Budget-Constrained Federated Bandits for Mobile Applications

Anran Xu[1,2], Zhenzhe Zheng[2†], Wenming Zheng[1], Fan Wu[2]
[1]China Telecom, China
[2]Shanghai Jiao Tong University, China
E-mail: {xuar2, zhengwenm}@chinatelecom.cn, zhengzhenzhe@sjtu.edu.cn, fwu@cs.sjtu.edu.cn

***Abstract*—With the development of federated learning and mobile computing, federated bandit frameworks have been proposed to enable multiple clients to explore and exploit collaboratively under the guarantee of data privacy. However, none of them take the widespread budget constraints into account. In this work, we introduce the Federated linear Bandits framework with Knapsacks (FBwK), where M clients can pull K arms with linear rewards and costs to minimize the total regret under the coordination of a central server. For FBwK, we propose a new definition of OPT solution and a bandit algorithm, namely FedUCBwK. Specifically, we design a uniform policy parameter update threshold for each client, which balances regret, communication, and computation costs. In this process, the central server uses a factor on knapsack constraints to pace the budget consumption and preserve the clients' privacy by transmitting model updates instead of raw data. We conduct a theoretical analysis and show that FedUCBwK achieves a sublinear regret with logarithmic communication and computation. Evaluation on real-world mobile application datasets demonstrates that FedUCBwK consistently outperforms existing methods across diverse tasks.**

***Index Terms*—federated bandits, bandits with knapsacks, federated learning**

I. Introduction

Many real-world mobile applications, such as personalized recommendations [1], [2], mobile advertising [3], [4], crowdsensing [5], [6], require sequential decision-making under uncertainty, where agents must select actions over time to maximize long-term cumulative rewards. The multi-armed bandit (MAB) framework [7] serves as a canonical model for such problems, characterizing the fundamental trade-off between exploration and exploitation. With the rise of federated learning (FL), decentralized decision-making has become viable across privacy-sensitive and bandwidth-constrained clients. Federated bandits [8], [9] extend MABs to this setting, allowing clients to interact with local environments while jointly optimizing global policies via a central server, without sharing raw data. However, existing work does not account for a key real-world challenge: resource and budget constraints that persist throughout the learning process [10], as shown in Fig. 1.

Fig. 1. Resource-Bounded Federated Bandits for Mobile Applications.

In many mobile applications, budget constraints are fundamental to how decisions are made and enforced in practice. These constraints are typically imposed by a central controller (*e.g.*, a server) based on overall financial, operational, or policy limits, and are consumed incrementally by the actions of individual clients. For instance, in mobile advertising [3], an advertiser allocates a global advertising budget, and each client consumes part of it whenever an ad is displayed. In wireless network configuration [11], a mobile network operator limits the total number of parameter changes to avoid service disruptions and energy overhead. Similarly, in crowdsourcing platforms [12], [13], the central platform allocates a fixed incentive budget, and each task assignment deducts from this budget to reward participating workers. These diverse scenarios share a common structure: each action incurs a cost, and the overall policy must optimize utility while satisfying global budget constraints. While prior FL works consider resource constraints, they predominantly focus on communication or computation overhead [14], [15] in mobile applications, which differ fundamentally from the hard budget bounds on action-level costs we consider.

To capture this structure, we formally model the problem as Federated Bandits with Knapsacks (FBwK), a novel formulation where multiple clients collaboratively learn arm-selection strategies under a shared global budget constraint. While prior work has made significant progress on Bandits with Knapsacks (BwK) [10], [16], [17] and FL [8], [9], [18], [19], their solutions cannot be directly applied to the FBwK

This work was supported in part by National Key R&D Program of China (No. 2023YFB4502400), in part by China NSF grant No. U2268204, 62322206, 62432007, 62132018, 62025204, 62272307, 62372296. Thanks to Xutong Liu, Shengjie Wang and Shuai Li for their valuable discussions and insightful comments. The opinions, findings, conclusions, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies or the government. †Zhenzhe Zheng is the corresponding author.

setting due to the following key challenges:

The first challenge lies in the high computational cost required to satisfy knapsack constraints on the client side. A standard approach to BwK[1] [10], [16] solves a linear program (LP) in every decision round. However, mobile clients typically lack the capacity to perform complex optimization procedures [20], making these methods impractical in federated environments, because the computational overhead becomes especially prohibitive as the number of clients or arms increases. Oracle-based alternatives for contextual BwK [17], [21] are theoretically attractive, but they rely on strong assumptions about supervised learning subroutines [22], and are rarely feasible in real-world mobile settings. Thus, to enable scalable and deployable FBwK algorithms, it is essential to design lightweight client-side solutions that reduce the number of LP computations while still offering provable regret bounds and competitive empirical performance.

The second challenge arises from the interplay between communication and computation costs in FBwK. Existing federated bandit algorithms [8], [23] primarily consider communication overhead, but overlook the significant computational burden imposed by solving BwK subproblems on clients. Effective learning in FBwK requires a synchronization scheme that jointly coordinates communication and local computation. If clients update policies purely locally without timely communication, they cannot leverage complementary information from others, leading to biased updates. Conversely, if clients communicate frequently without sufficient local training, the exchanged information may be poorly optimized and offer little value. Crucially, synchronization in FBwK is not a simple combination of existing approaches. This is because they use different intuitions to ensure a lower regret[2], and naively merging them may result in compounded errors and high cumulative regret. Therefore, a unified synchronization scheme is needed, which jointly optimizes communication, computation, and regret, with formal guarantees.

The third challenge comes from the need to preserve the privacy of the client. Prior works [8], [15], [25] often add noise or perturbations to raw data to ensure differential privacy. However, such mechanisms inevitably degrade model performance and may still risk privacy leakage in high-dimensional settings. To mitigate this, we follow the traditional federated learning paradigm [26] and transmit only model parameters instead of raw data. However, in the federated bandit setting, the transmitted parameters are local estimates rather than fully trained models, which may lack sufficient information to accurately recover global statistics or compute effective arm-selection strategies. This necessitates a new parameter aggregation and estimation approach tailored for federated bandits, one that preserves privacy without sacrificing decision quality.

In response to these challenges, we develop a unified solution that balances computation, communication, and privacy, while respecting global budget constraints and providing theoretical guarantees. We summarize our main contributions as follows.

- We formulate the federated linear bandits with knapsacks (FBwK) problem to model sequential decision-making in mobile applications under global knapsack-style resource constraints. We introduce a new regret benchmark (OPT) and prove that it serves as a valid upper bound on the global optimum by decomposing the FBwK problem across multiple clients.
- We propose a novel algorithm, FedUCBwK, to solve the FBwK problem under a linear reward/cost model and static context setting. FedUCBwK coordinates computation and communication via a unified epoch-based threshold: it triggers policy updates at clients and parameter aggregation at the server. To avoid budget overspending, a budget-control factor is integrated. Privacy is preserved by transmitting model updates rather than raw data. We further provide theoretical guarantees for FedUCBwK, including a regret bound of $O\left(d\sqrt{MT}\log(MT)\left(\frac{\mathrm{OPT}}{B}+1\right)\right)$ with high probability, and communication and computation costs of $O\left(Md^2(d+K)\log MT\right)$ and $O(Md\log MT)$, respectively.
- We conducted extensive experiments in two real-world scenarios: *mobile ad recommendation* and *mobile crowdsourcing*. The results show that FedUCBwK significantly reduces cumulative regret by up to 94. 47% compared to state-of-the-art federated bandit methods, while achieving logarithmic communication and computation costs.

II. Related work

The multi-armed bandit (MAB) problem is a classical framework for sequential decision-making under uncertainty, balancing exploration and exploitation [27]–[29]. In contextual bandits, each arm is associated with a context vector, and the expected reward is modeled as a function of the context [2], [30]. Linear contextual bandits, in particular, have been extensively studied [31], [32], achieving $\tilde{O}(\sqrt{T})$ regret bounds using UCB-style strategies. These models have been successfully applied to various real-world decision-making problems [33]–[36].

Distributed and Federated Bandits Recent efforts have extended MABs to distributed and federated settings, addressing privacy, coordination, and communication efficiency [8], [15], [23], [25], [37]–[40]. Early works explored collision-avoidance strategies [37] and cooperative estimation under delayed feedback and limited communication [23], [38]. More recently, federated bandits (FB), inspired by federated learning, have gained attention. Several works focused on privacy-preserving algorithms via secure aggregation or differential

[1]The BwK problem is the sub-problem solved locally by each client. Please refer to Section IV for more technical details.

