

Hierarchical Conversational Preference Elicitation with Bandit Feedback

Jinhang Zuo
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
jzuo@andrew.cmu.edu

Songwen Hu
Shanghai Jiao Tong University
Shanghai, China
hsw828@sjtu.edu.cn

Tong Yu
Adobe Research
San Jose, California, USA
tyu@adobe.com

Shuai Li*
Shanghai Jiao Tong University
Shanghai, China
shuaili8@sjtu.edu.cn

Handong Zhao
Adobe Research
San Jose, California, USA
hazhao@adobe.com

Carlee Joe-Wong
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
cjowong@andrew.cmu.edu

ABSTRACT

The recent advances of conversational recommendations provide a promising way to efficiently elicit users' preferences via conversational interactions. To achieve this, the recommender system conducts conversations with users, asking their preferences for different items or item categories. Most existing conversational recommender systems for cold-start users utilize a multi-armed bandit framework to learn users' preference in an online manner. However, they rely on a pre-defined conversation frequency for asking about item categories instead of individual items, which may incur excessive conversational interactions that hurt user experience. To enable more flexible questioning about key-terms, we formulate a new conversational bandit problem that allows the recommender system to choose either a key-term or an item to recommend at each round and explicitly models the rewards of these actions. This motivates us to handle a new exploration-exploitation (EE) trade-off between key-term asking and item recommendation, which requires us to accurately model the relationship between key-term and item rewards. We conduct a survey and analyze a real-world dataset to find that, unlike assumptions made in prior works, key-term rewards are mainly affected by rewards of representative items. We propose two bandit algorithms, Hier-UCB and Hier-LinUCB, that leverage this observed relationship and the hierarchical structure between key-terms and items to efficiently learn which items to recommend. We theoretically prove that our algorithm can reduce the regret bound's dependency on the total number of items from previous work. We validate our proposed algorithms and regret bound on both synthetic and real-world data.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Theory of computation** → **Online learning algorithms**.

*Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

Conference'17, July 2017, Washington, DC, USA

© 2022 Association for Computing Machinery.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM... \$15.00

<https://doi.org/10.1145/nmnnmnnn.nmnnmnnn>

KEYWORDS

conversational recommender system, online learning

ACM Reference Format:

Jinhang Zuo, Songwen Hu, Tong Yu, Shuai Li, Handong Zhao, and Carlee Joe-Wong. 2022. Hierarchical Conversational Preference Elicitation with Bandit Feedback. In *Proceedings of ACM Conference (Conference'17)*. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/nmnnmnnn.nmnnmnnn>

1 INTRODUCTION

Recommender systems have attracted much attention in a variety of areas over the past decades, e.g., to steer users towards movies they will enjoy [1] or suggest applications that they may find useful [2]. Most recommender systems utilize large amounts of historical data to learn user behaviour and preferences. However, for cold-start users who interact with the system for a short amount of time, there is not enough historical data to reliably learn user preferences, and thus the system has to interact with these users in an online manner in order to make informed recommendations. Successfully learning user preferences from this interaction, however, is challenging, as most users will not tolerate extensive exploration of potential items [3, 4]. The recent advances of conversational recommender systems (CRS) propose a new way to efficiently elicit users' preferences [3, 5–9]. The system is allowed to conduct conversations with users to get richer feedback information, which enables better preference elicitation and alleviates extensive explorations. These conversations are expected to be less burdensome for users than receiving irrelevant recommendations, allowing for less onerous exploration of item recommendations [7].

Research challenges. Although CRSs have achieved success in many fields (e.g., movie [5] and restaurant [6] recommendation), current conversation mechanisms still face challenges. Most mechanisms utilize a multi-armed bandit framework to decide what the agent should ask a user, based on the results of previous conversational interactions. Existing conversational bandits [3, 7, 8] rely on a pre-defined conversation frequency. Specifically, at some pre-determined conversation rounds, the recommender agent asks users questions about “key-terms,” i.e., their preferences for categories of items that could be recommended, and then receives feedback that it can use to infer users' item preferences. However, even though conversational interactions may be more tolerable to users than poor item recommendations [7], we should still limit the number of required interactions for finding an optimal item [11]. Thus, for

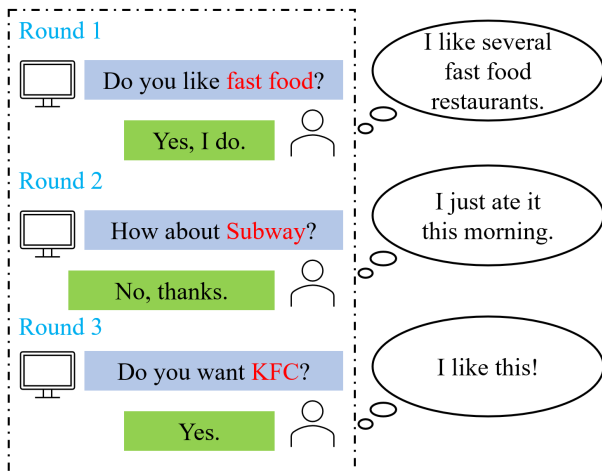


Figure 1: An example showing the importance of accurately modeling the relationship between the user preferences for key-terms and items. The user preference for ‘fast food’ is affected by some fast food restaurants (i.e., ‘KFC’ in this example), rather than all fast food restaurants. The user’s rejection of ‘Subway’ does not mean that the agent should always be penalized for recommending the key-term ‘fast food’. In fact, the user does like ‘fast food’ in this case. However, after round 2, existing approaches [3, 10, 11] will penalize the agent for recommending the key-term ‘fast food’ in round 1.

users whose preferences are easy to learn, we want to have fewer interactions, e.g., for users with clear preferences between key-term categories, more frequent key-term questions upfront may allow us to quickly identify a good key-term category containing items that the user likes. Prior work [11] takes a step in this direction by unifying key-term questions and item recommendation in the same action space, but their solutions do not have theoretical guarantees and do not explicitly consider the cost of key-term questions, which can still incur burdensome conversational interactions.

A natural approach to enable more flexible key-term question interactions is to treat each such interaction as an alternative to item recommendation. These interactions can then be associated with a reward indicating how much the user likes this category of items, just as item recommendations are associated with rewards that represent how much a user likes the item. We can then model the recommender agent’s actions as a sequence of actions, each drawn from an action space that includes both conversational key-term interactions and item recommendations. The goal is to choose a sequence of actions, informed by the results from prior actions, that maximizes the cumulative reward, i.e., user satisfaction. In doing so, we implicitly limit the number of required interactions before finding high-reward items (i.e., ones the user likes). To achieve this, **we formulate a new conversational bandit problem** that allows the agent to choose either a key-term or an item to recommend at each round and explicitly models the rewards of both actions.

This conversational bandit setting introduces a *new exploration-exploitation (EE) trade-off between key-term questions and item recommendation*, which can not be handled by previous work. Balancing such an EE trade-off is challenging as it is affected by the complicated relationship between item rewards and key-term rewards. One example of this relationship in conversational restaurant recommendations is illustrated by Figure 1. The recommender agent first recommends key-term “fast food” to the user, which satisfies the user as he likes fast food restaurants such as KFC, Chipotle, and Wendy’s. The agent then recommends “Subway”, but the user rejects that recommendation since he just ate it shortly before and does not want to eat it again. The agent finally recommends another fast food restaurant “KFC”, which is accepted by the user, and the interaction ends. In this example, the user’s preference for “fast food” is mainly determined by his enjoyment of “KFC”, but not his rejection of “Subway”. Intuitively, users’ preferences for one key-term (i.e., the rewards associated with that key-term) are primarily affected by their preferences for the best (i.e., most preferred) items within that key-term category. Another example where the key-term reward is determined by the rewards of the best items can be found in image retrieval [12]: given a specific query from the user, we may show the user different clusters (key-terms) of images. If a cluster contains any images the user is looking for, he should be satisfied with that cluster. Thus, the user’s satisfaction of a cluster is decided by whether it contains the desired images.

