Conversational Contextual Bandit: Algorithm and Application

Xiaoying Zhang*
CSE, The Chinese University of Hong Kong
xyzhang@cse.cuhk.edu.hk

Hong Xie
College of Computer Science, Chongqing University
xiehong2018@cqu.edu.cn

Hang Li
AI Lab, Bytedance
lihang.lh@bytedance.com

John C.S. Lui
CSE, The Chinese University of Hong Kong
cslui@cse.cuhk.edu.hk

ABSTRACT

Contextual bandit algorithms provide principled online learning solutions to balance the exploitation-exploration trade-off in various applications such as recommender systems. However, the learning speed of the traditional contextual bandit algorithms is often slow due to the need for extensive exploration. This poses a critical issue in applications like recommender systems, since users may need to provide feedbacks on a lot of uninterested items. To accelerate the learning speed, we generalize contextual bandit to *conversational contextual bandit*. Conversational contextual bandit leverages not only behavioral feedbacks on arms (e.g., articles in news recommendation), but also occasional conversational feedbacks on key-terms from the user. Here, a key-term can relate to a subset of arms, for example, a category of articles in news recommendation. We then design the Conversational UCB algorithm (ConUCB) to address two challenges in conversational contextual bandit: (1) which key-terms to select to conduct conversation, (2) how to leverage conversational feedbacks to accelerate the speed of bandit learning. We theoretically prove that ConUCB can achieve a smaller regret upper bound than the traditional contextual bandit algorithm LinUCB, which implies a faster learning speed. Experiments on synthetic data, as well as real datasets from Yelp and Toutiao, demonstrate the efficacy of the ConUCB algorithm.

ACM Reference Format:
Xiaoying Zhang, Hong Xie, Hang Li, and John C.S. Lui. 2020. Conversational Contextual Bandit: Algorithm and Application. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380148

1 INTRODUCTION

Contextual bandit serves as an invaluable tool for enhancing performance of a system through learning from interactions with the user while making trade-off between exploitation and exploration [1, 9, 11, 13]. The contextual bandit algorithms have been applied to recommender systems, for instance, to adaptively learn users' preference on items. In this application, the items are taken as the arms in contextual bandit, and the contextual vector of each arm/item contains the observed information about the user and the item at the time. The recommender system equipped with a contextual bandit algorithm sequentially recommends items to the user. The user provides a feedback (e.g., click) on the recommended item each round, which is viewed as a reward. The goal of the contextual bandit algorithm is to learn an item recommendation (arm selection) strategy to optimize the user's feedbacks in the long run (cumulative rewards), via utilizing the information of the user and items (contextual vectors) as well as the user's feedbacks (rewards). In general, the algorithm needs to make a trade-off between exploitation (i.e., leveraging the user's preference already known) and exploration (i.e., revealing the user's preference still unknown).

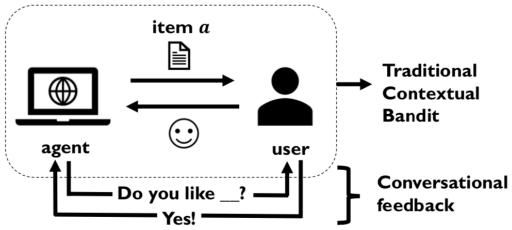


Figure 1: Conversational contextual bandit. The part in dashed box corresponds to traditional contextual bandit.

One shortcoming of the traditional contextual bandit algorithms [1, 9, 11, 13] lies in their slow learning speed. This is because they need to perform extensive exploration in order to collect sufficient feedbacks. For applications like recommender systems, it poses a critical issue, because it means that the user needs to provide feedbacks on a large number of items which she is not interested in.

Recently, a number of researchers propose the construction of conversational recommender systems that leverage conversations to elicit users' preference for better recommendation (e.g., [5, 6]). Inspired by this, we consider a novel contextual bandit setting in this paper, i.e., *conversational contextual bandit*, which incorporates a conversation mechanism into the traditional contextual bandit algorithm (as shown in Figure 1), for accelerating bandit learning.

To illustrate the benefits of the conversation mechanism, let us consider the news recommendation scenario with conversational contextual bandit. In this scenario, the agent/system also occasionally asks questions with regard to the user's preference on key-terms. For example, asking about the user's preference on a category: "Are you interested in news about *basketball*?", or asking about the user's preference on an entity: "Do you like to read news related to *LeBron James*?". There are two reasons why conversations can improve the learning speed. First, the key-terms

*The work was done when the first author was an intern at Bytedance AI Lab.

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380148

like "basketball" and "LeBron James" can be associated with a large number of articles. Thus feedback on one key-term may contain a large amount of information about the user's preference. Traditional contextual bandit algorithms [1, 9, 11, 13] may spend many interactions to collect the information, because many articles, that are related to the same key-term, may have different contextual vectors. For example, the contextual vector of an article about NBA games may be far from that of an article about basketball shoes, though they are all related to the key-term "basketball". Second, collecting explicit feedback from the user can help the system to capture the user's true preference faster. For example, the fact that the user reads an article of NBA game with "LeBron James" may be because she concerns about the result of the game or because she is a fan of LeBron James. To figure out which is more likely, instead of recommending articles related to each possibility, a more convenient way is to directly ask the user "Do you want to read more news about *LeBron James*?".

In this paper, the agent conducts conversations by asking the user's preference on key-terms. Here, a key-term is related to a subset of arms, and can be a category or topic. We assume that there exists a bipartite graph of key-terms and arms in which the relations between key-terms and arms are represented. The agent occasionally selects key-terms and asks the user about her preference over the key-terms. Then, the user shows her feedbacks to the agent, for example, indicating whether she is interested in the key-terms. The preference over the key-term is then "propagated" to the related arms, and is leveraged by the algorithm to select arms and update the bandit model. To the best of our knowledge, this is the first time such an approach is proposed.

There are two main challenges for conversational contextual bandit: *(1) which key-terms to select for querying, (2) how to leverage conversational feedbacks to build a more accurate bandit model.* We design the Conversational UCB algorithm (ConUCB) to address these challenges.

The ConUCB algorithm, as a generalization of the LinUCB algorithm [1], repeatedly conducts learning as follows: (1) If conversation is allowed at the round, given the current context and historical interactions on arms and key-terms, ConUCB selects the key-terms that reduce the learning error most, and enquires the user's preference over the key-terms; (2) It selects the arm with the largest upper confidence bound derived from both arm-level and key-term-level feedbacks, and receives a reward. The interaction record will then be leveraged for key-term selection. Theoretical analysis shows that ConUCB achieves a lower cumulative regret upper bound than the standard contextual bandit algorithm LinUCB. Note that a smaller regret upper bound means that the agent learns parameters more accurately within the same number of iterations, i.e., the speed of bandit learning is accelerated. Experimental results on a synthetic dataset, as well as real datasets from Yelp and Toutiao (the largest news recommendation platform in China) show that the ConUCB algorithm significantly outperforms the baseline algorithms including LinUCB. Note that the basic mechanism of adding conversation into bandit algorithms to speed up learning is generic. We demonstrate that ConUCB can be easily extended to other contextual bandit algorithms such as hLinUCB [13].

In summary, our contributions are as follows:

- We formulate the new conversational contextual bandit problem to improve the learning speed (Section 2).
- We design the ConUCB algorithm by generalizing LinUCB for conversational contextual bandit and theoretically prove that it can achieve a smaller regret upper bound than the conventional contextual bandit algorithm LinUCB, i.e., the learning speed is improved (Section 3).
- We empirically verify the improvement of ConUCB in learning speed using both a synthetic dataset and real-world datasets (Section 4 & 5).
- We also extend ConUCB by adding the conversation mechanism into hLinUCB (Section 6).

2 PROBLEM FORMULATION

We first introduce traditional contextual bandit and then we generalize it to obtain conversational contextual bandit. Without loss of generality, we take the UCB algorithm as an example algorithm of contextual bandit.

2.1 Contextual Bandit

In contextual bandit, the agent learns to maximize the cumulative reward in $T \in \mathbb{N}_+$ rounds through interaction with the user.

Consider a finite set of N arms denoted by $\mathcal{A}$. At each round $t = 1, \ldots, T$, the agent is given a subset of arms $\mathcal{A}_t \subseteq \mathcal{A}$, and each arm $a \in \mathcal{A}_t$ is associated with a d-dimensional contextual vector $\boldsymbol{x}_{a,t} \in \mathbb{R}^d$, which describes the observable information of arm a and the user at round t. The agent chooses an arm a_t on the basis of its contextual vector $\boldsymbol{x}_{a_t,t}$, shows it to the user, and receives a reward $r_{a_t,t} \in \mathcal{R}$. For example, $\mathcal{R} = \{0, 1\}$ represents a binary reward, and $\mathcal{R} = \mathbb{R}$ represents a continuous reward. The reward $r_{a_t,t}$ is a function of the contextual vector $\boldsymbol{x}_{a_t,t}$ and the parameter vector $\boldsymbol{\theta}$. The parameter vector $\boldsymbol{\theta}$ represents the user's preference and is what the agent wants to learn. At each round t, the agent takes the selected arms $a_1, \ldots, a_{t-1}$ and received rewards $r_{a_1,1}, \ldots, r_{a_{t-1},t-1}$ at the previous rounds as input, to estimate the reward and select the arm $a_t \in \mathcal{A}_t$.

