Online Learning and Detecting Corrupted Users for Conversational Recommendation Systems

Xiangxiang Dai, *Student Member, IEEE*, Zhiyong Wang, Jize Xie, Tong Yu, and John C. S. Lui, *Fellow, IEEE*

***Abstract*—Conversational recommendation systems (CRSs) are increasingly prevalent, but they are susceptible to the influence of corrupted user behaviors, such as deceptive click ratings. These behaviors can skew the recommendation process, resulting in sub-optimal results. Traditional bandit algorithms, which are typically oriented to single users, do not capitalize on implicit social connections between users, which could otherwise enhance learning efficiency. Furthermore, they cannot identify corrupted users in a real-time, multi-user environment. In this paper, we propose a novel bandit problem, Online Learning and Detecting Corrupted Users (OLDCU), to learn and utilize unknown user relations from disrupted behaviors to speed up learning and detect corrupted users in an online setting. This problem is non-trivial due to the dynamic nature of user behaviors and the difficulty of online detection. To robustly learn and leverage the unknown relations among potentially corrupted users, we propose a novel bandit algorithm RCLUB-WCU, incorporating a conversational mechanism. This algorithm is designed to handle the complexities of disrupted behaviors and to make accurate user relation inferences. To detect corrupted users with bandit feedback, we further devise a novel online detection algorithm, OCCUD, which is based on RCLUB-WCU's inferred user relations and designed to adapt over time. We prove a sub-linear regret bound for RCLUB-WCU, demonstrating its efficiency. We also analyze the detection accuracy of OCCUD, showing its effectiveness in identifying corrupted users. Through extensive experiments, we validate the performance of our methods. Our results show that RCLUB-WCU and OCCUD outperform previous bandit algorithms and achieve high corrupted user detection accuracy, providing robust and efficient solutions in the field of CRSs.**

***Index Terms*—Adversarial corruption, online learning, conversational recommendation, bandit feedback, clustering of bandits.**

Received 11 March 2024; revised 3 July 2024; accepted 17 August 2024. Date of publication 22 August 2024; date of current version 13 November 2024. The work of John C.S. Lui was supported in part by RGC GRF-14202923. Recommended for acceptance by L. Nie. *(Xiangxiang Dai and Zhiyong Wang contributed equally to this work.) (Corresponding author: Zhiyong Wang.)*

Xiangxiang Dai, Zhiyong Wang, and John C. S. Lui are with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Sha Tin, Hong Kong (e-mail: xxdai23@cse.cuhk.edu.hk; zywang21@cse.cuhk.edu.hk; cslui@cse.cuhk.edu.hk).

Jize Xie is with the Department of Industrial Engineering and Decision Analytics, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong (e-mail: jxiebj@connect.ust.hk).

Tong Yu is with Adobe Research, San Jose, CA 94107 USA (e-mail: tyu@adobe.com).

This article has supplementary downloadable material available at https://doi.org/10.1109/TKDE.2024.3448250, provided by the authors.

Digital Object Identifier 10.1109/TKDE.2024.3448250

I. Introduction

In TODAY'S world, recommendation systems have found extensive use across various domains. Traditional online recommendation systems often suffer from slow learning rates, necessitated by extensive exploratory phases to decode user preferences. To hasten this learning process and deliver more customized recommendations, the concept of the conversational recommendation system (CRS) has been introduced [1], [2], [3], [4]. A CRS engages users periodically to elicit explicit feedback on specific "key-terms," leveraging this additional conversational data to refine the understanding of user preferences [5], [6]. Fig. 1 shows a CRS within a movie recommendation context, where the learning agent, i.e., the platform, not only suggests movies, but also investigates user inclinations regarding certain themes, such as romance. This conversational dynamic enables the system to craft recommendations that more closely align with individual user tastes.

Despite recent advances in CRS, there is a continuous influx of data from numerous users [7], [8], [9], even involving conversational feedback level. Over time, the actions of these users, such as clicks and ratings, can be maliciously manipulated or disrupted [10], [11], [12], [13], [14]. Such disruptions can skew the learning agent's estimation of user preferences, leading the system to make less-than-ideal recommendations [9], [15], [16], thereby negatively impacting the user experience. Consequently, it's imperative to develop robust online learning strategies that can effectively learn from potentially manipulated user behaviors and identify corrupted users in real time. Some previous works propose bandit algorithms to interactively learn the *unknown* user preferences from corrupted feedback [10], [14], [17], [18].

Nonetheless, these initiatives are beset by two significant shortcomings. First, they are predominantly tailored for robust online preference learning on an individual user basis. In the more complex multi-user environments, these algorithms do not adequately harness the implicit inter-user relationships, which could be exploited to enhance learning efficiency amidst disrupted behaviors. Second, existing literature does not address the online identification of corrupted users within a multi-user framework. Although there are works dedicated to corrupted user detection [19], [20], [21], [22], [23], they primarily operate on the premise of pre-existing user information in an offline context, rendering them ineffective for online detection based on bandit feedback.

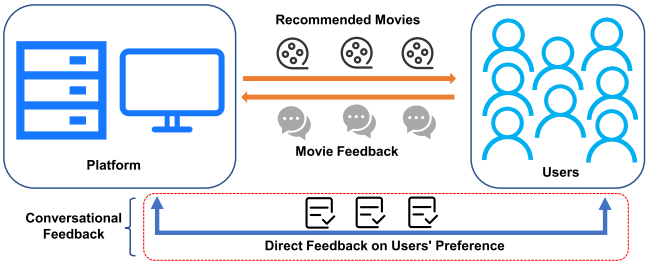


Fig. 1. A CRS illustration where users can offer feedback on two levels. The conversational feedback level is highlighted by the dashed red box.

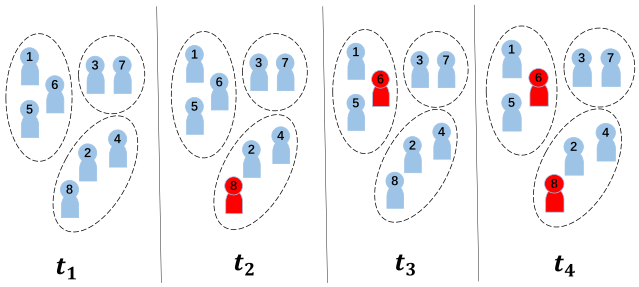


Fig. 2. Illustration of OLDCU. The *unknown* user relationships are depicted by dotted circles. For instance, users 3 and 7 share similar tastes, which could place them in the same user segment (i.e., cluster). Users 6 and 8 are corrupted users due to their fluctuating behaviors over time. For example, user 8's behaviors are normal at times t_1 and t_3 (shown in blue), but become adversarially corrupted at times t_2 and t_4 (shown in red) [10], [12], making it challenging to detect them online. The agent's task is to understand these user relationships to leverage information from similar users to enhance recommendation quality, and online identify corrupted users 6 and 8.

To overcome these limitations, as shown in Fig. 2, we propose a novel bandit problem titled "Online Learning and Detecting Corrupted Users from Bandit Feedback with Adversarial Corruption" (OLDCU), tailored for CRSs. To model and utilize the relationships among users, we posit an unknown clustering structure over users, where users with similar preferences are grouped into the same cluster [24], [25], [26]. This allows the agent to infer the clustering structure and leverage the information of similar users for better recommendations. Some users, known as corrupted users, occasionally exhibit corrupted behaviors to deceive the agent [10], [11], [12], [14], while most of the time they mimic the behaviors of normal users to avoid detection. The agent's task is not only to robustly learn the unknown user preferences and relationships from potentially disrupted feedback and balance the exploration-exploitation trade-off to maximize cumulative reward, but also to online detect corrupted users.

The OLDCU problem presents significant challenges in the realm of CRSs. First, corrupted behaviors can lead to inaccurate estimations of user preferences, which in turn can result in incorrect inferences about user relationships and sub-optimal recommendations. Second, detecting corrupted users from bandit feedback is a complex task, given the dynamic nature of their behaviors (sometimes regular while sometimes corrupted). This contrasts with offline settings, where static embeddings can capture corrupted users' information, and existing methods can perform binary classifications offline, which are not designed to adapt over time [27], [28]. To address these issues, we introduce an online learning framework mainly comprising two innovative algorithms:

1) *RCLUB-WCU:* To robustly estimate user preferences, learn the unknown relations from potentially corrupted behaviors, and perform high-quality recommendations, we propose a novel bandit algorithm "*Robust Clustering of Bandits with Corrupted Users*" (RCLUB-WCU), which maintains a dynamic graph over users to represent the learned clustering structure, where users linked by edges are inferred to be in the same cluster. RCLUB-WCU adaptively removes edges and recommends arms based on the aggregated interactive information within clusters. Key designs of RCLUB-WCU include: (i) Weighted ridge regression for robust user preference estimation, using the inverse of the confidence radius as weights to lessen the impact of potentially corrupted samples. (ii) A conversational query mechanism that utilizes interactive histories to adaptively select explorative key-terms, efficiently integrating information from both recommendations and conversations. (iii) A robust edge deletion rule that accounts for the potential impact of corruptions when determining cluster boundaries, ensuring that users within the same connected component are likely to belong to the same true cluster.
2) *OCCUD:* To detect corrupted users from bandit feedback, we leverage the learned clustering structure of RCLUB-WCU and develop a novel algorithm called "Online Cluster-based Corrupted User Detection" (OCCUD). By comparing each user's non-robust preference vector with the robust cluster estimate, OCCUD flags users as corrupted when the discrepancy exceeds a certain threshold. The underlying intuitions are as follows: Corrupted users, due to their misleading behaviors, would have non-robust preference estimations that deviate significantly from the ground truths. Conversely, with the accurate clustering provided by RCLUB-WCU, the robust preference estimations of users' inferred clusters should closely align with the ground truths. Therefore, for corrupted users, their non-robust estimates should significantly differ from the robust estimates of their inferred clusters.

In summary, this paper makes the following contributions.

- Considering challenges posed by the presence of adversarial corruption in user feedback within an online conversational recommendation system, we introduce the OLDCU problem, focusing on how to online detect corrupted users and maximize the cumulative reward.
- We propose the RCLUB-WCU and OCCUD algorithms within our novel online learning framework to address the OLDCU problem. RCLUB-WCU minimizes regret by leveraging social relations, while OCCUD detects corrupted users online based on inferred user relations.
- We prove the regret upper bound for RCLUB-WCU even with corrupted conversational bandit feedback, which

perfectly matches existing state-of-the-art results in several degenerate settings. We also give a theoretical performance guarantee for the detection algorithm OCCUD.

- Through experiments on both synthetic and real-world datasets, we demonstrate that our proposed algorithms outperform existing bandit algorithms and achieve high accuracy in detecting corrupted users.

II. Problem Formulation

This section formulates the problem of "*Online Learning and Detecting Corrupted Users for Conversational Recommendation System*" (OLDCU) (illustrated in Fig. 2).

