6}$$

where Δw_i^k is the gradient of client i in dimension k, and γ is a preset learning rate. In FL, client i upload Δw_i, with K dimensions, to the server. This is the full procedure of generating gradients in a neural network. Then we link the final result Δw_i to the Hoeffding's inequality.

Denote $z_{i,m}^{(k)}$ as the k-th dimension of gradient derived from the m-th datum in the i-th client, times the learning rate γ.

$$z_{i,m}^{(k)} = \gamma \frac{\partial Loss(d_{i,m})}{\partial w^{(k)}}. \tag{7}$$

Denote $z_i^{(k)}$ as the average value of $z_{i,m}^{(k)}$, based on the gradient calculation of deep learning in (6), we have

$$\begin{aligned} z_i^{(k)} &= \frac{1}{M_i} \sum_{m=1}^{M_i} \gamma \frac{\partial Loss(d_{i,m})}{\partial w^{(k)}} \\ &= \gamma \frac{\partial \frac{1}{M_i} \sum_{m=1}^{M_i} Loss(d_{i,m})}{\partial w^{(k)}} \\ &= \gamma \frac{\partial Cost_i}{\partial w^{(k)}} \\ &= \Delta w_i^{(k)}. \end{aligned} \tag{8}$$

Since $z_{i,m}^{(k)}$ are derived from different independent datum, they can be treated as independent random variables. In addition, the uploaded weight change $\Delta w_i^{(k)}$ is the average value of $z_{i,m}^{(k)}$. Therefore, we apply the Hoeffding's Inequality in (1) to $z_{i,m}^{(k)}$, and get p_i^k, the probability that k-dimension of weight change from client i diverges from its exception for a fixed value ϵ. Namely,

$$\begin{aligned} p_i^k &= \mathbb{P}(|\Delta w_i^{(k)} - \mathbb{E}(\Delta w_i^{(k)})|) \geq \epsilon) \\ &\leq 2exp\bigg(- \frac{2\epsilon^2 M_i^2}{\sum_{m=1}^{M_i} (b^{(k)} - a^{(k)})^2} \bigg) \\ &= 2exp\bigg(- \frac{2\epsilon^2 M_i}{(b^{(k)} - a^{(k)})^2} \bigg), \end{aligned} \tag{9}$$

where $b^{(k)}$ and $a^{(k)}$ are the upper and lower bounds of $z_{i,m}^{(k)}$. To make our estimate comparable for different client (i) and datum (m), we adopt the same bounds $b^{(k)}$ and $a^{(k)}$ across all clients instead of $b_{i,m}^{(k)}$ and $a_{i,m}^{(k)}$. We are safe to do this because (1) does not require a tight bound.

Recall that $z_{i,m}^{(k)}$ are weight changes derived by one datum. Although the server does not have knowledge about the datum $d_{i,m}$, we can estimate its skewness by estimating the skewness of $z_{i,m}^{(k)}$, which is a mapping of $d_{i,m}$. However, in federated learning, values of $z_{i,m}^{(k)}$ are also private. Therefore, FedACS does not require the clients to upload $z_{i,m}^{(k)}$, but uses the Hoeffding's inequality to estimate its expectation based on $\Delta w_i^{(k)}$.

TABLE I
IMPORTANT NOTATIONS

Notation	Definition
i, j	Index of client
m	Index of datum in one client
k	Index of parameter of the neural network
N	Number of client
M_i	Number of data in client i
K	Number of parameters of the neural network
$\hat{K}$	Number of parameters of the last layer
$d_{i,m}$	m-th datum of client i
$z_{i,m}$	Gradient derived by $d_{i,m}$
$z_{i,m}^{(k)}$	k-th dimension of $z_{i,m}$
$z_i^{(k)}$	Average of $z_{i,m}^{(k)}$ for all data in client i
$p_i^{(k)}$	Probalistic bound derived by the Hoeffding's inequality
$a^{(k)}$	Upper bound of $z_{i,m}^{(k)}$
$b^{(k)}$	Lower bound of $z_{i,m}^{(k)}$
Δw_i	Uploaded gradient from client i
$\overline{\Delta w}$	Average uploaded gradient
$\Delta w_i^{(k)}$	k-th dimension of Δw_i
$\overline{\Delta w}^{(k)}$	k-th dimension of $\overline{\Delta w}$
$P_i^{(k)}$	Skewness estimate of client i based on dimension k
P_i	Intermediate skewness estimate of client i
Q_i	Final skewness estimate of client i
ψ_i	Intrinsic skewness of client i
R_i	Reward of client i for bandit

B. Connection between client skewness and $p_i^{(k)}$

In this part, we link obtained p_i^k to client skewness, so that the bridge between the client insights and the client skewness estimation is completed.

We start with null and alternative hypotheses $\mathcal{H}_0$ and $\mathcal{H}_1$,

$\mathcal{H}_0$*: Data in client i is IID distributed.*
$\mathcal{H}_1$*: Data in client i is not IID distributed.*

In reality, there is no such a binary classification between IID and non-IID. These two hypotheses are never accepted and rejected in this paper. Instead, their probabilities of acceptance are calculated to infer the client skewness. To be specific, we conduct the hypothesis test to infer the client skewness: we first accept $\mathcal{H}_0$ anyway, so that we can calculate p_i^k. The obtained p_i^k then estimates how rare the situation is, which indicates the possibility to reject $\mathcal{H}_0$ (and accept $\mathcal{H}_1$). Finally, we can explore the skewness of clients based on the possibility to reject $\mathcal{H}_0$.

Following the aforementioned rationale, we first link the possibility of rejecting $\mathcal{H}_0$ to the value of $p_i^{(k)}$ derived by the Hoeffding's inequality, as presented in Lemma 1.

Lemma 1. *At least one of the alternatives must be satisfied: 1) clients with highers $p_i^{(k)}$ values are more likely to accept $\mathcal{H}_0$, or 2) clients with highers $p_i^{(k)}$ values are likely to have a lower skewness.*

Proof. See Appendix A-A □

In fact, the two alternatives in Lemma 1 indicates the same results: if we have a high possibility to believe that client i is

IID, and then client i is equivalently more likely to be with low skewness, because an IID client is defined to be with the lowest skewness. Therefore, we can consider that the value of $p_i^{(k)}$ has a negative relationship with client skewness anyway.

In addition, when the assumption is made that data in client i is IID, the expectation of $z_i^{(k)}$ should be equal to the global one $z^{(k)}$. Based on that, a new expression of $p_i^{(k)}$ is derived as a side product from Lemma 1, *i.e.*,

$$\begin{aligned} p_i^{(k)} &= \mathbb{P}(|\Delta w_i^{(k)} - \mathbb{E}(z^{(k)})|) \geq \epsilon) \\ &\leq 2exp\bigg(-\frac{2\epsilon^2 M_i}{(b^{(k)} - a^{(k)})^2}\bigg). \end{aligned} \tag{10}$$

C. Estimation of $\mathbb{E}(z^{(k)})$

According to (10), if we want to use this $p_i^{(k)}$ to measure skewness at the k-th dimension, we first need the expectation of $z^{(k)}$ to derive ϵ, and then $p_i^{(k)}$. The exact value of $\mathbb{E}(z^{(k)})$ is the average of $z_{i,m}^{(k)}$ of all data in all clients $(\forall i, m)$. However, not all clients participate in each round in the federated learning. Therefore, the exact value is impossible to obtain. Instead, we used the average of all data in all participating clients to estimate $\mathbb{E}(z^{(k)})$.

We present the rationale of estimating $\mathbb{E}(z^{(k)})$ into the following theorem.

Theorem 1. *The expectation of $z^{(k)}$ can be estimated by the average of uploaded weight change of all participating clients, weighted by the sample size of each client.*

Proof. See Appendix A-B □

As the estimation of $\mathbb{E}(z^{(k)})$ has been given, we are able to propose a more practical estimation of client skewness. Rewrite (10), we have

$$P_i^{(k)} = 2exp\bigg(-\frac{2(\epsilon^{(k)})^2 M_i}{(b^{(k)} - a^{(k)})^2}\bigg), \tag{11}$$

where

$$\epsilon^{(k)} = |\Delta w_i^{(k)} - \overline{\Delta w}^{(k)}|. \tag{12}$$

Here, $\overline{\Delta w}^{(k)}$ indicates the weighted average of uploaded weight change of all participating clients on dimension k.

