Federated Contextual Cascading Bandits with Asynchronous Communication and Heterogeneous Users

Hantao Yang[1], Xutong Liu[2*], Zhiyong Wang[2], Hong Xie[1], John C. S. Lui[2], Defu Lian[1], Enhong Chen[1]

[1]University of Science and Technology of China
[2]The Chinese University of Hong Kong
yanghantao@mail.ustc.edu.cn, liuxt@cse.cuhk.edu.hk, zywang21@cse.cuhk.edu.hk, xiehong2018@foxmail.com, cslui@cse.cuhk.edu.hk, liandefu@ustc.edu.cn, cheneh@ustc.edu.cn

Abstract

We study the problem of federated contextual combinatorial cascading bandits, where $|\mathcal{U}|$ agents collaborate under the coordination of a central server to provide tailored recommendations to the $|\mathcal{U}|$ corresponding users. Existing works consider either a synchronous framework, necessitating full agent participation and global synchronization, or assume user homogeneity with identical behaviors. We overcome these limitations by considering (1) federated agents operating in an asynchronous communication paradigm, where no mandatory synchronization is required and all agents communicate independently with the server, (2) heterogeneous user behaviors, where users can be stratified into $J \leq |\mathcal{U}|$ latent user clusters, each exhibiting distinct preferences. For this setting, we propose a UCB-type algorithm with delicate communication protocols. Through theoretical analysis, we give sub-linear regret bounds on par with those achieved in the synchronous framework, while incurring only logarithmic communication costs. Empirical evaluation on synthetic and real-world datasets validates our algorithm's superior performance in terms of regrets and communication costs.

Introduction

Contemporary recommendation systems commonly employ ordered lists to present recommended items, which cover applications in diverse domains like news, restaurants, and movies recommendations, as well as search engines (Lian et al. 2023; Wu et al. 2023). Recent attention has been directed towards the analysis of user interaction patterns and feedback in such ordered lists, leading to the development of a *cascade model* by Craswell et al. (2008). In this model, users engage with a list of items sequentially, clicking on the first satisfactory item encountered. Following a click, users cease further examination of the list. Learning agent receives feedback which indicates that the items before the clicked one were examined and deemed unsatisfactory, while the user's preference of items *after* the clicked entry remains unknown. Despite its apparent simplicity, cascade model has demonstrated efficacy in capturing user behaviors, as highlighted by Chuklin, Markov, and Rijke (2015).

Our study focuses on an online learning variant of the cascade model, referred to as *cascading bandits* (CB) (Kveton et al. 2015a,b). In the CB framework, the learning agent employs exploration-exploitation techniques to comprehend user preferences over items through interactions. Specifically, in each round, the agent recommends an item list, observes click feedback, and receives a reward of 1 if the user clicks on any item (otherwise, a reward of 0 is received). The agent's goal is to maximize cumulative rewards over T rounds (or equivalently minimize the T-round regret).

Motivated by large-scale applications with a huge number of items, previous studies (Li et al. 2016; Zong et al. 2016) have proposed the *contextual combinatorial cascading bandits* (C^3B) to integrate contextual information. This integration is achieved by adding a linear structure assumption, enhancing the scalability of CB. Later on, Li and Zhang (2018) extends C^3B to accommodate the heterogeneity of users' preferences for items. By assuming that users can be partitioned into J unknown user groups (i.e., clusters), they introduce the CLUB-cascade algorithm, where agents adaptively cluster users into groups, and harness the collaborative influence of these user groups to enhance the performance.

While the successes of C^3B in handling contextual information and user heterogeneity are noteworthy, prior investigations have primarily focused on either the single-agent paradigm (Li et al. 2016; Zong et al. 2016) or the synchronous setting which requires simultaneous data synchronization among agents (Li and Zhang 2018). In real-world scenarios, however, there could be a large number of participating agents, ranging from powerful servers to resource-constrained mobile devices. Furthermore, users can be highly heterogeneous in terms of preferences and arrival times. In this case, the feasibility of managing synchronous communication from all agents becomes questionable. This motivates us to study C^3B within an *asynchronous communication framework*, which can accommodate a large number of agents without requiring them to communicate at the same time, and allow agents to serve users with varying time of arrival and heterogenous preferences.

Our work introduces the first federated contextual combinatorial bandits (FedC^3B), a framework wherein a finite set of users $\mathcal{U}$ are served by $|\mathcal{U}|$ learning agents. Each agent is responsible for offering personalized recommendations to a specific user, and collaboration is facilitated through a central server. Due to the stochastic nature of user arrivals, communication between agents occurs asyn-

*Corresponding Author

Algorithm	Agent	User	Arm
OFUL (2011)	Single	Hom	Single
C^3-UCB (2016)	Single	Hom	Cascade
DisLinUCB (2019)	Syn	Hom	Single
CLUB-cascade (2018)	Syn	Heter	Cascade
Async-LinUCB (2021)	Par-Asy	Hom	Single
Async-LinUCB-AM (2021)	Par-Asy	Heter	Single
Phase-based FCLUB (2022a)	Par-Asy	Heter	Single
FedLinUCB (2022)	Asy	Hom	Single
FedC^{3}UCB-H (Ours)	Asy	Heter	Cascade

Table 1: Compared with existing works regarding agent communication ('Syn' is synchronous, 'Asy' is asynchronous and 'Par' is partially), user model ('Hom' is homogeneous and 'Heter' is heterogeneous), and arm selection.

Regret	Communication
$O(d\sqrt{T}\log T)$	N/A
$O(d\sqrt{KT}\log T)$	N/A
$O(d\sqrt{T}\log^2 T)$	$O(d\|\mathcal{U}\|^{1.5})$
$O(d\sqrt{JKT}\log T)$	$O(T)$
$O(d\sqrt{T}\log T)$	$O(d\|\mathcal{U}\|^2\log T)$
$O(d\sqrt{\|\mathcal{U}\|T}\log T)$	$O(d\|\mathcal{U}\|^2\log T)$
$O(d\|\mathcal{U}\|\sqrt{JT}\log^{1.5} T)$	$O(d\|\mathcal{U}\|J\log T)$
$O(d\sqrt{T}\log T)$	$O(d\|\mathcal{U}\|^2\log T)$
$O(d\sqrt{JKT}\log T)$	$O(d\|\mathcal{U}\|^2\log T+\log^2 T)$

Table 2: Each row of Table 2 is related to that of Table 1. Our framework generalizes existing works and achieves near-optimal regret with low communication cost.

chronously, eliminating the need for costly synchronizations and allowing independent communication with the central server. To account for user heterogeneity, we assume users can be categorized into $J \leq |\mathcal{U}|$ unknown clusters, each representing a group of users with similar user behaviors. The key challenge, therefore, is to accurately and swiftly determine these user clusters to facilitate effective cooperation among agents with similar users, while minimizing cross-contamination of information arising from users with different behaviors. To make the problem even more challenging, asynchronous communication introduces the additional complexity of potential delays and data inconsistencies, which shall be carefully handled to give meaningful regret/communication guarantees.

Our Contributions

To address the aforementioned challenges, this paper makes three key contributions as follows.