The goal of the agent is to maximize the expected cumulative rewards. Let $\sum_{t=1}^{T} \mathbb{E}[r_{a_t^*,t}]$ denote the maximum expected cumulative rewards in T rounds, where $a_t^* \in \mathcal{A}_t$ is the optimal arm at round t, i.e., $\mathbb{E}[r_{a_t^*,t}] \geq \mathbb{E}[r_{a,t}], \forall a \in \mathcal{A}_t$. The goal of contextual bandit learning is formally defined as minimization of the cumulative regret in T rounds:

$$R(T) \triangleq \sum_{t=1}^{T} \left(\mathbb{E}[r_{a_t^*,t}] - \mathbb{E}[r_{a_t,t}] \right). \tag{1}$$

The agent needs to make a trade-off between exploitation (i.e., choose the best arm estimated from received feedbacks) and exploration (i.e., seek feedbacks from arms that the agent is unsure about).

Let us consider the UCB (Upper Confidence Bound) algorithm. The agent selects the arm a_t at each round t as follows:

$$a_t = \arg\max_{a \in \mathcal{A}_t} R_{a,t} + C_{a,t},$$

where $R_{a,t}$ and $C_{a,t}$ are the estimated reward and confidence interval of arm a at round t respectively. The confidence interval $C_{a,t}$ measures the uncertainty of reward estimation of arm a at round t. In the LinUCB algorithm [1], the reward function is defined as a

linear function,

$$r_{a_t,t} = \boldsymbol{x}_{a_t,t}^T \boldsymbol{\theta} + \epsilon_t, \tag{2}$$

where $\boldsymbol{\theta}$ is a d-dimensional parameter vector, ϵ_t is a random variable representing the random noise. One can also generalize LinUCB by utilizing non-linear reward [7] or hidden features [13].

In news recommendation with contextual bandit, for example, the agent is the recommender system, each arm a is an article, the contextual vector $\boldsymbol{x}_{a,t}$ denotes the observable information of the user and the article a at round t, and the reward $r_{a,t}$ is whether the user clicks article a at round t. The parameter vector $\boldsymbol{\theta}$ represents the user's preference on articles. The goal of the system is to maximize the cumulative click-through rate (CTR) of the user.

2.2 Conversational Contextual Bandit

In conversational contextual bandit, the agent still learns to maximize the cumulative reward in T rounds through interacting with the user. In addition to collecting feedbacks on selected arms, the agent also occasionally conducts conversations with the user and learns from conversational feedbacks.

We consider that the agent conducts conversations by asking the user's preference on *key-terms*, where a key-term is related to a subset of arms, and can be a category, topic, etc. For example, in news recommendation, a key-term can be a keyword, a key-phrase, or a combination of multiple single keywords, extracted from articles. The agent may ask the user "Are you interested in news about *basketball*?" The user shows her answer to the question.

Problem formulation. Consider a finite set of $N \in \mathbb{N}_+$ arms denoted by $\mathcal{A}$ and a finite set of $K \in \mathbb{N}_+$ key-terms denoted by $\mathcal{K}$. The relationships between the arms and the key-terms are characterized by a weighted bipartite graph $(\mathcal{A}, \mathcal{K}, \boldsymbol{W})$, whose nodes are divided into two sets $\mathcal{A}$ and $\mathcal{K}$, and weighted edges are represented by the matrix $\boldsymbol{W} \triangleq [w_{a,k}]$. Here $w_{a,k}$ represents the relationship between arm a and key-term k. Without loss of generality, we normalize the total weights associated with each arm to be 1, i.e., $\sum_k w_{a,k} = 1$.

We also introduce a function $b(t)$ to model the frequency of conversations. Note that $b(t)$ determines: (1) whether to converse at round t; (2) the number of conversations until round t. To make it clearer, consider a function $q(t)$:

$$q(t) = \begin{cases} 1, & b(t) - b(t-1) > 0, \\ 0, & \text{otherwise.} \end{cases}$$

If $q(t) = 1$, then the agent conducts conversation with the user at round t; if $q(t) = 0$, then it does not. The agent conducts $b(t)$ conversations up to round t. For example, if $b(t) = k\lfloor \frac{t}{m} \rfloor, m \geq 1, k \geq 1$, then the agent makes k conversations in every m rounds. If $b(t) = \lfloor \log(t) \rfloor$, then the agent makes a conversation with a frequency represented by the logarithmic function of t. If $b(t) \equiv 0$, then there is no conversation between the agent and the user. Moreover, we assume that key-term-level conversations should be less frequent than arm-level interactions, i.e., $b(t) \leq t, \forall t$, in consideration of users' experience.

At each round $t = 1, \ldots, T$, the agent is given a subset of arms $\mathcal{A}_t \subseteq \mathcal{A}$, and each arm $a \in \mathcal{A}_t$ is specified by a contextual vector $\boldsymbol{x}_{a,t} \in \mathbb{R}^d$. Without loss of generality, we normalize the contextual vector such that $\|\boldsymbol{x}_{a,t}\|_2 = 1$. Based on the arm-level feedbacks and conversational feedbacks received in the previous $t-1$ rounds,

Algorithm 1: General algorithm of ConUCB

Input: arms $\mathcal{A}$, key-terms $\mathcal{K}$, graph $(\mathcal{A}, \mathcal{K}, \boldsymbol{W})$, $b(t)$.

1 **for** $t = 1, \ldots, T$ **do**
2 observe contextual vector $\boldsymbol{x}_{a,t}$ of each arm $a \in \mathcal{A}_t$;
3 If conversation is allowed at round t, i.e., $q(t) = 1$, select key-terms to conduct conversations and receive conversational feedbacks $\{\tilde{r}_{k,t}\}$;
4 select an arm $a_t = \arg\max_{a \in \mathcal{A}_t} \tilde{R}_{a,t} + C_{a,t}$;
5 receive a reward $r_{a_t,t}$;
6 update model ;

- If $q(t) = 1$, the agent conducts $\lfloor b(t) - b(t-1) \rfloor$ conversations with the user. In each conversation, the agent asks the user's preference on one selected key-term $k \in \mathcal{K}$, and gets the user's feedback $\tilde{r}_{k,t}$. For example, $\tilde{r}_{k,t}$ can be binary $\{0, 1\}$, where 0 and 1 stand for negative and positive feedback respectively.
- The agent chooses an arm $a_t \in \mathcal{A}_t$, presents it to the user, and receives the reward of arm a_t, denoted by $r_{a_t,t}$.

The agent tries to speed up the learning process by leveraging conversational feedbacks. The problems in conversational contextual bandit are to find (a) an effective arm selection strategy and (b) an effective key-term selection strategy, so that after T rounds of arm-level interactions and $b(T)$ times of conversations, the cumulative regret in Eq. (1) is significantly reduced.

General algorithm of conversational UCB (ConUCB). Algorithm 1 outlines the general algorithm of ConUCB, which will be described in detail in Section 3. One can see that when no conversation is performed between the agent and the user, conversational contextual bandit degenerates to standard contextual bandit. In ConUCB, the agent selects an arm with the following strategy

$$a_t = \arg\max_{a \in \mathcal{A}_t} \tilde{R}_{a,t} + C_{a,t}, \tag{3}$$

where $\tilde{R}_{a,t}$ and $C_{a,t}$ are the estimated reward and the confidence interval of arm a at round t respectively. As will be shown in Section 3 later, $\tilde{R}_{a,t}$ is inferred from *both* arm-level and key-term-level feedbacks, and $C_{a,t}$ contains *both* arm-level and key-term-level confidence interval.

In ConUCB, the user's feedback on key-term k, i.e., $\tilde{r}_{k,t}$, is estimated from the user's feedbacks on related arms, i.e.,

$$\mathbb{E}[\tilde{r}_{k,t}] = \sum_{a \in \mathcal{A}} \frac{w_{a,k}}{\sum_{a' \in \mathcal{A}} w_{a',k}} \mathbb{E}[r_{a,t}], \quad k \in \mathcal{K}. \tag{4}$$

Equivalently, $\tilde{r}_{k,t} = \sum_{a \in \mathcal{A}} \frac{w_{a,k}}{\sum_{a' \in \mathcal{A}} w_{a',k}} \mathbb{E}[r_{a,t}] + \tilde{\epsilon}_{k,t}$, where $\tilde{\epsilon}_{k,t}$ is a random variable representing the random noise in reward. When both the conversational feedback $\tilde{r}_{k,t}$ and arm-level feedback $r_{a,t}$ are binary, Eq. (4) has a probabilistic interpretation. Specifically, for binary feedback $c \in \{0, 1\}$, we have

$$\mathbb{P}(c|k,t) = \sum_{a \in \mathcal{A}} \mathbb{P}(a|k)\mathbb{P}(c|a,t), \tag{5}$$

where $\mathbb{P}(c = 1|k,t)$ and $\mathbb{P}(c = 0|k,t)$ represent the probabilities that the user gives a positive and negative feedback to the question with key-term k at round t respectively; $\mathbb{P}(c = 1|a,t)$ and $\mathbb{P}(c = 0|a,t)$ denote the probabilities that the user likes and dislikes arm a at round t respectively. Thus, if we take $\mathbb{E}[r_{a,t}] = \mathbb{P}(c = 1|a,t)$, $\mathbb{E}[\tilde{r}_{k,t}] = \mathbb{P}(c{=}1|k,t)$, and $\mathbb{P}(a|k){=}\frac{w_{a,k}}{\sum_{a' \in \mathcal{A}} w_{a',k}}$, we obtain Eq. (5).