[2]Methods to reduce communication costs in the federated setting are designed to address parameter estimation deviation caused by communication delays [18], [24], while those for reducing computation costs under knapsack constraints aim to manage the deviation due to budget loss [17], [21].

privacy [8], [25], while others designed communication-efficient protocols under master-worker or decentralized frameworks [15], [18], [19]. Huang *et al.* [9] proposed a linear contextual FB model with parameter-only transmission. Recent works also consider agent heterogeneity [39]–[41] and combinatorial extensions [42], [43]. However, none of these approaches consider the resource constraints (*e.g.*, budget or cost per action) that naturally arise in practical applications such as mobile recommendation or crowdsourcing, which motivates our study on federated bandits with knapsacks.

Bandits with Knapsacks BwK introduces budget or resource constraints into the MAB framework. BwK is first studied by Badanidiyuru *et al.* [10], who proved that the regret achieved by both algorithms is optimal up to polylogarithmic factors. Later, based on the optimal regret, Agrawal *et al.* [16] further proposed optimal algorithms for concave rewards and convex knapsacks. These works focus on static context or context-free bandits. The contextual BwK is first considered by Agrawal *et al.* [17], [21], who proposed computationally efficient solutions based on an optimization oracle. Recent studies expand into combinatorial [44], adversarial [45], [46], and other contextual extensions [47]. However, these assume centralized access to data and ignore client-server privacy and communication challenges.

III. Preliminaries

We consider a federated linear bandit with knapsack constraints under the server-clients framework. There are M clients, each accessing the same set of K actions (arms) denoted as $[K] := \{1, 2, \ldots, K\}$. At each time slot $t \in [T]$, each client $m \in [M]$ pulls an arm $a_{t,m} \in [K]$ and observes both a reward $r_{t,m}$ and a resource consumption $c_{t,m}$, which depend on the client-specific context vector $\boldsymbol{x}(a_{t,m}) \in \mathbb{R}^d$ associated with the selected arm. To simplify the budget allocation problem, we assume that each context vector $\boldsymbol{x}(a_{t,m})$ is drawn from a fixed distribution, *i.e.*, the context is static over time. This modeling assumption is practical in many scenarios; for example, an advertiser with a fixed budget often allocates it across similar advertisements or batches of the same product type. Clients collaboratively perform periodic policy updates and communicate under the coordination of a central server to optimize the global objective. A fixed total budget $B \in \mathbb{R}_+$ imposes a hard constraint on cumulative resource consumption. The algorithm terminates at the earliest time τ when the total consumption reaches B.

Linear Structure of Rewards and Costs To capture the correlation in arm outcomes across clients, we assume linear models for both rewards and costs. Specifically, the reward is given by $r_{t,m} = \langle \boldsymbol{\theta}, \boldsymbol{x}(a_{t,m}) \rangle + \eta^r_{t,m}$,where $\boldsymbol{\theta} \in \mathbb{R}^d$ is an unknown parameter vector and $\eta^r_{t,m}$ is zero-mean noise bounded in $[-1, 1]$. Similarly, the cost is modeled as $c_{t,m} = \langle \boldsymbol{\beta}, \boldsymbol{x}(a_{t,m}) \rangle + \eta^c_{t,m}$, where $\boldsymbol{\beta} \in \mathbb{R}^d$ is unknown and $\eta^c_{t,m}$ is zero-mean noise bounded in $[-1, 1]$. This linear assumption is widely adopted in contextual bandits [2], [9], [17], [30], offering both modeling flexibility and theoretical tractability.

Computation Each client updates its policy by solving a BwK subproblem formulated as a linear program (LP), as detailed in Section IV. We define the computation cost as the total number of optimization problems solved throughout the algorithm.

Communication We assume clients periodically communicate with the server under negligible latency. In each round, clients transmit locally accumulated parameter estimates to the server, which aggregates them into a global estimate and broadcasts it back. We define the communication cost as the total number of server–client communication rounds.

Privacy We assume static client contexts, *i.e.*, user characteristics and data distributions remain unchanged during a period. Although not time-varying, the context $\boldsymbol{x}(a_{t,m})$ represents personal information, and the reward $r_{t,m}$—such as user feedback in recommendation or advertising systems—is also sensitive. Therefore, both $\boldsymbol{x}(a_{t,m})$ and $r_{t,m}$ are kept private throughout this work.

Regret The objective of the central server is to minimize the expected cumulative regret across all clients:

$$\text{Regret}(T) = \text{OPT} - \left(\sum_{t=1}^{T} \sum_{m=1}^{M} \langle \boldsymbol{\theta}, \boldsymbol{x}(a_{t,m}) \rangle \right), \qquad (1)$$

subject to the global budget constraint: $\sum_{i=1}^{M} \sum_{t=1}^{T} c_{t,m} \leq B$. Here, OPT denotes the total expected reward for the optimal policy, as formally defined in Section IV. The goal is to design an FBwK algorithm that minimizes regret while ensuring efficient communication and computation, and satisfying client-side privacy constraints.

IV. OPT for FBwK

As mentioned above, we define regret as the gap between the total reward achieved by the bandit algorithm and the OPT reward obtained by an ideal dynamic policy, maximizes the expected cumulative reward under the budget constraint, assuming full knowledge of reward and cost distributions. To facilitate regret analysis in FBwK, we first introduce a linear programming (LP) relaxation of OPT, denoted as OPT-LP. Since our setting involves a distributed (federated) system, we further extend OPT-LP to capture the decentralized nature of the problem, and refer to this relaxation as OPT-FED.

A. OPT-LP

We define a linear programming relaxation for maximizing the expected total reward under a mixed policy over arms. Given expected reward vector $\boldsymbol{\theta}^\top \boldsymbol{X}$ and expected cost vector $\boldsymbol{\beta}^\top \boldsymbol{X}$ for each arm, the relaxed problem is formulated as:

$$\begin{array}{ll} \max_{\boldsymbol{p}} & \boldsymbol{\theta}^\top \boldsymbol{X} \boldsymbol{p} \\ \text{s.t.} & \boldsymbol{\beta}^\top \boldsymbol{X} \boldsymbol{p} \leq \frac{B}{T} \\ & \boldsymbol{p} \in \Delta \end{array}, \qquad (2)$$

where $\boldsymbol{X} \in \mathbb{R}^{d\times K}$ is the matrix of context vectors for the K arms, and $\Delta = \{\boldsymbol{p} : \sum_{i=1}^{K} p_i = 1, p_i \geq 0, i = 1,\ldots,K\}$ is the probability simplex over arm-selection probabilities, where p_i is the probability of pulling arm i. Let $\text{LP}(\boldsymbol{\theta}^\top \boldsymbol{X}, \boldsymbol{\beta}^\top \boldsymbol{X})$ denote the value of this LP-relaxation. It can be shown that [10]

$$\text{LP}(\boldsymbol{\theta}^\top \boldsymbol{X}, \boldsymbol{\beta}^\top \boldsymbol{X}) \geq \frac{\text{OPT}}{T}. \tag{3}$$

Hence, $T \cdot \text{LP}(\boldsymbol{\theta}^\top \boldsymbol{X}, \boldsymbol{\beta}^\top \boldsymbol{X})$ provides a valid upper bound for OPT and is denoted as OPT-LP.

B. OPT-FED

In the federated BwK setting with M clients, budget allocation across clients becomes essential. Under the static context assumption, the centralized BwK problem can be decomposed by distributing the budget equally among clients. We first allocate the total budget B evenly across T rounds, and further divide each per-round budget equally among M clients, yielding an individual client budget of $B/(MT)$. For each client $m \in [M]$, the LP relaxation becomes:

$$\begin{array}{ll} \max_{\boldsymbol{p_m}} & \boldsymbol{\theta}^\top \boldsymbol{X} \boldsymbol{p}_m \\ \text{s.t.} & \boldsymbol{\beta}^\top \boldsymbol{X} \boldsymbol{p}_m \leq B_m \\ & \boldsymbol{p}_m \geq \boldsymbol{0} \end{array}, \tag{4}$$

where B_m is the budget allocated to the client m, and $\boldsymbol{p}_m$ denotes its local arm-selection distribution. As the contexts are static, we can use a uniform $\boldsymbol{X}$ for all clients. We use $\boldsymbol{p}_m \geq \boldsymbol{0}$ instead of $\boldsymbol{p}_m \in \Delta$ for the convenience of proof, as the latter solution can be normalized as a specific solution for the former. The dual problem of Equation (4) is:

$$\begin{array}{ll} \min_{y_m} & B_m y_m \\ \text{s.t.} & \boldsymbol{\beta}^\top \boldsymbol{X} y_m \geq \boldsymbol{\theta}^\top \boldsymbol{X} \\ & y_m \geq 0 \end{array}. \tag{5}$$

Suppose there is an optimal solution for the above LP, then we have $B_m \geq \min(\boldsymbol{\beta}^\top \boldsymbol{X})$ for each client m. An intuitive understanding is that the allocated budget must be higher than the arm with the lowest cost. Otherwise, there will be no arm that can be pulled. Strong duality [48] ensures that if a feasible solution exists, then: $\boldsymbol{\theta}^\top \boldsymbol{X} \boldsymbol{p_m} = B_m y_m$. Because all clients share the same context matrix $\boldsymbol{X}$, they will obtain the same dual variable $y_m = y$. The total expected reward across M clients is:

$$R = \sum_{m\in M} B_m y_m = By = M\frac{B}{M} \cdot y_m. \tag{6}$$

This shows that the total reward is invariant to client-level budget decomposition, as long as all clients solve the same LP. In practice, once a client's budget is exhausted, it ceases to participate in arm selection.