Algorithmic Framework. We propose the Federated Contextual Combinatorial Cascading Upper Confidence Bound Algorithm for Heterogeneous Users (FedC^{3}UCB-H). FedC^{3}UCB-H contains three components: agent-side action selection based on the information received asynchronously, server-side graph-based heterogeneity testing for precise user clustering, and agent-server asynchronous communication strategies to manage potential data inconsistencies. The key challenge of the algorithm design is to manage the data inconsistency during user clustering. Existing matrix determinant-based protocols (Li and Wang 2021; Liu et al. 2022a; He et al. 2022) fail to achieve this goal due to insufficient communication, so we propose a novel communication protocol that employs a p_t-auxiliary protocol in conjunction with the matrix determinant-based protocol. This new protocol effectively controls data inconsistency, ensuring the correct operation of heterogeneity testing and action selection.

Theoretical Analysis. We prove that FedC^{3}UCB-H achieves a $O(d\sqrt{JKT}\log T)$ regret bound with a communication cost of $O(d|\mathcal{U}|\log T+\log^2 T)$ communication cost, showing that we use only logarithmic communication cost to achieve regret results on par with the synchronous paradigm that incurs $O(T)$ communication cost (Li and Zhang 2018). Moreover, our result generalizes the federated linear contextual bandits (He et al. 2022) with homogeneous users by allowing $J > 1$, while matching their regret and communication cost when $J = 1$. Our analysis tackles several technical challenges, such as showing that within a short time span and at a low cost of asynchronous communication, each agent will collaborate with the correct counterparts, and managing information gaps for asynchronous collaboration involving multiple arms (cascading arms). We believe our proof techniques are novel and may be of independent interest for analyzing related works that involve asynchronous communication, heterogeneous users, and cascading arms.

Empirical Evaluation. Finally, we conduct experiments on both synthetic and real data and show the effectiveness of the user clustering procedure and the p_t-auxiliary communication, where our FedC^{3}UCB-H achieves superior performance regarding regrets and communication cost.

Related Work

We review and compare with existing algorithms in the area of contextual linear bandits. We consider three primary dimensions (1) cascading bandit frameworks, (2) user heterogeneity considerations, and (3) distributed/federated agents. Detailed comparisons are summarized in Tables 1 and 2.

Cascading Bandit. Cascading bandits (CB) and their variants belong to the field of online learning to rank, which has a vast literature and covers a wide range of applications (Chuklin, Markov, and Rijke 2015; Kveton et al. 2015a,b; Li et al. 2016; Lian, Liu, and Chen 2020; Vial et al. 2022; Liu et al. 2022b, 2023a,b; Choi, Udwani, and Oh 2023). CB is first proposed by Kveton et al. (2015a) and then generalized by Kveton et al. (2015b) to accommodate combinatorial action spaces. To handle large-scale applications, Zong et al. (2016) integrates the contextual information and introduces the contextual cascading bandits. Based on the seminal work of linear bandits (Abbasi-Yadkori, Pál, and Szepesvári 2011) (row 1 in Table 1), which leverages the optimism in the face of uncertainty principle (OFUL), CascadeLinUCB is introduced, yielding a regret bound of $O(d\sqrt{T})$. Their results are generalized by Li et al. (2016) to accommodate combinatorial actions, yielding a regret bound of $O(d\sqrt{KT})$ (row 2 in Table 2), and is improved to $O(d\sqrt{T})$ quite recently by (Liu et al. 2023a) and (Choi, Udwani, and Oh 2023). All these

works deal with single-agent scenarios (where no communication is needed) and assume homogeneous users, leading to techniques that are different and simpler than ours.

Online Clustering of Bandits. Online clustering of bandits (CLUB) have emerged as a prominent line of works to address user heterogeneity in contextual bandits. The first work is proposed by Gentile, Li, and Zappella (2014), followed by an extension by Li and Zhang (2018), wherein users pull cascading arms (row 4 in Table 1). Both works, including ours, adopt the key idea of maintaining a user heterogeneity graph to adaptively refine user clusters, resulting in regret bounds that depend on total user groups J instead of individual users $|\mathcal{U}|$. However, the above two works consider either single-agent setting, or synchronous setting that requires $O(T)$ commmunication to obtain up-to-date user information for decision-making. Our work, our work deviates by focusing on asynchronous multi-agent settings, where managing potential information delays and data inconsistencies are pivotal considerations.

Federated/Distributed Contextual Linear Bandits. There has been growing interest in bandit learning with multiple agents. Our work belongs to the field of cooperative distributed/federated bandits, in which multiple agents collaborate to solve a linear contextual bandit problem over certain communication networks, e.g., peer-to-peer networks (Korda, Szorenyi, and Li 2016; Zhu et al. 2021; Xu and Klabjan 2023) or star-shaped networks (Dubey and Pentland 2020; Li and Wang 2021). Our work aligns with the latter category, where a central server engages with multiple agents. For this setting, Wang et al. (2019) proposes DisLinUCB algorithm and achieves near-optimal $O(d\sqrt{T})$ in the synchronous setting, where users/agents are homogeneous and mandatory synchronization is required (row 2 in Tables 1 and 2). Li and Wang (2021) explores the partially asynchronous setting, where no synchronization is mandated, but agents still do not communicate with the server independently and one agent's upload will trigger other agents' download (row 5 in Table 1). Notably, their work also considers heterogeneous users but treats them as totally different entities, and thus achieves sub-optimal regret that has a $\sqrt{|\mathcal{U}|}$ factor (row 6 in Table 2). Ours improves this to $\sqrt{J}$ regret factor. Later on, Liu et al. (2022a) considers the similarity between users by considering user groups, similar to our approach, yet their communication model remains partially asynchronous, and the regret bound has a $|\mathcal{U}|$ factor (row 7 in Table 2). The recent FedLinUCB algorithm by He et al. (2022) for homogeneous users in a fully asynchronous setting (row 8 in Table 1) provides inspiration for our work. We generalize their approach to encompass heterogeneous users and cascading arms, yielding matching results when $J = 1$ and $K = 1$.

Problem Setup

Notations. Let $n \in \mathbb{N}_+$ be a positive integer. $[n]$ denotes the set $\{1, ..., n\}$. For any set $\mathcal{S}$, $|\mathcal{S}|$ denotes the number of elements in $\mathcal{S}$. For any event $\mathcal{E}$, we use $\mathbb{I}\{\mathcal{E}\}$ to denote the indicator function, where $\mathbb{I}\{\mathcal{E}\} = 1$ if $\mathcal{E}$ holds and $\mathbb{I}\{\mathcal{E}\} = 0$ otherwise. We use boldface lowercase letters and boldface capitalized letters to represent column vectors and matrices, respectively. For vector norms, $\|\boldsymbol{x}\|_p$ denotes the ℓ_p norm of vector $\boldsymbol{x}$. For any symmetric positive semi-definite (PSD) matrix $\boldsymbol{M}$ (i.e., $\boldsymbol{x}^\top \boldsymbol{M}\boldsymbol{x} \geq 0, \forall \boldsymbol{x}$), $\|\boldsymbol{x}\|_{\boldsymbol{M}} = \sqrt{\boldsymbol{x}^\top \boldsymbol{M}\boldsymbol{x}}$ denotes the matrix norm of $\boldsymbol{x}$ regarding matrix $\boldsymbol{M}$.