To verify our above intuition of the relationship between item and key-term rewards, we conduct a survey (as detailed in Section 3.2) asking participants to rate restaurants (items) and corresponding categories (key-terms). We find that *the item and key-term reward relationship defined by previous work is inaccurate for many categories*. Specifically, [3, 8] consider the reward of a key-term to be the average of the rewards of all associated items. However, we find this average is usually smaller than the true key-term ratings from participants, which are closer to the ratings of the most highly rated associated items. These findings are consistent with an analysis of restaurant ratings from the Yelp dataset (Section 3.2).

Our contributions. Inspired by the above observation that key-term rewards are mainly affected by the rewards of representative items, we leverage this relationship and the hierarchical structure between key-terms and items to develop more efficient recommendation algorithms. Since maximizing the cumulative reward is achieved by quickly identifying the item that most appeals to the user, we can accelerate our search by first identifying the most appealing key-term (which, from our results above, will contain the most appealing item). To do so, we propose two algorithms, Hier-UCB (upper confidence bound) and Hier-LinUCB, for stochastic and contextual conversational bandit settings, respectively. We prove the theoretical regret bound of the proposed algorithm reduces the dependency on the total number of items, from $|\mathcal{A}|$ in previous work [3] to $|\mathcal{K}| + |\mathcal{A}_{k^*}|$, where $|\mathcal{A}|$ is the size of the item pool, $|\mathcal{K}|$ is the size of the key-term pool, and $|\mathcal{A}_{k^*}|$ is the number of constituent items of the best key-term. Our regret analysis is also different from the previous hierarchical Bayesian bandit works [13–17]: with a careful treatment of the regrets caused by choosing sub-optimal key-terms or items, we provide a stronger frequentist regret bound that, unlike the previous Bayes regret bounds, is agnostic to the quality of prior information. Experimental results on a

synthetic dataset show that Hier-UCB outperforms both traditional UCB and Hier-LinUCB with a moderated item pool. We run Hier-UCB with three real-world datasets with large item pools and find it outperforms baseline algorithms from previous works.

In summary, our **research contributions** are as follows:

- To enable flexible key-term question frequencies for preference elicitation, we *formulate a conversational bandit problem* that explicitly models the exploration-exploitation (EE) trade-off between key-term questions and item recommendation.
- We *study the relationship between item and key-term rewards based on a survey and analysis of a real-world dataset*. We have an essential observation for our algorithm design that the key-term rewards are mainly determined by the rewards of representative items.
- We propose *Hier-UCB and Hier-LinUCB*, efficient bandit algorithms that leverage the observed relationship and the hierarchical structure between attributes and items. We prove that Hier-UCB reduces the theoretical regret bound's dependency on the total number of items in prior work.
- We *validate our proposed algorithms and regret bounds* on both synthetic and real-world data.

The rest of the paper is organized as follows. We formulate the new conversational bandit problem and study the item-key-term reward relationship in Section 3. In Section 4, we propose Hier-UCB and Hier-LinUCB and analyze their regrets. The experimental setup and results are presented in Section 5. We discuss some related works in Section 2 and conclude the paper in Section 6.

2 RELATED WORK

2.1 Conversational Preference Elicitation

The recent advances of conversational recommender systems (CRS) provide a promising way to efficiently elicit users' preferences [3, 5–8, 18, 19]. By developing advanced models and efficient algorithms in CRS, we can promisingly derive more information about the user preferences and understand their preferences more accurately [20–29]. To determine when to have the conversations and what to ask during conversations, previous works formulate the task as a multi-step decision-making process and propose reinforcement learning based approaches [30–34]. The most related works of this paper are a series of studies that utilize a multi-armed bandit framework for preference elicitation from cold-start users in an online manner. Christakopoulou et al. [7] first introduce the use of bandit-based strategies for preference elicitation into CRS. Zhang et al. [3] formulate the conversational contextual bandit problem that integrates linear contextual bandits and CRS. They propose the ConUCB algorithm, one of our baselines, which achieves a smaller theoretical regret upper bound than the conventional contextual bandit algorithm LinUCB [35]. A follow-up [11] models key-term asking and item recommendation in a unified framework and proposes a bandit solution based on Thompson Sampling. Xie et al. [8] introduce a comparison-based CRS with relative feedback to the comparisons of key-terms. However, none of these works explicitly model the cost (i.e., incurred regret) of the conversations that ask users about their key-term preferences in the conversational bandit framework. Thus, they may let users suffer from excessive conversational interactions. To the best of our knowledge, we are the first to formally model

the regret of key-term asking and study the new EE trade-off between key-term asking and item recommendation, to enable more flexible conversations. We further revisit the relationship between key-term and item rewards and, based on our results, propose a new model for key-term rewards as determined by the rewards of representative items, instead of all items as assumed in prior work.

2.2 Bandits with Hierarchy

There is a branch of research that relates to our proposed bandit algorithms that take advantages of a hierarchy structure. Hierarchical Bayesian bandits have been studied in [13–17], with applications in meta-learning and multi-task learning. A hierarchical Bayesian framework is proposed in [13] to solve the metadata-based multi-task multi-armed bandit problem. Hong et al. [14] formulate a general hierarchical Bayesian bandit problem for solving multiple similar bandit tasks sequentially or concurrently, where each task is parameterized by task parameters drawn from a distribution parameterized by hyper-parameters. Hong et al. [16] study the bandit problem with a deep hierarchical structure other than the traditional 2-level Bayesian hierarchy. However, all these works rely on the prior knowledge of arms and specific Bayesian hierarchical structures that may not be practical in real-world applications. Our proposed algorithms do not require any prior information and utilize the simple hierarchical structure between key-terms and items that is available in most applications. Moreover, we provide a stronger frequentist regret bound than the Bayes regret bounds in previous works, which is agnostic to the quality of priors. Our analysis for the frequentist regret is also different from that for the Bayes regret: without the hierarchical Bayesian structure, we need to carefully bound the regrets caused by the 4 cases in Section 3.3.

3 PROBLEM FORMULATION

In this section, we will formulate conversational recommendation as a conversational bandit problem. We start by introducing the stochastic conversational bandit scenario where an agent can recommend either a key-term or an item to a user. We then discuss the limitations of the previous works' models of key-term rewards and propose a new key-term reward function, based on both human evaluation and data analysis. We also provide a contextual version of our new conversational bandit problem.

3.1 Stochastic Conversational Bandit

We formulate a novel stochastic conversational bandit problem that allows the recommender agent to choose either key-term asking or item recommendation at each round. It explicitly models the regret caused by key-term asking, which is ignored by prior work [3, 8, 11]. The goal of the agent is to maximize the cumulative reward in T rounds through repeated key-term and item recommendations.

Consider a finite set of items (e.g., restaurants) denoted by \mathcal{A} and a finite set of key-terms (e.g., restaurant categories) denoted by \mathcal{K} . The relationships between the arms and the key-terms are defined by a weighted bipartite graph $(\mathcal{A}, \mathcal{K}, W)$, where item $a \in \mathcal{A}$ is associated to a key-term $k \in \mathcal{K}$ with weight $W_{a,k} \geq 0$. Without loss of generality, we assume $\sum_{k \in \mathcal{K}} W_{a,k} = 1$. More discussion of the weights can be found in Section 3.2.

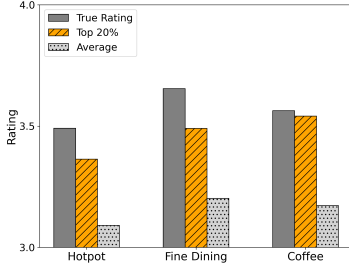


Figure 2: Restaurant category ratings from our survey are closer to the ratings of the top 20% of items in each category than the average ratings of all items in each category.

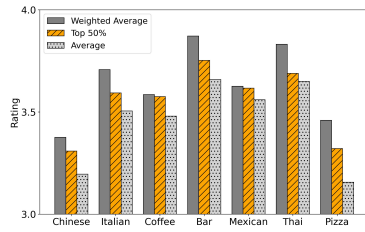


Figure 3: Restaurant category ratings from the Yelp dataset, showing similar category-item relationships as in Figure 2.