Finally, for any $\boldsymbol{\theta}^\top \boldsymbol{X}, \boldsymbol{\beta}^\top \boldsymbol{X}$, let $\text{LP-FED}(\boldsymbol{\theta}^\top \boldsymbol{X}, \boldsymbol{\beta}^\top \boldsymbol{X})$ denote the value of the following linear program.

$$\begin{array}{ll} \max_{\boldsymbol{p}} & \boldsymbol{\theta}^\top \boldsymbol{X} \boldsymbol{p} \\ \text{s.t.} & \boldsymbol{\beta}^\top \boldsymbol{X} \boldsymbol{p} \leq \frac{B}{MT}. \\ & \boldsymbol{p} \in \Delta \end{array} \tag{7}$$

We replace OPT-LP with $MT \cdot \text{LP-FED}(\boldsymbol{\theta}^\top \boldsymbol{X}, \boldsymbol{\beta}^\top \boldsymbol{X})$, which is denoted as OPT-FED in the analysis of regret.

V. Design of FedUCBwK

In this section, we introduce FedUCBwK, which balances regret, communication, and computation through epoch-based coordination. The core idea is to improve efficiency by jointly controlling the timing of policy updates and server–client communication. The design is based on a key observation that within a short period, the accumulation of new information (*i.e.*, selected arms and observed rewards/costs) is limited, leading to minimal changes in parameter estimates for $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$. Therefore, we can divide the time horizon into multiple epochs, where both LP-based policy updates and parameter aggregation are performed only once per epoch. To segment these epochs, FedUCBwK evaluates the change in information gain and applies a unified threshold that jointly accounts for estimation variance and the cost of computation and communication. Additionally, a budget control factor is introduced to prevent overspending and ensure feasibility under the global budget constraint. By coordinating these components, FedUCBwK achieves a principled trade-off among regret, communication, and computation, supported by rigorous theoretical guarantees.

We now describe how FedUCBwK ensures client-side privacy. The algorithm transmits only the estimated parameters $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$ between the server and clients, while keeping all raw data local to each client. Additionally, instead of sharing the global budget B, the server distributes a per-client budget of B/M, so that each client operates independently without access to the total budget.

The algorithms for client m and the central server in FedUCBwK are shown in Algorithm 1 and 2, respectively. At initialization, each client m pulls each arm once (Lines 3-4, Alg.1), and observes the feedback (the corresponding reward and cost) to initialize local estimates $\hat{\boldsymbol{\theta}}_{m,a}$ and $\hat{\boldsymbol{\beta}}_{m,a}$. After initialization, FedUCBwK proceeds in epochs until termination (*i.e.*, the time horizon T is reached or the budget is exhausted). In each epoch e, client m selects an arm based on the policy computed at the end of the previous epoch (Line 6, Alg. 1). It then records the number of times each arm a is played ($f(m,a)$), update the Gram matrix $\boldsymbol{V}_{t,m}$ (the sum of outer products of the pulled contexts), accumulates both the total reward $R_{m,a}$ and total cost $C_{m,a}$ for each arm (Lines 7-11, Alg.1). The Gram matrix $\boldsymbol{V}_{t,m}$ of client m captures the accumulated information about arm selection up to time t, and directly influences the variance of parameter estimates $\hat{\boldsymbol{\theta}}_{m,a}$ and $\hat{\boldsymbol{\beta}}_{m,a}$. To determine when to synchronize with the central server, FedUCBwK evaluates the following condition:

$$\frac{\det(\boldsymbol{V}_{t,m})}{\det(\boldsymbol{V}_{t_0,m})} > D, \tag{8}$$

where t_0 is the start time of the current epoch and D is a predefined threshold. The determinant ratio quantifies the relative increase in accumulated information during the epoch. Once the threshold is exceeded, communication with the server is triggered (Line 12, Alg. 1). Using the total number of times arm a is selected ($f(m,a)$) and the cor-

Algorithm 1: FedUCBwK: The Client m

Input: Time slots T, Threshold D, Budget factor ϵ

1 Receive the budget B/M, $e \leftarrow 0$, $t \leftarrow 0$;
2 **while** *not satisfy the stop condition* **do**
3 **if** $t \leq K$ **then**
4 Pull arm $a_{t,m} = t$;
5 **else**
6 Pull arm $a_{t,m}$ according to probability $\boldsymbol{p}^{e-1}$;
7 Update $f(m, a_{t,m}) \leftarrow f(m, a_{t,m}) + 1$;
8 Get the reward $r_{t,m}$ and the consumption $c_{t,m}$;
9 Update $\boldsymbol{V}_{t,m} \leftarrow \boldsymbol{V}_{t,m} + \boldsymbol{x}(a_{t,m})\boldsymbol{x}(a_{t,m})^\top$;
10 Update $R_{m,a} \leftarrow R_{m,a} + r_{t,m}$;
11 Update $C_{m,a} \leftarrow C_{m,a} + c_{t,m}$;
12 **if** $(\det \boldsymbol{V}_{t,m} / \det \boldsymbol{V}_{t_0,m}) > D$ *or* $t = K$ **then**
13 $\hat{\boldsymbol{\theta}}^e_{m,a} \leftarrow \left(\frac{R_{m,a}}{f(m,a)}\right) \frac{\boldsymbol{x}(a_{t,m})}{\|\boldsymbol{x}(a_{t,m})\|^2}$;
14 $\hat{\boldsymbol{\beta}}^e_{m,a} \leftarrow \left(\frac{C_{m,a}}{f(m,a)}\right) \frac{\boldsymbol{x}(a_{t,m})}{\|\boldsymbol{x}(a_{t,m})\|^2}$;
15 Send $\hat{\boldsymbol{\theta}}^e_{m,a}$, $\hat{\boldsymbol{\beta}}^e_{m,a}$, $f(m,a)$ for each a to the central server;
16 Receive $\hat{\boldsymbol{\theta}}^e$, $\hat{\boldsymbol{\beta}}^e$,$\boldsymbol{V}^e$ from the central server to calculate $\tilde{\boldsymbol{\theta}}^e$ and $\tilde{\boldsymbol{\beta}}^e$;
17 Solve $\begin{matrix}\max_{\boldsymbol{p}^e \in \Delta} & \boldsymbol{X}^\top \tilde{\boldsymbol{\theta}}^e \boldsymbol{p}^e \\ \text{s.t.} & \boldsymbol{X}^\top \tilde{\boldsymbol{\beta}}^e \boldsymbol{p}^e \leq (1-\epsilon)\frac{B}{TM}\end{matrix}$;
18 $\boldsymbol{V}_{t_0,m} \leftarrow \boldsymbol{V}_{t,m}$, $e \leftarrow e + 1$;
19 $t \leftarrow t + 1$;

Algorithm 2: FedUCBwK: The Central Server

Input: Time slots T, Number of clients M, Budget B

1 **Initialization:** Send B/M to each client;
2 $e \leftarrow 0$;
3 **while** *not reaching the time horizon* T **do**
4 **if** *A communication round is started* **then**
5 Receive $\hat{\boldsymbol{\theta}}^e_{m,a}$, $\hat{\boldsymbol{\beta}}^e_{m,a}$ from each client m;
6 $\boldsymbol{V}^e \leftarrow \sum_{m\in[M]} \sum_{a\in[K]} f(m,a) \frac{\hat{\boldsymbol{\theta}}^e_{m,a} (\hat{\boldsymbol{\theta}}^e_{m,a})^\top}{\|\hat{\boldsymbol{\theta}}^e_{m,a}\|^2}$;
7 $\hat{\boldsymbol{\theta}}^e \leftarrow (\boldsymbol{V}^e)^{-1} (\sum_{m\in[M]} \sum_{a\in[K]} f(m,a) \hat{\boldsymbol{\theta}}^e_{m,a})$;
8 $\hat{\boldsymbol{\beta}}^e \leftarrow (\boldsymbol{V}^e)^{-1} (\sum_{m\in[M]} \sum_{a\in[K]} f(m,a) \hat{\boldsymbol{\beta}}^e_{m,a})$;
9 Send $\hat{\boldsymbol{\theta}}^e$, $\hat{\boldsymbol{\beta}}^e$, and $\boldsymbol{V}^e$ to each client;
10 $e \leftarrow e + 1$;

responding cumulative reward $R_{m,a}$, client m estimates the reward parameter $\hat{\boldsymbol{\theta}}^e_{m,a}$ via projected linear regression:

$$\hat{\boldsymbol{\theta}}^e_{m,a} = \left(\frac{R_{m,a}}{f(m,a)}\right) \frac{\boldsymbol{x}(a_{t,m})}{\|\boldsymbol{x}(a_{t,m})\|^2}. \tag{9}$$

Similarly, the cost parameter $\hat{\boldsymbol{\beta}}^e_{m,a}$ is estimated using the cumulative consumption $C_{m,a}$ in the same manner (Lines 13–14, Alg. 1).