D. Combination of dimensions and formation of final skewness estimation

Equation (11) provides an estimation about the skewness of all clients. However, it does only make use of one dimension of weight changes. We include the skewness estimation by all dimensions of weight change to make the client skewness estimation more robust. The combination of estimation from different dimensions is not mathematically purposeful, but consider that the nature of $P_i^{(k)}$ is a probability. Multiplying $P_i^{(k)}$ among all dimensions is also plausible, as it can be understood as the logical operator "and".

A naïve approach is to apply the whole neural network to generate the skewness estimation as the prior version of FedACS did [1]. However, such an approach indicates that the whole neural network of the FA part has to be uploaded, introducing significant communication overheads. In this paper, we optimize it by only utilizing weights from the last layer of the neural network, which is sufficiently representative for the skewness estimation and only introduces neglectable overhead. A detailed overhead analysis is presented in Section VII.

The procedure of combining the dimensions and deriving the skewness estimate is present in Appendix A-C. As a result, we derive a final result Q_i as a practical estimate of client skewness.

$$Q_i = \sqrt{M_i}||\Delta w_i - \overline{\Delta w}||_2 \tag{13}$$

where,

$$\overline{\Delta w} = \frac{1}{\sum_{i \in N_t} M_i} \sum_{i \in N_t} M_i \Delta w_i. \tag{14}$$

The values of Q_i have a positive correlation to clients' skewness, *i.e.*, for all client pairs a and b,

$$\psi_a > \psi_b \iff Q_a > Q_b, \tag{15}$$

where ψ_a and ψ_b are intrinsic skewness of client a and b.

Q_i is an effective estimate of client skewness, and can be calculated based on uploaded FA insights from clients. If the prerequisite is satisfied that the number of data in each client is equal (which is practical by limiting the number of data participating in FL), the requirement of information about data size can be removed.

E. Analysis and Discussion

Computation complexity It can be easily concluded that the overall complexity of FedACS assisted FL is in the order $O(TM|\theta|)$, where T is the total number of rounds, M is the number of participating clients in one round, and $|\theta|$ is the number of parameters of the host neural network. FedACS exactly has the same computation complexity as vanilla FL.

Design benefit Since FedACS aims at assisting FL tasks here, the insight derivation (local analysis) part is designed as a gradient descent procedure with three benefits. First, it reuses the global model distributed by the server, and does not require extra downward communication. Second, in order to perform the host FL, each client must have the ability to perform gradient descent. FedACS does not require clients to install or deploy any other computation scheme. Last, the insight derived by clients is in the form of gradients (weight changes). Therefore, our skewness estimation maintains the same level of privacy reservation as its host task FL, and does not introduce an extra risk of privacy exposure.

Generality Though we mainly focus on distribution skewness analytics in this paper, following the line of non-IID FL studies [24]–[28], the gradient information we use as the proxy to the local data allows FedACS also capable of handling other data heterogeneity beyond class distribution. Essentially, any form of data heterogeneity, as long as it leads to a more diverged gradient, can be theoretically measured by FedACS. In addition, it is not a complicated task to adapt FedACS to assist other federated tasks. It can be easily completed by redefining $z_{i,m}^{(k)}$ in (7) with all other parts remain unchanged, because $z_{i,m}^{(k)}$ can be any mapping of raw data $d_{i,m}$. An easy

and good approach is to design $z_{i,m}^{(k)}$ as final or intermediate results of the host task, like the gradient for the FL task. Users are free to decide the number of dimensions $\hat{K}$. They can increase $\hat{K}$ by generating many kinds of insights, and therefore increase the accuracy of the skewness estimation.

VI. Analytics Informed Client Selection: a Bandit Approach

In this section, to demonstrate the benefit of the extracted skewness information derived in (13), we make use of this estimation to select clients with low skewness as participants of FL to help improve FL. We first demonstrate that leveraging this information to help client selection is non-trivial in Section VI-A. We then carefully formulate the client selection problem into a dueling bandit problem based on several findings we find in Section VI-B. We present our tailored dueling bandit solution in Section VI-C.

A. Challenges: drifting reward and limited sample size

At first glance, selecting clients with different skewness via MAB is plausible for this scenario, in which clients with different skewness levels refer to arms with different potential rewards [32]. Since MAB pursues arms with a high reward, and we are seeking those with low skewness (low Q_i), we simply use the inverse of Q_i as the reward of arm R_i, *i.e.*,

$$R_i = -Q_i = -\sqrt{M_i}||\Delta w_i - \overline{\Delta w}||_2 \tag{16}$$

Denote μ_i as the exception of R_i, and $\mu^* = \max_i \mu_i$. An MAB tries to select a sequence of arms that minimize the regret ρ,

$$\rho = T \cdot \mu^* - \sum_{t=1}^{T} R(t), \tag{17}$$

where $R(t)$ indicates R_i of the arms being selected at time t.

Although the bandit formation above seems plausible, we present two challenges that make the classical MAB solutions, *e.g.* upper confidence bound (UCB), $\epsilon - greedy$, perform poorly in identifying the right client.

Challenge 1: The reward value R_i in (16) is drifting, which violates a prerequisite of multi-armed stochastic bandit that the distribution of R_i should be stationary over time.

To demonstrate this, we use the defined uniform skewness environment model to reveal the changes of R_i over time. In this environment, all 200 clients are α-dominance clients. 140 clients are generated by Definition 3, filling up the environment. And the rest 60 clients are evenly divided into six observation groups, denoted as groups 1-6 (or G1-6). Clients in these groups are assigned α values with 0, 0.2, 0.4, 0.6, 0.8, and 1.0, respectively. In each round, all clients' rewards in observation groups are calculated using (16) as if they are selected for participating in the corresponding round.

The results are shown in Fig. 4. Several interesting insights can be derived from it. First, the values of R_i of all groups are drifting over time. Second, the quantitative relationship of groups holds over time, *i.e.*, client group with low skewness has higher reward than those with high skewness in any single round. Third, although the overall trend of R_i reflects the group

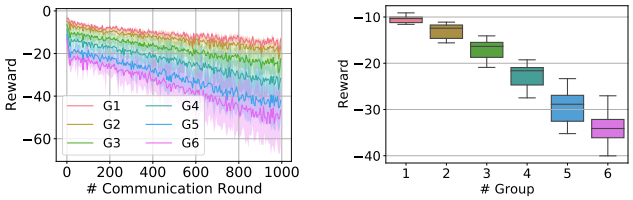


Fig. 4. R_i values of different groups, groups with smaller indexes are less skewed. Left: R_i values in different rounds, the lines indicate median reward of clients, and the shaded areas indicate the range. Right: R_i values of different groups in round #500.

skewness, irregulars exist. It is reflected by the overlap of the shaded area in Fig. 4(a), which means that in certain cases, clients with higher skewness have higher R_i. In addition, Fig. 4 further validates the correctness of the skewness estimation procedure in FedACS, since R_i values separate the clients with different skewness.

The drifting phenomenon of R_i has a theoretical cause. Eq. (13) shows that the calculation of skewness estimate derived is based on the global model for each round. As a result, the estimate is exactly not comparable in different rounds, since the global model in different rounds differs. In other words, there is no guarantee that values of R_i are comparable in different rounds, *i.e.*, we cannot guarantee the value of R_i of the same client (or client with same skewness) are stationary over time. The aforementioned insights greatly challenge the usage of stochastic MAB. Because traditional stochastic MAB algorithms, like ϵ-greedy and UCB, require the reward distribution of arms to remain unchanged over time [33].

Challenge 2: Curse of small sample size exists, which can result in poor performance.

If we use the aforementioned bandit solutions directly, as the bandit converges, we will repeatedly use a small number of clients with the lowest skewness. However, this naturally means that only a little portion of raw data is utilized by neural networks, which can be not sufficient to reach a satisfying convergence accuracy for neural network models.

We design another experiment to demonstrate the degrading effect of lacking raw samples. We fixed the number of IID samples held by each client to be 250, and the number of clients being selected for each round to be 10. Then, we adjust the number of clients in the environment to be 50, 100, 150, and 200. Since the CIFAR-10 dataset has 50,000 training samples at all, the four settings indicate that 25%, 50%, 75%, and 100% of raw samples are utilized.

As demonstrated in Fig. 5, with the size of data being utilized decreasing, the degrading effect becomes more critical. In addition, unlike the degrading effect of skewness (Fig. 2), lacking raw samples does not slow the convergence speed at the starting rounds.