At each round $t = 1, \dots, T$, the agent needs to choose an item $a_t \in \mathcal{A}$ or a key-term $k_t \in \mathcal{K}$ and shows it to the user. If the agent chooses an item, it will receive an item reward:

$$r_{a_t,t} = \mu_{a_t} + \epsilon_t, \quad (1)$$

where μ_{a_t} is the expected reward of item a_t and ϵ_t is a random variable representing the random noise. Similarly, if the agent chooses a key-term, it will receive a key-term reward:

$$\tilde{r}_{k_t,t} = \tilde{\mu}_{k_t} + \tilde{\epsilon}_t, \quad (2)$$

where $\tilde{\mu}_{k_t}$ is the expected reward of key-term k_t and $\tilde{\epsilon}_t$ is a random variable representing the random noise. For ease of presentation, if the agent chooses an item $a_t \in \mathcal{A}$, we assume it also chooses $k_t = \emptyset$; if the agent chooses a key-term $k_t \in \mathcal{K}$, we assume it also chooses $a_t = \emptyset$. With a slight abuse of notation, we let $r_{\emptyset,t} = \tilde{r}_{\emptyset,t} = 0$. We define the real obtained reward of the agent at round t as:

$$R_{a_t,k_t,t} = r_{a_t,t} + \tilde{r}_{k_t,t}. \quad (3)$$

The goal of the agent is to maximize the expected cumulative rewards. Let $\sum_{t=1}^T \mathbb{E}[R_{a_t^*,k_t^*,t}]$ denote the maximum expected cumulative reward in T rounds, where (a_t^*, k_t^*) is the optimal action at round t , i.e., $\mathbb{E}[R_{a_t^*,k_t^*,t}] \geq \mathbb{E}[R_{a,k,t}]$, $\forall a \in \mathcal{A}, b = \emptyset$ or $\forall k \in \mathcal{K}, a = \emptyset$. The goal of the agent is to maximize the expected cumulative reward or, equivalently, to minimize the expected cumulative regret in T rounds:

$$Reg(T) = \sum_{t=1}^T \left(\mathbb{E}[R_{a_t^*,k_t^*,t}] - \mathbb{E}[R_{a_t,k_t,t}] \right). \quad (4)$$

3.2 Key-term Reward Revisited

Since we allow the agent to choose from key-term or item recommendations, the relationship between key-term and item rewards plays an important role in deciding which action to take at each round. Previous work [3] proposes that the reward of key-term k is the weighted average of the rewards of all related items:

$$\mathbb{E}[\tilde{r}_{k,t}] = \sum_{a \in \mathcal{A}} \frac{W_{a,k}}{\sum_{a' \in \mathcal{A}} W_{a',k}} \mathbb{E}[r_{a,t}], \quad k \in \mathcal{K}. \quad (5)$$

However, it is unrealistic for the recommender agent to know the exact weight $W_{a,k}$ between item a and key-term k in advance. It is also impractical to learn these weights online. As a result, previous work considers simplified binary weights for real applications, i.e., $W_{a,k} = 1$ if item a belongs to key-term k , otherwise $W_{a,k} = 0$. For example, if there are only 3 restaurants a_1, a_2, a_3 belonging to key-term k_1 , then the expected reward of key-term k_1 is calculated as $\mathbb{E}[\tilde{r}_{k_1,t}] = \frac{\mathbb{E}[r_{a_1,t}] + \mathbb{E}[r_{a_2,t}] + \mathbb{E}[r_{a_3,t}]}{3}$. We call this key-term reward calculation method the "simple average", and we study its correctness from a survey and data analysis on a real dataset.

We conduct the survey by asking the participant questions. Each participant is asked to first rate 3 restaurant categories (key-terms), "Hotpot", "Fine Dining", and "Coffee", and then 10 restaurants (items) randomly sampled from 50 restaurants for each category, on a scale of 1 to 5. The rating of category has an extra option "I have never tried", and participants who check this option will not be counted into the analysis of the corresponding category. In total, there are 55 participants. The true ratings of different categories and the ratings calculated by a simple average are shown in Figure 2. We find the simple average ratings are always smaller than the true ratings, which suggests that simple average is not an accurate way to calculate key-term rewards. One potential explanation is that people's preferences for a restaurant category are mainly determined by their preferences for some representative restaurants, so the simple average method underestimates the key-term reward as it treats all associated restaurants in the same category equally.

To verify this idea, we propose a new key-term reward calculation method called "top- α average", which takes the average reward of the top α fraction of restaurants as the key-term reward. We choose $\alpha = 20\%$ for better comparison with the simple average ratings, while $\alpha \in [0.1, 0.5]$ generally provides similar results. As shown in Figure 2, top-20% average ratings are much closer to the true ratings than simple average ratings.

Since the scale of the survey is relatively small, to further verify our findings, we run a **data analysis on the Yelp dataset** [36]. We choose 7 restaurant categories with over 10,000 reviews for each. We calculate the weighted average ratings of them, by taking the review counts of restaurants as the weights. This key-term reward can be interpreted as the average score of a review for a restaurant in that category. Notice that the recommender agent should not know the review counts before its recommendations, so weighted average ratings are only used for evaluation but not recommendation. We compare weighted average ratings with simple average ratings and top-50% average ratings in Figure 3. Top-50% average ratings are closer to weighted average ratings than simple average ratings, which is consistent with our findings from our survey.

3.3 Contextual Conversational Bandit

We next introduce a contextual version of the new conversational bandit problem, which allows us to more easily accommodate large item pools. The agent plays with a user with unknown feature $\theta^* \in \mathbb{R}^d$ for T rounds. At each round t , the agent observes contextual vector $\mathbf{x}_{a,t}$ for all items $a \in \mathcal{A}$ and contextual vector $\tilde{\mathbf{x}}_{k,t}$ for all key-terms $k \in \mathcal{K}$. It then needs to choose an item $a_t \in \mathcal{A}$ or a key-term $k_t \in \mathcal{K}$ and shows it to the user, as in the non-contextual problem. We follow LinUCB [35] in modeling the reward as a linear function of the context. If the agent chooses an item, it will receive an item reward:

$$r_{a_t,t} = \mathbf{x}_{a_t,t}^T \theta^* + \epsilon_t, \quad (6)$$

where $\mathbf{x}_{a_t,t}^T \theta^*$ is the expected reward of item a_t and ϵ_t is a random variable representing the random noise. Similarly, if the agent chooses a key-term, it will receive a key-term reward:

$$\tilde{r}_{k_t,t} = \tilde{\mathbf{x}}_{k_t,t}^T \theta^* + \tilde{\epsilon}_t, \quad (7)$$

where $\tilde{\mu}_{k_t}$ is the expected reward of key-term k_t and $\tilde{\epsilon}_t$ is a random variable representing the random noise. The definitions of the obtained reward $R_{a_t,k_t,t}$ and the regret $Reg(T)$ are the same as those in Section 3.1. The goal of the agent is still to minimize the expected regret $Reg(T)$, defined as in (4).

4 ALGORITHM & THEORETICAL ANALYSIS

In this section, we propose two algorithms, Hier-UCB and Hier-LinUCB, for the stochastic and contextual conversational bandit settings, respectively. As discussed in Section 3.2, the reward of a key-term is mainly determined by the rewards of the representative items, and the proposed algorithms take advantages of this finding and the hierarchical structure between key-terms and items to achieve more efficient learning.

4.1 Hier-UCB

Inspired by the well-known upper-confidence bound (UCB) solution algorithm for traditional multi-armed bandit formulations, we propose a Hierarchical UCB (Hier-UCB) algorithm described in Algorithm 1 for the stochastic setting of our conversational bandit problem. The algorithm maintains the empirical mean $\hat{\mu}_a, \tilde{\mu}_k$ and a confidence radius $\rho_a, \tilde{\rho}_k$ for each item $a \in \mathcal{A}$ and key-term $k \in \mathcal{K}$. The confidence radius is designed to be large if item a or key-term k is not chosen often (T_a or \tilde{T}_k , which denote the number of times item a or key-term k has been recommended, is small) as is typical in UCB-based algorithms, so as to encourage exploration of rarely sampled arms or key-terms.