After local estimation, all clients send their calculated parameters and arm pull counts to the server for aggregation (Line 15, Alg.1). Due to privacy-preserving constraints, the server only has parameters $\hat{\boldsymbol{\theta}}^e_{m,a}$ and $\hat{\boldsymbol{\beta}}^e_{m,a}$ for each arm, without observing the corresponding context vectors $\boldsymbol{x}(a_{t,m})$. Since both $\hat{\boldsymbol{\theta}}^e_{m,a}$ and $\boldsymbol{x}(a_{t,m})$ are in the same direction, the server can recover the normalized context direction as $\hat{\boldsymbol{\theta}}^e_{m,a} / \left\|\hat{\boldsymbol{\theta}}^e_{m,a}\right\|$. Combined with the number of times each arm was pulled, the server can perform weighted aggregation to estimate global parameters without accessing any raw context information (Lines 6–8, Alg. 2). Then the server sends the aggregated global parameters $\boldsymbol{V}^e, \hat{\boldsymbol{\theta}}^e, \hat{\boldsymbol{\beta}}^e$ back to each client (Line 9, Alg. 2).

After receiving the aggregated parameters from the central server, each client confidence sets $\mathbb{C}^e_\theta \subseteq \mathbb{R}^d$ and $\mathbb{C}^e_\beta \subseteq \mathbb{R}^d$ for $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$, respectively, in epoch e. Following standard techniques in linear bandits [30], each confidence set takes the form of an ellipsoid centered at the global estimate. Specifically, the confidence set for $\boldsymbol{\theta}$ is defined as:

$$\mathbb{C}^e_\theta = \left\{\boldsymbol{\theta} \in \mathbb{R}^d : \left\|\hat{\boldsymbol{\theta}}^e - \boldsymbol{\theta}\right\|_{\boldsymbol{V}^e} \leq \ell_e\right\}, \tag{10}$$

where $\ell_e = \sqrt{2\log\left(\frac{\det(\boldsymbol{V}^e)^{1/2}\det(\lambda I)^{-1/2}}{\delta}\right)} + \lambda^{1/2}$. The set $\mathbb{C}^e_\beta$ is constructed analogously. This construction motivates the synchronization condition in Eq. (8): the volume of the confidence ellipsoids, which controls the uncertainty in parameter estimation, scales with $\det(\boldsymbol{V}_{t,m})$. If $\det(\boldsymbol{V}_{t,m})$ does not increase significantly during an epoch, the confidence bound remains nearly unchanged, and recomputation or communication is unnecessary. Thus, in epoch e, we replace $\det(\boldsymbol{V}_{t,m})$ with the epoch-level quantity $\det(\boldsymbol{V}^e)$ to quantify information accumulation. To select arms, we adopt the principle of *optimism in the face of uncertainty*, as used in LinUCB [2], [30] and UCB algorithm for BwK [16]. Specifically, each client constructs an optimistic estimate of the reward parameter $\boldsymbol{\theta}$ as:

$$\tilde{\boldsymbol{\theta}}^e = \mathrm{argmax}_{\boldsymbol{\theta}\in\mathbb{C}^e_\theta} \left(\max_{a\in[K]} \langle \boldsymbol{x}(a_{t,m}), \boldsymbol{\theta}\rangle \right), \tag{11}$$

and similarly obtains a pessimistic estimate $\tilde{\boldsymbol{\beta}}^e$ for the cost parameter by minimizing over $\mathbb{C}^e_\beta$. This yields an upper confidence bound (UCB) for reward and a lower confidence bound (LCB) for cost. Each client then selects the arm a that maximizes the ratio of these bounds, balancing reward gain against resource consumption (Line 16, Alg. 1).

Then each client then updates the arm-pulling policy for the next epoch by solving the following linear programming (Line 17, Alg. 1):

$$\begin{matrix}\max_{\boldsymbol{p}^e \in \Delta} & \boldsymbol{X}^\top \tilde{\boldsymbol{\theta}}^e \cdot \boldsymbol{p}^e \\ \text{s.t.} & \boldsymbol{X}^\top \tilde{\boldsymbol{\beta}}^e \cdot \boldsymbol{p}^e \leq (1-\epsilon)\frac{B}{TM}\end{matrix}, \tag{12}$$

where Δ denotes the probability simplex. The multiplicative factor $(1-\epsilon)$ introduces a safety margin to account for estimation errors in $\tilde{\boldsymbol{\beta}}^e$, preventing premature termination due to overestimation of cost. The detailed role of ϵ in regret and budget analysis will be discussed in the next section. After computing the policy $\boldsymbol{p}^e$, each client resets $\boldsymbol{V}_{t_0,m}$ and proceeds to the next epoch (Line 18, Alg. 1).

VI. Theoretical Analysis

In this section, we provide theoretical guarantees in terms of regret, communication, and computation costs.

A. Regret Analysis

Theorem 1. *With FedUCBwK, we can get a regret with a high probability* $1-\epsilon$, $\epsilon = \frac{d\sqrt{DMT}\log(MT)}{B}$,

$$\textit{Regret}(T) = O\left(d\sqrt{DMT}\log(MT)(\frac{OPT}{B}+1)\right). \quad (13)$$

Lemma 1. *We define the difference between the realized reward and the expected reward using the upper confidence bound of pulled arms as:*

$$\textit{Diff}(T) = \sum_{t=1}^{T}\sum_{m=1}^{M}\langle\tilde{\boldsymbol{\theta}}_{t,m}-\boldsymbol{\theta}, \boldsymbol{x}(a_{t,m})\rangle. \quad (14)$$

Then the $\textit{Diff}(T)$ *is bounded by* $O\big(d\sqrt{DMT}\log(MT)\big)$.

Proof. First, although the same estimator $\tilde{\boldsymbol{\theta}}^e$ is used throughout each epoch e, we denote it as $\tilde{\boldsymbol{\theta}}_{t,m}$ in the analysis to explicitly indicate the time step t and client m, *i.e.*, $\tilde{\boldsymbol{\theta}}_{t,m} = \tilde{\boldsymbol{\theta}}^e$ for all $t \in e$. Let the algorithm proceed over E epochs. We denote by $\boldsymbol{V}^e$ the aggregated Gram matrix in epoch e. By the design of the communication threshold, we have:

$$1 \le \frac{\det(\boldsymbol{V}_j)}{\det(\boldsymbol{V}_{j-1})} \le D,$$

Otherwise, there will be a synchronization. We imagine the MT pulls are all made by one client in a round-robin fashion. This client takes $a_{1,1}, a_{1,2}, \ldots, a_{1,M}, a_{2,1}, \ldots, a_{T,M}$). We use $\tilde{\boldsymbol{V}}_{t,m} = \lambda\boldsymbol{I} + \sum_{\{(p,q):(p<t)\vee(p=t\wedge q<m)\}}\boldsymbol{x}(a_{p,q})\boldsymbol{x}(a_{p,q})^T$ to denote the Gram matrix this imaginary client can collect when he gets to $\boldsymbol{x}(a_{t,m})$. And by our algorithm, each client m will use the random policy received from the central server, generated by the aggregated Gram matrix $\boldsymbol{V}^e$ in each communication round. Thus, we can bound the gap between the Gram matrix as:

$$1 \le \frac{\det(\tilde{\boldsymbol{V}}_{t,m})}{\det(\boldsymbol{V}^e)} \le \frac{\det(\boldsymbol{V}_j)}{\det(\boldsymbol{V}_{j-1})} \le D.$$