B. A dueling bandit formulation

For the first challenge of drifting reward, the most straightforward consideration to capture the drifting reward is modeling this problem as a non-stationary bandit. The non-stationary

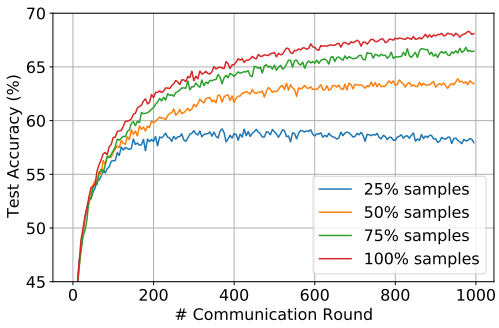


Fig. 5. Test accuracy v.s. communication rounds of FL with different size of utilized samples.

bandit is firstly proposed in [34], considering the bandit whose arms have reward distribution varying over time. However, the non-stationary bandit focuses on the potential intersection of rewards of different arms, which leads to the changing optimal arm [35]–[37]. In order to prevent the potential loss, non-stationary bandit algorithms frequently check whether the optimal arm is changed by increasing the proportion of exploration. This high proportion of exploration behavior sacrifices the bandit effectiveness.

Fortunately, in our case, since the data distribution of each client usually stays the same, we do not need to worry about changes of optimal clients. It is unlikely to exist a severe intersection of R_i curves for clients with different skewness. Furthermore, theoretically, skewness estimations in the same round are derived from the same global model, and are therefore comparable. Because a low skewness client is more likely (with possibility higher than 0.5) to have higher R_i than those with high skewness. We can convert the skewness estimate of clients in one round into comparisons of participating clients. Our problem for selecting a set of clients with low skewness is equivalent to selecting a set of clients that get the most wins in duelings. We can then formulate our problem into a multi-dueling bandit problem, whose preliminaries have been introduced in Section II-C.

As for the curse of the limited sample size, essentially, this challenge reflects the trade-off between the number of clients we expect to utilize and the skewness level of the selected clients we can tolerate in bandit. If the number of involved clients remains small, the degrading caused by client skewness is reduced, but the degrading caused by lacking raw samples is aggravated. On the contrary, when we increase the number of clients being utilized, the neural network will receive more raw samples, but the clients with higher skewness will "sneak in", and degrade the FL performance.

Based on the previous two insights, we present the formal formulation of our problem. Denote the set of clients as C. For each client $i \in C$, ψ_i indicates quantified intrinsic skewness of client i. Note that if all clients are α-dominance clients and α_i denotes the dominance level of a particular client i, then ψ_i has a positive correlation with α_i.

For each pair of clients c_i and c_j, if they are both given rewards from (16) in the same round or closed rounds, and then the quantitative comparison of R_i and R_j has a noisy negative correlation with ψ_i and ψ_j, *i.e.*,

$$\mathbb{P}(R_i > R_j) > 0.5 \iff \psi_i < \psi_j. \tag{18}$$

Similar to (2), we denote the stochastic preference between client i and j as $\phi(i, j)$,

$$\phi(i, j) = \mathbb{P}(R_i > R_j) - 0.5. \tag{19}$$

We formulate our client selection problem as,

$$\min_{S'} \left\{ \sum_{t=1}^{T} \sum_{i \in S'} \phi(i^{(*)}, i) \right\}, \tag{20}$$

where $i^{(*)}$ is the client with the lowest skewness, and S' indicates the set of desirable clients.

Notably, the utilized sample size is controlled by the size of desirable client pool S', where a larger S' allow users to utilize more clients.

C. Dueling based client selection algorithm

Algorithm 2 Select clients

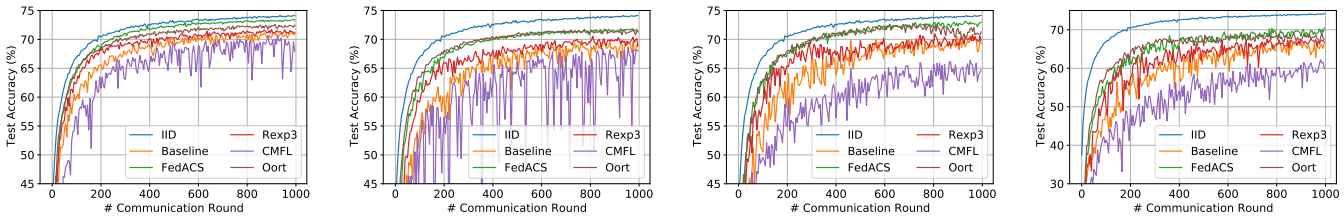
Input: A, B parameters of multi-dueling bandit, C set of clients, κ number of clients to be selected, skewness tolerance λ
Output: S set of selected clients

$S' \leftarrow$ empty set
$P \leftarrow |C| \cdot \lambda$
for $i = 1, 2, ..., P$ **do**
  for $c \in C$ **do**
    sample θ_c by $Beta(A_i + 1, B_i + 1)$
  end for
  $c^* \leftarrow \text{argmax}_c \theta_c$
  append c^* to S'
  remove c^* from C
end for
$S \leftarrow$ randomly draw κ clients from S'
return S

Algorithm 3 Update rewards

Input: A, B parameters of beta distribution of each client, R rewards of participating clients in this round, $R^{<h>}$ historical rewards, η learning rate
Output: A, B updated parameters

$R^{<all>} \leftarrow$ combination of R and $R^{<h>}$ (when one client has rewards in different rounds, use the latest one)
for $i \leftarrow$ clients in R **do**
  for $j \leftarrow$ clients in $R^{<all>}$ **do**
    if $R_i > R_j^{<all>}$ **then**
      $A_i \leftarrow A_i + \eta$
    else if $R_i < R_j$ **then**
      $B_i \leftarrow B_i + \eta$
    end if
  end for
end for
return A, B



(a) Uniform skewness environment (b) Inverse Pareto skewness environment (c) Dirichlet skewness environment (d) Few class skewness environment

Fig. 6. Test accuracy v.s. communication rounds on different skewness environments.

To solve the presented multi-dueling bandit problem, we extend the existing IndSelfSparring, proposed in multi-dueling bandit literature [23], to a more flexible version, named FlexSelfSparring, to better handle our problems. The multi-dueling bandit solution is based on Thompson sampling. It maintains a beta distribution with two parameters A and B for each arm. The bandit functions by sampling from the distribution to select the arms, and updating the parameters based on the dueling result of arms. Algo. 2 describes the procedure of selecting clients. It requires all clients to sample from their own beta distributions, and repeatedly choose clients with the highest sampling result. Algo. 3 describes the procedure of updating the bandit based on the received rewards. It shapes the beta distributions of participating clients. If a client is more likely to defeat others in duels, it will have a higher A value and a lower B value, and its beta distribution will be more likely to return higher results.

Three new features, named *reward transformation*, *sequential selection* and *historical comparison*, are further added to FlexSelfSparring, to improve its performance in FedACS.

Reward transformation In FlexSelfSparring, the raw input of dueling bandit is not the binary result of duelings, but the rewards of arms derived from (16), which is drifting over time. Therefore, we first transform derived rewards into binary dueling results by comparing their R_i before updating the bandits.

Sequential selection In order to increase the raw data utilized by the neural network, we apply parameter λ to explore and maintain a client pool with low skewness, whose size is larger than the number of clients selected each round. After that, clients are randomly sampled from that pool as participants of FL.

Historical comparison Since the raw evidence we obtained to train the bandit is exactly a drifting reward, instead of binary dueling results. Simply converting it into binary dueling in the method mentioned above will eventually lose some valuable information. To completely digest the knowledge from R_i, we create dueling results based on not only R_i in the same round, but also historical R_i in a few former rounds.

For the sequential selection, we emphasize that, unlike vanilla bandit which explores and exploits the top κ lowest skewness clients, our bandit tries to find $\lambda \cdot N$ clients, denoted as S', with low skewness, and randomly pick κ clients from them in each round. This is a simple but effective way to balance the trade-off between client skewness and lacking raw samples. When λ takes its minimum, κ/N, it becomes a vanilla multi-dueling bandit, which always tries to use top κ clients with the lowest skewness. On the contrary, when $\lambda = 1$, it falls back to randomly selecting clients (the default policy in the existing FL protocol). By the introduction of parameter λ, we can both restrict participating clients to be with low skewness, and provide the neural network with sufficient raw sample by extending the client pool.

The approach of historical comparison has both theoretical and experimental supports. Theoretically, dueling bandit allows "noisy" comparison instead of determined ones [22], and therefore admits comparison with historical results even if it is less credible. Experimentally, results from Fig. 4(a) show that rewards for the same kinds of arms do not change drastically in a few rounds.

We next prove that the improved FlexSelfSparring algorithm does not hurt the kernel of IndSelfSparring algorithm so that its theoretical guarantee can be preserved.