At each round, the algorithm first calculates key-term \bar{k}^* with the highest UCB value among all key-terms, and then finds item \bar{a}^* with the highest UCB value from all associated items of key-term \bar{k}^* (line 6). It then decides whether to select an item or a key-term based on a switching condition (line 7): if the condition is not satisfied, which indicates the key-terms have not been sufficiently explored, it will recommend key-term \bar{k}^* (line 8) for one time, then recommend item \bar{a}^* (line 12); otherwise, it will only recommend item \bar{a}^* (line 12). Note that both key-term and item recommendations can incur regret. Intuitively, the switching condition ensures that an item is chosen if a conservative estimate of the best item reward exceeds a generous estimate of the best key-term reward: in other words, if

Algorithm 1 Hier-UCB

```

1: Input: graph  $(\mathcal{A}, \mathcal{K}, W)$ ,  $\gamma$ .
2: Init:  $T_a = 0, \tilde{T}_k = 0, \hat{\mu}_a = 1, \tilde{\mu}_k = 1$ .
3: while  $t \leq T$  do
4:   For each item  $a$ ,  $\rho_a = \sqrt{\frac{3 \ln t}{2T_a}}$ .
5:   For each key-term  $k$ ,  $\tilde{\rho}_k = \sqrt{\frac{3 \ln t}{2\tilde{T}_k}}$ .
6:    $\bar{k}^* = \operatorname{argmax}_{k \in \mathcal{K}} \tilde{\mu}_k + \tilde{\rho}_k$ ,  $\bar{a}^* = \operatorname{argmax}_{a \in \mathcal{A}} W_{a,\bar{k}^*} (\hat{\mu}_a + \rho_a)$ 
7:   if not  $(\hat{\mu}_{\bar{a}^*} - \gamma \cdot \rho_{\bar{a}^*} \geq \tilde{\mu}_{\bar{k}^*} + \gamma \cdot \tilde{\rho}_{\bar{k}^*})$  then
8:     Choose key-term  $\bar{k}^*$ , receive reward  $\tilde{r}_{\bar{k}^*,t}$ .
9:     Update  $\tilde{T}_{\bar{k}^*}, \tilde{\mu}_{\bar{k}^*}$ :  $\tilde{T}_{\bar{k}^*} = \tilde{T}_{\bar{k}^*} + 1, \tilde{\mu}_{\bar{k}^*} = \tilde{\mu}_{\bar{k}^*} + (\tilde{r}_{\bar{k}^*,t} - \tilde{\mu}_{\bar{k}^*}) / \tilde{T}_{\bar{k}^*}$ .
10:     $t = t + 1$ .
11:   end if
12:   Choose item  $\bar{a}^*$ , receive reward  $r_{\bar{a}^*,t}$ .
13:   Update  $T_{\bar{a}^*}, \hat{\mu}_{\bar{a}^*}$ :  $T_{\bar{a}^*} = T_{\bar{a}^*} + 1, \hat{\mu}_{\bar{a}^*} = \hat{\mu}_{\bar{a}^*} + (r_{\bar{a}^*,t} - \hat{\mu}_{\bar{a}^*}) / T_{\bar{a}^*}$ .
14:    $t = t + 1$ .
15: end while

```

the best item's reward exceeds the best key-term's reward with high probability. The parameter γ in the switching condition controls how easily the condition can be satisfied, and Theorem 4.2 gives the appropriate values of γ .

4.2 Hier-LinUCB

We propose a Hierarchical LinUCB (Hier-LinUCB) algorithm, described in Algorithm 2, for the contextual setting. It is a modified version of Hier-UCB based on the LinUCB [35] algorithm for contextual bandits. Hier-LinUCB maintains user feature estimate $\theta, \hat{\theta}$ and confidence radius $C_{a,t}, \tilde{C}_{k,t}$ for each item $a \in \mathcal{A}$ and key-term $k \in \mathcal{K}$. At each round, the algorithm first calculates the key-term \bar{k}^* with the highest UCB value among all key-terms (line 7) and then finds the item \bar{a}^* with the highest UCB value from all associated items of key-term \bar{k}^* (line 8). It then decides whether to only explore on item \bar{a}^* based on a switching condition (line 9): if the condition is false, which indicates the key-terms have not been sufficiently explored, it will recommend key-term \bar{k}^* for one time (line 10), then recommend item \bar{a}^* for one time (line 14). Otherwise, it will only choose item \bar{a}^* to recommend (line 14). Again, the switching condition controls when to recommend key-terms or items and is based on comparing a conservative estimate of the best item reward with a generous estimate of the key-term reward. The parameter γ again controls how conservative or generous these estimates are.

4.3 Regret Analysis

In this section, we analyze the regret (4) of Hier-UCB in the stochastic conversational setting. Let $\mathcal{A}_k = \{a \in \mathcal{A}, W_{a,k} \neq 0\}$ denote the set of all associated items for key-term k , k^* denote the key-term with the highest expected reward, and $a_k^* = \operatorname{argmax}_{a \in \mathcal{A}_k} \mu_a$. Note that \bar{k}^* is the key-term with the current highest UCB value and may not be k^* . In order to derive the regret bound, we assume that $a_{\bar{k}^*}^* = \operatorname{argmax}_a \mathbb{E}[r_{a,t}]$, i.e., the best item in the key-term with the highest reward also has the highest reward among all items. This assumption is consistent with our observation in Section 3.2 that the key-term rewards are mainly affected by the rewards of the best

Algorithm 2 Hier-LinUCB

```

1: Input: graph  $(\mathcal{A}, \mathcal{K}, W)$ ,  $\gamma$ .
2: Init:  $\tilde{M} = I, \tilde{b} = 0, M = I, \mathbf{b} = 0$ .
3: while  $t \leq T$  do
4:    $\tilde{\theta} = \tilde{M}^{-1}\tilde{b}, \theta = M^{-1}\mathbf{b}$ .
5:   For each item  $a$ ,  $C_{a,t} = \alpha_t \|\tilde{x}_{k,t}\|_{\tilde{M}^{-1}}$ .
6:   For each key-term  $k$ ,  $\tilde{C}_{k,t} = \alpha_t \|\tilde{x}_{k,t}\|_{\tilde{M}^{-1}}$ .
7:    $\tilde{k}^* = \operatorname{argmax}_{k \in \mathcal{K}} \tilde{x}_{k,t}^T \tilde{\theta} + \tilde{C}_{k,t}$ .
8:    $\tilde{a}^* = \operatorname{argmax}_{a \in \mathcal{A}} W_{a,\tilde{k}^*} \left( x_{a,t}^T \theta + C_{a,t} \right)$ .
9:   if not  $(x_{\tilde{a}^*,t}^T \theta - \gamma C_{\tilde{a}^*,t} \geq \tilde{x}_{\tilde{k}^*,t}^T \tilde{\theta} + \gamma \tilde{C}_{\tilde{k}^*,t})$  then
10:    Choose key-term  $\tilde{k}^*$ , receive reward  $\tilde{r}_{\tilde{k}^*,t}$ .
11:    Update  $\tilde{M}, \tilde{b}$ :  $\tilde{M} = \tilde{M} + \tilde{x}_{\tilde{k}^*,t} \tilde{x}_{\tilde{k}^*,t}^T, \tilde{b} = \tilde{b} + \tilde{x}_{\tilde{k}^*,t} \tilde{r}_{\tilde{k}^*,t}^T$ .
12:     $t = t + 1$ .
13:   end if
14:   Choose item  $\tilde{a}^*$ , receive reward  $\tilde{r}_{\tilde{a}^*,t}$ .
15:   Update  $M, \mathbf{b}$ :  $M = M + x_{\tilde{a}^*,t} x_{\tilde{a}^*,t}^T, \mathbf{b} = \mathbf{b} + x_{\tilde{a}^*,t} \tilde{r}_{\tilde{a}^*,t}^T$ .
16:    $t = t + 1$ .
17: end while

```

associated items. It is also quite general compared to the first-order dominance assumption used to design previous hierarchical bandit algorithms [37], which requires the i^{th} best item in key-term k^* has larger reward than the i^{th} best item of key-term k , for any i, k . There are four cases in total that can cause regret in a given round:

- (1) \tilde{k}^* is sub-optimal and switching condition is not satisfied
- (2) \tilde{k}^* is sub-optimal and switching condition is satisfied
- (3) \tilde{k}^* is optimal and switching condition is not satisfied
- (4) \tilde{k}^* is optimal and switching condition is satisfied

We consider the regret decomposed into these four cases separately.