3 ALGORITHM & THEORETICAL ANALYSIS

In this section, we present the details of ConUCB by providing specific solutions to the two problems in Algorithm 1: (1) how to select key-terms to conduct conversation (Line 3), (2) how to select an arm, i.e., calculate $\tilde{R}_{a,t}$ and $C_{a,t}$ (Line 4). ConUCB is a generalization of LinUCB [1] in the sense that the arm-level reward function is the same as that of LinUCB in Eq. (2). We theoretically analyze the upper bound of its cumulative regret and discuss the impact of conversational feedbacks. We note that ConUCB has a generic mechanism of selecting key-terms and leveraging feedbacks on key-terms to speed up learning, which can be easily incorporated into a variety of contextual bandit algorithms such as CoFineUCB [16], hLinUCB [13]. In Section 6, we explain how to apply the same technique to the hLinUCB algorithm.

3.1 ConUCB Algorithm

The ConUCB algorithm is described in Algo. 2. It contains a key-term selection module to select key-terms (line 2-11) and an arm-selection module to select arms (line 12-15). The two modules collaborate with each other as follows:

- If conversation is allowed at round t, given the current context and interaction histories on both arms and key-terms, the key-term selection module repeatedly selects the key-term that minimizes the regret and asks the user's preference over it (line 5). Then the newly estimated key-term-level parameter vector ($\tilde{\boldsymbol{\theta}}_t$) is passed to the arm-selection module.
- Under the guidance of $\tilde{\boldsymbol{\theta}}_t$ and rewards received up to round t, the arm-selection module recommends an arm to the user, and receives a reward (line 12-15). The interaction record will then be leveraged by the key-term selection module.

3.1.1 ***Arm selection.*** At round t, ConUCB first estimates the user's preference at the key-term level, denoted as $\tilde{\boldsymbol{\theta}}_t$, by

$$\tilde{\boldsymbol{\theta}}_t = \arg\min_{\tilde{\boldsymbol{\theta}}} \sum_{\tau=1}^{t} \sum_{k\in\mathcal{K}_\tau} \left(\frac{\sum_{a\in\mathcal{A}} w_{a,k} \tilde{\boldsymbol{\theta}}^T \boldsymbol{x}_{a,\tau}}{\sum_{a\in\mathcal{A}} w_{a,k}} - \tilde{r}_{k,\tau} \right)^2 + \tilde{\lambda} \|\tilde{\boldsymbol{\theta}}\|_2^2,$$

where $\mathcal{K}_\tau$ denotes the set of key-terms queried at round τ. We set $\mathcal{K}_\tau = \emptyset$ if no key-term is queried, and we let $\mathcal{K}_\tau$ contain duplicated elements if querying a key-term multiple times at the round. The coefficient $\tilde{\lambda} \in \mathbb{R}$ controls regularization. Then $\tilde{\boldsymbol{\theta}}_t$ is used to guide the learning of the arm-level parameter vector at round t:

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}} \ \lambda \sum_{\tau=1}^{t-1} (\boldsymbol{\theta}^T \boldsymbol{x}_{a_\tau,\tau} - r_{a_\tau,\tau})^2 + (1-\lambda) \|\boldsymbol{\theta} - \tilde{\boldsymbol{\theta}}_t\|_2^2.$$

where $\lambda \in [0,1]$ balances learning from rewards at arm-level and learning from feedbacks at key-term-level (i.e., $\tilde{\boldsymbol{\theta}}_t$). Both optimization problems have closed-form solutions, $\tilde{\boldsymbol{\theta}}_t = \tilde{\boldsymbol{M}}_t^{-1} \tilde{\boldsymbol{b}}_t$ and $\boldsymbol{\theta}_t = \boldsymbol{M}_t^{-1}(\boldsymbol{b}_t + (1-\lambda)\tilde{\boldsymbol{\theta}}_t)$, where

$$\begin{aligned}
\tilde{\boldsymbol{M}}_t &= \sum_{\tau=1}^{t} \sum_{k\in\mathcal{K}_\tau} \left(\frac{\sum_{a\in\mathcal{A}} w_{a,k} \boldsymbol{x}_{a,\tau}}{\sum_{a\in\mathcal{A}} w_{a,k}} \right) \left(\frac{\sum_{a\in\mathcal{A}} w_{a,k} \boldsymbol{x}_{a,\tau}}{\sum_{a\in\mathcal{A}} w_{a,k}} \right)^T + \tilde{\lambda} \boldsymbol{I}, \\
\tilde{\boldsymbol{b}}_t &= \sum_{\tau=1}^{t} \sum_{k\in\mathcal{K}_\tau} \left(\frac{\sum_{a\in\mathcal{A}} w_{a,k} \boldsymbol{x}_{a,\tau}}{\sum_{a\in\mathcal{A}} w_{a,k}} \right) \tilde{r}_{k,\tau}, \\
\boldsymbol{M}_t &= \lambda \sum_{\tau=1}^{t-1} \boldsymbol{x}_{a_\tau,\tau} \boldsymbol{x}_{a_\tau,\tau}^T + (1-\lambda)\boldsymbol{I}, \\
\boldsymbol{b}_t &= \lambda \sum_{\tau=1}^{t-1} \boldsymbol{x}_{a_\tau,\tau} r_{a_\tau,\tau}.
\end{aligned} \tag{6}$$

To apply the arm-selection strategy in Eq. (3) , we need to derive the confidence interval $C_{a,t}$. Based on the closed-form solutions of

Algorithm 2: ConUCB algorithm

Input: graph $(\mathcal{A}, \mathcal{K}, \boldsymbol{W})$, conversation frequency function $b(t)$.
Init: $\tilde{\boldsymbol{M}}_0 = \tilde{\lambda}\boldsymbol{I}$, $\tilde{\boldsymbol{b}}_0 = \boldsymbol{0}$, $\boldsymbol{M}_0 = (1-\lambda)\boldsymbol{I}$, $\boldsymbol{b}_0 = \boldsymbol{0}$.

1 **for** $t = 1, 2, \ldots, T$ **do**
2 **if** $b(t) - b(t-1) > 0$ **then**
3 nq= $b(t) - b(t-1)$;
4 **while** $nq > 0$ **do**
5 Select a key-term $k \in \mathcal{K}$ according to Eq. (8), and query the user's preference over it ;
6 Receive the user's feedback $\tilde{r}_{k,t}$;
7 $\tilde{\boldsymbol{M}}_t = \tilde{\boldsymbol{M}}_{t-1} + \left(\frac{\sum_{a\in\mathcal{A}} w_{a,k} \boldsymbol{x}_{a,t}}{\sum_{a\in\mathcal{A}} w_{a,k}} \right) \left(\frac{\sum_{a\in\mathcal{A}} w_{a,k} \boldsymbol{x}_{a,t}}{\sum_{a\in\mathcal{A}} w_{a,k}} \right)^T$;
8 $\tilde{\boldsymbol{b}}_t = \tilde{\boldsymbol{b}}_{t-1} + \left(\frac{\sum_{a\in\mathcal{A}} w_{a,k} \boldsymbol{x}_{a,t}}{\sum_{a\in\mathcal{A}} w_{a,k}} \right) \tilde{r}_{k,t}$;
9 nq-=1
10 **else**
11 $\tilde{\boldsymbol{M}}_t = \tilde{\boldsymbol{M}}_{t-1}$, $\tilde{\boldsymbol{b}}_t = \tilde{\boldsymbol{b}}_{t-1}$;
12 $\tilde{\boldsymbol{\theta}}_t = \tilde{\boldsymbol{M}}_t^{-1} \tilde{\boldsymbol{b}}_t$, $\boldsymbol{\theta}_t = \boldsymbol{M}_t^{-1} \left(\boldsymbol{b}_t + (1-\lambda)\tilde{\boldsymbol{\theta}}_t \right)$;
13 Select $a_t = \arg\max_{a\in\mathcal{A}_t} \boldsymbol{x}_{a,t}^T \boldsymbol{\theta}_t + \lambda\alpha_t \|\boldsymbol{x}_{a,t}\|_{\boldsymbol{M}_t^{-1}} + (1-\lambda)\tilde{\alpha}_t \|\boldsymbol{x}_{a,t}^T \boldsymbol{M}_t^{-1}\|_{\tilde{\boldsymbol{M}}_t^{-1}}$;
14 Ask the user's preference on arm $a_t \in \mathcal{A}$ and receive the reward $r_{a_t,t}$;
15 $\boldsymbol{M}_t = \boldsymbol{M}_t + \lambda \boldsymbol{x}_{a_t,t} \boldsymbol{x}_{a_t,t}^T$, $\boldsymbol{b}_t = \boldsymbol{b}_t + \lambda \boldsymbol{x}_{a_t,t} r_{a_t,t}$;

$\boldsymbol{\theta}_t$ and $\tilde{\boldsymbol{\theta}}_t$ in Eq. (6), we can prove that Lemma 1 holds, and thus

$$C_{a,t} = \lambda\alpha_t \|\boldsymbol{x}_{a,t}\|_{\boldsymbol{M}_t^{-1}} + (1-\lambda)\tilde{\alpha}_t \|\boldsymbol{x}_{a,t}^T \boldsymbol{M}_t^{-1}\|_{\tilde{\boldsymbol{M}}_t^{-1}},$$

where α_t is defined in Lemma1, and $\tilde{\alpha}_t$ represents the estimation error of $\tilde{\boldsymbol{\theta}}_t$ and is determined by how the agent selects key-terms. We will discuss $\tilde{\alpha}_t$ later in Section 3.1.2, and show that $\tilde{\alpha}_t < \alpha_t$.