Theorem 2. *Under approximate linearity,* FlexSelfSparring *converges to the optimal set of clients S'.*

Proof. See Appendix A-D. □

Theorem 3. *Under approximate linearly,* FlexSelfSparring *converges to the optimal set of clients with an asymptotically optimal no-regret rate of $O(N \ln(T)/\delta)$, where T is the number of communication rounds, and δ is the difference between expected binary dueling results of best two clients.*

Proof. See Appendix A-E. □

In addition to the convergence of the bandit scheme, the assistance of FedACS does not break the convergence of the FL task. The theoretical convergence guarantee is based on [38], which proves the convergence of FedAvg under an arbitrary client selection scheme. Specifically, we can apply the client selection of FedACS in Algo. 2 into Definition 3.2 and Theorem 3.1 of [38] to derive the bound of convergence.

VII. Evaluation

Settings: FedACS is evaluated on a popular dataset, CIFAR-10 [30]. In our experiments, there are 200 clients following three settings of Definitions 3, 4, and 5. In Definition 4, the shape parameter $s = 2$. In Definition 5, the parameters for layering are $x_{med} = 0.2, x_{max} = 3$. Each clients holds $M = 2000$ samples. The simple CNN model from Pytorch tutorial

is used [31]. Local epoch and local batch size are set to $E = 5, B = 400$. Learning rate and learning rate decay per local epoch are set to $\gamma = 0.1, \gamma_d = 0.9993$. The FL model is trained for five repeat trials, and the medians are recorded. In Algo. 3, learning rate of dueling bandit $\eta = 0.2$, and five historical rounds are considered. If not mentioned, skewness tolerance parameter $\lambda = 0.4$.

Baseline and benchmarks: The baseline is the vanilla FL, where the participants are randomly selected. We also implement the case that the data are IID distributed in clients, which represents a theoretical upper bound of FedACS. In addition to these two, we implement three benchmarks from literature for comparison: Oort, CMFL, and Rexp3. Oort [32] and CMFL [26] are two state-of-the-art solutions designed for improving FL performance under the non-IID environment. Oort uses an empirical MAB to select clients, regrading training loss as a reward. CMFL calculates the similarity of clients' gradient and global gradient based on the sign count of dimensions, and removes "diverging" client gradients. Rexp3 [36] is an advanced non-stationary bandit algorithm with a strong regret bound. In our experiments, Rexp3 bandit will be fed with R_i in (16) as rewards, which is the same as FedACS. It can also be seen as an ablation study of FedACS.

A. Performance improvement in FL

Overall performance: Figs. 6(a)-(c) show the performance of FL in different skewness environments. We conclude several evaluation metrics of FedACS and other methods. The first set of metrics is about the test accuracy, we first calculate *terminal accuracy*, as the average test accuracy in the last 50 rounds. Based on it, we conclude *relative improvement*, as the improvement of terminal accuracy, compared to the degrading effect of the skewness environment. The second set of metrics are regarding the convergence speed. We first set the target accuracy of three skewness environments (uniform, inverse Pareto, Dirichlet) to be 70%, 68%, and 68%, respectively, and record rounds have taken for each method to reach the target accuracy. Based on that, we then calculate *speedup* of methods, compared to the baseline.

Results about test accuracy and convergence speed are summarized in Tables II and III[4] respectively. According to experiment results, FedACS significantly improves the performance of FL. In different skewness environments, the accuracy degrading is reduced for 54.1% - 78.2%, and the convergence speeds up for $2.0\times - 2.1\times$. Interestingly, one of our benchmarks, CMFL, turns out to hurt FL, because the manually selected feature it used cannot be an accurate measurement of the client skewness in our data environments. Although Rexp3 shares the skewness measurements with FedACS, it only has a slight improvement on FL. The reason is that the non-stationary bandit performs too conservatively in the exploration-exploitation dilemma to prevent the drifting of optimal arms. Therefore, it is less suitable for our bandit problem than dueling bandits. Oort performs much better than the former two benchmarks, providing a similar speedup to FedACS, but its performance regarding terminal accuracy is outperformed by FedACS.

[4]Some results are not available in the table because CMFL fails to reach the target accuracy

TABLE II
SUMMARY OF TERMINAL ACCURACY AND RELATIVE IMPROVEMENT

Environment	Method	Accuracy (%)	Improvement (%)
IID	-	74.0	100.0
Uniform	baseline	70.8	0.0
	FedACS	73.3	78.2
	Rexp3	71.3	15.1
	CMFL	68.3	−77.8
	Oort	72.3	45.9
Inverse Pareto	baseline	68.8	0.0
	FedACS	71.6	54.1
	Rexp3	69.7	16.2
	CMFL	66.2	−50.1
	Oort	71.3	47.5
Dirichlet	baseline	69.7	0.0
	FedACS	72.7	69.8
	Rexp3	70.1	9.8
	CMFL	64.8	−112.2
	Oort	71.3	37.4

TABLE III
SUMMARY OF ROUNDS TO TARGET AND RELATIVE SPEEDUP

Environment	Method	Rounds	Speedup
Uniform	baseline	490	1.0x
	FedACS	245	2.0x
	Rexp3	400	1.3x
	CMFL	775	0.6x
	Oort	270	1.8x
Inverse Pareto	baseline	520	1.0x
	FedACS	245	2.1x
	Rexp3	365	1.4x
	CMFL	835	0.6x
	Oort	205	2.5x
Dirichlet	baseline	465	1.0x
	FedACS	220	2.1x
	Rexp3	265	0.9x
	CMFL	-	-
	Oort	210	2.2x

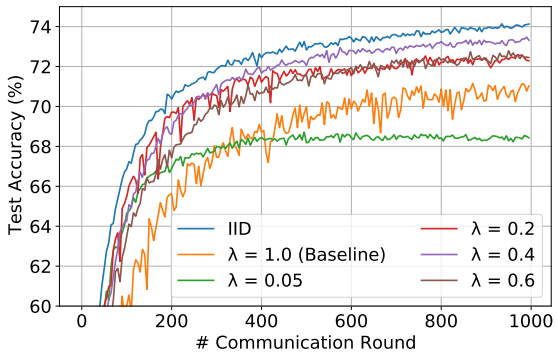


Fig. 7. Performance of FL with different values of λ.

Advantage over Oort: Among the three benchmarks, Oort is the only strong opponent of FedACS. In addition, an irregular is observed regarding the performance of Oort, that its test accuracy decreases in later rounds in Fig. 6(c). We find out that it is partially due to a special mechanism of Oort: it bans the clients whose rounds of participation exceeds a threshold. With this mechanism, some clients with low skewness are banned

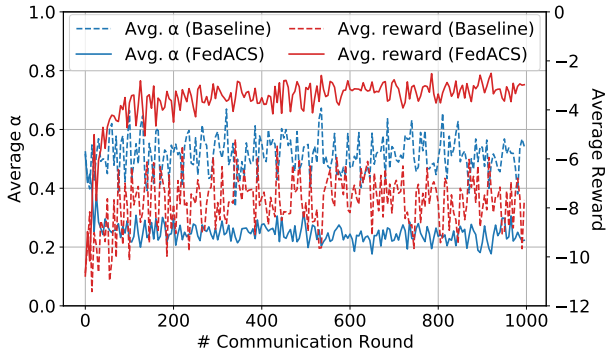


Fig. 8. Average α and R_i values of participating clients over time in uniform skewness environment.

in later rounds. Although removing this mechanism may help, we claim that the mechanism is necessary for Oort. Since Oort regards clients with high training loss as high quality, it will be undefended for potential poisoners. Consider a poisoner whose data are all mistakenly labeled, and it will of course gain high training loss. If no banning mechanism is present, the poisoner will be served as "high quality" and participate in many rounds, which is catastrophic for FL. Meanwhile, FedACS is naturally robust to these poisoners, because poisoners provide gradients diverge from average, and therefore receive a low reward in FedACS.

Severely skewed environment: In many real-world scenarios, the client skewness is expected to be much more severe than in our simulations above, and the clients are unlikely to possess data from all classes. To evaluate the performance under these severely skewed environments, we design the few class (uniform) skewness environment. In such an environment, each client is set similar to the uniform skewness environment, but it only has data from x classes where x is a uniform random variable ranging between 1 and 10. Experiment results in Fig. 6(d) show that FedACS can still outperform the benchmarks in the few class skewness environment, but its relative improvement (47.3%) is smaller than in other environments. The reason is that the clients in the desired pool generated by FedACS are more skewed due to the overall severely skewed environment.