4.3.1 Sub-optimal \tilde{k}^* , no switching. For sub-optimal key-term \tilde{k}^* , when the switching condition is not satisfied, we recommend key-term \tilde{k}^* and item \tilde{a}^* . In this case, we only need to consider that

$$\exists \tilde{k}^* \neq k^*, \tilde{\mu}_{\tilde{k}^*} + \tilde{\rho}_{\tilde{k}^*} \geq \tilde{\mu}_{k^*} + \tilde{\rho}_{k^*}. \quad (8)$$

By applying Hoeffding's inequality on $\tilde{\mu}_{\tilde{k}^*}$, we have $\tilde{\mu}_{\tilde{k}^*} - \tilde{\rho}_{\tilde{k}^*} \leq \tilde{\mu}_{\tilde{k}^*} \leq \tilde{\mu}_{\tilde{k}^*} + \tilde{\rho}_{\tilde{k}^*}$ with high probability (at least $1 - 2|\mathcal{K}|t^{-2}$). Then (8) becomes

$$\exists \tilde{k}^* \neq k^*, 2\tilde{\rho}_{\tilde{k}^*} \geq \tilde{\mu}_{k^*} - \tilde{\mu}_{\tilde{k}^*}. \quad (9)$$

Substituting $\tilde{\rho}_{\tilde{k}^*}$ with its definition $\tilde{\rho}_{\tilde{k}^*} = \sqrt{\frac{3 \ln t}{2\tilde{T}_{\tilde{k}^*}}}$, we have

$$\mathbb{E}[\tilde{T}_{\tilde{k}^*}] \leq \frac{6 \ln t}{(\tilde{\mu}_{k^*} - \tilde{\mu}_{\tilde{k}^*})^2}. \quad (10)$$

The regret caused by selecting sub-optimal key-terms and items is bounded as

$$\text{Reg}_1(T) \leq \sum_{k \neq k^*} \mathbb{E}[\tilde{T}_k] \cdot (\mu_{a^*} - \tilde{\mu}_k + 1). \quad (11)$$

4.3.2 Sub-optimal \tilde{k}^* , switching. By applying Hoeffding's inequality on $\tilde{\mu}_k$ and $\tilde{\mu}_a$, we have $\tilde{\mu}_k - \tilde{\rho}_k \leq \tilde{\mu}_k + \tilde{\rho}_k$ and $\mu_a - \rho_a \leq \tilde{\mu}_a \leq \mu_a + \rho_a$ with high probability. If the switching condition is

satisfied, we have

$$\mu_{\tilde{a}^*} + (1 - \gamma) \cdot \rho_{\tilde{a}^*} \geq \tilde{\mu}_{\tilde{k}^*} + (\gamma - 1) \cdot \tilde{\rho}_{\tilde{k}^*}, \quad (12)$$

$$\rho_{\tilde{a}^*} + \tilde{\rho}_{\tilde{k}^*} \leq \frac{1}{\gamma - 1} (\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*}). \quad (13)$$

We also have $T_{\tilde{a}^*} \leq \tilde{T}_{\tilde{k}^*} \leq \frac{6 \ln t}{(\tilde{\mu}_{k^*} - \tilde{\mu}_{\tilde{k}^*})^2}$ at the first time that the switching condition is satisfied, which gives us

$$\rho_{\tilde{a}^*} + \tilde{\rho}_{\tilde{k}^*} \geq \tilde{\mu}_{k^*} - \tilde{\mu}_{\tilde{k}^*}. \quad (14)$$

If $\gamma > \frac{\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*}}{\tilde{\mu}_{k^*} - \tilde{\mu}_{\tilde{k}^*}} + 1$, (13) and (14) will never be true at the same time.

Thus, by taking $\gamma \geq \max_{k \neq k^*} \frac{\mu_{\tilde{a}^*} - \tilde{\mu}_k}{\tilde{\mu}_{k^*} - \tilde{\mu}_k} + 1$, the switching condition will never be satisfied and the regret caused by this part is

$$\text{Reg}_2(T) \leq O(1). \quad (15)$$

4.3.3 Optimal \tilde{k}^* , no switching. Similar to the previous case, when the switching condition is not satisfied, we have

$$\mu_{\tilde{a}^*} - (\gamma + 1) \cdot \rho_{\tilde{a}^*} < \tilde{\mu}_{\tilde{k}^*} + (\gamma + 1) \cdot \tilde{\rho}_{\tilde{k}^*}, \quad (16)$$

$$\rho_{\tilde{a}^*} + \tilde{\rho}_{\tilde{k}^*} > \frac{1}{\gamma + 1} (\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*}). \quad (17)$$

We then bound $\mathbb{E}[\tilde{T}_{\tilde{k}^*}]$ with Lemma 1.

LEMMA 4.1.

$$\mathbb{E}[\tilde{T}_{\tilde{k}^*}] \leq \frac{6(\gamma + 1)^2 \ln t}{(\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*})^2} + \sum_{a \in \mathcal{A}_{\tilde{k}^*}, a \neq \tilde{a}^*} \frac{6 \ln t}{(\mu_{\tilde{a}^*} - \mu_a)^2}. \quad (18)$$

PROOF. For the sub-optimal key-term \tilde{k}^* , before the switching condition is satisfied, we have

$$T_{\tilde{a}^*} \geq \tilde{T}_{\tilde{k}^*} - \sum_{a \in \mathcal{A}_{\tilde{k}^*}, a \neq \tilde{a}^*} T_a \geq \tilde{T}_{\tilde{k}^*} - \sum_{a \in \mathcal{A}_{\tilde{k}^*}, a \neq \tilde{a}^*} \frac{6 \ln t}{(\mu_{\tilde{a}^*} - \mu_a)^2}. \quad (19)$$

Let us consider a threshold for $\tilde{T}_{\tilde{k}^*}$ and discuss two cases. If $\tilde{T}_{\tilde{k}^*} > \frac{6(\gamma+1)^2 \ln t}{(\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*})^2} + \sum_{a \in \mathcal{A}_{\tilde{k}^*}, a \neq \tilde{a}^*} \frac{6 \ln t}{(\mu_{\tilde{a}^*} - \mu_a)^2}$, with Eq.(19), we have

$$T_{\tilde{a}^*} \geq \frac{6(\gamma + 1)^2 \ln t}{(\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*})^2}. \quad (20)$$

From Eq.(17), we can also derive an upper bound for $T_{\tilde{a}^*}$:

$$T_{\tilde{a}^*} < \left(\frac{\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*}}{(\gamma + 1)\sqrt{3 \ln t/2}} - \frac{1}{\sqrt{\tilde{T}_{\tilde{k}^*}}} \right)^{-2}. \quad (21)$$

However, since $\tilde{T}_{\tilde{k}^*} > \frac{6(\gamma+1)^2 \ln t}{(\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*})^2} + \sum_{a \in \mathcal{A}_{\tilde{k}^*}, a \neq \tilde{a}^*} \frac{6 \ln t}{(\mu_{\tilde{a}^*} - \mu_a)^2}$, Eq.(21) becomes

$$T_{\tilde{a}^*} < \frac{6(\gamma + 1)^2 \ln t}{(\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*})^2} + \sum_{a \in \mathcal{A}_{\tilde{k}^*}, a \neq \tilde{a}^*} \frac{6 \ln t}{(\mu_{\tilde{a}^*} - \mu_a)^2}, \quad (22)$$

which is contrary to Eq.(19). Thus, we only need to consider the case that $\tilde{T}_{\tilde{k}^*} \leq \frac{6(\gamma+1)^2 \ln t}{(\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*})^2} + \sum_{a \in \mathcal{A}_{\tilde{k}^*}, a \neq \tilde{a}^*} \frac{6 \ln t}{(\mu_{\tilde{a}^*} - \mu_a)^2}$, which provides the upper bound

$$\mathbb{E}[\tilde{T}_{\tilde{k}^*}] \leq \frac{6(\gamma + 1)^2 \ln t}{(\mu_{\tilde{a}^*} - \tilde{\mu}_{\tilde{k}^*})^2} + \sum_{a \in \mathcal{A}_{\tilde{k}^*}, a \neq \tilde{a}^*} \frac{6 \ln t}{(\mu_{\tilde{a}^*} - \mu_a)^2}. \quad (23)$$

□

The regret caused by this part is bounded as

$$\text{Reg}_3(T) \leq \mathbb{E}[\tilde{T}_{k^*}] \cdot (\mu_{a^*} - \tilde{\mu}_{k^*} + 1). \quad (24)$$

4.3.4 Optimal \tilde{k}^* , switching. This part can be considered as the regret of the traditional UCB algorithm on the arm set \mathcal{A}_{k^*} .

$$\text{Reg}_4(T) \leq \sum_{a \in \mathcal{A}_{k^*}, a \neq a^*} \frac{6 \ln t}{\mu_{a^*} - \mu_a}. \quad (25)$$

4.3.5 Overall regret. We define $\Delta_a = \mu_{a^*} - \mu_a$, $\tilde{\Delta}_k = \tilde{\mu}_{k^*} - \tilde{\mu}_k$, $\Delta_k^* = \mu_{a_k^*} - \tilde{\mu}_k$. The overall regret of Hier-UCB is given below.