Consequently, ConUCB selects an arm according to the following strategy (Eq. (3)):

$$a_t = \arg\max_{a\in\mathcal{A}_t} \underbrace{\boldsymbol{x}_{a,t}^T \boldsymbol{\theta}_t}_{\tilde{R}_{a,t}} + \underbrace{\lambda\alpha_t \|\boldsymbol{x}_{a,t}\|_{\boldsymbol{M}_t^{-1}} + (1-\lambda)\tilde{\alpha}_t \|\boldsymbol{x}_{a,t}^T \boldsymbol{M}_t^{-1}\|_{\tilde{\boldsymbol{M}}_t^{-1}}}_{C_{a,t}}. \tag{7}$$

In Eq. (7), $\tilde{R}_{a,t}$ is the estimated reward of arm a at round t, based on the current estimated parameter vectors at arm-level and key-term level. It represents exploitation of currently promising arms. $C_{a,t}$ denotes the uncertainty in reward estimation of arm a, which contains two parts: (1) uncertainty from the noise in arm rewards received until round t (the first term) ; (2) uncertainty from the estimated key-term-level parameter vector $\tilde{\boldsymbol{\theta}}_t$ (the second term). It represents exploration of currently less promising arms. It is easy to verify that $C_{a,t}$ shrinks when more interactions between the agent and user are carried out, and thus the exploitation and exploration are balanced. Moreover, one can see that the second term of $C_{a,t}$ shrinks more quickly than the first term (more details can be fund at Eq. (11) in Appendix), indicating the benefit of conversation.

Lemma 1. *Let $\boldsymbol{\theta}_*$ and $\tilde{\boldsymbol{\theta}}_*$ denote the unknown true parameter vectors of the user at arm level and key-term level respectively. Assume that*

$\|\tilde{\theta}_t - \tilde{\theta}_*\|_{\tilde{M}_t} \le \tilde{\alpha}_t$, $\tilde{\epsilon}_{k,t}$ and ϵ_t are conditionally 1-sub-Gaussian, then for $\forall t, a \in \mathcal{A}$, with probability $1-\sigma$, the following inequality holds

$$|x_{a,t}^T(\theta_t - \theta_*)| \le \lambda\alpha_t\|x_{a,t}\|_{M_t^{-1}} + (1-\lambda)\tilde{\alpha}_t\|x_{a,t}^T M_t^{-1}\|_{\tilde{M}_t^{-1}},$$

where $\alpha_t = \sqrt{d\log\left((1+\frac{\lambda t}{(1-\lambda)d})/\sigma\right)}$, *and* $\|x\|_M = \sqrt{x^T M x}$.

3.1.2 ***Key-Term Selection.*** Next we describe how the ConUCB algorithm selects key-terms. Let $X_t \in \mathbb{R}^{|\mathcal{A}_t|\times d}$ denote the collection of contextual vectors of arms presented at round t, i.e, $X_t(a,:) = x_{a,t}^T, \forall a \in \mathcal{A}_t$. Given the current context X_t and interactions on both arms and key-terms up to round t, to minimize the cumulative regret, ConUCB needs to select a key-term so that θ_t can be learned as accurately as possible, since no regret would be induced if θ_t is equal to the unknown true arm-level parameter vector θ_*. Thus, a natural idea is to select the key-term that minimizes the estimation error $\mathbb{E}[\|X_t\theta_t - X_t\theta_*\|_2^2]$. As suggested by Theorem 2, this means to select key-terms according to Eq. (8).

Theorem 2. *Given the current context X_t and interactions at both arm-level and key-term level up to round t, to minimize $\mathbb{E}[\|X_t\theta_t - X_t\theta_*\|_2^2]$, one only needs to select the key-term k to minimize*

$$\mathrm{tr}\left(X_t M_t^{-1}(\tilde{M}_{t-1} + \tilde{x}_{k,t}\tilde{x}_{k,t}^T)^{-1}M_t^{-1}X_t^T\right).$$

In other words, it selects the key-term k as follows:

$$k = \arg\max_{k'} \quad \|X_t M_t^{-1}\tilde{M}_{t-1}^{-1}\tilde{x}_{k',t}\|_2^2 \Big/ \left(1 + \tilde{x}_{k',t}^T\tilde{M}_{t-1}^{-1}\tilde{x}_{k',t}\right). \quad (8)$$

where $\tilde{x}_{k,t} = \sum_{a\in\mathcal{A}} \frac{w_{a,k}}{\sum_{a'\in\mathcal{A}} w_{a',k}} x_{a,t}$.

We can observe from Eq. (8) that the key-term selection is depended on both the arms and key-terms selected in the previous rounds, i.e., M_t and $\tilde{M}_t$. Moreover, the essence of Theorem 2 is to select key-terms to minimize $\mathrm{tr}\left(X_t M_t^{-1}\tilde{M}_t^{-1}M_t^{-1}X_t^T\right)$, which can make the last term in the arm selection strategy, i.e., $\tilde{\alpha}_t\|x_{a,t}^T M_t^{-1}\|_{\tilde{M}_t^{-1}}$ in Eq. (7), shrink faster.

When selecting key-terms according to Eq. (8), the agent can calculate $\tilde{\alpha}_t$ using Lemma 3. Since $b(t) \le t$, $\tilde{\alpha}_t$ is at the order of $O(\sqrt{d+\log t})$, while α_t is at the order of $O(\sqrt{d\log t})$, implying $\tilde{\alpha}_t < \alpha_t$.

Lemma 3. *In selection of key-terms according to Eq. (8), with probability $1-\sigma$, the following inequality holds:*

$$\|\tilde{\theta}_t - \tilde{\theta}_*\|_{\tilde{M}_t} \le \tilde{\alpha}_t = \sqrt{2\left(d\log 6 + \log(\tfrac{2b(t)}{\sigma})\right)} + 2\sqrt{\tilde{\lambda}}\|\tilde{\theta}_*\|_2.$$

3.2 Regret Upper Bound of ConUCB

We can prove that the regret upper bound of ConUCB is as follows.

Theorem 4. *Assume that $\lambda \in [0, 0.5]$, $\tilde{\lambda} \ge \frac{2(1-\lambda)}{\lambda(1-\sqrt{\lambda})^2}$, with the key-term selection strategy defined in Eq. (8), then with probability $1-\sigma$, ConUCB has the following regret upper bound.*

$$R(T) \le 2\left(\sqrt{\lambda}\sqrt{d\log\left((1+\tfrac{\lambda T}{(1-\lambda)d})/\sigma\right)} + 2\sqrt{\tfrac{1-\lambda}{\lambda}}\|\tilde{\theta}_*\|_2 + (1-\sqrt{\lambda})\sqrt{d\log 6 + \log(\tfrac{2b(t)}{\sigma})}\right)\sqrt{Td\log(1+\tfrac{\lambda T}{d(1-\lambda)})}.$$

Since $b(t) \le t, \lambda \in [0, 0.5]$, the regret upper bound of ConUCB in Theorem 4 is at most $\dot{O}((1-\sqrt{\lambda})\sqrt{d+\log T} + \sqrt{\lambda d\log T})$, where $\dot{O}(a) = O(a\cdot\sqrt{dT\log T})$. It is smaller than the regret upper bound of LinUCB [1], i.e., $\dot{O}((1-\sqrt{\lambda})\sqrt{d\log T} + \sqrt{\lambda d\log T}) = \dot{O}(\sqrt{d\log T})$. Therefore, when d and $\log T$ are large, which is usually the case in practice, we can improve substantially by reducing $\sqrt{d\log T}$ to $\sqrt{d+\log T}$ in the first term.

4 EXPERIMENTS ON SYNTHETIC DATA

In this section, we describe experimental results on synthetic data.