Parameter sensitivity analysis: Skewness tolerance λ is a critical parameter for the performance of FedACS assisted FL. To investigate its influence, we test FedACS on a uniform skewness environment with different values of λ. Experiment results is concluded in Fig. 7. When λ takes its minimum, 0.05, although the harm of skewness is minimized, its performance turns out to be even lower than the baseline. It is due to a severe lacking of data. The performance increases when increasing λ from 0.05 to 0.4, owing to more utilized data, and decreases when changing λ from 0.4 to 1.0, due to the severer skewness of the participants. A good choice of λ should be neither too high, which connives skewed clients, nor too low, which limits the number of utilized samples. Among all values we used, the one with the best performance is 0.4, which is applied in other experiments.

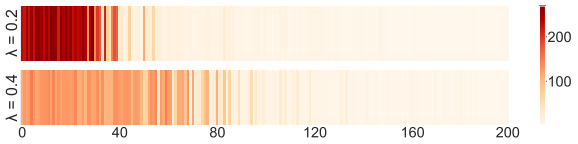


Fig. 9. Clients' participation summary in FL in uniform skewness environment, clients in the left has lower skewness, and vise versa.

B. Details about client selection

Functionality of the bandit: In Fig. 8, two metrics about client selection are presented. The first is the average α values of participating clients in each round. Since lower α indicates lower skewness, this metric helps us identify the overall skewness of participating clients in one round. The second is the average R_i values of participating clients to feed the bandit. This metric helps us identify whether the dueling bandit functions properly to increase the overall reward. Both metrics of Fig. 8 show that FedACS performs client selection successfully. The average α values in FedACS fall quickly, indicating that the skewness of participating clients becomes low. Also, the average R_i becomes higher than baseline, indicating that the bandit succeeds in pursuing a high reward. **Which clients are selected?** In this part, we investigate some details in the client selection procedure. We check whether FedACS functions like our expectation, letting clients with low skewness being more frequently selected. We used results from a uniform skewness environment to verify it, so that we can use α values of clients to represent client skewness. In Fig. 9, we sort and arrange the clients with their α values, where clients in the left have lower α, and therefore with lower skewness. Then, we record how many times each client is being selected as a participant in a whole FL process (1000 rounds). When $\lambda = 0.4$, approximately the first 80 clients are selected to form a client pool, and therefore, the first 80 clients are much more frequently selected than the rest. As for $\lambda = 0.2$, the phenomenon is similar, but the size of the client pool becomes 40. The results show that FedACS accurately locks on the clients with the lowest skewness, and lets them form a pool with fixed sizes as candidates of participants.

C. Reducing overheads

Overhead analysis: Both extra computation and communication are introduced for the FA procedure. For the computation, FedACS asks the clients to perform an extra training process, with only one local epoch, and maximal batch size. Obviously, one local epoch of BGD takes fewer communication resources than the mini-batch approach used by the host FL process. Since an FL epoch has five local epochs, we can derive a loose estimate that the computation overhead introduced by FedACS in our settings is less than 20%.

The communication overhead is derived by comparing the number of weights of the host neural network and the number of weights used by FedACS. In our settings, the neural network has 620,060 parameters, and the number of weights, which is utilized by FedACS, is 8500. Considering the fact that the former has to be transmitted twice (download and upload),

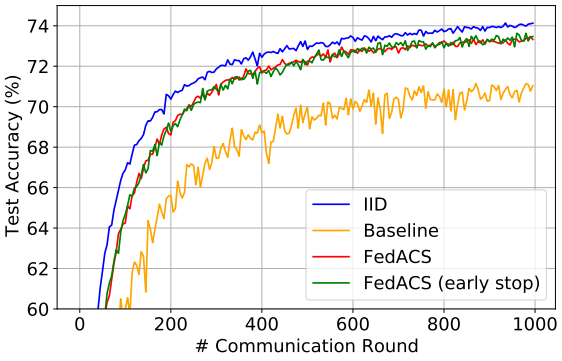


Fig. 10. Performance of early stop.

while the latter is only transmitted once, the communication overhead introduced by FedACS in our settings is 0.69%.

Early stop: The vanilla settings of FedACS require participants to derive and transmit insights every round. However, experiment results in Fig. 8 show that the dueling bandit converges much faster than the FL model, *i.e.*, the FedACS takes much fewer rounds to find out clients with low skewness. Given the finding above, we add a new feature on FedACS, called "early stop". In our experiments, the insight derivation and skewness estimation only happen in the first 200 rounds. Afterward, FedACS only provides client selection, and the participating clients do nothing for FedACS. The choice of the terminal round is based on results in Fig. 8. After round #200, the average α value of participating clients stops decreasing, which indicates the bandit has converged. To validate the performance of the early stop, we test it in a uniform skewness environment. According to results in Fig. 10, early stop does not degrade FL performance at all (the round-accuracy curve almost overlaps with the vanilla FedACS). The early stop is helpful in reducing overhead. **In our settings, with an early stop, FedACS only introduces a communication overhead of 0.14%, and a computation overhead of less than 4%., which are exactly 1/5 of the former values.**

VIII. Related Work

A. Federated paradigm

Both forms of the federated paradigm are provided with numerous applications. Traditional deep learning tasks, borrow the power of FL to exploit data in the edge [11], [12]. On the other hand, FA enables engineers to perform non-training tasks with edge data. For example, FA evaluates the performance of FL models [14], or completes traditional data mining tasks [15], [16].

In addition to FL, there is another concept heavily related to FA: distributed data mining (DDM) [39]. Although DDM considers a similar scenario, where data mining tasks are performed under the distributed scheme, there exists a crucial difference between FA and DDM, that the server and clients are usually owned by different stakeholders in FA, resulting in the untrusted environment. Furthermore, in DDM, data in the clients are usually assigned by the server, which is different from the tenet of FA. The difference between FA and its neighbor concepts are summarized in Table IV and V, respectively.

TABLE IV
Difference between FL and FA

	FL	FA
Goal	Training neural networks	Non-training tasks
Aggregation	FedAvg	Task dependent
Insight	Model weights	Task dependent

TABLE V
Difference between DDM and FA

	DDM	FA
Raw data transmission	Redistribution	Stay where it origins
Clients and server	Trusted	Untrusted
Heterogeneities	Little concerned	Focused

B. Application of the Hoeffding's inequality

The Hoeffding's inequality has been widely applied in various related fields. Data mining, which has a strong relationship with FA, is also the majority of application of the Hoeffding's inequality [40], [41]. In [40], the Hoeffding's inequality is applied in frequent data stream pattern mining. The Hoeffding's tree algorithm, firstly proposed in [41], creates a decision tree in streaming data, and uses the Hoeffding's inequality to decide when a new leaf node should be split. In addition, the Hoeffding's inequality is applied in one of our benchmarks, Oort [32]. In these applications, however, the Hoeffding's inequality is always utilized to calculate the number of samples required given a confidence level. On the contrary, the Hoeffding's inequality is given a novel usage in FedACS, which gives the number of samples (the number of local data) first, and then uses the derived confidence level to infer the likelihood to accept an assumption, which is linked to the client skewness.

C. FL under heterogeneities

The existence of heterogeneities is a key characteristic of FL, and therefore has attracted worldwide interest from researchers. Some researchers tackle the device heterogeneity, which harms the stability and performance of the FL system [42], [43]. On the other hand, being consistent with FedACS, many methods have been proposed to reduce the negative effect of data heterogeneity [8], [25]. In [8], the local datasets in the clients are augmented by the globally shared data to reduce the skewness of the clients. In [25], reinforcement learning helps find clients with higher potential benefits for FL. Personalized Federated Learning, as an emerging variation of traditional FL, breaks the limit of one global model and alleviates the data heterogeneity [24], [44]. FedACS is a universal framework for all kinds of federated tasks. Even if we only consider FedACS assisted FL, there exist several advantages over existing methods. FedACS improves FL performance via intelligent client selection, neither requires extra global data [8] or wastes the updates from the clients

[26]. Compared to its client-selection siblings, FedACS can complete the client selection online [25], and is intrinsically robust to the poisoners [32].

D. Non-stationary bandits and dueling bandits

Non-stationary bandit, originally proposed by Gittens in [34], assumes the case that reward distribution of arms is drifting over time. Several solutions or algorithms have been proposed to tackle the non-stationary bandit problem [35], [36]. In [35], discounted UCB and sliding-window UCB are proposed, with an upper-confidence bound analysis of rounds to suboptimal. In [36], a near-optimal algorithm named Rexp3 is proposed, borrowing the idea of the adversarial bandit to provide a pessimistic regret bound.