Therefore, by Lemma 6 in Appendix,

$$\begin{aligned}
\textit{diff}_{t,m} &\le O\left(\sqrt{d\log\frac{T}{\delta}}\right)\sqrt{\boldsymbol{x}(a_{t,m})^T(\boldsymbol{V}^e)^{-1}\boldsymbol{x}(a_{t,m})} \\
&\le O\left(\sqrt{d\log\frac{T}{\delta}}\right)\sqrt{\boldsymbol{x}(a_{t,m})^T\tilde{\boldsymbol{V}}_{t,m}^{-1}\boldsymbol{x}(a_{t,m})\cdot\frac{\det(\tilde{\boldsymbol{V}}_{t,m})}{\det(\boldsymbol{V}^e)}} \\
&\le O\left(\sqrt{d\log\frac{T}{\delta}}\right)\sqrt{D\boldsymbol{x}(a_{t,m})^T\tilde{\boldsymbol{V}}_{t,m}^{-1}\boldsymbol{x}(a_{t,m})}.
\end{aligned}$$

We now extend the single-client analysis to the federated setting by bounding the accumulated difference across all clients. Let $\mathcal{B}_e$ denote the set of (t,m) pairs belonging to epoch e. By applying Lemmas 2, 3, and 4 in the Appendix and setting $\gamma = dDMT\log\left(\frac{T}{\delta}\right)$, we obtain the following bound on the cumulative estimation error across all epochs:

$$\begin{aligned}
\textit{Diff}(T) &= \sum_{m}^{M}\sum_{e=1}^{E}\sum_{(t,m)\in\mathcal{B}_e}\textit{diff}_{t,m} \le \sqrt{MT\sum_{e=1}^{E}\sum_{(t,m)\in\mathcal{B}_e}\textit{diff}_{t,m}^2} \\
&\le O\left(\sqrt{\gamma\sum_{e=1}^{E}\sum_{(t,m)\in\mathcal{B}_e}\min\left(\|\boldsymbol{x}(a_{t,m})\|^2_{\tilde{\boldsymbol{V}}^{-2}_{t,m}},1\right)}\right) \\
&\le O\left(\sqrt{\gamma\log\left(\frac{\det(\boldsymbol{V}^E)}{\det(\boldsymbol{V}_0)}\right)}\right) \\
&\le O(d\sqrt{DMT}\log(MT)).
\end{aligned}$$

Next, we show that $\tilde{\boldsymbol{\theta}}_{t,m}$ and $\tilde{\boldsymbol{\beta}}_{t,m}$ estimates satisfy the following properties. With probability $1-M\delta$, let $\rho = O\left(d\sqrt{DMT}\log(MT)\right)$:

Property (1): $\quad \tilde{\boldsymbol{\theta}}_{t,m} \ge \boldsymbol{\theta}, \tilde{\boldsymbol{\beta}}_{t,m} \le \boldsymbol{\beta}$.

Property (2):

$$\left|\sum_{t=1}^{T}\sum_{m=1}^{M}\left(\boldsymbol{X}^\top\tilde{\boldsymbol{\theta}}_{t,m}\boldsymbol{p}_t - \langle\boldsymbol{\theta}, \boldsymbol{x}(a_{t,m})\rangle\right)\right| \le \rho,$$

$$\left|\sum_{t=1}^{T}\sum_{m=1}^{M}\left(\boldsymbol{X}^\top\tilde{\boldsymbol{\beta}}_{t,m}\boldsymbol{p}_t - \langle\boldsymbol{\beta}, \boldsymbol{x}(a_{t,m})\rangle\right)\right| \le \epsilon B.$$

Since $\mathbb{E}\left[\langle\tilde{\boldsymbol{\theta}}_{t,m}, \boldsymbol{x}(a_{t,m})\rangle \mid \boldsymbol{p}_t, \boldsymbol{X}^\top\tilde{\boldsymbol{\theta}}_{t,m}\right] = \boldsymbol{X}^\top\tilde{\boldsymbol{\theta}}_{t,m}\boldsymbol{p}_t$, we can obtain, using Lemma 5 in Appendix, with probability $1-\delta$,

$$\left|\sum_{t=1}^{T}(\langle\tilde{\boldsymbol{\theta}}_{t,m}, \boldsymbol{x}(a_{t,m})\rangle - \boldsymbol{X}^\top\tilde{\boldsymbol{\theta}}_{t,m}\boldsymbol{p}_t)\right| \le O(\sqrt{\log(d/\delta)\,T}).$$

Then, according to $\textit{Diff}(T)$ in Lemma 1, we can bound the gap between the realized reward of the pulled arm and the expected reward under the fractional solution using the upper confidence bound:

$$\left|\sum_{t=1}^{T}\sum_{m=1}^{M}\left(\boldsymbol{X}^\top\tilde{\boldsymbol{\theta}}_{t,m}\boldsymbol{p}_t - \langle\boldsymbol{\theta}, \boldsymbol{x}(a_{t,m})\rangle\right)\right| \le \rho.$$

A similar bound holds for the consumption estimation involving $\boldsymbol{\beta}$. Then we can set $\epsilon = \frac{d\sqrt{DMT}\log(MT)}{B}$. Till now, we can get Property (2). By Property (1), the definition and problem decomposition of OPT:

$$\begin{aligned}
\sum_{m=1}^{M}\sum_{t=1}^{T}\boldsymbol{X}^\top\tilde{\boldsymbol{\theta}}_{t,m}\boldsymbol{p}_t &= \sum_{m=1}^{M}\sum_{t=1}^{T}\mathrm{LP}\left(\boldsymbol{X}^\top\tilde{\boldsymbol{\theta}}_{t,m}, \boldsymbol{X}^\top\tilde{\boldsymbol{\beta}}_{t,m}, \epsilon\right) \\
&\ge MT\mathrm{LP}(\boldsymbol{\theta}^\top\boldsymbol{X}, \boldsymbol{\beta}^\top\boldsymbol{X}, \epsilon) \\
&\ge (1-\epsilon)\mathrm{OPT}, \\
\sum_{m=1}^{M}\sum_{t=1}^{T}\boldsymbol{X}^\top\tilde{\boldsymbol{\beta}}_{t,m}\boldsymbol{p}_t &\le (1-\epsilon)B.
\end{aligned} \quad (15)$$

Then, using Property (2) above and Equation (15), we can satisfy the hard constraint. *i.e.*,

$$\sum_{t=1}^{T}\sum_{m=1}^{M}\langle\boldsymbol{\beta}, \boldsymbol{x}(a_{t,m})\rangle \le B.$$

Thus, the algorithm will not terminate before time T. We denote the total realized reward we get is Rew, Rew $= \sum_{t=1}^{T}\sum_{m=1}^{M}\langle \boldsymbol{\theta}, \boldsymbol{x}(a_{t,m})\rangle$. Then

$$\text{Rew} \geq (1-\epsilon)\text{OPT} - O\left(d\sqrt{DMT}\log(MT)\right),$$

$$\epsilon\text{OPT} = O\big(\frac{d\sqrt{DMT}\log(MT)\text{OPT}}{B}\big),$$

$$\begin{aligned}\text{Regret}(T) &= \text{OPT} - \text{Rew}\\ &= O\big(d\sqrt{DMT}\log(MT)(\frac{\text{OPT}}{B}+1)\big).\end{aligned}$$

B. Computation Cost

Theorem 2. *Using the FedUCBwK algorithm, the computation cost is bound by* $O(Md\log\frac{MT}{D})$.

In Algorithms 1 and 2, FedUCBwK triggers LP computation only when the design matrix $\boldsymbol{V}$ has accumulated sufficient new information. This is determined by a threshold D: a new epoch begins when

$$\frac{\det(\boldsymbol{V}_{j+1})}{\det(\boldsymbol{V}_j)} > D.$$

This ensures computation is performed only when necessary, reducing redundant updates. Therefore, to analyze the total computation cost, it suffices to bound the number of such epochs triggered by this determinant-based condition. By Lemma 3 in the Appendix, we can get

$$P\log D < \sum_{j=0}^{P-1}\log\left(\frac{\det \boldsymbol{V}_{j+1}}{\det \boldsymbol{V}_j}\right) < d\log\left(1+\frac{MT}{\lambda d}\right),$$

where P denotes the total epochs. It will be bounded by

$$P < O(d\log\frac{MT}{D}). \tag{16}$$

LP calculation is done on each client. Thus, the computation cost is $O(Md\log\frac{MT}{D})$. This result shows that FedUCBwK achieves logarithmic computation cost scaling with the time horizon T.

C. Communication Cost

Theorem 3. *Using the FedUCBwK algorithm, the communication cost is bound by* $O\left(Md^2(d+K)\log\frac{MT}{D}\right)$.