4.1 Experiment Setting

Synthetic data. We create a key-term set $\mathcal{K} \triangleq \{1, 2, \ldots, K\}$ with size K and an arm pool (set) $\mathcal{A} \triangleq \{a_1, a_2, \ldots, a_N\}$ with size N. Each arm $a \in \mathcal{A}$ is associated with a d-dimensional feature vector x_a, and it is also related to a set of key-terms $\mathcal{Y}_a \subseteq \mathcal{K}$ with equal weight $1/|\mathcal{Y}_a|$. We assume that the more shared key-terms two arms have, the closer their feature vectors will be. We generate an arm's feature vector as follows: (1) we first generate a pseudo feature vector $\dot{x}_k$ for each key-term $k \in \mathcal{K}$, where each dimension of $\dot{x}_k$ is drawn independently from a uniform distribution $U(-1, 1)$; (2) for each arm a, we sample n_a key-terms uniformly at random from $\mathcal{K}$ without replacement as its related key-terms set $\mathcal{Y}_a$ with equal weight $1/n_a$, where n_a is a random number in $[1, M]$; (3) finally, each dimension i of x_a is independently drawn from $N(\sum_{k\in\mathcal{Y}_a}\dot{x}_k(i)/n_a, \sigma_g^2)$, where $\sigma_g \in \mathbb{R}_+$. We note that the information of key-terms is contained in x_a, and thus they are available for all algorithms. We then generate N_u users, each of whom is associated with a d-dimensional vector θ_u, i.e., the ground-truth of user u's preference. Each dimension of θ_u is drawn from a uniform distribution U(-1,1). We consider a general setting in which at each round t, the simulator only discloses a subset of arms in $\mathcal{A}$, denoted as $\mathcal{A}_t$, to the agent for selection, for example, randomly selecting 50 arms from $\mathcal{A}$ without replacement. The true arm-level reward $r_{a,t}$ as well as key-term level reward $\tilde{r}_{k,t}$ are generated according to Eq. (2) and Eq. (4) respectively. The noise ϵ_t is sampled from Gaussian distribution $\mathcal{N}(0, \sigma_g^2)$ only once for all arms at round t, so is the noise $\tilde{\epsilon}_{k,t}$. In the simulation, we set contextual vector $d = 50$, user number $N_u = 200$, arm size $N = 5000$, size of key-terms $K = 500$, $\sigma_g = 0.1$. We set $M = 5$, which means that each arm is related to at most 5 different key-terms, because the average values of M in Yelp and Toutiao are 4.47 and 4.49 respectively. Moreover, following similar procedures in paper [11], we tune the optimal parameters in each algorithm.
Baselines. We compare the proposed ConUCB algorithm with the following algorithms.

- **LinUCB** [9]: The state-of-art contextual bandit algorithm. LinUCB only works with arm-level feedback, and it does not consider conversational feedback.
- **Arm-Con**: Christoper et. al. [6] proposes to conduct conversation by asking the user whether she likes an additional item selected by a bandit algorithm. Arm-Con adopts the conversation format and leverages LinUCB for arm selection.
- **Var-RS**: A variant of ConUCB selecting a key-term *randomly.*

- **Var-MRC**: A variant of ConUCB that selects the key-term with the maximal related confidence under current context:

$$k = \arg\max_{k'} \sum_{a \in \mathcal{A}_t} \frac{w_{a,k'}}{\sum_{a' \in \mathcal{A}_t} w_{a',k'}} \tilde{\alpha}_t \|\boldsymbol{x}_{a,t}^T \boldsymbol{M}_t^{-1}\|_{\tilde{\boldsymbol{M}}_t^{-1}},$$

where $\tilde{\alpha}_t \|\boldsymbol{x}_{a,t}^T \boldsymbol{M}_t^{-1}\|_{\tilde{\boldsymbol{M}}_t^{-1}}$ is the part of confidence interval $C_{a,t}$ related to key-terms (Eq. (7)).

- **Var-LCR**: A variant of ConUCB that selects the key-term with the largest confidence reduction under current context:

$$k = \arg\max_{k'} \sum_{a \in \mathcal{A}_t} \frac{w_{a,k'}}{\sum_{a' \in \mathcal{A}_t} w_{a',k'}} \left(C_{a,t} - C_{a,t}^{k'}\right),$$

where $C_{a,t}^{k'}$ is the new confidence interval of arm a at round t, if we query key-term k' next.

Note that if we run LinUCB algorithm T rounds, then all other algorithms conduct T-round arm-level interactions and $b(T)$ conversations. Conversations are conducted at the same time for the algorithms.

4.2 Evaluation Results

We evaluate all algorithms in terms of cumulative regret defined in Eq. (1). We set the frequency function $b(t) = 5\lfloor \log(t) \rfloor$. At each round t, we randomly select 50 arms from $\mathcal{A}$ without replacement as $\mathcal{A}_t$. The same $\mathcal{A}_t$ are presented to all algorithms.

Cumulative regret comparison. We run the experiments 10 times, and calculate the average cumulative regret for each algorithm. The results plotted in Figure 3a show that LinUCB has the largest cumulative regret, indicating that the use of conversational feedbacks, either by querying items (Arm-Con) or by querying key-terms (Var-RS, ConUCB, Var-MRC, Var-LCR) can improve the learning speed of bandit, since they can learn parameters more accurately within the same number of iterations. Moreover, algorithms that query key-terms, i.e., Var-RS, ConUCB, Var-MRC, Var-LCR, have much smaller cumulative regret than Arm-Con, which asks the user's preference on additional items. It is reasonable, because feedback on a key-term should be more informative than feedback on an arm. Finally, the ConUCB algorithm has the smallest cumulative regret, demonstrating the effectiveness of its key-term selection strategy.

Accuracy of learned parameters. We next compare the accuracy of the learned parameters in different algorithms. For each algorithm, we calculate the average difference between the learned parameter vectors of users and the ground-truth parameter vectors of users, i.e., $\frac{1}{N_u} \sum_u \|\boldsymbol{\theta}_{u,t} - \boldsymbol{\theta}_{u,*}\|_2$, where $\boldsymbol{\theta}_{u,t}$ and $\boldsymbol{\theta}_{u,*}$ represent the learned and the ground-truth parameter vectors of user u respectively. A smaller $\frac{1}{N_u} \sum_u \|\boldsymbol{\theta}_{u,t} - \boldsymbol{\theta}_{u,*}\|_2$ implies a higher accuracy in learning of parameter vector. Figure 3b shows the average difference $\frac{1}{N_u} \sum_u \|\boldsymbol{\theta}_{u,t} - \boldsymbol{\theta}_{u,*}\|_2$ in *every* 50 iterations. One can observe that for all algorithms, the values of $\frac{1}{N_u} \sum_u \|\boldsymbol{\theta}_{u,t} - \boldsymbol{\theta}_{u,*}\|_2$ decrease in t. That is, all algorithms can learn the parameters more accurately with more interactions with users. Moreover, ConUCB is the best algorithm in learning of the parameters.

Impact of conversation frequency $b(t)$. Next we study the impact of conversation frequency. In principle, key-term-level conversations should be less frequent than arm-level interactions, i.e., $b(t) \leq t$. Thus we mainly consider two types of conversation frequency function $b(t)$: (1) $b(t) = Q_l \lfloor \log(t) \rfloor$: ask Q_l questions every time, while the span between two consecutive conversations gets larger and larger; (2) $b(t) = Q_l \lfloor \frac{t}{Q_q} \rfloor$: ask Q_l questions per Q_q iterations. For the first type of $b(t)$, we vary the value of Q_l, and obtain cumulative regrets at round 1000 shown in Figure 2a. For the second type of $b(t)$, we set $Q_q = 50$, vary the value of Q_l, and plot cumulative regrets at round 1000 in Figure 2b. We also run the experiments with $Q_q = 20$ and $Q_q = 100$, and the results are similar to that with $Q_q = 50$.

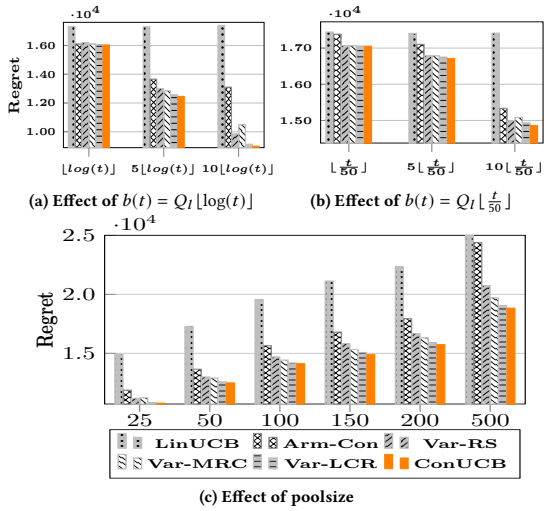


Figure 2: Effect of various factors: Figures (a) & (b) show the effect of $b(t)$; Figure (c) shows the effect of poolsize.