THEOREM 4.2. Assume that $\gamma > \max_{k \neq k^*} \frac{\tilde{\Delta}_k}{\Delta_k^*} + 1$, Hier-UCB has the following regret upper bound

$$\begin{aligned} \text{Reg}(T) &= \sum_{i=1}^4 \text{Reg}_i(T) \\ &\leq 6 \left[\sum_{k \neq k^*} \frac{\Delta_k^* + 1}{\tilde{\Delta}_k^2} + \sum_{a \in \mathcal{A}_{k^*}, a \neq a^*} \frac{\Delta_{k^*}^* + \Delta_a + 1}{\Delta_a^2} \right. \\ &\quad \left. + \sum_{a \in \mathcal{A}_{k^*}, a \neq a^*} \frac{(\gamma + 1)^2 (\Delta_{k^*}^* + 1)}{(\Delta_{k^*}^*)^2} \right] \ln T. \end{aligned} \quad (26)$$

Remark. Looking at the above distribution dependent bound, we have the $O((|\mathcal{K}| + |\mathcal{A}_{k^*}|) \ln T)$ regret, which is asymptotically tight in T . Since applying traditional UCB algorithms to item set \mathcal{A} leads to a regret of $O(|\mathcal{A}| \ln T)$, we improve the regret bound by reducing $|\mathcal{A}| \ln T$ to $(|\mathcal{K}| + |\mathcal{A}_{k^*}|) \ln T$, owing to the utilization of hierarchical structure between key-terms and items. Notice that ConUCB [3] in the stochastic setting will lead to the regret of $O(|\mathcal{A}| + \ln T)$ even without considering the regret caused by key-term asking. Thus, our proposed algorithms prevail over the others when the size of the item pool $|\mathcal{A}|$ is substantially large. It is possible to analyze the regret of Hier-LinUCB following similar steps of regret decomposition for Hier-UCB, but it is more involved due to users' feature estimation and we left it for the future work.

5 EXPERIMENTS

We conduct experiments on both synthetic and real-world data to validate our proposed algorithms.

5.1 Experimental Setup

In this section, we discuss our experimental setup. We first introduce two baseline algorithms, then describe the metrics used for evaluation. We also list three research questions that we would like to answer.

5.1.1 Baselines. We compare our algorithm with two baseline algorithms that do not properly model the relationship between user preferences to key-terms and items.

- **LinUCB** [35]: A standard non-conversational contextual bandit algorithm. This comparison demonstrates the benefits of Hier-LinUCB's ability to select key-terms.
- **ConUCB** [3]: A recent conversational bandit algorithm proposed in the contextual setting that employs a fixed frequency of choosing key-terms. This comparison demonstrates Hier-LinUCB's ability to adaptively select key-terms

or items depending on each user, accounting for the cost of choosing key-terms.

5.1.2 Metrics. We use the cumulative regret in Eq.(4) to measure the performance of algorithms. As a common metric for bandit problems, it captures the gap between the reward of always choosing the optimal action and the reward of the action chosen by a specific algorithm. Besides, in Section 5.3, we show the average reward over iterations, $\frac{1}{N} \sum_{t=1}^N R_{a_t, k_t, t}$.

5.1.3 Research Questions. To validate our approach, we design the following research questions:

- RQ1. In a real-world setting, what relationship can be observed between user preferences to key-terms and items?
- RQ2. Can our algorithms achieve less regret than baseline algorithms by leveraging the relationship and the hierarchical structure between key-terms and items?
- RQ3. Can our algorithms adapt to switch between key-term and item recommendations more flexibly than previous work?

The extensive study in Section 3.2 can answer RQ1. We observe that users' preference on a specific key-term is mainly affected by their preference on its representative items.

5.2 Synthetic Data

We consider a stochastic conversational bandit setting with 10 key-terms and 100 items. We create an item pool $\mathcal{A} = \{a_1, a_2, \dots, a_{100}\}$ and a key-term pool $\mathcal{K} = \{k_1, k_2, \dots, k_{10}\}$. We assume each item is only associated with one key-term, and let a_1, a_2, \dots, a_{10} associate with k_1 , $a_{11}, a_{12}, \dots, a_{20}$ associate with k_2 , etc. We set each item a_i 's expected reward as $\mu_{a_i} = i/100$. We assume that the reward of key-term k is equal to the discounted reward of the best associated item, i.e., $\mathbb{E}[\tilde{r}_{k,t}] = \lambda \cdot \max_a (W_{a,k} \mathbb{E}[r_{a,t}])$, where $\lambda \leq 1$ is a discount factor. Note that it is a special form of the general assumption required by Hier-UCB in Section 4.3. We choose $\lambda = 0.5$, then for each key-term k_i , $\tilde{\mu}_{k_i} = i/2$. The rewards of items and key-terms are generated from Bernoulli distributions. We run Hier-UCB, Hier-LinUCB and traditional UCB algorithms on these items and key-terms. We set $\gamma = 1$ for both Hier-UCB and Hier-LinUCB, and $\alpha_t = 1$ for Hier-LinUCB. For Hier-LinUCB, as it is designed for the contextual setting, we consider 100-dimensional one-hot contextual vectors for all items, and the algorithm will try to estimate a 100-dimensional feature vector. We repeat the experiment 50 times and show the average cumulative regret with 95% confidence interval.

Answer to RQ2. Figure 4 shows that Hier-UCB achieves less regret than traditional UCB and Hier-LinUCB in the stochastic setting, owing to the strategic balance between key-term and item recommendations: it spends around 2000 iterations on key-term learning, then switch to explore on a smaller item pool, thus achieves much less regret than UCB. This also validates our regret bound. As Hier-UCB is specifically designed for the stochastic setting with moderate-size item pools, it also outperforms Hier-LinUCB, which is designed for the contextual setting: the least square estimation of feature vectors in Hier-LinUCB may incur higher regrets than directly estimating the expected reward of each arm.

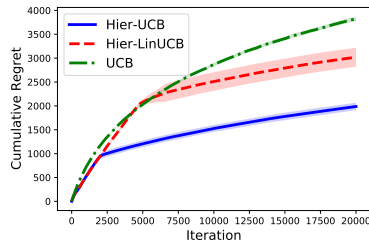


Figure 4: The result validates the regret bound of our algorithm in the stochastic setting.

5.3 Real-world Datasets

5.3.1 Dataset Description. In this section, we evaluate Hier-LinUCB on three real-world datasets, and compare its performance with LinUCB and ConUCB. The original datasets are released by Yelp¹, MovieLens², and LastFM³. In our experiments, we use the public source data preprocessed by [3] for the Yelp dataset and [8] for the MovieLens and LastFM datasets. The Yelp dataset has 1000 key-terms, 5000 items and 200 users, where the key-terms are restaurant categories (e.g., “Burgers”) and the items are restaurants. The LastFM dataset has 2726 key-terms, 2000 items and 100 users, where the key-terms are artist tags (e.g., “Rock”) and the items are artists. The MovieLens dataset has 5585 key-terms, 2000 items and 200 users, where the key-terms are movie tags (e.g., “Action”) and the items are movies. We aim to show our algorithms perform well on the Yelp and the LastFM datasets and that, although MovieLens has more key-terms than items, which may reduce the benefit of key-term asking, our proposed algorithm can still achieve comparable performance to the previous baselines. All datasets provide the hierarchical relationship between key-terms and items, the contextual vectors of key-terms and items, and the feature vectors of all users (detailed data generation methods can be found in Section 5.1 of [3] and Section 4.3 of [8]).