Communication is only required at the end of each epoch when each client sends $O(dK)$ parameters to the server and then receives $O(d^2)$ parameters. Therefore, in each epoch, the communication cost is $O\left(Md(d+K)\right)$. Hence, further according to the upper bound of the number of epochs in Equation (16), the total communication cost is $O\left(Md^2(d+K)\log\frac{MT}{D}\right)$.

Combining the above theorems and setting D as a constant, we can get a regret bound of $O\big(d\sqrt{MT}\log(MT)(\frac{\text{OPT}}{B}+1)\big)$ with a high probability, under $O\left(Md^2(d+K)\log MT\right)$ communication cost and $O(Md\log MT)$ computation cost. Thus, FedUCBwK achieves logarithmic communication cost scaling with the time horizon T.

VII. Experiment Results

We evaluate the performance of the proposed **FedUCBwK** algorithm on two real-world datasets across three task settings, comparing it against five baseline algorithms. FedUCBwK-FullCom refers to the variant where clients communicate and solve the LP problem at every time step, which can be regarded as an optimal bandit solution in a centralized setting. FedUCBwK-FewCom performs only infrequent communication and computation, using a large threshold D. We further compare FedUCBwK with other federated and distributed bandits algorithms, *e.g.*, FedUCB [8] (which is equivalent to DisLinUCB [23] by letting the privacy budget $1/\epsilon$ go to 0). We also evaluated the performance of FedUCB under full communication. In all experiments, we conducted 10 repeated trials and calculated the mean and standard deviation.

A. Datasets and Tasks

MovieLens-100K [49]: This dataset contains movie ratings from 943 users on 1682 movies. To handle data sparsity, we use collaborative filtering to complete the rating matrix and then employ non-negative matrix factorization [9] with 10 latent factors to get $R = WH$, where the user matrix $W \in \mathbb{R}^{943\times 10}$, and the movie matrix $H \in \mathbb{R}^{10\times 1682}$. We simulate two ad allocation tasks. The first scenario involves recommending the proper ad for a specific group of users [50]. We regard each ad as an arm that corresponds to a certain amount of revenue and deduction. The objective is to identify the optimal ad for a specific group of users that maximizes the reward within a range of advertiser budgets. We cluster W's columns into 20 user groups, select one as the target group u, centroid with θ_u, and the corresponding cost vector β_u is randomly generated. Ads (arms) are clustered from H's rows into $K = 10$ groups. The second scenario involves finding the proper targeted user for a specific category of ads [51]. Each user is treated as an arm. We cluster H's rows into 20 ad groups, select one as the target ad a, and assign its centroid to θ_a, with a random β_a. User arms are then formed by clustering W's rows into $K = 10$ groups. In the above experiments, we set $K = 20$, $d = 10$, $M = 10$, and $T = 1000$.

Human Activity Recognition (HAR) Dataset (AMU-HAR) [52]: This dataset was collected using smartphone sensors (accelerometer, magnetometer, and gyroscope). Each of the 9 activity classes (e.g., running, cycling) contains 403,605 samples, with each sample represented by 12 sensor-derived features. We simulate a crowdsourcing scenario [12], where each activity is treated as an arm and each task involves recognizing the corresponding activity. Assigning a task incurs a cost, and the objective is to select the optimal task (activity) for a group of workers to maximize reward under a limited budget. We randomly generate a reward and cost for each sample, and compute the average reward θ and cost β for each arm. The class center is used as the arm's context vector for calculating the OPT solution. In our experiments, we set $K = 9$, $d = 12$, $M = 10$, and $T = 1000$.

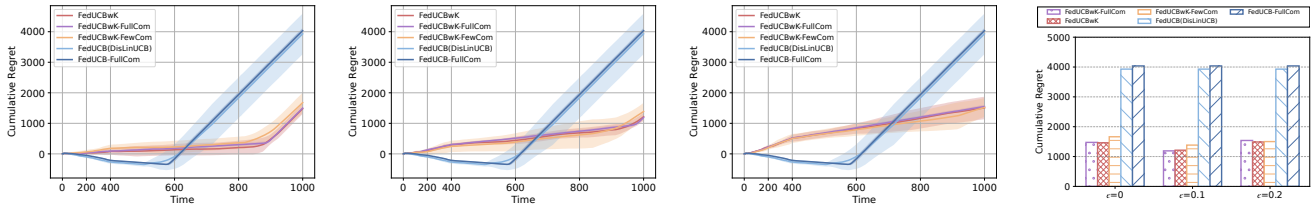

Fig. 2. Mobile Ad Recommendation: Ad Recommendation. From left to right, ϵ= 0, 0.05, 0.1, respectively.

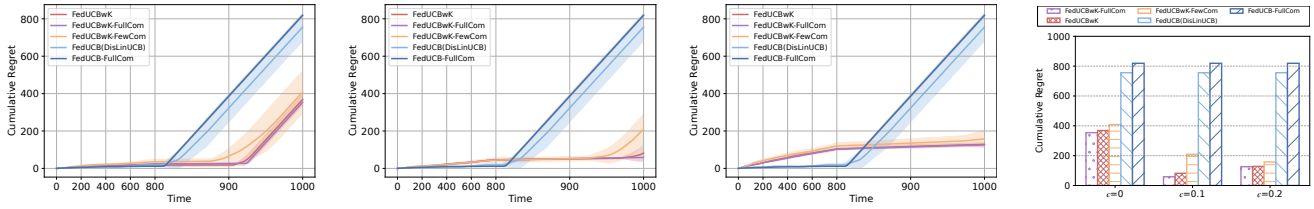

Fig. 3. Mobile Ad Recommendation: Targeted User Recommendation. From left to right, ϵ= 0, 0.05, 0.1, respectively.

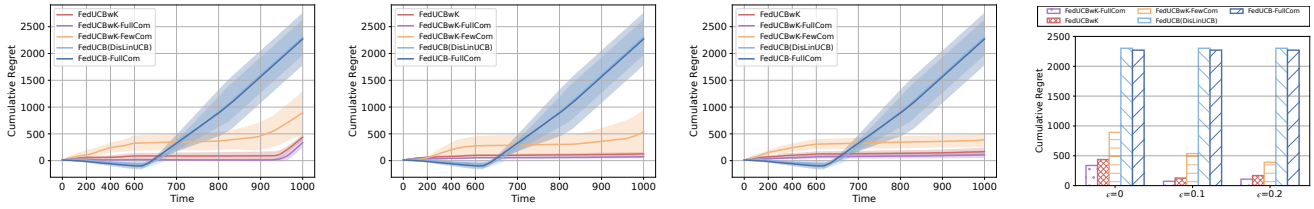

Fig. 4. Mobile Crowd-Sourcing. From left to right, ϵ= 0, 0.05, 0.1, respectively.

In most practical tasks, K tends to be much larger than d. This occurs because mobile computing scenarios typically involve a vast number of arms (e.g., users, items, or tasks), while feature dimensions are inherently limited or can be effectively reduced via feature engineering.

B. Cumulative Regret Results under Limited Budget

In this subsection, we evaluate the cumulative regret performance of different algorithms under the scenario of limited budgets. We first illustrate why the cumulative regret curve in FBwK deviates from that of traditional bandits. A key distinction lies in the budget constraint: in FBwK, once the budget is exhausted, the bandit algorithm stops pulling arms, whereas the OPT continues to accrue rewards, resulting in a steep regret increase. This behavior is typical in BwK problems. Algorithms such as FedUCB (DisLinUCB) and FedUCB-FullCom, which ignore budget constraints, tend to pull arms with higher rewards early on. As a result, they achieve low regret in the initial phase. However, once the budget is depleted at time step τ, these algorithms terminate prematurely, causing a sharp regret increase beyond τ.

1) Mobile Ad Recommendation: For the ad recommendation task, as shown in Fig. 2, FedUCBwK consistently achieves regret close to FedUCBwK-FullCom and significantly outperforms other baseline methods. By selecting arms with high reward and low cost, FedUCBwK avoids early termination and maintains low cumulative regret. In contrast, algorithms without budget awareness (*e.g.*, FedUCB, FedUCB-FullCom) tend to over-consume early and terminate prematurely, leading to steep regret growth in later rounds. As ϵ increases (left to right in Fig. 2), FedUCBwK becomes more conservative in estimating consumption, reducing the risk of early budget exhaustion. However, overly large ϵ may overweight cost and cause suboptimal arm selection. We find $\epsilon = 0.1$ achieves the best trade-off. Interestingly, regret can be negative early on when FedUCB selects high-reward arms not constrained by budget—exceeding the reward of OPT—but this comes at the cost of exhausting the budget early and suffering high regret later.