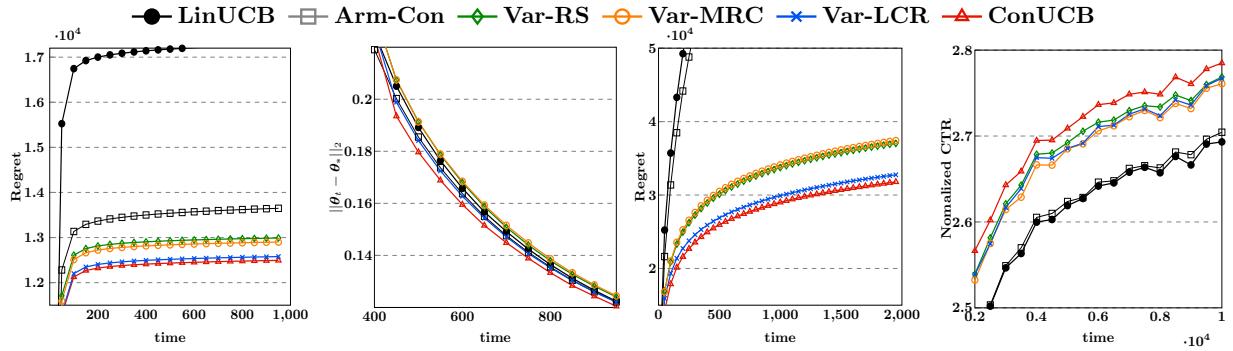
Figure 2b and Figure 2a show that asking more questions can help reduce the cumulative regrets more. For example, in Figure 2a, the cumulative regret is the largest when $b(t) = \lfloor \log(t) \rfloor$, while the cumulative regret is the smallest when $b(t) = 10\lfloor \log(t) \rfloor$. Similarly, in Figure 2b, the cumulative regret is the largest when $b(t) = \lfloor \frac{t}{50} \rfloor$, while it is the smallest when $b(t) = 10\lfloor \frac{t}{50} \rfloor$.

Comparing the cumulative regrets with $b(t) = 5\lfloor \log(t) \rfloor$ and $b(t) = 5\lfloor \frac{t}{50} \rfloor$, we can observe that although the agent asks more questions with $b(t) = 5\lfloor \frac{t}{50} \rfloor$, its cumulative regret is much larger than that with $b(t) = 5\lfloor \log(t) \rfloor$. The reason seems to be that with $b(t) = 5\lfloor \log(t) \rfloor$ the agent can ask more questions at the beginning, quickly capture users' preference, and then gradually reduce the cumulative regret afterward.

Impact of poolsize $|\mathcal{A}_t|$. We change the size of $\mathcal{A}_t$ from 25 to 500 while fixing all the parameters as described in Section 4.1. Figure 2c plots the cumulative regrets under different poolsizes. One can observe that as the poolsize increases, the cumulative regrets of all algorithms also increase, since it is more difficult to select the best arm when the poolsize becomes larger. Again, we observe similar trends in different poolsizes: (1) Using conversational feedbacks, either by querying additional arms (Arm-Con), or by querying key-terms (Var-RS, Var-MRC, Var-LCR, ConUCB) can reduce regret; Querying key-terms is more effective and has smaller regrets than querying arms. (2) The regret of ConUCB is the smallest.

5 EXPERIMENTS ON REAL DATA

In this section, we describe empirical evaluations of the ConUCB algorithm on two real-world datasets.



(a) Cumulative regret on synthetic dataset (b) Accuracy of learned parameters (c) Cumulative regret on Yelp dataset (d) Normalized CTR on Toutiao dataset

Figure 3: Experimental results on all datasets.

5.1 Experiments on Yelp Dataset

Yelp dataset. The public Yelp dataset[1] contains users' reviews of restaurants on Yelp. We only keep users with no less than 100 reviews, and restaurants with no less than 50 reviews. The final Yelp dataset has 1,998 users, 25,160 restaurants, and 342,237 reviews.

Experiment settings. We take each restaurant as an arm. Each restaurant in the dataset is associated with a number of categories. For example, one restaurant named *"Garage"* is associated with the following categories: { *"Mexican", "Burgers", "Gastropubs"*}. We take each category as a key-term. Each arm is equally related to its associated key-terms. There are in total 1,004 categories in the dataset, i.e. $|\mathcal{K}| = 1004$. We construct 50-dimensional arms' contextual vectors via applying PCA on feature vectors generated from restaurants' attributes, including: (1) geographic features: 330 different cities; (2) categorical features: 1,004 different categories; (3) average rating and total review count; (4) attributes: 34 different attributes, such as whether the restaurant serves alcohol or WIFI. We also normalize all the contextual vectors, i.e., $\|\boldsymbol{x}_a\|_2 = 1, \forall a$. The original 5-scale ratings are converted to a binary-valued feedback between restaurants and users, i.e., high ratings (4 and 5) as positive(1) and low ratings (≤ 3) as negative(0). We derive users' true parameters based on ridge regression. We fix the size of $\mathcal{A}_t$ to 50. During the simulation, the true arm-level reward $r_{a,t}$ as well as key-term level reward $\tilde{r}_{k,t}$ are generated according to Eq. (2) and Eq. (4) respectively. Note that key-terms (i.e., categorical features) are part of arms' feature sets, thus the information about key-terms is also available for all algorithms.

Evaluation results. We compare all six algorithms in terms of cumulative regret. We adopt the frequency function $b(t) = 5\lfloor log(t) \rfloor$. We run the experiments 10 times, and the average result for each algorithm is shown in Figure 3c. We observe similar results as those on the synthetic dataset. That is, all algorithms that query key-terms, i.e., Var-RS, Var-MRC, Var-LCR, ConUCB, have smaller cumulative regrets than LinUCB and Arm-Con. Moreover, the ConUCB algorithm has the smallest cumulative regret, followed by Var-LCR, and then Var-MRC and Var-RS.

5.2 Experiments on Toutiao Dataset

Toutiao dataset. This is a real-world news recommendation dataset, obtained from Toutiao[2], which is the largest news recommendation platform in China. The dataset contains 2,000 users' interaction records in December 2017. There are 1,746,335 news articles and 8,462,597 interaction records.

Experiment settings. We take each news article as an arm. Each article is associated with several categories, such as, *"news_car"* and *"news_sports"*. Each article is also associated with several keywords automatically extracted from the article. We filter the keywords occurring less than 1,000 times. In total, there are 2, 384 keywords and 573 categories. We take them as key-terms $\mathcal{K}$. We assume that an article is equally related to its associated key-terms. To get the contextual vector of an arm, we first represent the article by a vector of 3,469 features (e.g., topic distribution, categories, keywords, etc.). We then use PCA to conduct dimension reduction, and take the first 100 principal components as the contextual vectors, i.e., $d = 100$. We infer each user's feedback on an article through the user's reading behavior: if the user reads the article, then the feedback is 1, otherwise the feedback is 0. The feedback is also the reward of the article. We infer each user's feedback on key-terms by simulation. Specifically, we pre-process the interacted articles of the 2,000 users in November 2017 as above, and employ ridge regression to infer users' true preference based on interaction records in the period. Then we generate the ground-truth key-term-level feedbacks according to Eq. (4). Note that the information about key-terms is also available for LinUCB and Arm-Con algorithms, in the sense that key-terms (i.e., categories, keywords) are in arms' feature sets.

Comparison method. The unbiased offline evaluation protocol proposed in [10] is utilized to compare different algorithms. The unbiased offline evaluation protocol only works when the feedback in the system is collected under a random policy. Hence, we simulate the *random* policy of the system by generating a candidate pool as follows. At each round t, we store the article presented to the user (a_t) and its received feedback r_{a_t}. Then we create $\mathcal{A}_t$ by including the served article along with 49 extra articles the user has interacted with (hence $|\mathcal{A}_t| = 50, \forall t$). The 49 extra articles are drawn uniformly at random so that for any article a the user interacted with, if a occurs in some set $\mathcal{A}_t$, this article will be the one served by the system 1/50 of the times. The performance of algorithms is evaluated by Click Through-Rate (CTR), the ratio between the number of clicks an algorithm receives and the number of recommendations it makes. Specifically, we use the average CTR in every 500 iterations (not the cumulative CTR) as the evaluation

[1] http://www.yelp.com/academic_dataset
[2] https://www.Toutiao.com/

metric. Following [9], we normalize the resulting CTR from different algorithms by the corresponding logged random strategy's CTR.

Evaluation results. Figure 3d shows the normalized CTRs of different algorithms over 2000 users. One can observe that algorithms that querying key-terms can achieve higher CTRs than LinUCB and Arm-Con. Again ConUCB achieves the best performance. Moreover, on the Toutiao dataset, Var-MRC and Var-LCR perform worse than Var-RS. This is because they tend to select key-terms related to a large group of arms repeatedly. One can also observe that Arm-Con only outperforms LinUCB slightly. This is because Toutiao dataset contains more negative feedbacks than positive feedbacks, and some negative feedbacks are caused by that a user has read something similar, rather than this user does not like the article. However, articles with such negative feedbacks may be queried by Arm-Con using additional questions with larger probability, due to articles with similar contents received positive feedbacks from the same user. This brings disturbance to the learning of the user's parameter vector, decreasing the efficiency of additional questions.