A. Online Learning and Detecting Corrupted Users

In the context of conversational recommendation systems (CRS), here are u users to be served, which we denote by the set $\mathcal{U} = \{1, 2, \ldots, u\}$. A subset of users, denoted $\tilde{\mathcal{U}} \subseteq \mathcal{U}$, may exhibit corrupted behaviors. These users attempt to blend in with normal users to avoid detection, while occasionally engaging in actions that lead the system to make poor recommendations. Each user $i \in \mathcal{U}$, no matter a normal one or corrupted one, has an associated preference feature vector $\boldsymbol{\theta}_i \in \mathbb{R}^d$, which is *unknown* to the system and bounded such that $\|\boldsymbol{\theta}_i\|_2 \leq 1$. Users are thought to be organized into an *unknown* clustering structure based on preference similarities, which the system must learn through interaction. The set of users $\mathcal{U}$ is divided into m clusters, $V_1, V_2, \ldots, V_m$, each cluster containing users with identical preference vectors and users from different clusters having distinct preference vectors. Specifically, the set of users $\mathcal{U}$ can be partitioned into m ($m \ll u$) clusters, $V_1, V_2, \ldots V_m$, where $\cup_{j\in[m]} V_j = \mathcal{U}$, and $V_j \cap V_{j'} = \emptyset$, for $j \neq j'$. Users in the same cluster have the same preference feature vector, while users in different clusters have different preference vectors. We use $\boldsymbol{\theta}^j$ to denote the common preference vector shared by users in the j-th cluster V_j, and use $j(i)$ to denote the index of cluster user i belongs to (i.e., $i \in V_{j(i)}$). Then we have: for any two users $k, i \in \mathcal{U}$, if $k \in V_{j(i)}$, then $\boldsymbol{\theta}_k = \boldsymbol{\theta}^{j(i)} = \boldsymbol{\theta}_i$; otherwise $\boldsymbol{\theta}_k \neq \boldsymbol{\theta}_i$. We assume that the arm set $\mathcal{A} \subseteq \mathbb{R}^d$ is finite. Each arm $a \in \mathcal{A}$ is associated with a feature vector $\boldsymbol{x}_a \in \mathbb{R}^d$ with $\|\boldsymbol{x}_a\|_2 \leq 1$. We denote $\|\boldsymbol{x}\|_{\boldsymbol{M}} = \sqrt{\boldsymbol{x}^\top \boldsymbol{M} \boldsymbol{x}}$, $[m] = \{1, \ldots, m\}$, the number of elements in set $\mathcal{A}$ as $|\mathcal{A}|$.

The agent's learning process operates as follows. In each round $t \in [T]$, a user $i_t \in \mathcal{U}$ arrives and the learning agent must choose from a subset of arms, $\mathcal{A}_t \subseteq \mathcal{A}$. The agent determines the user's cluster V_t from past interactions and selects an arm $a_t \in \mathcal{A}_t$ using information aggregated from V_t. When the user receives a_t, they provide a reward, such as click-through rate (CTR), expected to be $\boldsymbol{x}_{a_t}^\top \boldsymbol{\theta}_{i_t}$. We utilize a linear model for online recommendations due to its computational efficiency, with options to include non-linear relationships by incorporating deep learning architectures like DNNs or Transformers. This is facilitated by a feature mapping function $\varphi : \mathcal{A} \to \mathbb{R}^d$, representing each arm as a feature vector $\boldsymbol{x}_{a_t} = \varphi(a_t)$, which effectively captures complex user-item interactions [6], [29]. However, corrupted users can manipulate the feedback. Adopting the approach from previous works [10], [12], we model corrupted users as occasionally manipulating rewards to misdirect the agent towards sub-optimal arms. Specifically, at each round t, if the current served user is a corrupted user (i.e., $i_t \in \tilde{\mathcal{U}}$), the user can corrupt the reward by c_t. To summarize, the reward that the agent receives at round t is modeled as:

$$r_t = \boldsymbol{x}_{a_t}^\top \boldsymbol{\theta}_{i_t} + \eta_t + c_t, \tag{1}$$

where $c_t = 0$ if i_t is a normal user, (i.e., $i_t \notin \tilde{\mathcal{U}}$), and η_t is 1-sub-Gaussian random noise. As the number of corrupted users is usually small (i.e., $|\tilde{\mathcal{U}}| \ll u$), and they only corrupt the rewards occasionally with small magnitudes to make themselves hard to be detected, we assume the sum of corruption magnitudes in all rounds is upper bounded by the *corruption level* C, i.e., $\sum_{t=1}^{T} |c_t| \leq C$ [10], [14], [18].

The learning agent's objectives are twofold. One objective of the learning agent is to minimize the expected cumulative regret, which quantifies the difference between the expected cumulative rewards gained from the optimal policy and the algorithm used by the agent:

$$R(T) = \mathbb{E}\left[\sum_{t=1}^{T} (\boldsymbol{x}_{a_t^*}^\top \boldsymbol{\theta}_{i_t} - \boldsymbol{x}_{a_t}^\top \boldsymbol{\theta}_{i_t})\right], \tag{2}$$

where $a_t^* \in \arg\max_{a\in\mathcal{A}_t} \boldsymbol{x}_a^\top \boldsymbol{\theta}_{i_t}$ denotes an optimal arm with the highest expected reward at round t. The second objective is to identify corrupted users online using bandit feedback. Specifically, at each round t, the agent identifies a set of users, denoted as $\tilde{\mathcal{U}}_t$, suspected to be corrupted. The goal is to make $\tilde{\mathcal{U}}_t$ as close as possible to the actual set of corrupted users $\tilde{\mathcal{U}}$.

B. Conversational Contextual Bandit Feedback

Compared to traditional recommendation systems, in the context of CRSs, the system not only provides recommendations but also possesses the capability to intermittently solicit direct feedback from users regarding specific "key-terms" to gain a better understanding of user preferences. A "key-term" is a keyword or topic that is associated with a subset of arms. For instance, the key-term "movie" could be related to arms such as comedy, horror, action, etc. Let us denote a finite set of such key-terms as $\mathcal{K}$, with a total count of K key-terms. The association between arms and key-terms is depicted through a weighted bipartite graph $(\mathcal{A}, \mathcal{K}, \boldsymbol{W})$, where $\boldsymbol{W}$, defined as $\boldsymbol{W} \triangleq [w_{a,k}], a \in \mathcal{A}, k \in \mathcal{K}$, represents the weight matrix that signifies the strength of the relationship between each arm $a \in \mathcal{A}$ and key-term $k \in \mathcal{K}$. A weight $w_{a,k} \geq 0$ indicates the association level, with the assumption that every key-term k is positively connected to at least some arms (i.e., $\sum_{a\in\mathcal{A}} w_{a,k} > 0$ for all $k \in \mathcal{K}$), and the sum of weights for each arm is normalized to 1 (i.e., $\sum_{k\in\mathcal{K}} w_{a,k} = 1$ for each $a \in \mathcal{A}$). The feature vector of a key-term k is constructed as $\tilde{\boldsymbol{x}}_k = \sum_{a\in\mathcal{A}} \frac{w_{a,k}}{\sum_{a'\in\mathcal{A}} w_{a',k}} \boldsymbol{x}_a$. The feedback mechanism for a key-term k at time t from a user

$i_t \in \mathcal{U}$, who may be either normal or corrupted, is mathematically expressed as:

$$\tilde{r}_{k,t} = \tilde{\boldsymbol{x}}_k^\top \boldsymbol{\theta}_{i_t} + \tilde{\eta}_t + \tilde{c}_t, \tag{3}$$

where $\tilde{\eta}_t$ is assumed to be 1-sub-Gaussian random noise, and $\tilde{c}_t$ is corrupted conversational reward, with $\tilde{c}_t = 0$ if $i_t \notin \tilde{\mathcal{U}}$. Here, we also assume the sum of corruption magnitudes in all rounds is upper bounded by the *corruption level*. Building on the insights from prior works [5], [6], [30], [31], the unknown user preference vector $\boldsymbol{\theta}_{i_t}$ is essentially assumed to be the same at both the arm level and the key-term level. However, a notable departure from earlier conversational bandit frameworks is that our model acknowledges the possibility of corruption even at the key-term feedback level.

To maintain a positive user experience, it is crucial to regulate the frequency of the system's conversational engagements. To this end, we introduce a conversation frequency function $b_{i_t}(t)$ tailored for the user currently being served, i_t. This function governs the number of conversational prompts initiated by the system. At each round t, the system may engage in $q(t) = \lfloor b_{i_t}(t) - b_{i_t}(t-1) \rfloor$ conversations with the user i_t, provided that the condition $b_{i_t}(t) - b_{i_t}(t-1) > 0$ is satisfied. For example, if $b_{i_t}(t) = k\lfloor \frac{t}{m} \rfloor$ with $m \geq 1$ and $k \geq 1$, the system will initiate k conversations at every m-round interval. Under this model, the system will engage in $b_{i_t}(t)$ conversational interactions with the user i_t.

III. Algorithm Design

In this section, we present our algorithms designed to address the OLDUC problem. RCLUB-WCU (see in Algorithm 1) is a bandit algorithm that effectively learns the unknown user clustering structure and preferences, even in the presence of potentially corrupted user behaviors. Expect utilizing the cluster-based information, it also conducts occasional conversations with users via key-term selection, to enhance recommendation quality (see in Algorithm 2). Building on the clustering structure discerned by RCLUB-WCU, the OCCUD algorithm (see in Algorithm 3) is capable of accurately identifying corrupted users based on bandit feedback.

A. RCLUB-WCU

The corrupted user behaviors may cause inaccurate user preference estimations, leading to erroneous relation inference and sub-optimal recommendations. In this case, how to learn and utilize the unknown user relations to make good recommendations becomes non-trivial. Motivated by this, we design RCLUB-WCU as follows.

Assign the inferred cluster V_t for user i_t: RCLUB-WCU maintains a dynamic undirected graph $G_t = (\mathcal{U}, E_t)$ over users, which is initialized to be a complete graph (Algorithm 1 Line 2). Users with similar learned preferences will be connected with edges in E_t. The connected components in the graph represent the inferred clusters by the algorithm. At round t, user i_t comes to be served with a feasible arm set $\mathcal{A}_t$ for the agent to choose from (Line 4). In Line 6, RCLUB-WCU detects the connected component V_t in the graph containing user i_t to be the current inferred cluster for i_t.

Algorithm 1: RCLUB-WCU.

1: **Input:** Regularization parameter λ, confidence radius parameter β, threshold parameter α, edge deletion parameter α_1, $f(T) = \sqrt{(1 + \ln(1 + b_i(T) + T))/(1 + b_i(T) + T)}$.
2: **Initialization:**
- $\boldsymbol{M}_{i,0} = \mathbf{0}_{d\times d}, \boldsymbol{b}_{i,0} = \mathbf{0}_{d\times 1}$, $\tilde{\boldsymbol{M}}_{i,0} = \mathbf{0}_{d\times d}, \tilde{\boldsymbol{b}}_{i,0} = \mathbf{0}_{d\times 1}, T_{i,0} = 0, \forall i \in \mathcal{U}$;
- A complete graph $G_0 = (\mathcal{U}, E_0)$ over $\mathcal{U}$.