Federated Contextual Cascading Bandits. In this section, we formulate the setting of "Federated Contextual Combinatorial Cascading Bandits" (FedC^{3}B). We consider a finite set $\mathcal{I} \triangleq \{1, ..., I\}$ of *ground items* to be selected with $I \in \mathbb{N}_+$, referred to as *base arms*. Let $\Pi(\mathcal{I}) = \{(a_1, ..., a_k) : k \geq 1, a_1, ..., a_k \in \mathcal{I}, a_i \neq a_j \text{ for any } i \neq j\}$ be the set of all possible tuples of distinct items selected from $\mathcal{I}$, where we refer each of such tuples as an *action*. For any action $\boldsymbol{a} \in \mathcal{A}$, it involves at most K items, i.e., $\text{len}(\boldsymbol{a}) \leq K$, where $K \leq I$.

In FedC^{3}B, there is a finite set $\mathcal{U}$ of users, to be served by $|\mathcal{U}|$ (local) learning agents individually. Each learning agent serves one user u in order to provide personalized service to that user.[1] Each user $u \in \mathcal{U}$ is associated with an *unknown* preference vector $\boldsymbol{\theta}_u \in \mathbb{R}^d$, with $\|\boldsymbol{\theta}_u\|_2 \leq 1$, where $d \in \mathbb{N}_+$. To characterize the heterogeneity inherent in the user population, we assume users are partitioned into disjoint clusters, and two users share the same preference vector, if and only if they belong to the same cluster. Specifically, let $\mathcal{V}_1, \dots \mathcal{V}_J$ denote the partitioned clusters, where $J \in \mathbb{N}_+, J \leq |\mathcal{U}|$, $\cup_{j\in[J]} \mathcal{V}_j = \mathcal{U}$ and $\mathcal{V}_j \cap \mathcal{V}_{j'} = \emptyset$, for $j \neq j'$. Moreover, $\boldsymbol{\theta}_u = \boldsymbol{\theta}_{u'}$ if and only if there exists $j \in [J]$ such that $\{u, u'\} \subseteq \mathcal{V}_j$. For the ease of presentation, let $\boldsymbol{\theta}^j$ denote the shared preference vector for cluster $\mathcal{V}_j$. These clusters, termed *ground-truth clusters*, remain *unknown* to the agent. Importantly, in the scenario where $J = 1$, it is implied that $\mathcal{U}$ constitutes homogeneous users.

We consider a total number of $T \in \mathbb{N}_+$ learning rounds. In each round $t \in [T]$, a single user identified as u_t arrives to receive service. The user u_t is presented with a finite set of items $\mathcal{I}_t \subseteq \mathcal{I}$ and feasible action set $A_t \subseteq \Pi(\mathcal{I}_t)$. Let $\boldsymbol{x}_{t,i} \in \mathbb{R}^d$ denote the feature vector of item $i \in \mathcal{I}_t$, where $\|\boldsymbol{x}_{t,i}\|_2 \leq 1$, which are revealed to the agent u_t responsible for user u_t. The agent serving u_t takes a feasible action $\boldsymbol{a}_t = (a_{1,t}, ..., a_{\texttt{len}(\boldsymbol{a}_t),t}) \in \mathcal{A}_t$, i.e., recommends a list of items $\boldsymbol{a}_t$ to the user.[2] The user u_t checks these items $\boldsymbol{a}_t$ in sequence, beginning with the first, clicking on the first attractive item, and stopping checking items after the clicked item. We use the Bernoulli random variable $w_t(i) \in \{0, 1\}$ to indicate whether item $i \in \mathcal{I}_t$ would be clicked or not once checked. The learning agent observes the index of the first clicked item or $+\infty$ (if no item is clicked), defined as

$$O_t = \min\{1 \leq k \leq \texttt{len}(\boldsymbol{a}_t) : w_t(a_{k,t}) = 1\} \quad (1)$$

Here O_t fully determines $w_t(a_{k,t}) = 1 - \mathbb{I}\{k < O_t\}$ for $k = 1, ..., \min\{O_t, \texttt{len}(\boldsymbol{a}_t)\}$. Accordingly, we say that item i is observed in round t, if $O_t < \infty$ and $a_{t,O_t} = i$. Since the user clicks are shaped by their personal inclinations, the random vector $(w_t(i))_{i\in\mathcal{I}_t}$ is presumed to be independent of users other than u_t. We assume $w_t(i)$'s across different items are conditionally independent with expectation

$$\bar{w}_t(i) \triangleq \mathbb{E}[w_t(i) \mid \mathcal{H}_t^{(u_t)}] = \langle \boldsymbol{\theta}_{u_t}, \boldsymbol{x}_{t,i} \rangle. \quad (2)$$

[1] We also use u to identify the agent who serves user u.

[2] We omit superscript u_t for $\boldsymbol{a}_t^{(u_t)}$, etc., when contexts are clear.

where $\mathcal{H}_t^{(u_t)}$ is the history associated with user u_t: $\mathcal{H}_t^{(u_t)} \triangleq \{\boldsymbol{x}_{t,i}\}_{i\in\mathcal{I}_t} \cup \{\{\boldsymbol{x}_{s,i}\}_{i\in\mathcal{I}_t}, \boldsymbol{a}_s, O_s, w_s(a_{k,s})\}_{k\le O_s, s<t, u_s=u_t}$. We make the following assumptions.

Assumption 1 (Minimum heterogeneity gap)**.** *The gap between any two preference vectors for different ground-truth clusters is at least γ, $\|\boldsymbol{\theta}^j - \boldsymbol{\theta}^{j'}\|_2 \ge \gamma, \forall j, j' \in [J], j \ne j'$, where $\gamma > 0$ is an unknown positive constant.*

Assumption 2 (Random arrival of users)**.** *At each round t, a user u_t comes uniformly at random from $\mathcal{U}$ with probability $1/|\mathcal{U}|$, independent of the past rounds.*

Assumption 3 (Item regularity)**.** *At each time step t, the feature vector $\boldsymbol{x}_{t,a}$ of each arm $a \in E$ is drawn independently from a fixed but unknown distribution ρ over $\{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 \le 1\}$, where $\mathbb{E}_{\boldsymbol{x}\sim\rho}[\boldsymbol{x}\boldsymbol{x}^\top]$ is full rank with minimal eigenvalue $\lambda_x > 0$. Additionally, at any time t, for any fixed unit vector $\boldsymbol{\theta} \in \mathbb{R}^d$, $(\boldsymbol{\theta}^\top \boldsymbol{x})^2$ has sub-Gaussian tail with variance upper bounded by σ^2.*

Remark. All these assumptions align with previous works on CB (Gentile, Li, and Zappella 2014; Li and Zhang 2018; Liu et al. 2022a; Wang et al. 2023). Notably, Assumption 3 stands as a more practical and less restrictive alternative compared to previous CB works that impose restrictions on the variance upper bound σ^2. Assumption 2, our results can easily generalize to arbitrary distributions with minimum arrival probability greater than $p_{\min} > 0$.