Our novel usage of dueling bandit exactly derives higher performance, compared to the non-stationary bandit. Dueling bandit learns from binary dueling result of arm pairs instead of reward values [22]. Traditional dueling bandit algorithms aim at finding a single "strong" arm called Condorect winner. An extension of the dueling bandit is the multi-dueling bandit, which breaks the limit of comparison of two arms [23], [45]. In [45], more than two arms can participate in one dueling, but the target remains to be finding the Condorect winner. In INDSELFSPARRING algorithm, which is applied in FedACS, the target is extended from single Condorect winner to a set of "strong" arms [23].

IX. CONCLUSION

Data heterogeneity is a critical challenge for IIoT deployments and greatly affects the performances of federated learning in IIoT applications. In this paper, we follow the framework of FA to present the first work on federated skewness analytic and client selection, referred to as FedACS. FedACS aims at building an ideal IID environment from massive clients with diverged severity of data heterogeneity. Our proposed FedACS could serve as a standalone federated analytic tool for the distribution characterization purpose or symbiosis with other host federated tasks to improve their quality of services. Specifically, FedACS first uses local-derived insights to infer about clients' data heterogeneity with privacy protection based on the Hoeffding's inequality. After that, it intelligently selects low skewness clients based on a carefully designed dueling bandit solution. Extensive experiments demonstrate that, when assisting federated learning, FedACS reduces the accuracy degrading by ~78.2%, and accelerates the FL's convergence for ~ 2.4×.

REFERENCES

[1] Z. Wang, Y. Zhu, D. Wang, and Z. Han, "Fedacs: Federated skewness analytics in heterogeneous decentralized data environments," in *IEEE/ACM Int. Symp. Qual. Service*, virtual event, Jun. 2021.

[2] Industrial IoT revenue expected to nearly double through 2025; data generation to triple. S&P Global. [Online]. Available: https://f.hubspotusercontent10.net/hubfs/5413615/451_Reprint_Industrial-IoT_29APR2021.pdf

[3] Y. Guo, T. Ji, Q. Wang, L. Yu, G. Min, and P. Li, "Unsupervised anomaly detection in iot systems for smart cities," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2231–2242, Oct. 2020.

[4] A. Hazra, M. Adhikari, T. Amgoth, and S. N. Srirama, "Joint computation offloading and scheduling optimization of iot applications in fog networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 3266–3278, Oct. 2020.

[5] W. Li, Y. Chai, F. Khan, S. R. U. Jan, S. Verma, V. G. Menon, X. Li *et al.*, "A comprehensive survey on machine learning-based big data analytics for iot-enabled smart healthcare system," *Mobile Netw. Appl.*, vol. 26, no. 3, pp. 234–252, Jan. 2021.

[6] 2018 reform of EU data protection rules. European Commission. [Online]. Available: https://ec.europa.eu/commission/sites/beta-political/files/data-protection-factsheet-changes_en.pdf

[7] California consumer privacy act. California Government. [Online]. Available: https://www.oag.ca.gov/privacy/ccpa

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

[9] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the Convergence of FedAvg on Non-IID Data," in *Proc. Int. Conf. Learn. Represent.*, Addis Ababa, Ethiopia, Apr. 2020.

[10] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[11] Y. Liu, A. Huang, Y. Luo, H. Huang, Y. Liu, Y. Chen, L. Feng, T. Chen, H. Yu, and Q. Yang, "Fedvision: An online visual object detection platform powered by federated learning," in *Proc. AAAI Conf. Artif. Intell.*, New York, NY, Apr. 2020, pp. 13 172–13 179.

[12] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, "Applied federated learning: Improving google keyboard query suggestions," *arXiv preprint arXiv:1812.02903*, 2018.

[13] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A survey on federated learning for resource-constrained iot devices," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 1–24, 2021.

[14] Federated analytics: Collaborative data science without data collection. Google AI. [Online]. Available: https://ai.googleblog.com/2020/05/federated-analytics-collaborative-data.html

[15] W. Zhu, P. Kairouz, B. McMahan, H. Sun, and W. Li, "Federated heavy hitters discovery with differential privacy," in *Proc. Int. Conf. Artif. Intell. Statist.*, Palermo, Italy, Jun. 2020, pp. 3837–3847.

[16] D. Ravichandran and S. Vassilvitskii, "Evaluation of Cohort Algorithms for the FLoC API." [Online]. Available: https://github.com/google/ads-privacy/raw/master/proposals/FLoC/FLOC-Whitepaper-Google.pdf

[17] E. Bagdasaryan, P. Kairouz, S. Mellem, A. Gascón, K. Bonawitz, D. Estrin, and M. Gruteser, "Towards sparse federated analytics: Location heatmaps under distributed differential privacy with secure aggregation," *arXiv*, 2021.

[18] Z. Wang, Y. Zhu, D. Wang, and Z. Han, "Fedfpm: A unified federated analytics framework for collaborative frequent pattern mining," in *Proc. IEEE Conf. Comput. Commun.*, 2022.

[19] D. K. Dennis, T. Li, and V. Smith, "Heterogeneity for the win: One-shot federated clustering," in *Proc. Int. Conf. Mach. Learn.*, virtual event, Jul. 2021, pp. 2611–2620.

[20] P. Kairouz, B. McMahan, and V. Smith. Federated Learning Tutorial. [Online]. Available: https://sites.google.com/view/fl-tutorial/home

[21] W. Hoeffding, "Probability inequalities for sums of bounded random variables," in *The Collected Works of Wassily Hoeffding*. Springer, 1994, pp. 409–426.

[22] Y. Yue, J. Broder, R. Kleinberg, and T. Joachims, "The k-armed dueling bandits problem," *J. Comput. Syst. Sci.*, vol. 78, no. 5, pp. 1538–1556, Sep. 2012.

[23] Y. Sui, V. Zhuang, J. W. Burdick, and Y. Yue, "Multi-dueling bandits with dependent arms," *arXiv preprint arXiv:1705.00253*, 2017.

[24] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang, "Personalized cross-silo federated learning on non-iid data," in *Proc. AAAI Conf. Artif. Intell.*, virtual event, Feb. 2021, pp. 7865–7873.

[25] 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.*, Toronto, Canada, Jul. 2020, pp. 1698–1707.

[26] W. Luping, W. Wei, and L. Bo, "CMFL: Mitigating communication overhead for federated learning," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, Dallas, TX, Jul. 2019, pp. 954–964.

[27] T. Yu, E. Bagdasaryan, and V. Shmatikov, "Salvaging federated learning by local adaptation," *arXiv preprint arXiv:2002.04758*, 2020.

[28] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," in *Neurips Workshop Federated Learn.*, Vancouver, Canada, Dec. 2019.

[29] I. B. Aban, M. M. Meerschaert, and A. K. Panorska, "Parameter estimation for the truncated pareto distribution," *J. Amer. Statist. Assoc.*, vol. 101, no. 473, pp. 270–277, Jan. 2006.

[30] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," *Tech. Rep.*, 2009.

[31] "Training a classifier - pytorch tutorials 1.7.1 documentation." [Online]. Available: https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial

[32] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection," in *Proc. USENIX Symp. Oper. Sys. Des. Implementation*, virtual event, Jul. 2021, pp. 19–35.

[33] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2-3, pp. 235–256, May 2002.

[34] J. Gittins, "A dynamic allocation index for the sequential design of experiments," *Progress in statistics*, pp. 241–266, 1974.

[35] A. Garivier and E. Moulines, "On upper-confidence bound policies for switching bandit problems," in *Proc. Int. Conf. Algorithmic Learn. Theory*, Espoo, Finland, Oct. 2011, pp. 174–188.

[36] O. Besbes, Y. Gur, and A. Zeevi, "Stochastic multi-armed-bandit problem with non-stationary rewards," in *Proc. Adv. Neural Inf. Process. Syst.*, Montreal, Canada, Dec. 2014, pp. 199–207.

[37] A. Slivkins and E. Upfal, "Adapting to a changing environment: the brownian restless bandits." in *Proc. Conf. Learn. Theory*, Helsinki, Finland, Jul. 2008, pp. 343–354.

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

[39] L. Zeng, L. Li, L. Duan, K. Lu, Z. Shi, M. Wang, W. Wu, and P. Luo, "Distributed data mining: a survey," *Inf. Technol. Manage.*, vol. 13, no. 4, pp. 403–409, May 2012.