For Hier-LinUCB, we set $\gamma = 0.5$, $\alpha_t = 1$ on Yelp and $\gamma = 0.5$, $\alpha_t = 0.25$ on LastFM and MovieLens. For LinUCB and ConUCB, we generally follow the original experimental settings in [3, 8], while some experimental configurations are changed to fit our new scenario, which will be discussed in the next section. We choose $b(t) = 10\lfloor \log(t) \rfloor$ as the conversation frequency function for ConUCB, which leads to the smallest regret in their experiments.

5.3.2 Experimental Settings.

Key-term reward. We assume that the reward of key-term k is equal to the discounted reward of the best associated item, i.e., $\mathbb{E}[\tilde{r}_{k,t}] = \lambda \cdot \max_a (W_{a,k} \mathbb{E}[r_{a,t}])$, where $\lambda \leq 1$ is a discount factor. It is a special form of the general assumption in Section 4.3. We set $\lambda = 0.5$ and re-calculate the expected rewards and feature vectors of key-terms.

Item pool size. ConUCB [3] randomly chooses an arm pool of 50 items at each round in their original experiments. However,

such an arm pool is too small compared to the whole arm pool with thousands of items, and it is very likely to exclude many good items with high rewards. Instead, we use the full item pool in our experiments, which also makes the learning problem more challenging. To encourage exploration, we set $\alpha_t = 1$ for all baseline algorithms. We observe higher cumulative regrets than the original experiments, which is consistent with Figure 2 in [3].

5.3.3 Results. In our experiments, we compare the performance of Hier-LinUCB with LinUCB and ConUCB. We also conduct a case study of two users in the Yelp dataset with ConUCB.

Answer to RQ2. We show the cumulative regrets and averaged rewards of different algorithms in Figures 5 and 6. Notice that the system recommends one key-term or item to one user at each interaction; when the averaged reward almost converges after 20000 interactions for Yelp, the average number of key-term asking and item recommendations per user is 100, which is reasonable after a few days or weeks. Figure 5 shows that Hier-LinUCB achieves lower regrets than LinUCB and ConUCB after iteration ends. At the very beginning, due to more exploration on key-terms, Hier-LinUCB may not achieve lower regrets than LinUCB and ConUCB. However, as long as it learns the correct key-terms, it will converge to the optimal actions much faster than others. We observe that the regret of ConUCB periodically increases, which comes from key term conversations predefined by the conversation frequency function [3]. To better observe the early state rewards and understand the behavior of different algorithms in cold-start recommender systems, we further compare the averaged reward with the logarithmic x-axis in Figure 6. The results are shown in Figure 6. Although LinUCB can achieve good rewards at the early stage, it may not find the optimal item eventually compared to ConUCB and Hier-LinUCB, showing the necessity of key-term asking. Besides, the averaged rewards of ConUCB and Hier-LinUCB converge to the similar levels once they successfully identify the best key-terms. However, in Figures 6a and 6b, Hier-LinUCB searches faster than ConUCB and has higher rewards for the early iterations, since the number of key-terms is similar to (or less than) the number of items for the Yelp and LastFM datasets. When the number of key-terms is larger than the number of items (MovieLens), as shown in Figure 6c, Hier-LinUCB has to explore more at first (resulting in initially lower rewards), before eventually achieving higher reward than ConUCB. Notice that the former case (i.e., number of key-terms is limited) is common in practice, since the key-terms of the items need to be labeled carefully, which usually leads to a lot of human efforts.

Answer to RQ3. We pick two users from the Yelp dataset and monitor their regrets in Figure 7. We find Hier-LinUCB has different switch points for different users: before the switch point, it explores more on key-terms; after the switch point, it only chooses items and the regret converges quickly. By contrast, ConUCB has a fixed key-term conversation period as highlighted in red. Hier-LinUCB thus enables more flexible and personalized switching between key-term asking and item recommendations. As discussed before, the key-term exploration of Hier-LinUCB is based on the switching condition, while the key-term exploration of ConUCB is based on a fixed conversation frequency function. Hier-LinUCB treats User

¹<https://www.yelp.com/dataset/>

²<https://www.grouplens.com>

³<https://www.lastfm.com>

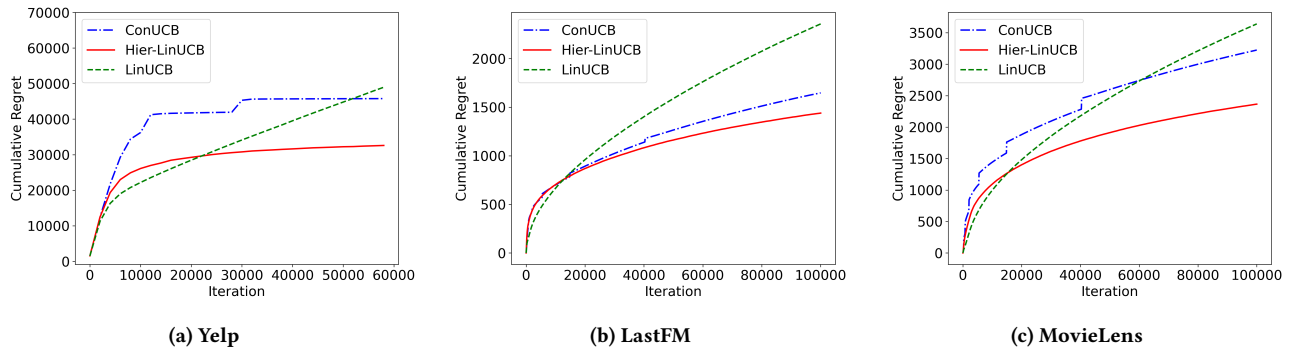


Figure 5: Comparison of the cumulative regrets on real-world datasets using Hier-LinUCB, LinUCB and ConUCB. Hier-LinUCB has the lowest converged regret after more than 30000 iterations. The improvement is most apparent on Yelp, likely because the MovieLens and LastFM datasets have a large number of key terms, requiring more key-term exploration.

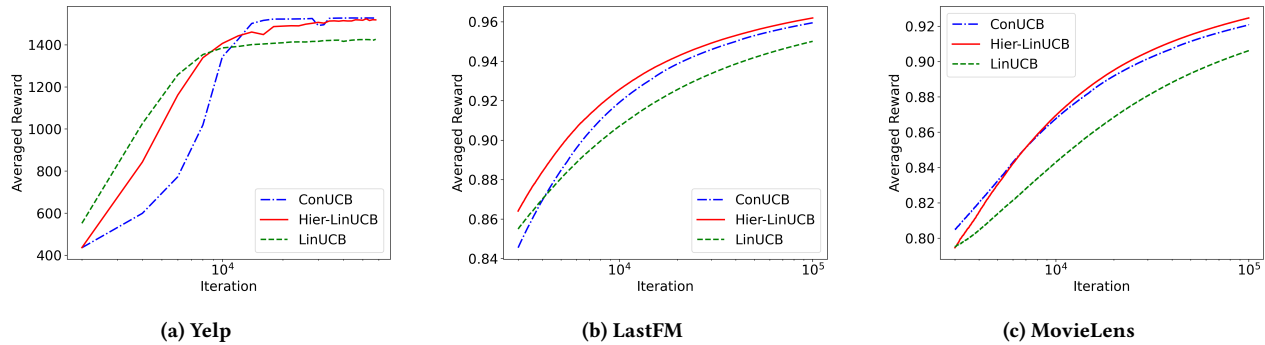


Figure 6: Comparison of the averaged rewards on real-world datasets using Hier-LinUCB, LinUCB and ConUCB. The averaged reward of Hier-LinUCB grows faster than ConUCB on Yelp and LastFM, while slightly slower than ConUCB on MovieLens at the early stage. This is likely due to the large number of key-terms for MovieLens.