3: **for all** $t = 1, 2, \ldots, T$ **do**
4: Receive the index of the current served user $i_t \in \mathcal{U}$, get the feasible arm set at this round $\mathcal{A}_t$.
5: Select key-terms to conduct conversations and receive feedback if conversation is allowed (Algorithm 2);
6: Determine the connected components V_t in the current maintained graph $G_{t-1} = (\mathcal{U}, E_{t-1})$ such that $i_t \in V_t$.
7: Calculate robustly estimated statistics for the cluster V_t:

$$\boldsymbol{M}_{V_t,t-1} = \lambda \boldsymbol{I} + \sum_{i\in V_t} \boldsymbol{M}_{i,t-1}\,,$$

$$\boldsymbol{b}_{V_t,t-1} = \sum_{i\in V_t} \boldsymbol{b}_{i,t-1}\,, \hat{\boldsymbol{\theta}}_{V_t,t-1} = \boldsymbol{M}_{V_t,t-1}^{-1} \boldsymbol{b}_{V_t,t-1}\,;$$

8: Recommend an arm a_t with the largest UCB index as in (6) and receive the corresponding reward r_t;
9: Update the statistics for robust estimation of user i_t:

$$\boldsymbol{M}_{i_t,t} = \boldsymbol{M}_{i_t,t-1} + w_{i_t,t-1}\boldsymbol{x}_{a_t}\boldsymbol{x}_{a_t}^\top\,,$$

$$\boldsymbol{b}_{i_t,t} = \boldsymbol{b}_{i_t,t-1} + w_{i_t,t-1} r_t \boldsymbol{x}_{a_t}\,, T_{i_t,t} = T_{i_t,t-1} + 1\,,$$

$$\hat{\boldsymbol{\theta}}_{i_t,t} = (\lambda \boldsymbol{I} + \boldsymbol{M}_{i_t,t})^{-1} \boldsymbol{b}_{i_t,t}\,,$$

$$w_{i_t,t} = \min\left\{1, \frac{\alpha}{\|\boldsymbol{x}_{a_t}\|_{\boldsymbol{M}_{i_t,t}^{-1}}}\right\};$$

10: Keep robust estimation of other users unchanged:

$$\boldsymbol{M}_{\ell,t} = \boldsymbol{M}_{\ell,t-1}, \boldsymbol{b}_{\ell,t} = \boldsymbol{b}_{\ell,t-1}, T_{\ell,t} = T_{\ell,t-1},$$

$$\hat{\boldsymbol{\theta}}_{\ell,t} = \hat{\boldsymbol{\theta}}_{\ell,t-1}, \text{ for all } \ell \in \mathcal{U}, \ell \neq i_t;$$

11: Delete the edge $(i_t, \ell) \in E_{t-1}$, if

$$\left\|\hat{\boldsymbol{\theta}}_{i_t,t} - \hat{\boldsymbol{\theta}}_{\ell,t}\right\|_2 \geq \alpha_1 \left(f(T_{i_t,t}) + f(T_{\ell,t}) + (1+K)\alpha C\right),$$

and get an updated graph $G_t = (\mathcal{U}, E_t)$;
12: Detect corrupted users by using OCCUD (Algorithm 3).
13: **end for**

Robust preference estimation of cluster V_t: After determining the cluster V_t for user i_t, RCLUB-WCU estimates the common preferences for users in cluster V_t using the historical feedback of all users within V_t and recommends an arm to i_t accordingly. The corrupted behaviors could cause inaccurate

preference estimates, which can mislead the agent to make sub-optimal recommendations. To address this issue, inspired by [12], [32], we use the weighted ridge regression to make estimations robust to corruptions. Specifically, RCLUB-WCU robustly estimates the common preference vector of cluster V_t by solving the following weighted ridge regression

$$\hat{\boldsymbol{\theta}}_{V_t,t-1} = \underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\arg\min} \sum_{\substack{s\in[t-1]\\ i_s\in V_t}} w_{i_s,s}(r_s - \boldsymbol{x}_{a_s}^\top \boldsymbol{\theta})^2 + \lambda \|\boldsymbol{\theta}\|_2^2 , \quad (4)$$

where $\lambda > 0$ is a regularization coefficient. This optimization problem has a closed-form solution $\hat{\boldsymbol{\theta}}_{V_t,t-1} = \boldsymbol{M}_{V_t,t-1}^{-1}\boldsymbol{b}_{V_t,t-1}$, where $\boldsymbol{M}_{V_t,t-1} = \lambda \boldsymbol{I} + \sum_{\substack{s\in[t-1]\\ i_s\in V_t}} w_{i_s,s}\boldsymbol{x}_{a_s}\boldsymbol{x}_{a_s}^\top$, $\boldsymbol{b}_{V_t,t-1} = \sum_{\substack{s\in[t-1]\\ i_s\in V_t}} w_{i_s,s} r_{a_s}\boldsymbol{x}_{a_s}$.

In the above equations, we set the weight for user i_s in V_t at round s as $w_{i_s,s} = \min\{1, \alpha/\|\boldsymbol{x}_{a_s}\|_{M_{i_s,s-1}^{-1}}\}$, where α is the threshold coefficient to be determined later. The intuitions of designing these weights are as follows. The term $\|\boldsymbol{x}_{a_s}\|_{M_{i_s,s-1}^{-1}}$ is the confidence radius of the arm a_s for user i_s at round s, representing the confidence that the algorithm has about the estimation of the user i_s's preference in arm a_s in s. Specifically, if $\|\boldsymbol{x}_{a_s}\|_{M_{i_s,s-1}^{-1}}$ is large, it means that the learning agent is uncertain of user i_s's preference on a_s, intuitively indicating that this sample is more likely to be a corrupted one. Therefore, at round s, we use the inverse of confidence radius $\alpha/\|\boldsymbol{x}_{a_s}\|_{M_{i_s,s-1}^{-1}}$ to assign a small weight to the sample at this round if it is potentially corrupted. By doing this, uncertain interactive information for each user in cluster V_t is assigned with less importance when estimating the preference vector for V_t, which could help relieve the estimation inaccuracy caused by those uncertain samples that might be corrupted. For technical details, please refer to the theoretical analysis in Section IV-A and the proofs in the Appendix, available online.

Conversational query with key-terms: Interactions at the key-term level are shown in Algorithm 2. At round t, the agent initially determines the feasibility of conversations using $b_{i_t}(t)$. If conversations are permitted, the agent requests the user's feedback on $q(t)$ key-terms and employs this feedback to update the system parameters. A key-term with an extensive confidence radius indicates that the recommendation system has not fully delved into the user's preferences related to its corresponding items. This implies that such a key-term is ideal for further exploration. With this insight, we tactically choose key-terms with the most extensive confidence radius to enable adaptive, exploratory conversations. Specifically, when a conversation is allowed at a given round t, a key-term is selected according to the following equation:

$$k \in \underset{k\in\mathcal{K}_t}{\arg\max}\, \beta_t \|\tilde{\boldsymbol{x}}_k\|_{\boldsymbol{M}_{i_t,t}^{-1}} , \quad (5)$$

where $\mathcal{K}_t \subseteq \mathcal{K}$ represents the potentially time-dependent set of key-terms available at t and $\beta_t = \sqrt{\lambda} + \sqrt{2\log(T) + d\log(1 + \frac{b_{i_t}(t)+t}{\lambda d}) + (1+K)\alpha C}$ is the confidence radius parameter. By opting for key-terms with the largest confidence radius, Algorithm 2 can adaptively solicit user feedback on key-terms whose associated areas have been least explored to date. This approach effectively utilizes interactive information to guide the selection of key-terms, thereby enhancing the exploration of user preferences. We denote the total selected key-terms as $\mathcal{K}_t' \subseteq \mathcal{K}_t$ at round t.

Algorithm 2: Conversational Query With Key-Terms (At Round t, Used in Line 5 in Algorithm 1).

1: **Input:** Graph $(\mathcal{A}, \mathcal{K}, \boldsymbol{W})$, conversation frequency function $b_{i_t}(t)$.
2: **if** $b_{i_t}(t) - b_{i_t}(t-1) > 0$ **then**
3: $q(t) = \lfloor b_{i_t}(t) - b_{i_t}(t-1) \rfloor$;
4: **while** $q(t) > 0$ **do**
5: Select a key-term $k \in \mathcal{K}_t$ according to (5) and query the user's preference over it;
6: Receive the user's feedback $\tilde{r}_{k,t}$;
7: $\boldsymbol{M}_{i_t,t} = \boldsymbol{M}_{i_t,t-1} + \tilde{w}_{i_t,t-1}\tilde{\boldsymbol{x}}_k\tilde{\boldsymbol{x}}_k^\top$, $\boldsymbol{b}_{i_t,t} = \boldsymbol{b}_{i_t,t-1} + \tilde{w}_{i_t,t-1}\tilde{\boldsymbol{x}}_k\tilde{r}_{k,t}$, $\tilde{w}_{i_t,t} = \min\left\{1, \frac{\alpha}{\|\tilde{\boldsymbol{x}}_k\|_{\boldsymbol{M}_{i_t,t}^{-1}}}\right\}$;
8: $q(t) -= 1$;
9: **end while**
10: **else**
11: $\boldsymbol{M}_{i_t,t} = \boldsymbol{M}_{i_t,t-1}$, $\boldsymbol{b}_{i_t,t} = \boldsymbol{b}_{i_t,t-1}$;
12: **end if**

Recommend a_t with estimated preference of cluster V_t: Based on the corruption-robust preference estimation $\hat{\boldsymbol{\theta}}_{V_t,t-1}$ of cluster V_t, in Line 8, the agent recommends an arm using the upper confidence bound (UCB) strategy to balance exploration and exploitation

$$a_t = \underset{a\in\mathcal{A}_t}{\operatorname{argmax}} \underbrace{\boldsymbol{x}_a^\top \hat{\boldsymbol{\theta}}_{V_t,t-1}}_{\hat{R}_{a,t}} + \underbrace{\beta_t \|\boldsymbol{x}_a\|_{\boldsymbol{M}_{V_t,t-1}^{-1}}}_{C_{a,t}} , \quad (6)$$

where $\hat{R}_{a,t}$ denotes the estimated reward of arm a at t and $C_{a,t}$ denotes the confidence radius of arm a at t, respectively. The design of $C_{a,t}$ theoretically relies on Lemma 2 that will be given in Section IV.

Update the robust estimation of user i_t: After receiving the reward r_t, the algorithm updates the estimation statistics of user i_t, while keeping the statistics of other users unchanged (Line 9 and Line 10). Specifically, RCLUB-WCU estimates the preference vector of user i_t by solving the following weighted ridge regression

$$\hat{\boldsymbol{\theta}}_{i_t,t} = \underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\arg\min} \sum_{\substack{s\in[t]\\ i_s=i_t}} w_{i_s,s}(r_s - \boldsymbol{x}_{a_s}^\top \boldsymbol{\theta})^2 + \lambda \|\boldsymbol{\theta}\|_2^2 . \quad (7)$$

This has a closed-form solution $\hat{\boldsymbol{\theta}}_{i_t,t} = (\lambda \boldsymbol{I} + \boldsymbol{M}_{i_t,t})^{-1}\boldsymbol{b}_{i_t,t}$, with the weights designed in the same way as before by the same reasoning, where $\boldsymbol{M}_{i_t,t} = \sum_{\substack{s\in[t]\\ i_s=i_t}} w_{i_s,s}\boldsymbol{x}_{a_s}\boldsymbol{x}_{a_s}^\top$, $\boldsymbol{b}_{i_t,t} = \sum_{\substack{s\in[t]\\ i_s=i_t}} w_{i_s,s} r_{a_s}\boldsymbol{x}_{a_s}$.

Update the dynamic graph: Finally, with the updated preference estimation of user i_t, RCLUB-WCU checks whether the current inferred user i_t's preference similarities with other users are still true, and updates the maintained graph accordingly.

Algorithm 3: OCCUD (At Round t, Used in Line 12 in Algorithm 1).