6 EXTENSION OF ALGORITHM

ConUCB incorporates conversations into LinUCB. We demonstrate that the technique is generic and can be applied to other contextual bandit algorithms as well. Specifically, we show how to extend the hLinUCB algorithm [13] with conversational feedbacks.

Conversational hLinUCB. The hLinUCB algorithm [13] is one of the recent contextual bandit algorithms, which utilizes a set of l-dimensional hidden features ($\boldsymbol{v}_a \in \mathbb{R}^l$) that affects the expected reward, in addition to the contextual features ($\boldsymbol{x}_{a,t} \in \mathbb{R}^d$). Formally,

$$r_{a,t} = (\boldsymbol{x}_{a,t}, \boldsymbol{v}_a)^T \boldsymbol{\theta}_u + \eta_t, \tag{9}$$

where $\boldsymbol{\theta}_u \in \mathbb{R}^{l+d}$ is the parameter of user u, and η_t is drawn from a zero-mean Gaussian distribution $\mathcal{N}(0, \gamma^2)$.

We then design the hConUCB algorithm to extend the hLinUCB algorithm to incorporate conversational feedbacks. At round t, hConUCB first infers user u's current key-term-level preference $\tilde{\boldsymbol{\theta}}_{u,t}$ solely based on her conversational feedbacks, and then use $\tilde{\boldsymbol{\theta}}_{u,t}$ to guide the learning of user u's arm-level preference $\hat{\boldsymbol{\theta}}_{u,t}$, with the estimated hidden features $\{\hat{\boldsymbol{v}}_{a,t}\}_{a\in\mathcal{A}}$ fixed. Then, fixing $\tilde{\boldsymbol{\theta}}_{u,t}$ and $\hat{\boldsymbol{\theta}}_{u,t}$, we use both conversational feedbacks and arm-level feedbacks to update the hidden features of arms $\{\hat{\boldsymbol{v}}_{a,t}\}_{a\in\mathcal{A}}$. One can also derive the confidence bound following a similar procedure as in Lemma 1, and choose the arm with the maximal upper confidence bound value. The key-term-selection strategy of hConUCB follows the procedure in Theorem 2. Moreover, we modify the baselines in Section 4.1 in a similar way. For example, similar to Arm-Con, the hArm-Con algorithm conducts conversations by querying whether a user likes the item selected by hLinUCB.

Experiment evaluations. We then compare hConUCB with the modified baselines on the previous three datasets. The experimental settings on the three datasets are the same as those in Section 4 and in Section 5. The only difference is that in this section, features are randomly partitioned into an observable part and a hidden part. We fix the dimensionaliy l of hidden features to 5, and set the dimensionality d of observable features to 45 on synthetic dataset and Yelp dataset, and $d = 95$ on Toutiao dataset. We set $\gamma = 0.1$. Note that only observable features are showed to the algorithms.

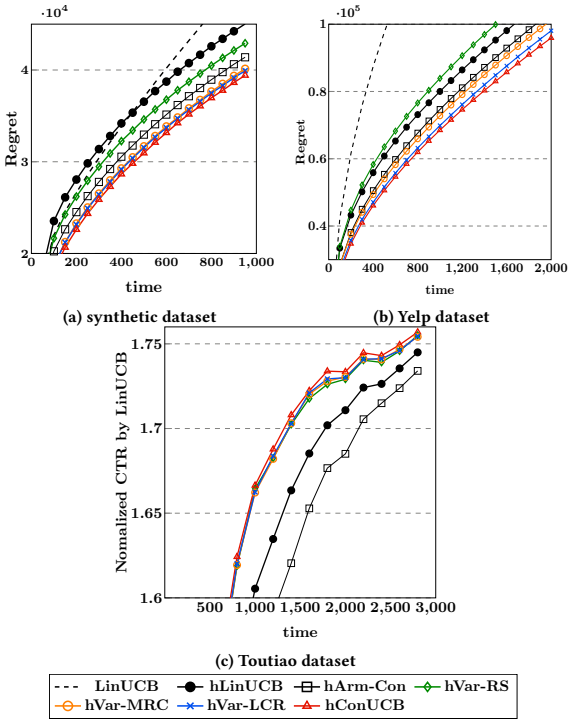


Figure 4: Experimental results of conversational hLinUCB.

The arm-level feedback $r_{a,t}$ and the key-term-level feedback $\tilde{r}_{k,t}$ are generated similarly as in Section 4 and in Section 5, except the arm-level reward model is in Eq. (9).

Figure 4a and 4b show the cumulative regrets (defined in Eq. (1)) on synthetic dataset and Yelp dataset respectively. One can observe that on both datasets, the algorithms that query key-terms except hVar-RS outperform LinUCB, hLinUCB and hArm-Con. Moreover, hConUCB has the smallest cumulative regret. The poor performance of hVar-RS is because randomly selecting key-terms cannot effectively contribute to the inference of arms' hidden features.

For the experiments on Toutiao dataset, we normalize the CTRs from different algorithms by the corresponding CTR of LinUCB. Figure 4c shows the normalized CTRs of different algorithms on Toutiao dataset. The values on the vertical axis in Figure 4c are all larger than 1, indicating that all the algorithms outperform LinUCB. Also, we can observe that all algorithms that query key-terms have higher CTRs than hLinUCB and hArm-Con, and hConUCB achieves the highest CTR. Moreover, in this experiment, the performance of hArm-Con is worse than that of hLinUCB. The reason might be the same as that Arm-Con does not outperform LinUCB, as shown in Figure 3d. In summary, the results demonstrate the effectiveness of conversational mechanism incorporated into bandit algorithms.

7 RELATED WORK

To the best of our knowledge, this is the first work to study the conversational contextual bandit problem. Our work is closely related to the following two lines of research.

Contextual bandit algorithms. Contextual bandit is a popular technique to address the exploration-exploitation trade-off, in various application tasks such as recommendation. LinUCB [9] and Thompson Sampling [2] are two representative algorithms for contextual bandits. A large number of algorithms have been proposed to accelerate the learning speed of LinUCB. For example, Bianchi et. al. [4, 11, 14] leverage relationship among users. Wang et. al. [13] learn hidden features, and Zheng et.al. [17] make use of a time-changing reward function. The CoFineUCB [16] algorithm proposed by Yue et. al. performs a coarse-to-fine hierarchical exploration. Different from them, our work tries to accelerate the learning speed through conversation. More importantly, as shown in Section 6, our approach is generic and can be applied to many existing algorithms.

It should be possible to incorporate the conversation mechanism into Thompson Sampling [2]. Thompson Sampling is a probability matching algorithm that samples $\boldsymbol{\theta}_t$ from the posterior distribution. Thus, one can define a hierarchical sampling approach that first samples $\tilde{\boldsymbol{\theta}}_t$ according to conversational feedbacks, and then samples $\boldsymbol{\theta}_t$ around $\tilde{\boldsymbol{\theta}}_t$ while considering arm-level feedbacks.

Conversational recommender systems. Christakopoulou *et. al.* [6] introduce the idea of conversational recommender systems. They conduct conversations by querying whether the user likes the items selected by the bandits algorithm. As shown in this paper, item-level conversations are less efficient than key-term-level conversations. Other researchers [5, 12, 18] further leverage recent advances in natural language understanding to generate conversations and assist recommendation. Their works utilize conversation in user preference learning (via deep learning or reinforcement learning) without theoretical guarantee, while our work utilizes conversation to speed up contextual bandit learning with theoretical guarantee. Yu *et. al.* [15] propose a dialog approach to speed up bandit learning in recommender systems. Their dialog approach recommends multiple items to the user and the user needs to provide feedback on why she likes or dislikes the recommended items. In this paper, we explore another approach of using conversations, i.e, querying key-terms occasionally to actively explore the user's preference. Bu et. al [3] also leverage conversations to assist item recommendation, however, their algorithm is only applied to offline learning, while our algorithm, based on the bandit technique, is an online learning algorithm. Furthermore, our algorithm adaptively optimizes the question strategy through interactions and conversations.

8 CONCLUSION

We formulate the *conversational contextual bandit* problem by incorporating a conversational mechanism into contextual bandit. We design the ConUCB algorithm to adaptively optimize the arm selection strategy and the key-term selection strategy through conversations and arm-level interactions with the user. Theoretical analysis shows that ConUCB can achieve a lower regret upper bound. Extensive experiments on synthetic dataset, real datasets from Yelp and Toutiao demonstrate that ConUCB indeed has a faster learning speed. The generality of the approach of incorporating conversations into bandit algorithms is also demonstrated.

ACKNOWLEDGMENTS

The work is supported by National Nature Science Foundation of China (61902042) and the GRF 14201819.