[40] J. X. Yu, Z. Chong, H. Lu, and A. Zhou, "False positive or false negative: mining frequent itemsets from high speed transactional data streams," in *Proc. Int. Conf. Very Large Data Bases*, vol. 4, Toronto, Canada, Aug. 2004, pp. 204–215.

[41] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proc. ACM Int. Conf. Knowl. Discovery Data Mining*, Boston, MA, Aug. 2000, pp. 71–80.

[42] C. Wang, Y. Yang, and P. Zhou, "Towards efficient scheduling of federated mobile devices under computational and statistical heterogeneity," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 2, pp. 394–410, Feb. 2020.

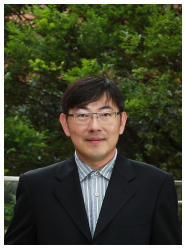
[43] W. Wu, L. He, W. Lin, and R. Mao, "Accelerating federated learning over reliability-agnostic clients in mobile edge computing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1539–1551, Jul. 2020.

[44] Q. Wu, X. Chen, Z. Zhou, and J. Zhang, "Fedhome: Cloud-edge based personalized federated learning for in-home health monitoring," *IEEE Trans. Mobile Comput. (early access)*, Dec. 2020.

[45] Y. Du, S. Wang, and L. Huang, "Dueling bandits: From two-dueling to multi-dueling," in *Proc. 19th Int. Conf. Auton. Agents MultiAgent Syst.*, Auckland, New Zealand, May 2020, pp. 348–356.

Zibo Wang received the B.E. degree in Electrical and Computer Engineering from Shanghai Jiao Tong University in 2020. He is currently pursuing the Ph.D. degree in the same institute. His research interests include federated learning, federated analytics, and privacy computing.

Yifei Zhu (Member, IEEE) is currently an Assistant Professor at Shanghai Jiao Tong University, China. He received the B.E. degree from Xi'an Jiaotong University, Xian, China, in 2012, and the M.Phil. degree from The Hong Kong University of Science and Technology, Hong Kong, China, in 2015, and the Ph.D degree in Computer Science from the Simon Fraser University, BC, Canada, in 2020. His current research interests include edge computing, multimedia networking, and distributed machine learning systems.



Dan Wang (Senior Member, IEEE)'s research falls in general computer networking and systems, where he published in ACM SIGCOMM, ACM SIGMETRICS and IEEE INFOCOM, and many others. He is the steering committee chair of IEEE/ACM IWQoS. He served as the TPC co-Chair of IEEE/ACM IWQoS 2020. His recent research focus on smart energy systems. He won the Best Paper Awards of ACM e-Energy 2018 and ACM Buildsys 2018. He has served as a TPC co-Chair of the ACM e-Energy 2020 and he will serve as General co-Chair of the ACM e-Energy 2022. He is a steering committee member of ACM e-Energy. He serves as a founding area editor of ACM SIGEnergy Energy Informatics Review. His research has been adopted by industry, e.g., Henderson, Huawei, and IBM. He won the Global Innovation Award, TechConnect, in 2017. He got his B.Sc., M.Sc., Ph.D. from Peking University, Case Western Reserve University and Simon Fraser University, all in Computer Science.

Zhu Han (Fellow, IEEE) received the B.S. degree in electronic engineering from Tsinghua University, in 1997, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, in 1999 and 2003, respectively.

From 2000 to 2002, he was an R&D Engineer of JDSU, Germantown, Maryland. From 2003 to 2006, he was a Research Associate at the University of Maryland. From 2006 to 2008, he was an assistant professor at Boise State University, Idaho. Currently, he is a John and Rebecca Moores Professor in the Electrical and Computer Engineering Department as well as in the Computer Science Department at the University of Houston, Texas. His research interests include wireless resource allocation and management, wireless communications and networking, game theory, big data analysis, security, and smart grid. Dr. Han received an NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society in 2011, the EURASIP Best Paper Award for the Journal on Advances in Signal Processing in 2015, IEEE Leonard G. Abraham Prize in the field of Communications Systems (best paper award in IEEE JSAC) in 2016, and several best paper awards in IEEE conferences. Dr. Han was an IEEE Communications Society Distinguished Lecturer from 2015-2018, AAAS fellow since 2019 and ACM distinguished Member since 2019. Dr. Han is 1% highly cited researcher since 2017 according to Web of Science. Dr. Han is also the winner of 2021 IEEE Kiyo Tomiyasu Award, for outstanding early to mid-career contributions to technologies holding the promise of innovative applications, with the following citation: "for contributions to game theory and distributed management of autonomous communication networks."

APPENDIX A
OMITTED PROOFS AND INFERENCE PROCEDURES

A. Proof of lemma 1

Denote the distribution of data in client i as $\mathcal{D}(d_i)$, and the global distribution in all clients as $\mathcal{D}(d)$. Similarly, we denote the distribution of $z_{i,m}^{(k)}$ in each clients and the global distribution as $\mathcal{D}(z_i^{(k)})$ and $\mathcal{D}(z^{(k)})$, respectively.

Supposed we have accepted $\mathcal{H}_0$, then distribution of $d_{i,m}$ in client i is identical to distribution of sum-up data in all clients,

$$\mathcal{D}(d_i) = \mathcal{D}(d). \tag{21}$$

As a mapping of $d_{i,m}$, distribution of $z_{i,m}^{(k)}$ is also identical to the overall distribution,

$$\mathcal{D}(z_i^{(k)}) = \mathcal{D}(z^{(k)}). \tag{22}$$

First, the exception of z_i^k is equal to the exception of $z_{i,m}^k$, since the former is simply arithmetic average of M_i samples of latter, *i.e.*,

$$\mathbb{E}(\Delta w_i^{(k)}) = \mathbb{E}(z_i^{(k)}) = \mathbb{E}(z_{i,m}^{(k)}). \tag{23}$$

Equation (22) gives that distribution of $z_{i,m}^k$ is identical to $z^{(k)}$ for all i, m, their exception is also equal:

$$\mathbb{E}(z_{i,m}^{(k)}) = \mathbb{E}(z^{(k)}). \tag{24}$$

Given these insights, we can rewrite (9) as:

$$\begin{aligned} p_i^{(k)} &= \mathbb{P}(|\Delta w_i^{(k)} - \mathbb{E}(\Delta w_i^{(k)})| \geq \epsilon) \\ &= \mathbb{P}(|\Delta w_i^{(k)} - \mathbb{E}(z^{(k)})| \geq \epsilon) \\ &\leq 2exp\bigg(-\frac{2\epsilon^2 M_i}{(b^{(k)} - a^{(k)})^2}\bigg). \end{aligned} \tag{25}$$

Recall the definition of $\mathcal{H}_0$:

$\mathcal{H}_0$*: Data in a client is IID distributed.*

We define a set of events $\mathcal{A}(\epsilon)$:

$$\mathcal{A}(\epsilon)\text{: } |\Delta w_i^{(k)} - \mathbb{E}(z^{(k)})| \geq \epsilon.$$

When estimating the skewness of a client, we measure the value of ϵ to satisfy $\mathcal{A}(\epsilon)$, and target at the satisfaction of $\mathcal{H}_0$. In other words, we calculates the posterier possibility of $\mathcal{H}_0$ from $\mathcal{A}(\epsilon)$, or $\mathbb{P}(\mathcal{H}_0|\mathcal{A}(\epsilon))$. On the other hand, $p_i^{(k)}$ exactly represents the possibility of $\mathcal{A}(\epsilon)$ assuming the satisfaction of $\mathcal{H}_0$, or $\mathbb{P}(\mathcal{A}(\epsilon)|\mathcal{H}_0)$. Bayesian theorem yields

$$\mathbb{P}(\mathcal{H}_0|\mathcal{A}(\epsilon)) = \frac{\mathbb{P}(\mathcal{A}(\epsilon)|\mathcal{H}_0)\mathbb{P}(\mathcal{H}_0)}{\mathbb{P}(\mathcal{A}(\epsilon))} \tag{26}$$

Since Hoeffding's inequality only provides a possibility bound, instead of the exact value, we conduct a heuristic inference on (26). If the value of $\mathbb{P}(\mathcal{A}(\epsilon)|\mathcal{H}_0)$, or $p_i^{(k)}$, increases, since prior possibility $\mathbb{P}(\mathcal{H}_0)$ is independent form ϵ, at least one of the alternative must be satisfied for the enforcement of (26): 1) $\mathbb{P}(\mathcal{H}_0|\mathcal{A}(\epsilon))$ increases, or 2) $\mathcal{A}(\epsilon)$ increases.