1: Initialize $\tilde{\mathcal{U}}_t = \emptyset$; Input probability parameter δ.
2: Update the statistics for non-robust estimation of user i_t:

$$\tilde{\boldsymbol{M}}_{i_t,t} = \tilde{\boldsymbol{M}}_{i_t,t-1} + \boldsymbol{x}_{a_t}\boldsymbol{x}_{a_t}^\top + \sum_{k\in\mathcal{K}'_t} \tilde{\boldsymbol{x}}_k\tilde{\boldsymbol{x}}_k^\top\,,$$

$$\tilde{\boldsymbol{b}}_{i_t,t} = \tilde{\boldsymbol{b}}_{i_t,t-1} + r_t\boldsymbol{x}_{a_t} + \sum_{k\in\mathcal{K}'_t} \tilde{\boldsymbol{x}}_k\tilde{r}_{k,t}\,,$$

$$\tilde{\boldsymbol{\theta}}_{i_t,t} = (\lambda\boldsymbol{I} + \tilde{\boldsymbol{M}}_{i_t,t})^{-1}\tilde{\boldsymbol{b}}_{i_t,t}\,.$$

3: Keep non-robust estimation of other users unchanged:

$$\tilde{\boldsymbol{M}}_{\ell,t} = \tilde{\boldsymbol{M}}_{\ell,t-1}, \tilde{\boldsymbol{b}}_{\ell,t} = \tilde{\boldsymbol{b}}_{\ell,t-1},$$

$$\tilde{\boldsymbol{\theta}}_{\ell,t} = \tilde{\boldsymbol{\theta}}_{\ell,t-1}, \text{ for all } \ell\in\mathcal{U}, \ell\neq i_t.$$

4: **for all** connected component $V_{j,t}\in G_t$ **do**
5: Calculate robust estimation statistics for the cluster $V_{j,t}$:

$$\boldsymbol{M}_{V_{j,t},t} = \lambda\boldsymbol{I} + \sum_{\ell\in V_{j,t}} \boldsymbol{M}_{\ell,t}\,, T_{V_{j,t},t} = \sum_{\ell\in V_{j,t}} T_{\ell,t}\,,$$

$$\boldsymbol{b}_{V_{j,t},t} = \sum_{\ell\in V_{j,t}} \boldsymbol{b}_{\ell,t}\,, \hat{\boldsymbol{\theta}}_{V_{j,t},t} = \boldsymbol{M}_{V_{j,t},t}^{-1}\boldsymbol{b}_{V_{j,t},t}\,;$$

6: **for all** user $i\in V_{j,t}$ **do**
7: Detect user i to be a corrupted user and add user i to the set $\tilde{\mathcal{U}}_t$ if the following holds:

$$\left\|\tilde{\boldsymbol{\theta}}_{i,t} - \hat{\boldsymbol{\theta}}_{V_{i,t},t-1}\right\|_2 > \frac{g(T_{i,t})}{\sqrt{\lambda_{\min}(\tilde{\boldsymbol{M}}_{i,t})+\lambda}} + \frac{g(T_{V_{i,t},t})+(1+K)\alpha C}{\sqrt{\lambda_{\min}(\boldsymbol{M}_{V_{i,t},t})}}\,. \tag{8}$$

8: **end for**
9: **end for**

Precisely, if the l_2-norm of the difference between the updated estimated preference vector $\hat{\boldsymbol{\theta}}_{i_t,t}$ of user i_t and the estimation $\hat{\boldsymbol{\theta}}_{\ell,t}$ of user ℓ is larger than a threshold defined in Line 11, RCLUB-WCU will delete the edge (i_t,ℓ) in G_{t-1} to separate them apart. This threshold is carefully designed, considering the estimation uncertainty caused by both stochastic noises and potentially corrupted behaviors. The updated graph $G_t=(\mathcal{U},E_t)$ will be used in the next round.

B. OCCUD

Based on the inferred clustering structure of RCLUB-WCU, we propose a novel algorithm, OCCUD, that can detect corrupted users in an online manner. We summarize how OCCUD works at round t in Algorithm 3. The design ideas and process of OCCUD are as follows.

Besides the robust preference estimations (with weighted ridge regression) of users and clusters kept by RCLUB-WCU, OCCUD also maintains the non-robust estimations for each user by regular online ridge regression without weights (Line 2 and Line 3). Specifically, at round t, OCCUD updates the non-robust estimation of user i_t by solving the following online regression without weights:

$$\tilde{\boldsymbol{\theta}}_{i_t,t} = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \sum_{\substack{s\in[t]\\ i_s=i_t}} (r_s - \boldsymbol{x}_{a_s}^\top\boldsymbol{\theta})^2 + \lambda\left\|\boldsymbol{\theta}\right\|_2^2\,, \tag{9}$$

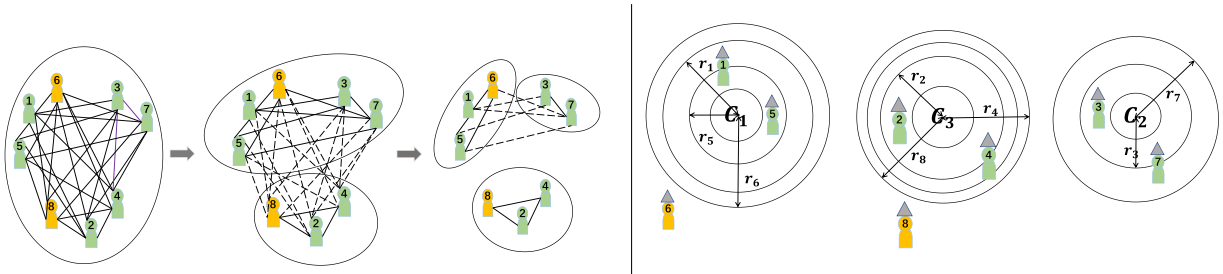
which has a closed-form solution $\tilde{\boldsymbol{\theta}}_{i_t,t}=(\lambda\boldsymbol{I}+\tilde{\boldsymbol{M}}_{i_t,t})^{-1}\tilde{\boldsymbol{b}}_{i_t,t}$, where $\tilde{\boldsymbol{M}}_{i_t,t}=\sum_{\substack{s\in[t]\\ i_s=i_t}}\boldsymbol{x}_{a_s}\boldsymbol{x}_{a_s}^\top$, $\tilde{\boldsymbol{b}}_{i_t,t}=\sum_{\substack{s\in[t]\\ i_s=i_t}} r_{a_s}\boldsymbol{x}_{a_s}$.

With the robust and non-robust preference estimations, OCCUD does the following to detect corrupted users based on the clustering structure inferred by RCLUB-WCU. First, OCCUD finds the connected components in the graph kept by RCLUB-WCU, which represent the inferred clusters for the users. Then, for each inferred cluster $V_{j,t}\in G_t$: (1) OCCUD computes its robustly estimated preferences vector $\hat{\boldsymbol{\theta}}_{V_{i_t},t}$ (Line 5). (2) Denote $g(x)=\sqrt{d\log(1+\frac{b_i(t)+x}{\lambda d})+2\log(T)+\sqrt{\lambda}}$ and $\lambda_{\min}(\cdot)$ as the minimum eigenvalue of the matrix argument. For each user i whose inferred cluster is $V_{j,t}$ (i.e.,$i\in V_{j,t}$), OCCUD calculates the difference between user i's non-robustly estimated preference vector $\tilde{\boldsymbol{\theta}}_{i,t}$ and the robustly estimated preference vector $\hat{\boldsymbol{\theta}}_{V_{i,t},t}$ for user i's inferred cluster $V_{j,t}$. If the difference is larger than a carefully designed threshold, OCCUD will detect user i as a corrupted user and add user i to the detected corrupted user set $\tilde{\mathcal{U}}_t$ (Line 7).

The intuitions of the OCCUD algorithm are as follows. On the one hand, after some interactions, the RCLUB-WCU algorithm will infer the user clustering structure accurately. Thus, at round t, the robustly-estimated preference vector $\hat{\boldsymbol{\theta}}_{V_{i,t},t}$ for user i's inferred cluster should be pretty close to user i's ground-truth preference vector $\boldsymbol{\theta}_i$. On the other hand, since the behaviors of normal users are always regular, at round t, if user i is a normal user, the non-robustly estimated preference vector $\tilde{\boldsymbol{\theta}}_{i,t}$ should also be close to the ground-truth $\boldsymbol{\theta}_i$. However, the non-robustly estimated preference vector of a corrupted user should be quite far from the ground truth due to the disruptions of the corrupted behaviors. Based on the above reasoning, to detect the corrupted users, for each user, OCCUD compares the user's non-robustly-estimated preference vector and the robustly-estimated preference vector of the user's inferred cluster. If the difference exceeds a carefully designed threshold, then with a high probability, this user is a corrupted user. For theoretical support and technical details about this threshold, please refer to the discussions and proof in Section IV-B and Appendix, available online. Simple illustrations of our proposed algorithms can be found in Fig. 3. Note that while our algorithms are initially developed within the context of the CRS framework, their applicability extends beyond this scope. The broader relevance and adaptability of our algorithms are elaborated in Section V-B.

IV. Theoretical Analysis

In this section, we theoretically analyze the performances of our proposed algorithms, RCLUB-WCU and OCCUD. For ease



(a) RCLUB-WCU (b) OCCUD

Fig. 3. Algorithm illustrations. Users from 1 to 8 correspond to the 8 users in Fig. 2, where users 6 and 8 can be corrupted at some time steps (orange), while the other users are never corrupted (green). (a) illustrates RCLUB-WCU, which starts with a fully connected user graph, and adaptively deletes edges between users (dashed lines) with dissimilar robustly learned preferences. The corrupted behaviors of users 6 and 8 may cause inaccurate user preference estimations, leading to erroneous relation inference. In this case, how to delete edges correctly becomes non-trivial, and our algorithm addresses this challenge (detailed in Section III-A). (b) illustrates OCCUD. We use person icons with triangle hats to represent the non-robust user preference estimations. In this illustration, the gap between the non-robust estimation of user 6 and the robust estimation of user 6's inferred cluster (circle C_1) exceeds the threshold r_6 (from Line 7 in Algorithm 3), then OCCUD detects user 6 to be corrupted.

of exposition, we put the proofs in the Appendix, available online. We first make the following assumptions about the clusters, users, and items, which are consistent with the settings from previous works on clustering of bandits [24], [25], [33].

Assumption 1 (Gap Between Different Clusters): Impractical to assume that only cameras with identical feature vectors form a group for configuration sharing, the gap between any two preference vectors for different clusters is at least γ

$$\left\|\boldsymbol{\theta}^{j}-\boldsymbol{\theta}^{j'}\right\|_2 \geq \gamma > 0\,, \forall j, j' \in [m]\,, j \neq j'\,,$$

where γ is an *unknown* positive constant.

Assumption 2 (Uniform Arrival of Users): At each round t, a user i_t comes uniformly at random from $\mathcal{U}$ with probability $1/u$, independent of the past rounds.

Assumption 3 (Item Regularity): At each round t, the feature vector $\boldsymbol{x}_a$ of each arm $a \in \mathcal{A}_t$ is drawn independently from a fixed *unknown* distribution ρ over $\{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 \leq 1\}$, where $\mathbb{E}_{\boldsymbol{x}\sim\rho}[\boldsymbol{x}\boldsymbol{x}^\top]$'s minimal eigenvalue $\lambda_x > 0$. At $\forall t$, for any fixed unit vector $\boldsymbol{z} \in \mathbb{R}^d$, $(\boldsymbol{\theta}^\top \boldsymbol{z})^2$ has sub-Gaussian tail with variance no greater than σ^2.

A. Regret Analysis of RCLUB-WCU

In this section, we present a theoretical performance guarantee for RCLUB-WCU by establishing an upper bound on its expected regret, as defined in (2). Regret is defined as the discrepancy between the cumulative rewards accrued by the agent and those obtained by an oracle strategy. By standard practice, the conversation frequency is bounded by $b_i(t) \leq t, I \in \mathcal{U}$. For the purposes of our analysis, we will therefore assume a linear relationship of the form $b_i(t) = b_i \cdot t$, where b_i is a constant that lies within the open interval (0, 1).

First, we prove the following lemma which gives a sufficient time, after which RCLUB-WCU can cluster all the users correctly with high probability.

Lemma 1: With the robust preference estimations, RCLUB-WCU will gather enough information for every user after:

$$t \geq O\left(u\left(\frac{Cd}{\gamma^2\tilde{\lambda}_x}+\frac{1}{\tilde{\lambda}_x^2}\right)\ln(T)\right)\,, \tag{10}$$

where $\tilde{\lambda}_x \triangleq \int_0^{\lambda_x}(1-e^{-\frac{(\lambda_x-x)^2}{2\sigma^2}})^c dx$, $|\mathcal{A}_t| \leq c, \forall t \in [T]$.

After correct clustering, the following lemma gives a high-probability upper bound of the gap between $\hat{\boldsymbol{\theta}}_{V_t,t-1}$ and the ground-truth $\boldsymbol{\theta}_{i_t}$ in the direction of the action vector $\boldsymbol{x}_a$ for RCLUB-WCU, supporting the design of the confidence radius $C_{a,t}$ in (6).