Learning Objective. At round t, the reward of each action $\boldsymbol{a} \in \mathcal{A}_t$ with random vector $\boldsymbol{w}_t \triangleq (w_t(i))_{i\in\mathcal{I}_t}$ is defined as:

$$f(\boldsymbol{a}, \boldsymbol{w}_t) \triangleq 1 - \prod_{k=1}^{\texttt{len}(\boldsymbol{a})}(1 - w_t(a_k)). \tag{3}$$

Let $\mathcal{H}_t = \bigcup_{u\in\mathcal{U}} \mathcal{H}_t^u$ be the total historical information up to time t, then by independence assumption, it is easy to verify that the expected reward is $\mathbb{E}[f(\boldsymbol{a}, \boldsymbol{w}_t)|\mathcal{H}_t] = f(\boldsymbol{a}, \bar{\boldsymbol{w}}_t)$. Let $\boldsymbol{a}_t^* = \operatorname{argmax}_{\boldsymbol{a}\in\mathcal{A}_t} f(\boldsymbol{a}, \bar{\boldsymbol{w}}_t)$ be the optimal action at t. Given the complexity of computing the optimal action $\boldsymbol{a}_t^*$ for a general feasible action set $\mathcal{A}_t$, even when $\boldsymbol{\theta}_{u_t}$ is known, we introduce an offline α-approximation oracle. This oracle, provided to the agent, generates an action $\widetilde{\boldsymbol{a}}$ for any weight vector $\boldsymbol{w} \in \mathbb{R}^{I_t}$ such that $f(\widetilde{\boldsymbol{a}}, \boldsymbol{w}) \ge \alpha \max_{\boldsymbol{a}\in\mathcal{A}_t} f(\boldsymbol{a}, \boldsymbol{w})$. The goal of the agents is to collaboratively minimize the cumulative α-approximate regret defined as

$$\text{Reg}(T) = \mathbb{E}\left[\sum_{t=1}^T \left(\alpha f(\boldsymbol{a}_t^*, \boldsymbol{w}_t) - f(\boldsymbol{a}_t, \boldsymbol{w}_t)\right)\right], \tag{4}$$

where the expectation is taken over the user arrival, the contexts, the weights, and the algorithm itself.

Communication model. We adopt the server-client communication protocol with $|\mathcal{U}|$ local agents and one central server, where each agent can communicate with the server by uploading and downloading data. Unlike peer-to-peer communication (Korda, Szorenyi, and Li 2016; Zhu et al. 2021), local agents do not communicate with each other directly. Moreover, we adopt the *asynchronous communication* paradigm: (1) there is no mandatory synchronization, (2) the communication between an agent and the server operates independently of other agents, without triggering additional communication. Finally, we define the communication cost as $\text{Com}(T) = \mathbb{E}[\sum_{t=1}^T \mathbb{I}\{\text{agent } u_t \text{ communications with the server}\}]$.

Algorithm 1: FedC^{3}UCB-H

1: **Input:** Communication and deletion thresholds $\alpha_c, \alpha_d > 0$, probability $p_t = 3\log t/t$, regularizer λ
2: **Initialize server:** complete graph $G_0 = (\mathcal{U}, E_0)$ over agents. $T_{u,0}^{ser} = 0$, $\boldsymbol{b}_{u,0}^{ser} = \boldsymbol{b}_{u,0}^{clu} = \boldsymbol{0}_{d\times 1}$, $\hat{\boldsymbol{\theta}}_{u,0}^{ser} = \hat{\boldsymbol{\theta}}_{u,0}^{clu} = \boldsymbol{0}_{d\times 1}$, $\boldsymbol{\Sigma}_{u,0}^{ser} = \boldsymbol{\Sigma}_{u,0}^{clu} = \boldsymbol{0}_{d\times d}$
3: **Initialize agents**: $T_{u,0}^{loc} = 0$, $\boldsymbol{b}_{u,0} = \boldsymbol{b}_{u,0}^{loc} = \boldsymbol{0}_{d\times 1}$, $\hat{\boldsymbol{\theta}}_{u,0} = \boldsymbol{0}_{d\times 1}$, $\boldsymbol{\Sigma}_{u,0} = \boldsymbol{\Sigma}_{u,0}^{loc} = \boldsymbol{0}_{d\times d}$
4: **for** round $t = 1, ..., T$ **do**
5: User u_t arrives to be served
6: ComInd $\leftarrow$`LocalAgent`(t, u_t, p_t, α_c)
7: **if** ComInd==1 **then**
8: Agent u_t sends $\boldsymbol{\Sigma}_{u_t,t}^{loc}$, $T_{u_t,t}^{loc}$ and $\boldsymbol{b}_{u_t,t}^{loc}$ to server
9: Run `Server`(t, u_t, α_d)
10: Server sends $\boldsymbol{\Sigma}_{u_t,t+1}^{clu}$ and $\hat{\boldsymbol{\theta}}_{u_t,t+1}^{clu}$ back to agent u_t
11: Agent u_t update: $\boldsymbol{\Sigma}_{u_t,t+1}^{loc} = \boldsymbol{0}_{d\times d}$, $\boldsymbol{b}_{u_t,t+1}^{loc} = \boldsymbol{0}_{d\times 1}$, $T_{u_t,t}^{loc} = 0$, $\boldsymbol{\Sigma}_{u_t,t+1} = \boldsymbol{\Sigma}_{u_t,t+1}^{clu}$, $\hat{\boldsymbol{\theta}}_{u_t,t+1} = \hat{\boldsymbol{\theta}}_{u_t,t+1}^{clu}$
12: **end if**
13: **end for**

The Proposed Algorithm

This section presents our algorithm, named "Federated Contextual Combinatorial Cascading Upper Confidence Bound Algorithm for Heterogeneous Users" (FedC^{3}UCB-H), which is outlined in Algorithm 1. High levelly, the main task of FedC^{3}UCB-H is to identify which agents can collaborate based on user heterogeneity. Agents who can collaborate aggregate their data asynchronously to address the agent-side cascading bandit problems. FedC^{3}UCB-H consists of three main components as follows. For simplicity, we summarize the related notations in Table 3.

Notation	Meaning
$\hat{\boldsymbol{\theta}}_{u,t}, \boldsymbol{\Sigma}_{u,t}$	Data for agent u's decision
$\boldsymbol{\Sigma}_{u,t}^{loc}, \boldsymbol{b}_{u,t}^{loc}, T_{u,t}^{loc}$	Local data of agent u
$\boldsymbol{\Sigma}_{u,t}^{ser}, \boldsymbol{b}_{u,t}^{ser}, T_{u,t}^{ser}$	Data stored at the server for agent u
$\hat{\boldsymbol{\theta}}_{u,t}^{clu}, \boldsymbol{\Sigma}_{u,t}^{clu}, \boldsymbol{b}_{u,t}^{clu}$	Data of the cluster that contains u

Table 3: Notations used in Algorithm 1, 2 and 3.