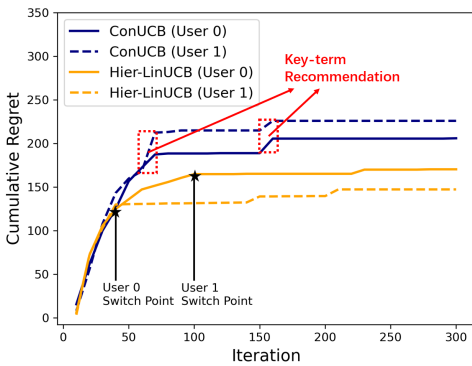


Figure 7: A case study of two users on the Yelp dataset. ConUCB has fixed exploration on key-terms (i.e., switching from item recommendations to key-term asking) for user 0 and user 1 (marked by red color), while Hier-LinUCB enables more flexible and personalized switching between key-term asking and item recommendations.

1 and User 0 differently, while ConUCB uses a fixed exploration policy and results in worse regrets.

6 CONCLUSION

To avoid the excessive conversational interactions incurred by the pre-defined conversation frequency of existing conversational bandit algorithms, we formulate a new conversational bandit problem that allows the recommender system to choose either a key-term or an item to recommend at each round. To balancing the new exploration-exploitation (EE) trade-off between key-term asking and item recommendation, we need accurately understand the relationship between key-term and item rewards. We study such relationship based on a survey and analysis on a real-world dataset. We observe that key-term rewards are mainly affected by rewards of representative items. We propose two bandit algorithms based on this observation and the hierarchical structure between key-terms and items. We prove the theoretical regret bound of our algorithm and validate them on synthetic and real-world data.

ACKNOWLEDGMENTS

This work was supported by ONR Grant N000142112128.

REFERENCES

- [1] Yashar Deldjoo, Mihai Gabriel Constantin, Hamid Eghbal-Zadeh, Bogdan Ionescu, Markus Schedl, and Paolo Cremonesi. Audio-visual encoding of multimedia content for enhancing movie recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 455–459, 2018.
- [2] Bin Liu, Deguang Kong, Lei Cen, Neil Zhenqiang Gong, Hongxia Jin, and Hui Xiong. Personalized mobile app recommendation: Reconciling app functionality and user privacy preference. In *Proceedings of the eighth ACM international conference on web search and data mining*, pages 315–324, 2015.
- [3] Xiaoying Zhang, Hong Xie, Hang Li, and John CS Lui. Conversational contextual bandit: Algorithm and application. In *Proceedings of The Web Conference 2020*, pages 662–672, 2020.
- [4] Bo Liu, Ying Wei, Yu Zhang, Zhixian Yan, and Qiang Yang. Transferable contextual bandit for cross-domain recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.
- [5] Jeffrey Dalton, Victor Ajayi, and Richard Main. Vote goat: Conversational movie recommendation. In *The 41st international acm sigir conference on research & development in information retrieval*, pages 1285–1288, 2018.
- [6] Nicola Sardella, Claudio Biancalana, Alessandro Micarelli, and Giuseppe Sansonetti. An approach to conversational recommendation of restaurants. In *International Conference on Human-Computer Interaction*, pages 123–130. Springer, 2019.
- [7] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. Towards conversational recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 815–824, 2016.
- [8] Zhihui Xie, Tong Yu, Canzhe Zhao, and Shuai Li. Comparison-based conversational recommender system with relative bandit feedback. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1400–1409, 2021.
- [9] Canzhe Zhao, Tong Yu, Zhihui Xie, and Shuai Li. Knowledge-aware conversational preference elicitation with bandit feedback. In *Proceedings of the ACM Web Conference 2022*, pages 483–492, 2022.
- [10] Junda Wu, Canzhe Zhao, Tong Yu, Jingyang Li, and Shuai Li. Clustering of conversational bandits for user preference learning and elicitation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2129–2139, 2021.
- [11] Shijun Li, Wenqiang Lei, Qingyun Wu, Xiangnan He, Peng Jiang, and Tat-Seng Chua. Seamlessly unifying attributes and items: Conversational recommendation for cold-start users. *ACM Transactions on Information Systems (TOIS)*, 39(4):1–29, 2021.
- [12] Xiaowei Jia, Handong Zhao, Zhe Lin, Ajinkya Kale, and Vipin Kumar. Personalized image retrieval with sparse graph representation learning. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2735–2743, 2020.
- [13] Runzhe Wan, Lin Ge, and Rui Song. Metadata-based multi-task bandits with bayesian hierarchical models. *Advances in Neural Information Processing Systems*, 34, 2021.
- [14] Joey Hong, Branislav Kveton, Manzil Zaheer, and Mohammad Ghavamzadeh. Hierarchical bayesian bandits. *arXiv preprint arXiv:2111.06929*, 2021.
- [15] Branislav Kveton, Mikhail Konobeev, Manzil Zaheer, Chih-wei Hsu, Martin Mladenov, Craig Boutilier, and Csaba Szepesvári. Meta-thompson sampling. In *International Conference on Machine Learning*, pages 5884–5893. PMLR, 2021.
- [16] Joey Hong, Branislav Kveton, Sumeet Katariya, Manzil Zaheer, and Mohammad Ghavamzadeh. Deep hierarchy in bandits. *arXiv preprint arXiv:2202.01454*, 2022.
- [17] Soumya Basu, Branislav Kveton, Manzil Zaheer, and Csaba Szepesvári. No regrets for learning the prior in bandits. *Advances in Neural Information Processing Systems*, 34, 2021.
- [18] Konstantina Christakopoulou, Alex Beutel, Rui Li, Sagar Jain, and Ed H Chi. Q&R: A two-stage approach toward interactive recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 139–148, 2018.
- [19] Bilih Priyogi. Preference elicitation strategy for conversational recommender system. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 824–825, 2019.
- [20] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th acm international conference on information and knowledge management*, pages 177–186, 2018.
- [21] Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. Towards knowledge-based recommender dialog system. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1803–1813, 2019.
- [22] Jie Zou, Yifan Chen, and Evangelos Kanoulas. Towards question-based recommender systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 881–890, 2020.
- [23] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. Towards deep conversational recommendations. *Advances in neural information processing systems*, 31, 2018.
- [24] Zeming Liu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, and Ting Liu. Towards conversational recommendation over multi-type dialogs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1036–1049, 2020.
- [25] Tong Yu, Yilin Shen, and Hongxia Jin. A visual dialog augmented interactive recommender system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 157–165, 2019.
- [26] Ruiyi Zhang, Tong Yu, Yilin Shen, Hongxia Jin, and Changyou Chen. Text-based interactive recommendation via constraint-augmented reinforcement learning. *Advances in neural information processing systems*, 32, 2019.
- [27] Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1006–1014, 2020.
- [28] Pan Xu, Zheng Wen, Handong Zhao, and Quanquan Gu. Neural contextual bandits with deep representation and shallow exploration. In *The Tenth International Conference on Learning Representations*, 2022.
- [29] Kun Zhou, Yuanhang Zhou, Wayne Xin Zhao, Xiaoke Wang, and Ji-Rong Wen. Towards topic-guided conversational recommender system. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4128–4139, 2020.
- [30] Yang Deng, Yaliang Li, Fei Sun, Bolin Ding, and Wai Lam. Unified conversational recommendation policy learning via graph-based reinforcement learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1431–1441, 2021.
- [31] Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 304–312, 2020.
- [32] Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. Interactive path reasoning on graph for conversational recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2073–2083, 2020.
- [33] Yueming Sun and Yi Zhang. Conversational recommender system. In *The 41st international acm sigir conference on research & development in information retrieval*, pages 235–244, 2018.
- [34] Kerui Xu, Jingxuan Yang, Jun Xu, Sheng Gao, Jun Guo, and Ji-Rong Wen. Adapting user preference to online feedback in multi-round conversational recommendation. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 364–372, 2021.
- [35] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670, 2010.
- [36] Yelp dataset. <https://www.yelp.com/dataset>. [Online].
- [37] Matthieu Jedor, Vianney Perchet, and Jonathan Louedec. Categorized bandits. *Advances in Neural Information Processing Systems*, 32, 2019.