Lemma 2: With probability at least $1-5\delta$ for some $\delta \in (0, \frac{1}{5})$ after correctly clusering, for each user i_t, we have:

$$\left|\boldsymbol{x}_a^{\mathrm{T}}(\hat{\boldsymbol{\theta}}_{V_t,t-1}-\boldsymbol{\theta}_{i_t})\right| \leq \beta\left\|\boldsymbol{x_a}\right\|_{\boldsymbol{M}_{V_t,t-1}^{-1}} \triangleq C_{a,t}\,,$$

where $\beta = \sqrt{\lambda}+\sqrt{2\log(\frac{1}{\delta})+d\log(1+\frac{b_{i_t}(T)+T}{\lambda d})}+(1+K)\alpha C$.

With Lemmas 1 and 2, we prove the following main theorem about the regret upper bound of RCLUB-WCU, which gives the first sub-linear regret bound for the OLDCU problem.

Theorem 3: (Regret Upper Bound of RCLUB-WCU) With the assumptions in Section II and $\alpha = \frac{\sqrt{d}+\sqrt{\lambda}}{C}$, the expected regret of the RCLUB-WCU algorithm for T rounds satisfies:

$$R(T) \leq O\left(\left(\frac{Cd}{\gamma^2\tilde{\lambda}_x}+\frac{1}{\tilde{\lambda}_x^2}\right)u\log(T)\right)+O\left(d\sqrt{mT}\log(T)\right) + O\left(mCd\log^{1.5}(T)\right)\,. \tag{11}$$

The regret upper bound shown in (11) is composed of three terms. The first term is the sufficient time for correctly clustering (defined in Lemma 1) needed to get enough information for accurate robust preference estimations such that the algorithm could cluster all users correctly afterward with high probability, where the number of users u relies on this term. Note that this term is related to the *corruption level* C, which is inevitable since, intuitively, if there are more corrupted user behaviors, it

will be harder for the algorithm to learn the underlying clustering structure correctly. The last two terms correspond to the regret after correctly clustering with the correct learned clustering structure. Specifically, the second term is caused by the stochastic noises when leveraging the aggregated information within clusters to make recommendations; the third term is the regret caused by the disruption of corrupted user behaviors, which is associated with the *corruption level* C.

When the *corruption level* C is *unknown*, we can use its estimated upper bound $\hat{C} \triangleq \sqrt{T}$ to replace C in the algorithm. In this way, if $C \leq \hat{C}$, the result of regret bound will be replacing C with $\hat{C}$ in (11); in the case when $C > \sqrt{T}$, $R(T) = O(T)$, which as supported by [12], is already optimal for a large class of bandit algorithms.

Discussions about the tightness of our regret bound: Since our work is the first study on the OLDCU problem, we will compare our regret upper bound with several degenerated cases to show the tightness of our result.

- First, in the case when $C = 0$, i.e., all users are normal, our setting degenerates to the classic CB problem [24]. In this case the bound in Theorem 3 becomes $O\big(\frac{1}{\tilde{\lambda}_x^2} u \log(T)\big) + O\big(d\sqrt{mT}\log(T)\big)$, perfectly matching the state-of-the-art results in CB [24], [25], [26].
- Second, in the case when $m = 1$ and $u = 1$, i.e., there is only one user, our setting degenerates to linear bandits with adversarial corruptions [12], [17], and the bound in Theorem 3 becomes $O\big(d\sqrt{T}\log(T)\big) + O\left(Cd\log^{1.5}(T)\right)$, it also perfectly matches the nearly optimal result in [12].

The above comparisons show the tightness of the regret bound of our proposed RCLUB-WCU algorithm, indicating nearly optimal recommendation performance of RCLUB-WCU.

B. Theoretical Performance Guarantee for OCCUD

We theoretically prove the following theorem, which gives a performance guarantee of the corrupted user detection algorithm OCCUD.

Theorem 4: (Theoretical Guarantee for OCCUD) With the assumptions in Section II, the carefully designed threshold in Algorithm 3 Line 7 after correctly clusering, for any detected corrupted user $i \in \tilde{\mathcal{U}}_t$, with probability at least $1 - 5\delta$, this user i is indeed a corrupted user.

This theorem theoretically guarantees that after RCLUB-WCU learns the clustering structure accurately, with high probability, the corrupted users detected by OCCUD are indeed corrupted, showing the high detection accuracy of our proposed OCCUD algorithm.

V. Experiments

We carry out a thorough set of experiments to resolve the ensuing research questions:

- *RQ1:* How effectively can RCLUB-WCU learn and utilize user preferences and relationships to deliver personalized online recommendations with corrupted users?
- *RQ2:* Is OCCUD capable of accurately detecting corrupted users in real-time, given potentially compromised bandit feedback?
- *RQ3:* Does RCLUB-WCU demonstrate greater robustness compared to baseline algorithms when faced with different user preferences, levels of corruption, conversation frequency and arm set sizes?

To thoroughly assess the versatility of our proposed algorithms, we conduct two distinct scenarios: one devoid of conversational feedback to ascertain the algorithm's universal applicability even in traditional recommendation systems, and another enriched with diverse forms of conversational feedback to underscore the algorithm's enhanced performance within the CRS framework.

A. Experiment Setup

1) Datasets: We perform our experiments utilizing both synthetic and real-world datasets. Below is the detailed information regarding the publicly available real-world datasets utilized in our paper:

- *Movielens* [34]: This dataset comprises ratings from 2,113 users across 10,197 movies. The Movielens dataset does not provide explicit labels for fraudulent users; therefore, we have manually identified such users by adopting the methodology in [35].
- *Amazon* [36]: We utilize a subset of the Amazon musical instrument reviews dataset, which includes contributions from 1,429 users pertaining to 900 items. In the Amazon dataset, we categorize users as normal if they have received helpful votes exceeding 80%, and as fraudulent if they have received less than 20%, in line with [21].
- *Yelp* [37]: The Yelp dataset is extensive, featuring 1,987,929 users and 150,346 items, primarily focusing on restaurant reviews. It comes with authentic labels for users, distinguishing between normal and fraudulent activities based on their review patterns.
- *Last.fm* [38]: Originating from the online music service Last.fm, this dataset encompasses 186,479 tag assignments, linking 1,892 users to 17,632 artists. Potential corrupted users are identified based on their tagging patterns and deviations from the norm [35].

The datasets exhibit varying proportions of fraudulent users, specifically 10%, 3.5%, 30.9% and 8.5% for the Movielens, Amazon, Yelp, and Last.fm datasets, respectively. To accommodate the diverse scales of these datasets, we strategically select subsets of the most active users (those who provide the highest number of ratings) and the items that have garnered the most ratings, aligning with [6], [25], [26]. Additionally, corrupt users are considered to provide negative rewards based on the normal rewards from the aforementioned datasets, thus allowing us to assess the performance of our algorithm even in the most extreme scenarios.

2) Evaluation Metrics: To evaluate the performance of the RCLUB-WCU recommendation, we use cumulative regret, a standard metric in bandit scenarios [7], [29], [39]. To assess

the detection of corrupted users by OCCUD, we use the AUC metric [21], [33], which measures the model's ability to differentiate between corrupted and normal users. To measure user experience in the presence of potentially corrupted feedback, we use the sum of all the ratings of random users from the real dataset as a metric [25]. All the experiments are conducted on PCs with Intel(R) Xeon(R) Gold 6240C @ 2.60 GHz, and AMD Ryzen 7 4800H with Radeon graphics @ 2.90 GHz.

3) Evaluation Parameters: In alignment with the publicly available code of the previous works [12], [40], we adopt a unified parameter setting to facilitate a direct and equitable comparison. Specifically, we set the regularization parameter λ to 1, the confidence radius parameter β to 1.5, the threshold parameter α to 0.2, and the edge deletion parameter α_1 to 1. Additionally, the experiment also includes an ablation analysis to examine parameter sensitivity.

B. Evaluation Without Conversational Feedback

In this section, we explore an extreme scenario under CRS where $b_i(t) \equiv 0, \forall i \in \mathcal{U}$. This specific circumstance enables us to focus exclusively on evaluating the performance of RCLUB-WCU and OCCUD under the condition of no conversational feedback, using both synthetic and real-world datasets.

1) Baseline Comparisons: To evaluate the effectiveness of our approach, we benchmark against several existing methods. Traditional offline detection techniques [20], [21], [27], [28] rely on comprehensive user data acquired prior to creating user encodings for classification, which is not feasible for online detection in bandit feedback environments. Therefore, we focus on comparing our algorithm with six online recommendation baselines and one baseline specifically designed for online corrupted user detection within bandit feedback.

- *LinUCB* [41]: Use a single non-robust estimated vector to represent the preference of each user.
- *LinUCB-ind*: Use a separate LinUCB for each user.
- *CW-OFUL* [12]: A state-of-the-art bandit approach with robust estimated vector for all users.
- *CW-OFUL-ind*: Use a separate CW-OFUL for each user.
- *CLUB* [24]: A graph-based clustering for multiple users.
- *SCLUB* [26]: A set-based clustering of bandits approach for multiple users without corruption.

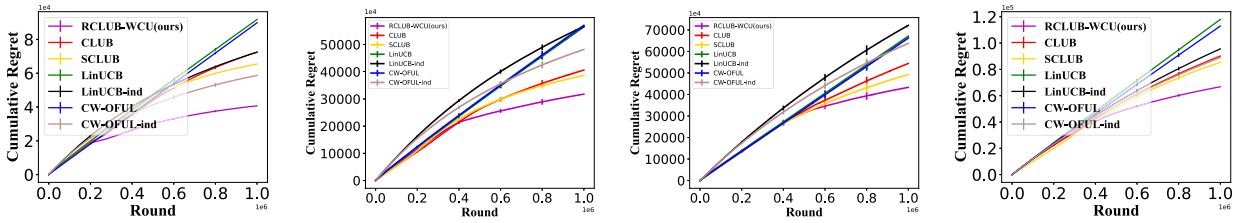
Moreover, our approach to online corrupted user detection is a pioneering study in this area. Consequently, there are no pre-existing baselines for direct comparison. Therefore, we introduce our OCCUD algorithm and compare it with two novel baseline methods. The first method, GCUD, utilizes a graph-based clustering structure. It identifies corrupted users by selecting those with the largest euclidean distance between their current and previous user parameters, represented as $\|\hat{\boldsymbol{\theta}}_{i,t} - \hat{\boldsymbol{\theta}}_{V_{i,t},t-1}\|_2$. This method assumes a known proportion of user corruption, which is unrealistic in many practical situations. The second baseline, NCUD, employs a simpler approach by comparing non-robust estimators of user parameters without incorporating weighted regression. Both of these baseline methods rely on the RCLUB-WCU framework to detect corrupted users. This comparison aims to demonstrate the non-trivial nature of OCCUD's design and its effectiveness in identifying corrupted users without requiring extensive pre-obtained data, unlike some offline methods [19], [20] that are incompatible with our approach.

2) Dataset Generation and Preprocessing: Synthetic Dataset: In alignment with the methodology outlined in [26], we simulate an environment with $u = 1,000$ users, grouped into $m = 10$ clusters, each comprising 100 users. A subset of 100 users is randomly designated as corrupted. User preferences and item vectors are generated in $d-1$ dimensions following a standard Gaussian distribution, normalized, and extended by one dimension with a constant value of 1, then scaled by $\sqrt{2}$, setting $d = 50$. An arm set of $|\mathcal{A}| = 1,000$ items is fixed, from which 20 items are randomly chosen in each round t to form a selection set $\mathcal{A}_t$. The corruption mechanism and the reward flipping for the initial rounds are implemented in [26], [32], with $T = 1,000,000$ total rounds and corruption level $C = 20,000$.