Agent-side combinatorial action selection. This component aims to select combinatorial action based on data received from the server. In each round $t \in [T]$, a random user u_t arrives, and the corresponding agent u_t will interact with this user (line 5 in Algorithm 1). The agent u_t then calls the subroutine Algorithm 2 (line 6 in Algorithm 1). During this process, u_t receives the feature vector $\boldsymbol{x}_{t,i}$ for each item $i \in \mathcal{I}_t$ (line 1 in Algorithm 2). Utilizing the estimated preference vector $\hat{\boldsymbol{\theta}}_{u_t,t}$ and the gram matrix $\boldsymbol{\Sigma}_{u_t,t}$, the agent calculates an optimistic UCB value $U_t^{(u_t)}(i)$ for the true weight $\bar{w}_t(i)$ of each item i (line 2 in Algorithm 2). This UCB value balances exploration and exploitation, with $\beta \|\boldsymbol{x}_{t,i}\|_{\boldsymbol{\Sigma}_{u_t,t}^{-1}}$ accounting for the uncertainty of the estimated weight $\langle \hat{\boldsymbol{\theta}}_{u_t,t}, \boldsymbol{x}_{t,i} \rangle$ and encouraging more exploration when

Algorithm 2: `LocalAgent`(t, u_t, p_t, α_c)

1: Receives context $\boldsymbol{x}_{t,i}, \forall i \in \mathcal{I}_t$
2: $U_t(i) \leftarrow \min\left\{\hat{\boldsymbol{\theta}}_{u_t,t}^T \boldsymbol{x}_{t,i} + \beta \left\|\boldsymbol{x}_{t,i}\right\|_{\boldsymbol{\Sigma}_{u_t,t}^{-1}}, 1\right\}, \forall i \in \mathcal{I}_t$
3: $\boldsymbol{a}_t \leftarrow oracle((U_t(i))_{i \in \mathcal{I}_t})$
4: Play $\boldsymbol{a}_t$ and observe $O_t, w_t(a_{k,t})$ for $k \leq O_t$, and receive reward $f(\boldsymbol{a}_t, \boldsymbol{w}_t)$
5: $T_{u_t,t}^{loc} = T_{u_t,t-1}^{loc} + O_t$
6: $\boldsymbol{\Sigma}_{u_t,t}^{loc} = \boldsymbol{\Sigma}_{u_t,t-1}^{loc} + \sum_{k=1}^{O_t} \boldsymbol{x}_{t,a_{k,t}} \boldsymbol{x}_{t,a_{k,t}}^\top$, $\boldsymbol{b}_{u_t,t}^{loc} = \boldsymbol{b}_{u_t,t-1}^{loc} + \sum_{k=1}^{O_t} w_t(a_{k,t}) \boldsymbol{x}_{t,a_{k,t}}$
7: **for** $u \neq u_t$ **do**
8: $\quad T_{u,t}^{loc} = T_{u,t-1}^{loc}$
9: $\quad \boldsymbol{\Sigma}_{u,t}^{loc} = \boldsymbol{\Sigma}_{u,t-1}^{loc}, \boldsymbol{b}_{u,t}^{loc} = \boldsymbol{b}_{u,t-1}^{loc}$
10: $\quad \boldsymbol{\Sigma}_{u,t+1} = \boldsymbol{\Sigma}_{u,t}, \boldsymbol{b}_{u,t+1} = \boldsymbol{b}_{u,t}, \hat{\boldsymbol{\theta}}_{u,t+1} = \hat{\boldsymbol{\theta}}_{u,t}$
11: **end for**
12: Generate a uniform random number $p \in [0, 1]$
13: **if** $\det(\boldsymbol{\Sigma}_{u_t,t} + \boldsymbol{\Sigma}_{u_t,t}^{loc}) > (1 + \alpha_c)\det(\boldsymbol{\Sigma}_{u_t,t})$ **or** $p < p_t$ **then**
14: $\quad$ **Return:** 1
15: **else**
16: $\quad \boldsymbol{\Sigma}_{u_t,t+1} = \boldsymbol{\Sigma}_{u_t,t}, \boldsymbol{b}_{u_t,t+1} = \boldsymbol{b}_{u_t,t}, \hat{\boldsymbol{\theta}}_{u_t,t+1} = \hat{\boldsymbol{\theta}}_{u_t,t}$
17: $\quad G_{t+1} = G_t$
18: $\quad$ **Return:** 0
19: **end if**

Algorithm 3: `Server`(t, u_t, α_d)

1: From graph $G_t = (\mathcal{U}, E_t)$, server identifies the connected component $\mathcal{V}_t$ that user u_t belongs to
2: $\boldsymbol{\Sigma}_{u_t,t+1}^{clu} = \lambda I + \sum_{u \in \mathcal{V}_t} \boldsymbol{\Sigma}_{u,t}^{ser}$, $\boldsymbol{b}_{u_t,t+1}^{clu} = \sum_{u \in \mathcal{V}_t} \boldsymbol{b}_{u,t}^{ser}$, $\hat{\boldsymbol{\theta}}_{u_t,t+1}^{clu} = (\boldsymbol{\Sigma}_{u_t,t+1}^{clu})^{-1} \boldsymbol{b}_{u_t,t+1}^{clu}$
3: $T_{u_t,t}^{ser} = T_{u_t,t-1}^{ser} + T_{u_t,t}^{loc}$
4: $\boldsymbol{\Sigma}_{u_t,t}^{ser} = \boldsymbol{\Sigma}_{u_t,t-1}^{ser} + \boldsymbol{\Sigma}_{u_t,t}^{loc}$, $\boldsymbol{b}_{u_t,t}^{ser} = \boldsymbol{b}_{u_t,t-1}^{ser} + \boldsymbol{b}_{u_t,t}^{loc}$, $\hat{\boldsymbol{\theta}}_{u_t,t}^{ser} = (\lambda I + \boldsymbol{\Sigma}_{u_t,t}^{ser})^{-1} \boldsymbol{b}_{u_t,t}^{ser}$
5: Reset deletion set $\tilde{E} = \emptyset$
6: **for** $u \neq u_t$ and $(u, u_t) \in E_t$ **do**
7: $\quad T_{u,t}^{ser} = T_{u,t-1}^{ser}$
8: $\quad \boldsymbol{\Sigma}_{u,t}^{ser} = \boldsymbol{\Sigma}_{u,t-1}^{ser}$, $\boldsymbol{b}_{u,t}^{ser} = \boldsymbol{b}_{u,t-1}^{ser}$, $\hat{\boldsymbol{\theta}}_{u,t}^{ser} = (\lambda I + \boldsymbol{\Sigma}_{u,t}^{ser})^{-1} \boldsymbol{b}_{u,t}^{ser}$
9: $\quad$ **if** $\left\|\hat{\boldsymbol{\theta}}_{u_t,t}^{ser} - \hat{\boldsymbol{\theta}}_{u,t}^{ser}\right\| > \alpha_d\left(\sqrt{\frac{1+\ln(1+T_{u_t,t}^{ser})}{1+T_{u_t,t}^{ser}}} + \sqrt{\frac{1+\ln(1+T_{u,t}^{ser})}{1+T_{u,t}^{ser}}}\right)$ **then**
10: $\quad\quad$ Server puts edge (u_t, u) into deletion set $\tilde{E}$
11: $\quad$ **end if**
12: **end for**
13: Update $E_{t+1} = E_t \backslash \tilde{E}$ and obtain new graph $G_{t+1} = (\mathcal{U}, E_{t+1})$

it is large. It is crucial to note that $\hat{\boldsymbol{\theta}}_{u_t,t}$ and $\boldsymbol{\Sigma}_{u_t,t}$ not only consist of data from user u_t but also from users within the same estimated cluster $\mathcal{V}$ that includes u_t, which is identified by the heterogeneity testing at the server to be introduced later. Due to the data inconsistency caused by the asynchronous communication, the choice of β is carefully designed and theoretically supported by Lemma 1.