If the first alternative is true, it directly comes to the first alternative of Lemma 1. Supposed that the first alternative is false, then the client is no longer IID. Therefore, its expectation of $z_{i,m}^{(k)}$ now differs from the global one, $z^{(k)}$. We denote the expectation as $\mathbb{E}(z^{(k)}) + \Delta z_i^{(k)}$.

The shape of distribution of $\mathbb{P}(\mathcal{A}(\epsilon))$ is similar to that of $\mathbb{P}(\mathcal{A}(\epsilon)|\mathcal{H}_0)$, while centering at different expectations. With these insights, we can review the second alternative: for any value of $\Delta w_i^{(k)}$, when it is approaching $\mathbb{E}(z^{(k)})$, which leads to an increase of $p_i^{(k)}$, it is also likely to be approaching $\mathbb{E}(z^{(k)}) + \Delta z_i^{(k)}$, which leads to an increase of $\mathbb{P}(\mathcal{A}(\epsilon))$. Obviously, it is only possible when $\Delta z_i^{(k)}$ is relatively small compared to ϵ. Therefore, we can expect a smaller $\Delta z_i^{(k)}$ when we shrink the value of ϵ, which indicates a lower skewness.

In conclusion, the two alternatives to enforce (26) can be inferred as the two alternatives of Lemma 1, that higher $p_i^{(k)}$ indicates a higher likelihood to accept $\mathcal{H}_0$, or directly reveals lower client skewness.

B. Proof of theorem 1

Denote the clients participated in round t as N_t. Consider (7) and (8), we have:

$$\begin{aligned} \mathbb{E}(z^{(k)}) &= \frac{1}{\sum_{i=1}^N M_i} \sum_{i=1}^N \sum_{m=1}^{M_i} z_{i,m}^{(k)} \\ &\approx \frac{1}{\sum_{i\in N_t} M_i} \sum_{i\in N_t} \sum_{m=1}^{M_i} z_{i,m}^{(k)} \\ &= \frac{1}{\sum_{i\in N_t} M_i} \sum_{i\in N_t} M_i \Delta w_i^{(k)} \\ &= \overline{\Delta w}^{(k)}. \end{aligned} \tag{27}$$

From (27), we conclude that the estimation of $\mathbb{E}(z^{(k)})$ is given by the weighted average of uploaded weight changes, weighted by their numbers of data.

Credibility of $\overline{\Delta w}^{(k)}$ as an estimation of $\mathbb{E}(z^{(k)})$ can be analyzed via the Hoeffding's inequality. Recall (27), $\overline{\Delta w}^{(k)}$ is the average of $z_{i,m}^{(k)}$ in all clients in N_t. Equation (1) yields,

$$\begin{aligned} &\mathbb{P}(|\overline{\Delta w}^{(k)} - \mathbb{E}(\overline{\Delta w}^{(k)})| \geq \epsilon) \\ &= \mathbb{P}(|\overline{\Delta w}^{(k)} - \mathbb{E}(z^{(k)})| \geq \epsilon) \\ &\leq 2exp\bigg(-\frac{2\epsilon^2 M_{N_t}}{(b^{(k)} - a^{(k)})^2}\bigg), \end{aligned} \tag{28}$$

where,

$$M_{N_t} = \sum_{i\in N_t} M_i \tag{29}$$

It may seem illogical that the estimation of $\mathbb{E}(z^{(k)})$ in (25), which will be used to give a probabilistic bound by the Hoeffding's inequality, is also bounded by Hoeffding's inequality. However, it is numerically reasonable, because the latter bound is much tighter than the former. When (25) and (28) are given the same confidence level, bound of ϵ in the latter estimate will be M_{N_t}/M_i tighter than the former. Therefore, in (25), the uncertainty given by estimating $\mathbb{E}(z^{(k)})$ is comparatively negligible.

As is discussed in Section V-B, another issue is raised that the credibility of $\overline{\Delta w}^{(k)}$ requires that the distribution of data in

clients of N_t should be identical with the global distribution, which is exactly impractical. However, with our effort to improve the performance of FL, the skewness of participating clients is gradually decreasing, making our estimation effective in practice.

C. Derivation of Q_i

The skewness estimation of client i by multiplying all utilized dimensions are given by

$$\begin{aligned} P_i &= \prod_{k=1}^{\hat{K}} 2exp\bigg(- \frac{2(\epsilon^{(k)})^2 M_i}{(b^{(k)} - a^{(k)})^2} \bigg) \\ &= 2^{\hat{K}} \prod_{k=1}^{\hat{K}} exp\bigg(- \frac{2(\epsilon^{(k)})^2 M_i}{(b^{(k)} - a^{(k)})^2} \bigg). \end{aligned} \tag{30}$$

Recall that higher $P_i^{(k)}$ indicates lower skewness, and the range of $P_i^{(k)}$ is $[0, 1]$. Therefore, a higher P_i also indicates lower skewness.

$b^{(k)}$ and $a^{(k)}$ are the upper and lower bounds of $z_{i,m}^k$. A normal method is requesting the minimum and maximum from all participating clients and deriving the tightest bound. However, it increases the communication overhead by 2×, and breaks the strict privacy restriction of FL. Therefore, we use a looser bound, which is equal for all dimensions. Denote them as b_{max} and a_{min}, *i.e.*,

$$b_{max} = \max_{\forall i,m,k} \left(z_{i,m}^{(k)}\right),\ a_{min} = \min_{\forall i,m,k} \left(z_{i,m}^{(k)}\right) \tag{31}$$

Since (1) only requires a and b as bounds, without requirement of tightness, we are able to use b_{max} and a_{min} to take place of $b^{(k)}$ and $a^{(k)}$ in all $\hat{K}$ dimensions, without loss of mathematical correctness. Rewrite (30),

$$P_i = 2^{\hat{K}} \prod_{k=1}^{\hat{K}} exp\bigg(- \frac{2(\epsilon^{(k)})^2 M_i}{(b_{max} - a_{min})^2} \bigg). \tag{32}$$

Take the logarithm on both sides, and simplify the form,

$$\frac{(\hat{K} \ln 2 - P_i)(b_{max} - a_{min})^2}{2} = M_i \sum_{k=1}^{\hat{K}} \left((\epsilon^{(k)})^2\right) \tag{33}$$

Recall (12), we can find that the sum of $(\epsilon^{(k)})^2$ among all dimensions is the square of the L_2 norm between Δw_i and $\overline{\Delta w}$, and denote the final form as Q_i:

$$\begin{aligned} Q_i &= \sqrt{\frac{(K \ln 2 - P_i)(b_{max} - a_{min})^2}{2}} \\ &= \sqrt{M_i} ||\Delta w_i - \overline{\Delta w}||_2 \end{aligned} \tag{34}$$

where,

$$\overline{\Delta w} = \frac{1}{\sum_{i \in N_t} M_i} \sum_{i \in N_t} M_i \Delta w_i \tag{35}$$

Eq. (34) shows that Q_i is an inverse transformation of P_i, *i.e.*, lower Q_i indicates lower skewness.

D. Proof sketch of theorem 2

We emphasize that compared to IndSelfSparring, FlexSelfSparring only breaks the limit that the size of the target arm set (λN) should be equal to the number of arms selected each round (κ). With our modification, the kernel of IndSelfSparring, conversion to the Thompson sampling, remains unchanged. Therefore, the proof of Theorem 2 is consistent with the proof of Theorem 1 in [23].

Here we briefly sketch the proof: first we show that in the infinite horizon, each client will be selected for infinite times. As a result, parameters of beta distribution (A and B) will permanently increase, and the beta distributions will eventually converge to the Dirac distribution. Therefore, the bandit will converge to optimal clients, since the sample result of the Dirac distribution is constant. The detailed proof is available in [23].

E. Proof sketch of theorem 3

As is described in Appendix A-D, our modifications on FlexSelfSparring does not hurt the Thompson sampling process in IndSelfSparring. Therefore, the proof of Theorem 3 is consistent with the proof of Theorem 2 in [23].

Here we briefly sketch the proof: FlexSelfSparring utilizes Thompson sampling, where the dueling results are considered as binary rewards, and is handled in the way of stochastic bandits. First, we show that compared to someone drawing clients with a fixed probability distribution, a player who utilizes Thompson sampling can have the optimal regret, which is the theoretical basis of each stochastic bandit algorithm. Then, the superiority of Thompson sampling extends from playing against a fixed distribution to a drifting but converging probability distribution. Finally, by merging the above insight and Theorem 2, the no-regret rate is calculated based on the definition of approximate linearity. The detailed proof is available in [23].