Real-world Datasets: On real-world datasets, we generate the preference and item vectors as in [26], [42], [43]. We first construct the binary feedback matrix through the users' ratings: if the rating is greater than 3, then the feedback is 1; otherwise, the feedback is 0. Then we use SVD to decompose the extracted binary feedback matrix. $R_{u\times m} = \boldsymbol{\theta} S X^{\mathrm{T}}$, where $\boldsymbol{\theta} = (\boldsymbol{\theta}_i), i \in [u]$ and $X = (\boldsymbol{x}_j), j \in [m]$, with dimensions $d = 50$ selected for both. To ensure a fair comparison with baseline algorithms, similar to [24], [27], [28], [35]), we employ identical real-world datasets, including the Movielens dataset (1,000 users, 1,000 items), the Amazon dataset (1,400 users, 800 items), and the Yelp dataset (2,000 users, 2,000 items). For experiments involving a larger number of users, please refer to the Appendix, available online. We form 10 clusters in the Movielens and Amazon datasets and 20 clusters in the Yelp dataset. The corruption of user feedback in these real-world datasets is the same manner as in the synthetic dataset.

3) Performance Evaluation and Analysis: Regret of Multiple Datasets: The performance results for our recommendation system are presented in Fig. 4(a)–(d). On the Movielens and Amazon datasets, characterized by smaller user gaps, LinUCB initially outperforms LinUCB-Ind. However, over time, LinUCB-Ind shows a tendency to catch up and potentially exceed LinUCB. For instance, on the Amazon dataset, which has the smallest proportion of corrupted users, the edge of RCLUB-WCU over other baselines is less pronounced, reflecting the lesser influence of corruption. Overall, RCLUB-WCU consistently exhibits lower regret across all datasets when compared to baseline methods, underscoring its effective adaptation to user preferences and inter-user relationships. The fact that RCLUB-WCU surpasses CW-OFUL-ind, despite both employing robust preference estimation techniques, underscores RCLUB-WCU's enhanced ability to utilize user information robustly and effectively.

Detection of Corrupted Users: The results of the detection of corrupted users are detailed in Table I. We test the AUC of OCCUD, GCUD, and NCUD for detection results every 200, 000 rounds. Over time, the performance of OCCUD improves significantly, outperforming GCUD and NCUD as it detects corrupted

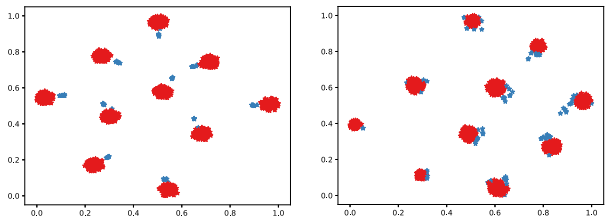


(a) Synthetic (b) Movielens (c) Amazon (d) Yelp

Fig. 4. Cumulative regret of recommendations in synthetic and real-world datasets without conversational feedback.

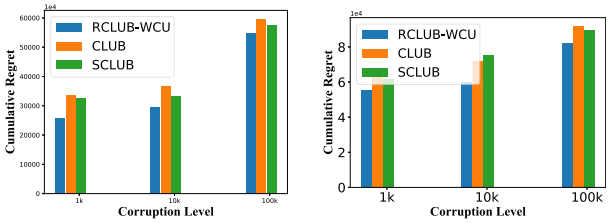
TABLE I
DETECTION RESULTS (AUC ± STANDARD DEVIATION) ON SYNTHETIC AND REAL DATASETS

Dataset	Algorithm \ Round	0.2M	0.4M	0.6M	0.8M	1M
Synthetic	**OCCUD**	0.599±0.0025	0.651±0.0032	0.777±0.0041	0.812±0.0049	**0.855**±0.005
	GCUD	0.477±0.0007	0.478±0.0007	0.483±0.0008	0.484±0.0008	0.502±0.001
	NCUD	0.452±0.0006	0.455±0.0008	0.460±0.0014	0.460±0.0015	0.464±0.0017
Movielens	**OCCUD**	0.650±0.004	0.750±0.0045	0.785±0.0047	0.830±0.0052	**0.850**±0.0063
	GCUD	0.450±0.0021	0.474±0.0026	0.485±0.0025	0.489 ±0.0030	0.492±0.0034
	NCUD	0.430±0.0007	0.430±0.0009	0.438±0.0012	0.442±0.0013	0.449±0.0016
Amazon	**OCCUD**	0.639±0.0011	0.735±0.0011	0.761±0.0020	0.802±0.0023	**0.840**±0.0022
	GCUD	0.480±0.0005	0.480±0.0013	0.486±0.0014	0.500±0.0017	0.518±0.0026
	NCUD	0.460±0.0012	0.461±0.0012	0.461±0.0018	0.465±0.0019	0.469±0.0021
Yelp	**OCCUD**	0.452±0.0015	0.489±0.0017	0.502±0.0021	0.578±0.0026	**0.628**±0.0032
	GCUD	0.473±0.0026	0.481±0.0027	0.496±0.0039	0.500±0.0042	0.510±0.0047
	NCUD	0.470±0.0017	0.475±0.0017	0.489±0.0022	0.502±0.0022	0.509±0.0024



(a) Synthetic (b) Movielens

Fig. 5. Visualization of the non-robust estimation of users by t-SNE. Red represents normal users, and blue represents corrupted users.



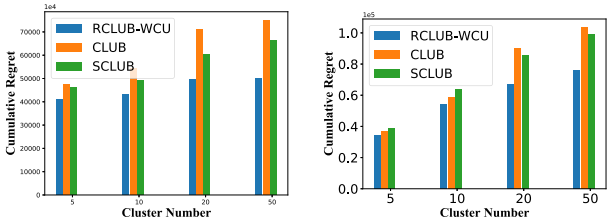
(a) Amazon (b) Yelp

Fig. 6. Cumulative regret under different corruption levels.

users only relying on robust estimations. Specifically, OCCUD achieves an AUC of 0.850 on the Movielens dataset, 0.840 on the Amazon dataset, and 0.628 on the Yelp dataset. AUC scores obtained are comparatively high, especially when benchmarked against recent literature on offline settings [27], [28]. In Fig. 5, we apply t-SNE [44] to analyze the non-robust estimations of preference vectors across the synthetic and Movielens datasets (due to space constraints, we will then evaluate the Amazon and Yelp datasets at the corruption level). The results show that the users with similar preferences form clusters, with a tendency for separation between normal and corrupted users.

Different Corruption Levels: We conduct experiments on the Amazon and Yelp datasets to assess the robustness of RCLUB-WCU, CLUB, and SCLUB against different levels of corruption, for these two baselines perform the best among all the baselines. Following the corruption mechanism described in Section V-B2, we vary the number of corrupted rounds T_c at 1,000; 10,000; and 100,000 to simulate increasing degrees of corruption. The outcomes, illustrated in Fig. 6, indicate a decline in all algorithms' performance with rising corruption levels. RCLUB-WCU maintains superiority with T_c up to 10,000, but at $T_c = 100,000$, its performance also degrades, consistent with our theoretical expectations. Across all tests, RCLUB-WCU outshines CLUB and SCLUB, demonstrating a more gradual increase in regret.

Varying Cluster Numbers: In line with [25], we evaluate the performance of cluster-based algorithms (RCLUB-WCU, CLUB, SCLUB) as the underlying cluster number varies. We set m to 5, 10, 20, and 50. The results, presented in Fig. 7, reveal that the performance of all these algorithms declines as the number of clusters increases, which is consistent with our



(a) Amazon (b) Yelp

Fig. 7. Cumulative regret with varying cluster numbers.

theoretical findings. The performance of CLUB and SCLUB deteriorates much more rapidly than that of RCLUB-WCU.

C. Evaluation With Conversational Feedback

In this section, we delve into user-interactive scenarios under sophisticated conversational recommendation frameworks, employing both synthetic and real-world datasets to assess the efficacy of our proposed algorithms.

1) Baseline Comparisons: For a more equitable evaluation within the CRS framework, and to underscore the strengths of our algorithm that consistently integrates conversational feedback, we have incorporated the following baseline algorithms that also facilitate direct conversations with users:

- *Arm-Con* [45]: This conversational bandit algorithm initiates dialogues about arms without key-term consideration, employing LinUCB for arm selection.
- *ConUCB* [5]: A fundamental conversational bandit algorithm that, when a conversation is permissible, chooses a key-term to minimize a certain estimation error.
- *ConLinUCB* [43]: A suite of algorithms varying the key-term selection strategy. It includes:
  - *ConLinUCB-BS*, which calculates the barycentric spanner of key terms for exploration.
  - *ConLinUCB-MCR*, which utilizes historical key-term selection data to choose terms with the largest confidence radius.
  - *ConLinUCB-UCB*, which employs a LinUCB-inspired strategy to select key terms with the highest upper confidence bound, combining the estimated mean with the confidence radius.

2) Dataset Generation and Preprocessing: The generation and preprocessing of synthetic and real-world datasets follow a process similar to that described in Section V-B, with corruption level $C = 40,000$. Consequently, we focus more on detailing the generation and setting of key-terms at the conversational level, following [5], [43], [46].

Synthetic Dataset: Each user preference vector $\boldsymbol{\theta}_i$ and each arm feature vector $\boldsymbol{x}_a$ are generated by independently drawing from the standard normal distribution $\mathcal{N}(-1, 1)$. Subsequently, these vectors are normalized. The weight matrix $\boldsymbol{W} \triangleq [w_{a,k}]$ is generated in the following manner: Initially, for each key-term k, an integer n_k is randomly selected from the range $\{1, 2, \ldots, 5\}$. A subset of n_k arms, denoted as $\mathcal{A}_k$, is then randomly chosen to be the related arms for key-term k. For each arm a that is related to a set of n_a key-terms $\mathcal{K}_a$, equal weights are assigned such that

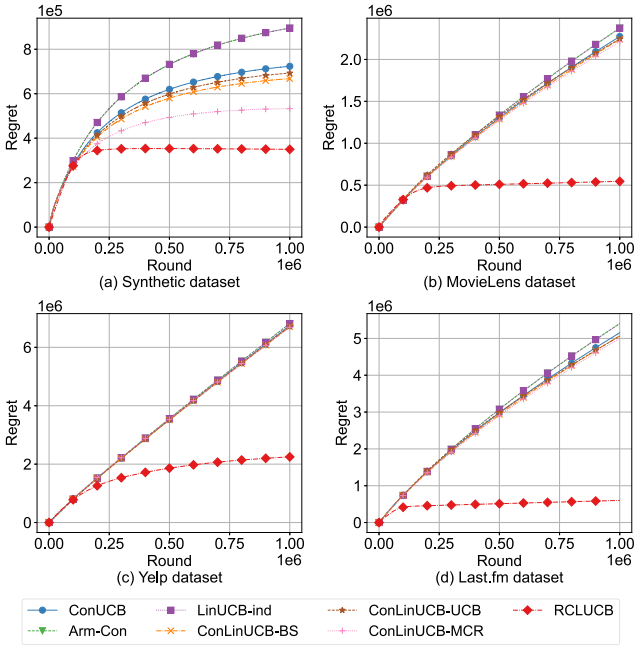


Fig. 8. Cumulative regret of recommendations in synthetic and real-world datasets with conversational feedback.

$w_{a,k} = \frac{1}{n_a}$ for all $k \in \mathcal{K}_a$. the feature vector for each key-term k is calculated as $\tilde{\boldsymbol{x}}_k = \sum_{a \in \mathcal{A}} \frac{w_{a,k}}{\sum_{a' \in \mathcal{A}} w_{a',k}} \boldsymbol{x}_a$. The key-term-level feedback is then generated in accordance with (3).