After computing UCB values for each arm, the agent employs the computation oracle to generate a ranked list $\boldsymbol{a}_t \in \mathcal{A}_t$ (line 3 in Algorithm 2). The user then scans through this list $\boldsymbol{a}_t$, receives the reward $f(\boldsymbol{a}_t, \boldsymbol{w}_t)$ (Eq. (3)) according to the random weight $\boldsymbol{w}_t$, and observes the partial feedback O_t as defined in Eq. (1). After receiving the feedback, agent u_t updates its local data and stores them in the local buffer (lines 5-6 in Algorithm 2). Finally, based on certain conditions, u_t evaluates whether it should communicate with the server (line 13 in Algorithm 2), which will be covered later in our asynchronous communication protocol.

Server-side user heterogeneity testing. Different from Wang et al. (2019); He et al. (2022) where users are assumed to be homogeneous, the server in our setting must address the user heterogeneity. Specifically, the server's task is to partition users into distinct clusters. Users from different clusters are viewed as heterogeneous and should not collaborate with each other. To address this task, the server maintains an undirected graph $G_t = (\mathcal{U}, E_t)$ over agents. This graph connects agents via edges if they are estimated to belong to the same cluster. Initially, G_0 is initialized as a complete graph, and it will be updated adaptively based on the uploaded information after each communication. At round t, if the communication condition in line 13 of Algorithm 2 is satisfied with communication indicator ComInd $= 1$, u_t uploads its local data (line 8 in Algorithm 1). The server then updates the agent u_t's information (lines 2-4 in Algorithm 3). Next, the server verifies the estimated heterogeneity between user u_t and other users based on the updated estimation. In particular, for each user u connected to u_t via edge $(u_t, u) \in E_{t-1}$, if the gap between her estimated preference vectors $\hat{\boldsymbol{\theta}}_{u,t}^{ser}$ and $\hat{\boldsymbol{\theta}}_{u_t,t}^{ser}$ exceeds a certain threshold (line 9 in Algorithm 3), the server removes edge (u_t, u) to separate them (line 10 in Algorithm 3). This threshold is carefully set to ensure that, with high probability, edges between heterogeneous users (those not in the same *ground-truth clusters*) are deleted after a short span of time steps. Conversely, edges connecting users within the same *ground-truth clusters* are retained. After user clusters are correctly identified, our algorithm will leverage collaborative information between homogeneous users while avoiding misuse of information from heterogeneous users to reduce the overall uncertainty. In line 13 of Algorithm 3, the server uses graph G_t to identify the connected component $\mathcal{V}_t$ in which u_t resides, and sends back to u_t the updated collaborative information of $\mathcal{V}_t$ for future decisions.

Asynchronous communication protocol. This protocol addresses the key question of when u_t should communicate with the server to share her information and to keep her updated with the newest information. Striking the right balance is the main challenge because excessive communication incurs high communication costs, while insufficient communication results in inaccurate estimation for the local agents and prevents the server from updating graph G_t effectively.

The communication protocol involves a two-step verification process to decide whether the agent needs to activate a communication. Inspired by He et al. (2022), the first step is to verify a matrix determinant-based condition (line

13 in Algorithm 2). This condition gauges the gap between information accumulated locally and collaborative information shared during the previous communication. If satisfied, it signifies that enough local data have been collected to significantly reduce global model uncertainty. Thus, agent u_t triggers a communication with the server, leading to global model updates (line 2 in Algorithm 3). Subsequently, the server provides the latest collaborative information to agent u_t (line 10 in Algorithm 1). Notably, this communication indirectly aids the update of graph G_t (line 13 in Algorithm 3).

Our communication protocol involves a two-step verification process to decide whether a communication needs to be activated. Inspired by He et al. (2022), the first step is to verify a matrix determinant-based condition (line 13 in Algorithm 2). This condition measures the gap between the information accumulated locally and the collaborative information shared during the previous communication. If satisfied, it means that enough local data have been collected to significantly reduce global model uncertainty. Thus, agent u_t will trigger a communication with the server, leading to the update of the global model at the server (line 2 in Algorithm 3). Subsequently, the server returns the lastest collaborative information to agent u_t (line 10 in Algorithm 1). Note that as a by-product, this communication will also help to update the graph G_t (line 13 in Algorithm 3). However, this by-product communication isn't enough to ensure that the server's heterogeneity testing of all agents is completed in sublinear time, which motivates our second condition to increase the number of communications. Specifically, we introduce a certain probability p_t that user u_t activates an auxiliary communication at round t, as a complement to the matrix determinant-based condition. We refer to this protocol as *p_t-auxiliary communication protocol*. The key challenge is to determine suitable p_t, and through careful analysis, we set $p_t = 3\log t/t$. A deterministic version of this protocol, the force-communication protocol, is also offered for uncertainty-averse applications. Note that we also provide a deterministic version of the ϵ-auxiliary condition, i.e., the force-communication protocol, for uncertainty-averse applications. However, our theory and experiments show that the force-communication may yield worse performance both in regret and communication, whose details and analysis are postponed to the Appendix. In cases where neither the matrix determinant-based condition nor the p_t-auxiliary condition is met, communication between the agent and the server is not activated, and local data remain local. For inactive agents, their data also remains unchanged.

Theoretical Analysis

Theorem 1. *Under Assumptions 1-3 stated in the section of Problem Setup, if we set β as in Lemma 1 and $\tilde{\lambda}_x \triangleq \int_0^{\lambda_x}(1-e^{-(\lambda_x-x)^2/2\sigma^2})^I dx$, then the expected cumulative regret of Algorithm 1 is upper bounded by*

$$Reg(T) \leq O\big(d\sqrt{(J+|\mathcal{U}|\alpha_c)(1+|\mathcal{U}|^2\alpha_c)KT}\log T + |\mathcal{U}|(d\gamma^{-2}\tilde{\lambda}_x^{-1}+\tilde{\lambda}_x^{-2})\log T\big). \quad (5)$$

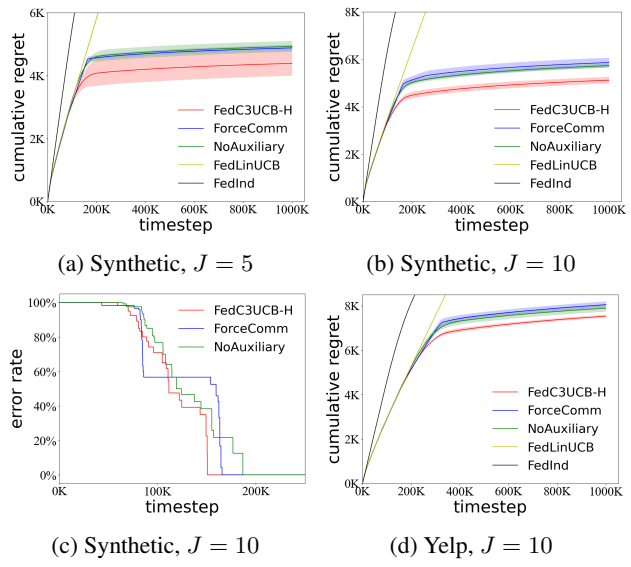
The communication cost is bounded by $Com(T) \leq O\left(d(|\mathcal{U}|+1/\alpha_c)\log T + \log^2 T\right)$