As for targeted user recommendation, we can find similar results in Fig.3. FedUCBwK again balances reward and cost effectively and achieves up to 89.17% regret reduction over FedUCB with $\epsilon = 0.1$. Under both tasks, due to the limited number of synchronization rounds, FedUCBwK-FewCom hinders effective information aggregation and strategy updates, leading to higher variance and larger regret. In addition, due to inaccurate estimation, FedUCBwK-FewCom often requires a larger ϵ to control the budget. Typically, when $\epsilon = 0.2$, FedUCBwK-FewCom can manage the budget and avoid early termination.

2) Mobile Crowd-sourcing: We present the experimental results for mobile crowd-sourcing in Fig. 4. The overall trends are consistent with the previous tasks. Methods in the FedUCB series, which are unaware of budget constraints, initially achieve negative regret by aggressively selecting high-reward arms. However, they rapidly exhaust the budget and terminate early, leading to steep regret growth in later stages. In contrast, the FedUCBwK variants allocate the budget more conservatively, enabling sustained learning and more stable performance. As shown in Fig. 4, setting $\epsilon = 0.1$ achieves the best balance. where FedUCBwK reduces regret by up to 94.47% compared to FedUCB. When $\epsilon = 0$, the overly tight constraint leads to early termination, limiting the

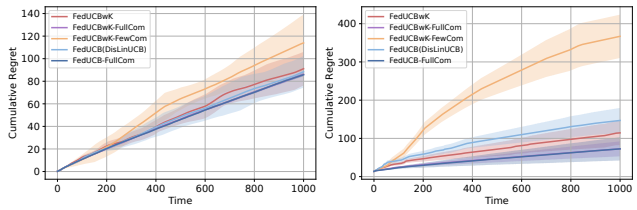

Fig. 5. Sufficient Budget

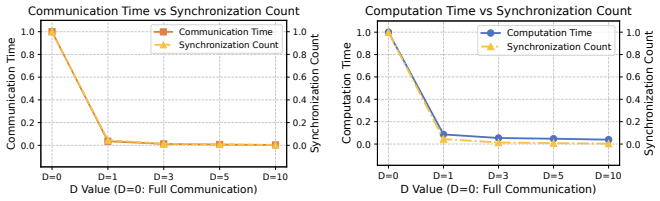

Fig. 6. computation cost and communication cost

opportunity to identify better arms and resulting in higher final regret.

C. Cumulative Regret Results under Abundant Budget

We further evaluate the cumulative regret under an abundant budget setting (Even if the arm with the maximum cost is selected in each round, the budget will not be exhausted). In Fig. 5, the left subfigure corresponds to the Mobile Ad Recommendation task, and the right to Mobile Crowd-Sourcing. When the budget is sufficient, all methods achieve comparable performance, and the main performance differences arise from the frequency of communication and synchronization. Among them, FedUCBwK-FullCom achieves the lowest regret and variance due to frequent updates, while FedUCBwK-FewCom exhibits the highest regret and largest variance, owing to limited communication and slower policy adaptation. Notably, the performance gap between FedUCBwK-FewCom and other methods is more significant in the crowd-sourcing task, where even fewer communication rounds are performed, leading to less effective learning.

D. Computation and Communication Cost

We compare the communication time, computation time, and synchronization count under different values of D, where $D = 0$ corresponds to FedUCBwK-FullCom (*i.e.*, computation and communication at every round). As shown in Fig. 6, compared to full communication, FedUCBwK achieves logarithmic communication and computation complexity. With $T = 1000$ time steps, we find that by setting a reasonable threshold (typically $D = 3$), FedUCBwK performs only around 10 communication rounds. It is worth noting that in real-world distributed systems, where communication latency is typically higher than the single-machine distributed communication in the experiment, the benefit of reduced communication would be even more substantial.

VIII. Conclusion

In this work, we propose FedUCBwK, a practical and communication-efficient framework for solving budget-constrained federated bandits in mobile application scenarios. Our approach enables M distributed clients to collaboratively optimize arm selection under a constrained budget, while preserving data privacy through model update exchanges. To address the challenges of constrained resources and costly synchronization, we introduce an epoch-based policy update mechanism controlled by a unified threshold, which significantly reduces communication and computation overhead. In addition, a conservative budget adjustment factor is incorporated to prevent premature budget exhaustion. We provide rigorous theoretical guarantees showing that FedUCBwK achieves near-optimal regret with logarithmic communication and computation complexity. Extensive experiments on real-world datasets demonstrate the effectiveness of FedUCBwK in mobile settings, showing substantial regret reduction compared to baseline methods, especially under tight budget constraints.

Appendix

For the sake of completeness, we include the lemma from the previous work that was utilized in our proof process.

Lemma 2. *For any $\delta > 0$, with probability $1 - M\delta$, $\boldsymbol{\theta}$ always lies in the constructed $\mathbb{C}_{t,m}$ for all t and all m.*

Lemma 3. *Let $\{X_t\}_{t=1}^{\infty}$ be a sequence in $\mathbb{R}^d$, $\boldsymbol{V}$ is a $d \times d$ positive definite matrix and define $\boldsymbol{V}_t = \boldsymbol{V} + \sum_{s=1}^{t} \boldsymbol{X}_s \boldsymbol{X}_s^\top$. Then we have that*

$$\log\left(\frac{\det(\boldsymbol{V}_n)}{\det(\boldsymbol{V})}\right) \leq \sum_{t=1}^{n} \|\boldsymbol{X}_t\|^2_{\boldsymbol{V}_{t-1}^{-1}}.$$

If $\|\boldsymbol{X}_t\|_2 \leq L$ for all t, we have that

$$\begin{aligned}\sum_{t=1}^{n} \min\left\{1, \|\boldsymbol{X}_t\|^2_{\boldsymbol{V}_{t-1}^{-1}}\right\} &\leq 2\left(\log\det(\boldsymbol{V}_n) - \log\det \boldsymbol{V}\right) \\ &\leq 2\left(d\log\left(\left(\mathrm{trace}(\boldsymbol{V}) + nL^2\right)/d\right) - \log\det \boldsymbol{V}\right).\end{aligned}$$

Lemma 4. *With probability $1 - M\delta$, single step difference $diff_{t,m} = \langle \tilde{\boldsymbol{\theta}}_{t,m} - \boldsymbol{\theta}, \boldsymbol{x}(a_{t,m})\rangle$ is bounded by*

$$\begin{aligned}diff_{t,m} &\leq 2\left(\sqrt{2\log\left(\frac{\det(\boldsymbol{V}_{t,m})^{\frac{1}{2}}}{\delta \det(\lambda I)^{\frac{1}{2}}}\right)} + \lambda^{\frac{1}{2}}\right)\|\boldsymbol{x}(a_{t,m})\|_{\boldsymbol{V}_{t,m}^{-1}} \\ &\leq O\left(\sqrt{d\log\frac{t}{\delta}}\right)\|\boldsymbol{x}(a_{t,m})\|_{\boldsymbol{V}_{t,m}^{-1}}.\end{aligned}$$

Lemma 5. *(Azuma-Hoeffding inequality). If a supermartingale $(Y_t; t \geq 0)$, corresponding to filtration $\mathcal{F}_t$, satisfies $|Y_t - Y_{t-1}| \leq c_t$ for some constant c_t, for all $t = 1, \ldots, T$, then for any $a \geq 0$,*

$$\Pr\left(Y_T - Y_0 \geq a\right) \leq e^{-\frac{a^2}{2\sum_{t=1}^{Tc} c_t^2}}.$$

Lemma 6. *Let $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$ be positive semi-definite matrices such that $\boldsymbol{A} = \boldsymbol{B} + \boldsymbol{C}$. Then, we have that*

$$\sup_{\boldsymbol{x} \neq \boldsymbol{0}} \frac{\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{B}\boldsymbol{x}} \leq \frac{\det(\boldsymbol{A})}{\det(\boldsymbol{B})}.$$

References

[1] C. Zeng, Q. Wang, S. Mokhtari, and T. Li, "Online context-aware recommendation with time varying multi-armed bandit," in *KDD*, 2016, pp. 2025–2034.

[2] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *WWW*, 2010, pp. 661–670.

[3] A. Nuara, F. Trovò, N. Gatti, and M. Restelli, "A combinatorial-bandit algorithm for the online joint bid/budget optimization of pay-per-click advertising campaigns," in *AAAI*, 2018, pp. 2379–2386.

[4] M. Guo, W. Zhang, C. Yuan, B. Jia, G. Song, H. Hua, S. Wang, and Q. Zhang, "A bayesian multi-armed bandit algorithm for bid shading in online display advertising," in *CIKM*, 2024, pp. 4506–4513.