Real-World Datasets: Key-terms are extracted based on movie genres, business categories, or tag IDs from the MovieLens, Yelp, and Last.fm datasets, respectively. Specifically, we extract the top 2,000 arms ($|\mathcal{A}| = 2,000$) with the most user-assigned tags and the top 500 users ($N_u = 500$) who have assigned the most tags. For each arm, we retain a maximum of 20 tags that are associated with the most arms, and these are considered as the arm's associated key-terms. All the retained key-terms linked with the arms constitute the key-term set $\mathcal{K}$. The Last.FM dataset has $|\mathcal{K}| = 2,726$ key-terms, while the Movielens dataset has 5,585 key-terms. The Yelp has $|\mathcal{K}| = 805$. For each arm, the weights of all related key-terms are set to be equal. Following the approach in [5], the feature vectors of key-terms are computed as $\tilde{\boldsymbol{x}}_k = \sum_{a \in \mathcal{A}} \frac{w_{a,k}}{\sum_{a' \in \mathcal{A}} w_{a',k}} \boldsymbol{x}_a$. Eq. (3) continues to be used for generating the key-term-level feedback is then still generated.

3) Performance Evaluation and Analysis: Regret Analysis Across Multiple Datasets: We conduct an evaluation of cumulative regret across four distinct datasets, comparing the performance of RCLUCB-WCU against six baseline algorithms amidst the context of users arriving at random. The findings are presented in Fig. 8, with the number of rounds plotted on the x-axis and the cumulative regret on the y-axis Consistent patterns emerge across all datasets, corroborating prior works. Specifically, each algorithm demonstrates sublinear regret as the number of rounds increases. Algorithms that do not incorporate querying of key terms, i.e., LinUCB-ind and Arm-Con, exhibit

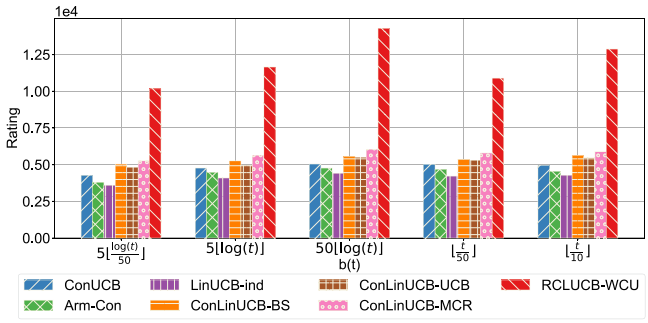


Fig. 9. Rating under different conversation functions.

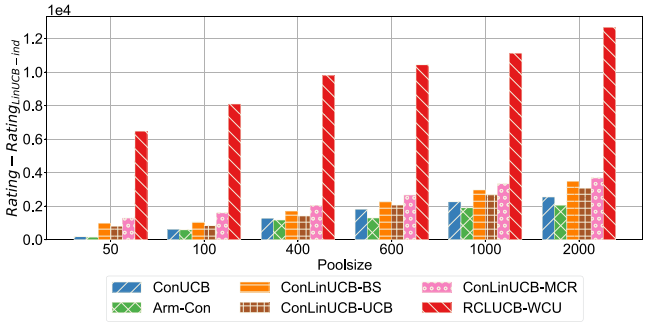


Fig. 10. Evaluation of user ratings across various arm poolsizes.

the highest cumulative regret, underscoring the value of conversational interactions for improved performance. Notably, our RCLUCB-WCU algorithm surpasses all competing algorithms, securing the improvement of 34.3%, 75.3%, 66.3%, 87.9%, over ConLinUCB-MCR—the strongest baseline—across all four datasets, respectively, under potentially corrupted feedback on both arm and conversation interaction levels. This enhancement in performance lends empirical support to our theoretical findings and highlights the efficacy of incorporating conversational elements into the recommendation process.

Impact of Conversation Frequency: To compare the impact of the conversation frequency on all algorithms since different users arrive randomly, we maintain the same unified conversation function for all algorithms, which are $\{5\lfloor\frac{\log(t)}{50}\rfloor, 5\lfloor\log(t)\rfloor, 50\lfloor\log(t)\rfloor, \lfloor\frac{t}{50}\rfloor, \lfloor\frac{t}{10}\rfloor\}$, respectively. We then assess the cumulative rating of the items recommended to all users who arrive randomly by the time rounds $T = 10000$, averaged under 6 random trials. A higher value of $b(t)$ allows the agent to engage in more conversations. The outcomes of this setup are shown in Fig. 9 under the largest dataset Yelp. As $b(t)$ increases, our algorithms demonstrate a corresponding rise in ratings, which underscores the beneficial impact of more frequent conversations. In every scenario, RCLUCB-WCU outperforms the ConUCB, Arm-Con, ConLinUCB-UCB, ConLinUCB-MCR, and ConLinUCB-BS, showcasing its robustness across different conversation frequencies.

Different Arm Set Sizes: We evaluate the influence of arm set poolsize on algorithm efficacy by varying the size of $|\mathcal{A}_t|$ to include {50, 100, 400, 600, 1000, 2000} options within the Yelp dataset which boasts the largest user base. Note that an increase in $|\mathcal{A}_t|$ typically makes the task of identifying the optimal arm more challenging. To underscore the superiority of our algorithms under conversation interactions, we measure the rating differential between LinUCB-ind and the other contenders, denoted as $Rating - Rating_{\text{LinUCB-ind}}$. This differential quantifies the enhancement in user feedback attributable to the conversational bandit algorithms relative to LinUCB-ind. As shown in Fig. 10, our proposed algorithm, RCLUCB-WCU, demonstrates increasingly pronounced benefits even as $|\mathcal{A}_t|$ grows. Specifically, with an arm set size of 50, RCLUCB-WCU achieves improvements by factors of 36.9×, 43.9×, 7.2×, 4.0×, and 5.6× over ConUCB, Arm-Con, ConLinUCB-UCB, ConLinUCB-MCR, and ConLinUCB-BS, respectively. When the arm set size expands to 2000, the improvements by RCLUCB-WCU are by factors of 4.0×, 5.2×, 3.1×, 2.4×, and 2.7×, respectively, over the same benchmarks. Given that recommendation applications often involve a large arm set size, RCLUCB-WCU is expected to deliver a substantial performance edge over baselines in practical settings.

TABLE II
CUMULATIVE REGRET ACROSS DIFFERENT PARAMETER CONFIGURATIONS OF THE RCLUCB-WCU SERIES

Dataset \ Algorithm	RCLUCB-WCU (a)	RCLUCB-WCU (b)	RCLUCB-WCU (c)	RCLUCB-WCU (d)	RCLUCB-WCU (e)
Synthetic	48140	47435	47674	46637	46168
MovieLens	43380	43839	43255	41799	44707
Last.fm	61282	61584	61511	62781	61923
Yelp	74581	72063	71010	70772	72035

Ablation Study on Parameter Sensitivity: We then present an ablation study designed to investigate the sensitivity of various parameters, including the regularization parameter λ, the confidence radius parameter β, the threshold parameter α, and the edge deletion parameter α_1. Specifically, we have configured the following parameter settings $(\lambda, \beta, \alpha, \alpha_1)$ for the RCLUCB-WCU series: RCLUCB-WCU (a) with $(1, 1.5, 0.2, 1)$, RCLUCB-WCU (b) with $(0.9, 1.4, 0.3, 0.2)$, RCLUCB-WCU (c) with $(1.1, 1.3, 0.25, 0.25)$, RCLUCB-WCU (d) with $(0.8, 1.2, 0.4, 0.3)$, and RCLUCB-WCU (e) with $(1.2, 1.1, 0.5, 0.15)$. We have set the experimental rounds T and the corruption level C at 10,000 for each configuration. Table II summarizes the results on cumulative regret across four distinct datasets. These results illustrate the consistent performance under various parameter configurations, underscoring the robustness of our algorithm RCLUCB-WCU.

VI. Related Work

Conversational Recommendation Systems: Conversational recommendation systems (CRSs) engage in conversations by asking users if they like the items chosen by the bandit algorithm [45]. Further improvement has been achieved by taking advantage of recent developments in deep learning or reinforcement learning to generate conversations and assist recommendations, but without a theoretical guarantee [2], [47]. Conversational feedback on key terms is used to help elicit user preferences and accelerate online recommendations [5].

Some subsequent works attempt to improve the performance of CRSs with the aid of additional information, such as relative feedback [6], self-generated key-terms [42], and knowledge graph [30]. Unlike these works, we consider learning unknown user relations via bandit feedback and leveraging these learned relations to improve quality of CRSs.

Bandits With Adversarial Corruptions: The work [10] first studies the problem of bandits with adversarial corruptions, where the rewards are corrupted with the sum of corruption magnitudes in all rounds constrained by the *corruption level* C. The paper [14] proposes an improved algorithm with a tighter regret bound. The work [17] first studies stochastic linear bandits with adversarial corruptions. To tackle the contextual linear bandit setting where the arm set changes over time, the work [18] proposes a variant of the OFUL [39] that achieves a sub-linear regret. A recent work [12] proposes the CW-OFUL algorithm that achieves a nearly optimal regret bound. All these works focus on designing robust bandit algorithms for a single user; none consider how to robustly learn and leverage the implicit relations among potentially corrupted users for more efficient learning. Moreover, none of them considers how to online detect corrupted users in the multi-user case. We expand upon the findings of [48], moving from simple arm-level reward corruption to more complex two-level corruption affecting both arm and key levels.

Fraud Detection: The goal of fraud detection is to distinguish fraudsters from normal users. Various efforts have been made to detect offline fraud [20], [21]. With the development of the graph learning architecture GNN, the paper [19] proposes the SemiGNN model, which applies a GNN-based hierarchical attention mechanism to do fraud detection in financial applications. To tackle the issue of label imbalance, the paper [22] proposes the PC-GNN model with node resampling, and the work [23] devises the AO-GNN model with edge pruning. A recent work [49] proposes the H2-FDetector model considering the influence of homophilic and heterophilic connections. All these works are based on offline static *known* information of users and user relations. These works have not explored detecting corrupted users in an online setting with bandit feedback, where such an approach is crucial due to dynamic user characteristics [42] and the privacy concerns associated with collecting user data in recommendation systems [50]. Furthermore, they also have not considered optimizing the balance between exploration and exploitation for long-term rewards, which significantly differentiates our work.

Bandits with Multiple Users: Some works study how to leverage user relations to speed up the bandit learning process in the case of multiple users. The work [51] leverages a *known* user adjacency graph to share context and rewards among neighboring users. To adaptively learn and utilize the *unknown* user relations, the paper [24] formulates the clustering of bandits (CB) problem where there is an *unknown* user clustering structure to be learned by the agent. A follow-up work [40] uses the collaborative effects on items to guide the clustering of users. The work [25] studies the CB problem in a cascading bandit setting. The paper [26] considers the setting where users have different arrival frequencies. A recent work [33] studies the problem of federated clustering of bandits. All these works only consider leveraging the relations among normal users; none of them have considered how to robustly learn the user relations from potentially disrupted behaviors, and thus would easily be misled by corrupted users.

To the best of our knowledge, this paper is the first work to study the problem of (i) learn the *unknown* user relations and preferences from potentially corrupted user behaviors; (ii) adaptively detect the corrupted users online from both selection and conversational feedback. Compared to neural-based online learning methods, such as NeuralUCB, our approach offers relatively lower computational costs [52].

VII. Conclusion and Future Work

In this paper, we introduce the OLDCU problem within a conversational recommendation framework, dealing with users who have unknown preferences and relations, some of whom may perform corrupted actions. Our novel bandit algorithm, RCLUB-WCU, optimizes item and key-term selection through user interactions, while OCCUD detects corrupted users based on learned user relations. We provide theoretical performance analysis, including a sublinear regret upper bound for RCLUB-WCU and an evaluation of OCCUD's corrupted user detection accuracy. Our extensive experiments demonstrate that our algorithms outperform existing bandit algorithms and achieve high accuracy in detecting corrupted users.

For future work, we intend to incorporate user-centered evaluations to align our proposed algorithms more closely with real-world user interactions and needs. Additionally, investigating the integration of our bandit learning techniques with large language models (LLMs) in conversational systems could prove intriguing.