Discussion and Comparison. Looking at the above theorem, by setting $\alpha_c = 1/|\mathcal{U}|^2$, we can bound the regret by $\tilde{O}(d\sqrt{JKT})$, while maintaining a near-constant $\tilde{O}(d|\mathcal{U}|^2)$ communication cost, which is exactly the result presented in row 8 of Table 2. Compared with the state-of-the-art CLUB-cascade algorithm (Li and Zhang 2018) which incurs a $O(T)$ communication cost (row 4 of Table 2), our algorithm significantly reduces the communication cost to logarithmic terms regarding T, while obtaining the same regret bounds. In the scenario where users are homogeneous ($J = 1$) and only a single arm is selected per round ($K = 1$), our regret matches He et al. (2022), incurring only an additional $O(\log^2 T)$ communication cost due to the p_t-auxiliary communication. As for the lower bound, Liu et al. (2022a) establishes a lower bound of $\Omega(\sqrt{dJT})$ with unlimited communication, where our result is near-optimal and matches this lower bound up to a factor $O(\sqrt{dK})$. For the communication cost, He et al. (2022) shows $\Omega(|\mathcal{U}|)$ communication is needed to achieve near-optimal regret, and removing the $O(d|\mathcal{U}|)$ communication gap is still a challenging open problem.

Proof for Theorem 1. Due to space limit, we give a sketched proof here. Our regret analysis mainly contains two parts. The first part upper bounds the number of exploration rounds required to achieve accurate user partitioning at the server. In particular, leveraging assumptions of item regularity and user arrivals, we prove that the user groups would be correctly partitioned if all user data are synchronized and uploaded to the server exactly at (or after) $T_0 = O(|\mathcal{U}|(d\gamma^{-2}\tilde{\lambda}_x^{-1}+\tilde{\lambda}_x^{-2})\log T)$ timesteps. However, since γ and $\tilde{\lambda}_x$ are unknown (which makes T_0 unknown) to the agents or the server, we devise a p_t-auxiliary communication that is agnostic to T_0 and ensures that after at most $T_1 = O(|\mathcal{U}|\log T + \sqrt{T})$ extra rounds, the server obtains the correct user partition. For the second part of our analysis after T_0+T_1, agents sharing the same $\boldsymbol{\theta}$ begin to collaborate with each other under server coordination and we prove the following concentration bound:

Lemma 1. *Let $\beta = \sqrt{\lambda} + (\sqrt{1+|\mathcal{U}|\alpha_c} + |\mathcal{U}|\sqrt{2\alpha_c})\sqrt{d\log(1+\frac{KT}{\alpha_c\lambda d}) + 2\log(1/\delta) + 4\log(T|\mathcal{U}|)}$, for $t > T_0 + T_1$ and every $u \in \mathcal{U}$ in true user cluster $j(u)$, it holds with probability at least $1-\delta$ that $\left\|\hat{\boldsymbol{\theta}}_{u,t} - \boldsymbol{\theta}^{j(u)}\right\|_{\boldsymbol{\Sigma}_{u,t}} \leq \beta$.*

The key challenge of proving Lemma 1 is to handle the gap between the delayed $\boldsymbol{\Sigma}_{u,t}$ and up-to-date information had agents been synchronized. We address the challenge by bounding this gap by a factor of $O(J + |\mathcal{U}|\alpha_c)$ thanks to the matrix determinant condition in line 13 of Algorithm 2. Based on this concentration bound, we conduct a contextual combinatorial cascading bandit regret analysis to bound the final regret. For the communication cost, the matrix-determinant condition and the p_t-auxiliary communication contribute $O(d(|\mathcal{U}|+1/\alpha_c)\log T)$ and $O(\log^2 T)$, respectively. For detailed proofs, please refer to our Appendix. □



(a) Synthetic, $J = 5$ (b) Synthetic, $J = 10$

(c) Synthetic, $J = 10$ (d) Yelp, $J = 10$

Figure 1: Cumulative regret and error rates of user clustering

Empirical Evaluation

In this section, we compare FedC^{3}UCB-H against four baseline algorithms: (1) FedLinUCB (He et al. (2022)), adapted to accommodate cascading arms while maintaining a single estimated vector for all agents, (2) FedInd, where agents operate independent cascading bandit algorithms locally without any communication, (3) NoAuxiliary, a variant of FedC^{3}UCB-H that omits the p_t-auxiliary communication condition, and (4) ForceComm, which replaces the p_t-auxiliary communication with a deterministic force-communication protocol. In the ForceComm protocol, each agent communicates with the server upon its arrival at time steps $\{1, 2, 4, ..., 2^n\}$. The baselines (1) and (2) aim to evaluate the significance of user cluster identification, while the baselines (3) and (4) show the importance of our p_t-auxiliary communication. We report the averaged regret, communication cost, and error rates of user clustering (the number of incorrect users divided by $|\mathcal{U}|$; users in a correctly partitioned cluster are considered correct, while others are incorrect) over three independent runs with standard deviations.

Synthetic Experiments. We conduct experiments in a synthetic environment with $|\mathcal{U}| = 40$ users, $J = 5$ or $J = 10$ clusters with orthogonal $\boldsymbol{\theta}$, and $T = 1000,000$ rounds. The preference and feature vectors are of dimension $d = 20$, with each entry drawn from a standard Gaussian distribution, and are normalized to vectors with $\|.\|_2 = 1$ (Li et al. 2019). At round t, a user u_t randomly chosen from $\mathcal{U}$ and and $I_t = 200$ items are generated, where each item is associated with a random $\boldsymbol{x}_{t,i} \in \mathbb{R}^d$ generated as above. The agent needs to recommend $K = 4$ items as cascading arms $\boldsymbol{a}_t$ to the user, leading to a random reward $f(\boldsymbol{a}_t, \boldsymbol{w}_t)$ as defined in Eq. (3). The agents then observed feedback according to Eq. (1). Results for the $J = 5$ scenario are shown in Fig. 1a, while Fig. 1b depicts the results for $J = 10$. In both cases, FedC^{3}UCB-H outperformes the baseline algorithms. For the $J = 10$ setting, FedC^{3}UCB-H achieves an 82% reduction in regret compared to FedLinUCB and a 64% reduction compared to FedInd. This shows the effectiveness of our user clustering procedure. Compared with NoAuxiliary, FedC^{3}UCB-H achieves an 10% reduction in regret, indicating the importance of our p_t-auxiliary communication. This is confirmed by Fig. 1c, where NoAuxiliary exhibited slower convergence in correctly identifying user clusters. When compared to ForceComm, FedC^{3}UCB-H achieved 12% less regret and 5% less communication. Although ForceComm initially identifies user clusters more rapidly (Fig. 1c), it incurs unnecessary communication after users are correctly partitioned, causing it to perform worse over time. FedC^{3}UCB-H, instead, strikes a better balance. The $J = 10$ scenario produces consistent results, reaffirming the observations in the $J = 5$ scenario.