[5] H. Wang, Y. Yang, E. Wang, W. Liu, Y. Xu, and J. Wu, "Truthful user recruitment for cooperative crowdsensing task: A combinatorial multi-armed bandit approach," *IEEE Transactions on Mobile Computing*, vol. 22, no. 7, pp. 4314–4331, 2023.

[6] Y. Ouyang, F. Zeng, N. N. Xiong, A. Liu, and W. Pedrycz, "Mwrs: A mab-based worker recruitment scheme with tripartite stackelberg game for reliable mobile crowdsensing," *IEEE Transactions on Mobile Computing*, 2025.

[7] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3-4, pp. 285–294, 1933.

[8] A. Dubey and A. Pentland, "Differentially-private federated linear bandits," in *NeurIPS*, 2020, pp. 6003–6014.

[9] R. Huang, W. Wu, J. Yang, and C. Shen, "Federated linear contextual bandits," in *NeurIPS*, 2021, pp. 27 057–27 068.

[10] A. Badanidiyuru, R. Kleinberg, and A. Slivkins, "Bandits with knapsacks," in *FOCS*, 2013, pp. 207–216.

[11] I. Siomina, P. Värbrand, and D. Yuan, "Automated optimization of service coverage and base station antenna configuration in UMTS networks," *IEEE Wireless Communications*, vol. 13, no. 6, pp. 16–25, 2006.

[12] Y. Song and H. Jin, "Minimizing entropy for crowdsourcing with combinatorial multi-armed bandit," in *INFOCOM*, 2021, pp. 1–10.

[13] Y. Xu, M. Xiao, J. Wu, S. Zhang, and G. Gao, "Incentive mechanism for spatial crowdsourcing with unknown social-aware workers: A three-stage stackelberg game approach," *IEEE Transactions on Mobile Computing*, vol. 22, no. 8, pp. 4698–4713, 2023.

[14] Y. Li, F. Li, S. Yang, and Y. Wang, "Bgefl: Enabling communication-efficient federated learning via bandit gradient estimation in resource-constrained networks," *IEEE Transactions on Networking*, pp. 1–16, 2025.

[15] T. Li and L. Song, "Privacy-preserving communication-efficient federated multi-armed bandits," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 3, pp. 773–787, 2022.

[16] S. Agrawal and N. R. Devanur, "Bandits with concave rewards and convex knapsacks," in *EC*, 2014, pp. 989–1006.

[17] ——, "Linear contextual bandits with knapsacks," in *NeurIPS*, 2016, pp. 3450–3458.

[18] C. Shi and C. Shen, "Federated multi-armed bandits," in *AAAI*, 2021, pp. 9603–9611.

[19] C. Shi, C. Shen, and J. Yang, "Federated multi-armed bandits with personalization," in *AISTATS*, 2021, pp. 2917–2925.

[20] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[21] S. Agrawal, N. R. Devanur, and L. Li, "An efficient algorithm for contextual bandits with knapsacks, and an extension to concave objectives," in *COLT*, 2016, pp. 4–18.

[22] D. Foster and A. Rakhlin, "Beyond ucb: Optimal and efficient contextual bandits with regression oracles," in *ICML*, 2020, pp. 3199–3210.

[23] Y. Wang, J. Hu, X. Chen, and L. Wang, "Distributed bandit learning: Near-optimal regret with efficient communication," in *ICLR*, 2019.

[24] A. Dubey and A. S. Pentland, "Differentially-private federated linear bandits," in *NeurIPS*, 2020, pp. 6003–6014.

[25] Z. Zhu, J. Zhu, J. Liu, and Y. Liu, "Federated bandit: A gossiping approach," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 5, no. 1, pp. 02:1–02:29, 2021.

[26] 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," *Foundations and Trends in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[27] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2, pp. 235–256, 2002.

[28] J. Langford and T. Zhang, "The epoch-greedy algorithm for multi-armed bandits with side information," in *NeurIPS*, 2007, pp. 817–824.

[29] S. Agrawal and N. Goyal, "Thompson sampling for contextual bandits with linear payoffs," in *ICML*, 2013, pp. 127–135.

[30] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári, "Improved algorithms for linear stochastic bandits," in *NeurIPS*, 2011, pp. 2312–2320.

[31] W. Chu, L. Li, L. Reyzin, and R. E. Schapire, "Contextual bandits with linear payoff functions," in *AISTATS*, 2011, pp. 208–214.

[32] S. Agrawal and N. Goyal, "Thompson sampling for contextual bandits with linear payoffs," in *ICML*, 2013, pp. 127–135.

[33] X. Wang, J. Ye, and J. C. S. Lui, "Decentralized task offloading in edge computing: A multi-user multi-armed bandit approach," in *INFOCOM*, 2022, pp. 1199–1208.

[34] S. Seo, S. Kim, S. Kook, S. Baek, and S. Kim, "Constructing 3-dimensional 5g coverage map for real-time airborne missions," in *MobiCom*, 2020, pp. 81:1–81:3.

[35] H. Gupta, J. Chen, B. Li, and R. Srikant, "Online learning-based rate selection for wireless interactive panoramic scene delivery," in *INFOCOM*, 2022, pp. 1799–1808.

[36] A. Xu, Z. Zheng, F. Wu, and G. Chen, "Online data valuation and pricing for machine learning tasks in mobile health," in *INFOCOM*, 2022, pp. 850–859.

[37] D. M. Kalathil, N. Nayyar, and R. Jain, "Decentralized learning for multiplayer multiarmed bandits," *IEEE Transactions on Information Theory*, vol. 60, no. 4, pp. 2331–2345, 2014.

[38] P. Landgren, V. Srivastava, and N. E. Leonard, "Distributed cooperative decision making in multi-agent multi-armed bandits," *Automatica*, vol. 125, p. 109445, 2021.

[39] L. Yang, Y. J. Chen, M. H. Hajiesmaili, J. C. S. Lui, and D. Towsley, "Distributed bandits with heterogeneous agents," in *INFOCOM*, 2022, pp. 200–209.

[40] R. Chawla, D. Vial, S. Shakkottai, and R. Srikant, "Collaborative multi-agent heterogeneous multi-armed bandits," in *ICML*, 2023, pp. 4189–4217.

[41] H. Yang, X. Liu, Z. Wang, H. Xie, J. C. Lui, D. Lian, and E. Chen, "Federated contextual cascading bandits with asynchronous communication and heterogeneous users," in *AAAI*, vol. 38, no. 18, 2024, pp. 20 596–20 603.

[42] F. Fourati, M.-S. Alouini, and V. Aggarwal, "Federated combinatorial multi-agent multi-armed bandits," in *ICML*, 2024, pp. 13 760–13 782.

[43] X. Wu and B. Li, "Achieving regular and fair learning in combinatorial multi-armed bandit," in *INFOCOM*, 2024, pp. 361–370.

[44] K. A. Sankararaman and A. Slivkins, "Combinatorial semi-bandits with knapsacks," in *AISTATS*, 2018, pp. 1760–1770.

[45] N. Immorlica, K. A. Sankararaman, R. E. Schapire, and A. Slivkins, "Adversarial bandits with knapsacks," in *FOCS*, 2019, pp. 202–219.

[46] M. Bernasconi, M. Castiglioni, A. Celli, and F. Fusco, "Beyond primal-dual methods in bandits with stochastic and adversarial constraints," in *NeurIPS*, 2024, pp. 8541–8568.

[47] Z. Chen, R. Ai, M. Yang, Y. Pan, C. Wang, and X. Deng, "Contextual decision-making with knapsacks beyond the worst case," in *NeurIPS*, 2024, pp. 88 147–88 193.

[48] B. Gärtner and J. Matousek, *Understanding and using linear programming*, ser. Universitext. Springer, 2007.

[49] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems*, vol. 5, no. 4, pp. 1–19, 2015.

[50] G. Theocharous, P. S. Thomas, and M. Ghavamzadeh, "Ad recommendation systems for life-time value optimization," in *WWW*, 2015, pp. 1305–1310.

[51] L. Guo, J. Jin, H. Zhang, Z. Zheng, Z. Yang, Z. Xing, F. Pan, L. Niu, F. Wu, H. Xu *et al.*, "We know what you want: An advertising strategy recommender system for online advertising," in *KDD*, 2021, pp. 2919–2927.

[52] Y. A. Khan, S. Imaduddin, Y. P. Singh, M. Wajid, M. Usman, and M. Abbas, "Artificial intelligence based approach for classification of human activities using mems sensors data," *Sensors*, vol. 23, no. 3, p. 1275, 2023.