References

[1] Y. Sun and Y. Zhang, "Conversational recommender system," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 235–244.
[2] Y. Zhang, X. Chen, Q. Ai, L. Yang, and W. B. Croft, "Towards conversational search and recommendation: System ask, user respond," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, 2018, pp. 177–186.
[3] S. Li, W. Lei, Q. Wu, X. He, P. Jiang, and T.-S. Chua, "Seamlessly unifying attributes and items: Conversational recommendation for cold-start users," *ACM Trans. Inf. Syst.*, vol. 39, no. 4, pp. 1–29, 2021.
[4] C. Gao, W. Lei, X. He, M. de Rijke, and T.-S. Chua, "Advances and challenges in conversational recommender systems: A survey," *AI Open*, vol. 2, pp. 100–126, 2021.
[5] X. Zhang, H. Xie, H. Li, and J. C. S. Lui, "Conversational contextual bandit: Algorithm and application," in *Proc. Web Conf.*, 2020, pp. 662–672.
[6] Z. Xie, T. Yu, C. Zhao, and S. Li, "Comparison-based conversational recommender system with relative bandit feedback," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 1400–1409.
[7] W. Chu, L. Li, L. Reyzin, and R. Schapire, "Contextual bandits with linear payoff functions," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 208–214.
[8] P. Kohli, M. Salek, and G. Stoddard, "A fast bandit algorithm for recommendation to users with heterogenous tastes," in *Proc. AAAI Conf. Artif. Intell.*, 2013, pp. 1135–1141.
[9] E. Garcelon et al., "Adversarial attacks on linear contextual bandits," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 14362–14373.
[10] T. Lykouris, V. Mirrokni, and R. Paes Leme, "Stochastic bandits robust to adversarial corruptions," in *Proc. 50th Annu. ACM SIGACT Symp. Theory Comput.*, 2018, pp. 114–122.

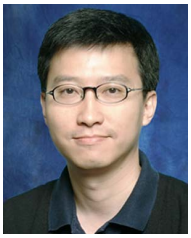
[11] Y. Ma, K.-S. Jun, L. Li, and X. Zhu, "Data poisoning attacks in contextual bandits," in *Proc. Int. Conf. Decis. Game Theory Secur.*, Springer, 2018, pp. 186–204.
[12] J. He, D. Zhou, T. Zhang, and Q. Gu, "Nearly optimal algorithms for linear contextual bandits with adversarial corruptions," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 34614–34625.
[13] M. Hajiesmaili et al., "Adversarial bandits with corruptions: Regret lower bound and no-regret algorithm," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 19943–19952.
[14] A. Gupta, T. Koren, and K. Talwar, "Better algorithms for stochastic bandits with adversarial corruptions," in *Proc. Conf. Learn. Theory*, 2019, pp. 1562–1578.
[15] K.-S. Jun, L. Li, Y. Ma, and J. Zhu, "Adversarial attacks on stochastic bandits," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 3644–3653.
[16] F. Liu and N. Shroff, "Data poisoning attacks on stochastic bandits," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4042–4050.
[17] Y. Li, E. Y. Lou, and L. Shan, "Stochastic linear optimization with adversarial corruption," 2019, *arXiv: 1909.02109*.
[18] Q. Ding, C.-J. Hsieh, and J. Sharpnack, "Robust stochastic linear contextual bandits under adversarial attacks," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2022, pp. 7111–7123.
[19] D. Wang et al., "A semi-supervised graph attentive network for financial fraud detection," in *Proc. IEEE Int. Conf. Data Mining*, 2019, pp. 598–607.
[20] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 315–324.
[21] G. Zhang et al., "FRAUDRE: Fraud detection dual-resistant to graph inconsistency and imbalance," in *Proc. IEEE Int. Conf. Data Mining*, 2021, pp. 867–876.
[22] Y. Liu et al., "Pick and choose: A GNN-based imbalanced learning approach for fraud detection," in *Proc. Web Conf.*, 2021, pp. 3168–3177.
[23] M. Huang et al., "AUC-oriented graph neural network for fraud detection," in *Proc. ACM Web Conf.*, 2022, pp. 1311–1321.
[24] C. Gentile, S. Li, and G. Zappella, "Online clustering of bandits," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 757–765.
[25] S. Li and S. Zhang, "Online clustering of contextual cascading bandits," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 3554–3561.
[26] S. Li, W. Chen, S. Li, and K.-S. Leung, "Improved algorithm on online clustering of bandits," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 2923–2929.
[27] Q. Li, Y. He, C. Xu, F. Wu, J. Gao, and Z. Li, "Dual-augment graph neural network for fraud detection," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.*, 2022, pp. 4188–4192.
[28] Z. Qin, Y. Liu, Q. He, and X. Ao, "Explainable graph-based fraud detection via neural meta-graph search," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.*, 2022, pp. 4414–4418.
[29] T. Lattimore and C. Szepesvári, *Bandit Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2020.
[30] C. Zhao, T. Yu, Z. Xie, and S. Li, "Knowledge-aware conversational preference elicitation with bandit feedback," in *Proc. ACM Web Conf.*, 2022, pp. 483–492.
[31] Z. Li, M. Liu, and J. Lui, "FedConPE: Efficient federated conversational bandits with heterogeneous clients," in *Proc. Int. Joint Conf. Artif. Intell.*, 2024, pp. 4533–4541.
[32] H. Zhao, D. Zhou, and Q. Gu, "Linear contextual bandits with adversarial corruptions," 2021, *arXiv:2110.12615*.
[33] X. Liu, H. Zhao, T. Yu, S. Li, and J. C. Lui, "Federated online clustering of bandits," in *Proc. 38th Conf. Uncertainty Artif. Intell.*, 2022, pp. 1221–1231.
[34] F. M. Harper and J. A. Konstan, "The MovieLens datasets: History and context," *ACM Trans. Interactive Intell. Syst.*, vol. 5, no. 4, pp. 1–19, 2015.
[35] S. Liu, B. Hooi, and C. Faloutsos, "HoloScope: Topology-and-spike aware fraud detection," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 1539–1548.
[36] J. J. McAuley and J. Leskovec, "From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 897–908.
[37] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 985–994.
[38] I. Cantador, P. Brusilovsky, and T. Kuflik, "Second workshop on information heterogeneity and fusion in recommender systems (HetRec2011)," in *Proc. 5th ACM Conf. Recommender Syst.*, 2011, pp. 387–388.
[39] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári, "Improved algorithms for linear stochastic bandits," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 2312–2320.
[40] S. Li, A. Karatzoglou, and C. Gentile, "Collaborative filtering bandits," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2016, pp. 539–548.
[41] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 661–670.
[42] J. Wu, C. Zhao, T. Yu, J. Li, and S. Li, "Clustering of conversational bandits for user preference learning and elicitation," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, 2021, pp. 2129–2139.
[43] Z. Wang, X. Liu, S. Li, and J. C. Lui, "Efficient explorative key-term selection strategies for conversational contextual bandits," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 10288–10295.
[44] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.
[45] K. Christakopoulou, F. Radlinski, and K. Hofmann, "Towards conversational recommender systems," in *Proc. 22nd ACM SIGKDD Int. Conf.*, 2016, pp. 815–824.
[46] W. Lei, X. He, M. de Rijke, and T.-S. Chua, "Conversational recommendation: Formulation, methods, and evaluation," in *Proc. 43rd Int. ACM SIGIR Conf.*, 2020, pp. 2425–2428.
[47] K. Christakopoulou, A. Beutel, R. Li, S. Jain, and E. H. Chi, "Q&R: A two-stage approach toward interactive recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 139–148.
[48] Z. Wang, J. Xie, T. Yu, S. Li, and J. Lui, "Online corrupted user detection and regret minimization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2024, pp. 33262–33287.
[49] F. Shi, Y. Cao, Y. Shang, Y. Zhou, C. Zhou, and J. Wu, "H2-FDetector: A GNN-based fraud detector with homophilic and heterophilic connections," in *Proc. ACM Web Conf.*, 2022, pp. 1486–1494.
[50] F. Liu, Z. Cheng, H. Chen, Y. Wei, L. Nie, and M. Kankanhalli, "Privacy-preserving synthetic data generation for recommendation systems," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2022, pp. 1379–1389.
[51] Q. Wu, H. Wang, Q. Gu, and H. Wang, "Contextual bandits in a collaborative environment," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2016, pp. 529–538.
[52] Y. Qi, Y. Ban, T. Wei, J. Zou, H. Yao, and J. He, "Meta-learning with neural bandit scheduler," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2024, Art. no. 2796.

Xiangxiang Dai (Student Member, IEEE) received the BE degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology (HUST), Wuhan, China, in 2023. He is currently working toward the PhD degree with the Department of Computer Science and Engineering, Chinese University of Hong Kong, advised by Prof. John C. S. Lui. His research interests include online learning theory and their algorithm design for various applications, such as web recommendation systems, video analytics, and computer networks.

Zhiyong Wang received the bachelor's degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology, in 2021. He is currently working toward the PhD degree with the Department of Computer Science and Engineering, Chinese University of Hong Kong, advised by Prof. John C. S. Lui. His research interests include machine learning, reinforcement learning theory, online learning, and bandits.

Jize Xie received the bachelor's degree from the Department of Electrical Engineering, Shanghai Jiao Tong University, in 2019. He is currently working toward the PhD degree with the Department of Industrial Engineering and Decision Analytics, Hong Kong University of Science and Technology, Hong Kong, SAR, from 2023. His research interests include machine learning, operations research, and bandits.

Tong Yu received the PhD degree from the Department of Electrical and Computer Engineering, Carnegie Mellon University. He is a senior research scientist with Adobe Research. His current research focuses on LLMs, generative models, multimodal learning and reinforcement learning, with applications in dialog systems and recommender systems. He has published more than 60 papers in conferences and journals, including NeurIPS, ICML, ICLR, ACL, EMNLP, NAACL, CVPR, AAAI, IJCAI, AISTATS, UAI, KDD, ACM Web Conference, SIGIR, VLDB, CIKM, the *Journal of Machine Learning Research*, etc.



John C. S. Lui (Fellow, IEEE) received the PhD degree in computer science from UCLA. He is currently the Choh-Ming Li chair professor with the Department of Computer Science and Engineering, Chinese University of Hong Kong. His current research interests include online learning algorithms and applications (e.g., multi-armed bandits, reinforcement learning), quantum Internet, machine learning on network sciences and networking systems, large scale data analytics, network/system security, network economics, large scale storage systems, and performance evaluation theory. He is currrently the senior editor of *IEEE/ACM Transactions on Networking*, and has been serving in the editorial board of the *ACM Transactions on Modeling and Performance Evaluation of Computing Systems, IEEE Transactions on Network Science & Engineering, IEEE Transactions on Mobile Computing, IEEE Transactions on Computers, IEEE Transactions on Parallel and Distributed Systems, Journal of Performance Evaluation* and so on. He is a member of the review panel in the IEEE Koji Kobayashi Computers and Communications Award committee, and has served at the IEEE Fellow Review Committees. He served the associate dean of research with the College of Engineering, CUHK (2014–2018) and the chairman of the CSE Department from 2005–2011. He received various departmental teaching awards and the CUHK Vice-Chancellor's Exemplary Teaching Award. He also received the CUHK Faculty of Engineering Research Excellence Award (2011–2012). He is a co-recipient of the best paper award in the IFIP WG 7.3 Performance 2005, IEEE/IFIP NOMS 2006, SIMPLEX 2013, and ACM RecSys 2017. He is an elected member of the IFIP WG 7.3, fellow of ACM, senior research fellow of the Croucher Foundation, fellow of the Hong Kong Academy of Engineering Sciences (HKAES), and was the past chair of the ACM SIGMETRICS (2011–2015). His personal interests include films and general reading.