	Synthetic		Yelp
	$J = 5$	$J = 10$	$J = 10$
FedC^{3}UCB-H	818.3	1070.7	951.3
ForceComm	909.3	1125.3	1034.0
NoAuxiliary	666.7	899.3	789.3
FedLinUCB	479.7	482.3	486.0

Table 4: Communication cost for different datasets.

Experiments on Real Dataset. We conduct experiments on the Yelp dataset, which contains 4.7 million ratings of 1.57×10^5 restaurants from 1.18 million users.[3] Since Yelp dataset is very sparse, we extract 1000 items with the most ratings and 1000 users who rate most. This subset forms a new rating matrix $\boldsymbol{M}^{1000\times1000}$. We use matrix $\boldsymbol{M}$ to generate feature vectors and preference vectors of $d = 20$ dimension for all items and users by singular-value decomposition (SVD) (Li and Zhang 2018; Li et al. 2019; Liu et al. 2022a), respectively. Then we randomly sample $|\mathcal{U}| = 40$ users and use k-means clustering (Ahmed, Seraj, and Islam 2020) to generate $J = 10$ user clusters. Each cluster's center vector represents the true preference vector for users within that cluster. The experiment's remaining parameters match those of the synthetic experiments. Fig. 1d shows the regret comparison and Table 4 shows the communication cost. The results are consistent with the synthetic experiments. FedC^{3}UCB-H achievs a 67% reduction in regret compared to FedLinUCB, 43% compared to FedInd, 5% compared to NoAuxiliary, and 6% compared to ForceComm. As for the communication cost, FedC^{3}UCB-H has higher communication costs than NoAuxiliary and FedLinUCB due to auxiliary communication, but manages to reduce communication by 8% compared to ForceComm.

Future Directions

It would be interesting to eliminate the $O(\sqrt{dK})$ gap in the regret and the $O(d|\mathcal{U}|)$ gap in the communication. It would also be valuable to generalize the linear structure by considering non-linear structures or model mis-specifications.

[3] http://www.yelp.com/dataset challenge

Acknowledgments

We thank the anonymous reviewers for their helpful comments. The work of John C.S. Lui was supported in part by the RGC GRF 14202923.

References

Abbasi-Yadkori, Y.; Pál, D.; and Szepesvári, C. 2011. Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems*, 24.

Ahmed, M.; Seraj, R.; and Islam, S. M. S. 2020. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8): 1295.

Choi, H.; Udwani, R.; and Oh, M.-h. 2023. Cascading Contextual Assortment Bandits. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Chuklin, A.; Markov, I.; and Rijke, M. d. 2015. Click models for web search. *Synthesis lectures on information concepts, retrieval, and services*, 7(3): 1–115.

Craswell, N.; Zoeter, O.; Taylor, M.; and Ramsey, B. 2008. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*, 87–94.

Dubey, A.; and Pentland, A. 2020. Differentially-private federated linear bandits. *Advances in Neural Information Processing Systems*, 33: 6003–6014.

Gentile, C.; Li, S.; and Zappella, G. 2014. Online clustering of bandits. In *International Conference on Machine Learning*, 757–765. PMLR.

He, J.; Wang, T.; Min, Y.; and Gu, Q. 2022. A Simple and Provably Efficient Algorithm for Asynchronous Federated Contextual Linear Bandits. *Advances in neural information processing systems*.

Korda, N.; Szorenyi, B.; and Li, S. 2016. Distributed clustering of linear bandits in peer to peer networks. In *International conference on machine learning*, 1301–1309. PMLR.

Kveton, B.; Szepesvari, C.; Wen, Z.; and Ashkan, A. 2015a. Cascading bandits: Learning to rank in the cascade model. In *International Conference on Machine Learning*, 767–776. PMLR.

Kveton, B.; Wen, Z.; Ashkan, A.; and Szepesvári, C. 2015b. Combinatorial cascading bandits. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*, 1450–1458.

Li, C.; and Wang, H. 2021. Asynchronous Upper Confidence Bound Algorithms for Federated Linear Bandits. *arXiv preprint arXiv:2110.01463*.

Li, S.; Chen, W.; Li, S.; and Leung, K.-S. 2019. Improved Algorithm on Online Clustering of Bandits. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, 2923–2929. AAAI Press. ISBN 9780999241141.

Li, S.; Wang, B.; Zhang, S.; and Chen, W. 2016. Contextual combinatorial cascading bandits. In *International conference on machine learning*, 1245–1253. PMLR.

Li, S.; and Zhang, S. 2018. Online clustering of contextual cascading bandits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Lian, D.; Gao, Z.; Song, X.; Li, Y.; Liu, Q.; and Chen, E. 2023. Training Recommenders Over Large Item Corpus With Importance Sampling. *IEEE Transactions on Knowledge and Data Engineering*.

Lian, D.; Liu, Q.; and Chen, E. 2020. Personalized Ranking with Importance Sampling. In *Proceedings of The Web Conference 2020*, 1093–1103.

Liu, X.; Zhao, H.; Yu, T.; Li, S.; and Lui, J. 2022a. Federated Online Clustering of Bandits. In *The 38th Conference on Uncertainty in Artificial Intelligence*.

Liu, X.; Zuo, J.; Wang, S.; Joe-Wong, C.; Lui, J.; and Chen, W. 2022b. Batch-size independent regret bounds for combinatorial semi-bandits with probabilistically triggered arms or independent arms. *Advances in Neural Information Processing Systems*, 35: 14904–14916.

Liu, X.; Zuo, J.; Wang, S.; Lui, J. C.; Hajiesmaili, M.; Wierman, A.; and Chen, W. 2023a. Contextual combinatorial bandits with probabilistically triggered arms. In *International Conference on Machine Learning*, 22559–22593. PMLR.

Liu, X.; Zuo, J.; Xie, H.; Joe-Wong, C.; and Lui, J. C. 2023b. Variance-adaptive algorithm for probabilistic maximum coverage bandits with general feedback. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*, 1–10. IEEE.

Vial, D.; Sanghavi, S.; Shakkottai, S.; and Srikant, R. 2022. Minimax Regret for Cascading Bandits. *arXiv preprint arXiv:2203.12577*.

Wang, Y.; Hu, J.; Chen, X.; and Wang, L. 2019. Distributed Bandit Learning: Near-Optimal Regret with Efficient Communication. *arXiv e-prints*.

Wang, Z.; Xie, J.; Liu, X.; Li, S.; and Lui, J. 2023. Online clustering of bandits with misspecified user models. *arXiv preprint arXiv:2310.02717*.

Wu, C.; Lian, D.; Ge, Y.; Zhu, Z.; and Chen, E. 2023. Influence-Driven Data Poisoning for Robust Recommender Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Xu, M.; and Klabjan, D. 2023. Decentralized Randomly Distributed Multi-agent Multi-armed Bandit with Heterogeneous Rewards. In *Advances on Neural Information Processing Systems*.

Zhu, Z.; Zhu, J.; Liu, J.; and Liu, Y. 2021. Federated bandit: A gossiping approach. In *Abstract Proceedings of the 2021 ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems*, 3–4.

Zong, S.; Ni, H.; Sung, K.; Ke, N. R.; Wen, Z.; and Kveton, B. 2016. Cascading bandits for large-scale recommendation problems. *arXiv preprint arXiv:1603.05359*.