A APPENDIX

Proof of Lemma 1:

Proof. According to the closed-form solution of $\boldsymbol{\theta}_t$, we can get:

$$\begin{aligned}\boldsymbol{\theta}_t &= \boldsymbol{M}_t^{-1}\boldsymbol{b}_t\\ &= \left(\lambda\sum_{\tau=1}^{t-1}\boldsymbol{x}_{a,\tau}\boldsymbol{x}_{a,\tau}^T + (1-\lambda)\boldsymbol{I}\right)^{-1}(\lambda\sum_{\tau=1}^{t-1}\boldsymbol{x}_{a,\tau}r_{a,\tau} + (1-\lambda)\tilde{\boldsymbol{\theta}}_t)\\ &= \boldsymbol{\theta}_* - (1-\lambda)\boldsymbol{M}_t^{-1}\boldsymbol{\theta}_* + \lambda\boldsymbol{M}_t^{-1}(\textstyle\sum_{\tau=1}^{t-1}\boldsymbol{x}_{a,\tau}\epsilon_\tau) + (1-\lambda)\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{\theta}}_t.\end{aligned}$$

The equality holds since $r_{a,\tau} = \boldsymbol{x}_{a,\tau}^T\boldsymbol{\theta}_* + \epsilon_\tau$. With $\boldsymbol{\theta}_* \approx \tilde{\boldsymbol{\theta}}_*$,

$$\boldsymbol{\theta}_t - \boldsymbol{\theta}_* = (1-\lambda)\boldsymbol{M}_t^{-1}(\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*) + \lambda\boldsymbol{M}_t^{-1}(\textstyle\sum_{\tau=1}^{t-1}\boldsymbol{x}_{a,\tau}\epsilon_\tau). \quad (10)$$

Thus, with $\|\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\|_{\tilde{\boldsymbol{M}}_t} \le \tilde{\alpha}_t$, we can get:

$$\begin{aligned}&|\boldsymbol{x}_{a,t}^T\boldsymbol{\theta}_t - \boldsymbol{x}_{a,t}^T\boldsymbol{\theta}_*|\\ &= (1-\lambda)|\boldsymbol{x}_{a,t}^T\boldsymbol{M}_t^{-1}(\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*)| + \lambda|\boldsymbol{x}_{a,t}^T\boldsymbol{M}_t^{-1}(\sum_{\tau=1}^{t-1}\boldsymbol{x}_{a,\tau}\epsilon_\tau)|\\ &\le (1-\lambda)\tilde{\alpha}_t\|\boldsymbol{x}_{a,t}^T\boldsymbol{M}_t^{-1}\|_{\tilde{\boldsymbol{M}}_t^{-1}} + \lambda\|\boldsymbol{x}_{a,t}\|_{\boldsymbol{M}_t^{-1}}\|\textstyle\sum_{\tau=1}^{t-1}\boldsymbol{x}_{a,\tau}\epsilon_\tau\|_{\boldsymbol{M}_t^{-1}}.\end{aligned}$$

Let α_t denote the upper bound of $\|\sum_{\tau=1}^{t-1}\boldsymbol{x}_{a,\tau}\epsilon_\tau\|_{\boldsymbol{M}_t^{-1}}$, then Theorem 1 in paper [1] suggests that

$$\alpha_t = \sqrt{2\log\left(\frac{\det(\boldsymbol{M}_t)^{1/2}\det((1-\lambda)\boldsymbol{I})^{-1/2}}{\sigma}\right)} \le \sqrt{d\log\left((1+\tfrac{\lambda t}{(1-\lambda)d})/\sigma\right)}.$$

This proof is then complete. □

Proof of Theorem 2:

Proof. According to Theorem 5, to minimize $\mathbb{E}[\|\boldsymbol{X}_t\boldsymbol{\theta}_t - \boldsymbol{X}_t\boldsymbol{\theta}_*\|_2^2]$, the key-term selection strategy needs to select key-term k to minimize

$$\operatorname{tr}\left(\boldsymbol{X}_t\boldsymbol{M}_t^{-1}(\tilde{\boldsymbol{M}}_{t-1} + \tilde{\boldsymbol{x}}_{k,t}\tilde{\boldsymbol{x}}_{k,t}^T)^{-1}\boldsymbol{M}_t^{-1}\boldsymbol{X}_t^T\right).$$

Then, using Woodbury matrix identity, it is equal to minimize:

$$\operatorname{tr}\left(\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_{t-1}^{-1}\boldsymbol{M}_t^{-1}\boldsymbol{X}_t^T\right) - \frac{\|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_{t-1}^{-1}\tilde{\boldsymbol{x}}_{k,t}\|_2^2}{1+\tilde{\boldsymbol{x}}_{k,t}^T\tilde{\boldsymbol{M}}_{t-1}^{-1}\tilde{\boldsymbol{x}}_{k,t}}.$$

With the interaction history, $\operatorname{tr}\left(\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_{t-1}^{-1}\boldsymbol{M}_t^{-1}\boldsymbol{X}_t^T\right)$ is a constant. Thus, the proof is complete. □

Theorem 5. *Given the interaction history at both arm-level and key-term level up to round t, we have:*

(1) $\min_k \mathbb{E}[\|\boldsymbol{X}_t\boldsymbol{\theta}_t - \boldsymbol{X}_t\boldsymbol{\theta}_*\|_2^2] \Leftrightarrow \min_k \mathbb{E}[\|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}(\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*)\|_2^2]$.

(2) $\mathbb{E}[\|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}(\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*)\|_2^2] \le (\|\boldsymbol{\theta}_*\|_2^2 + 1)\operatorname{tr}\left(\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1}\boldsymbol{M}_t^{-1}\boldsymbol{X}_t^T\right)$.

Proof. With Eq. (10), we can get:

$$\begin{aligned}&\min_k \mathbb{E}\|\boldsymbol{X}_t\boldsymbol{\theta}_t - \boldsymbol{X}_t\boldsymbol{\theta}_*\|_2^2\\ &= \min_k \mathbb{E}\|(1-\lambda)\boldsymbol{X}_t\boldsymbol{M}_t^{-1}(\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*) + \lambda\boldsymbol{X}_t\boldsymbol{M}_t^{-1}(\textstyle\sum_{\tau=1}^{t-1}\boldsymbol{x}_\tau\epsilon_\tau)\|.\end{aligned}$$

Note that key-term selection does not affect $\{\epsilon_\tau\}_{\tau=1}^{t-1}$, thus we can get the first observation:

$$\min_k \mathbb{E}\|\boldsymbol{X}_t\boldsymbol{\theta}_t - \boldsymbol{X}_t\boldsymbol{\theta}_*\|_2^2 \Leftrightarrow \min_k \mathbb{E}\|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}(\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*)\|_2^2.$$

For the second observation, according to the closed-form of $\tilde{\boldsymbol{\theta}}_t$, we can further infer that:

$$\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_* = -\tilde{\lambda}\tilde{\boldsymbol{M}}_t^{-1}\tilde{\boldsymbol{\theta}}_* + \tilde{\boldsymbol{M}}_t^{-1}\left(\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\tilde{\boldsymbol{x}}_{k,\tau}\tilde{\epsilon}_{k,\tau}\right).$$

Thus, we can get:

$$\begin{aligned}&\mathbb{E}\|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}(\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*)\|_2^2\\ &\le \underbrace{\tilde{\lambda}\,\|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1}\tilde{\boldsymbol{\theta}}_*\|_2^2}_{A_1} + \underbrace{\mathbb{E}\|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1}(\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\tilde{\boldsymbol{x}}_{k,\tau}\tilde{\epsilon}_{k,\tau})\|_2^2}_{A_2}\end{aligned}$$

- **Bound** A_1**.** The first term is bounded by:

$$\begin{aligned}\|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1}\tilde{\boldsymbol{\theta}}_*\|_2^2 &\le \|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1/2}\|_F^2\|\tilde{\boldsymbol{M}}_t^{-1/2}\tilde{\boldsymbol{\theta}}_*\|_2^2\\ &\le \frac{\|\tilde{\boldsymbol{\theta}}_*\|_2^2}{\tilde{\lambda}}\,\mathrm{tr}(\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1}\boldsymbol{M}_t^{-1}\boldsymbol{X}_t^T)\end{aligned}$$

- **Bound** A_2**.** We denote $\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\tilde{\boldsymbol{x}}_{k,\tau}\tilde{\epsilon}_{k,\tau} = \tilde{\boldsymbol{X}}_t\tilde{\boldsymbol{\epsilon}}_t$, where $\tilde{\boldsymbol{X}}_t \in R^{d\times b(t)}$. The second term is bounded by:

$$\begin{aligned}&\mathbb{E}\|\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1}(\textstyle\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\tilde{\boldsymbol{x}}_{k,\tau}\tilde{\epsilon}_{k,\tau})\|_2^2\\ &= \mathbb{E}[\mathrm{tr}(\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1}\tilde{\boldsymbol{X}}_t\tilde{\boldsymbol{\epsilon}}_t\tilde{\boldsymbol{\epsilon}}_t^T\tilde{\boldsymbol{X}}_t^T\tilde{\boldsymbol{M}}_t^{-1}\boldsymbol{M}_t^{-1}\boldsymbol{X}_t^T)]\\ &\le \mathrm{tr}(\boldsymbol{X}_t\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1}\boldsymbol{M}_t^{-1}\boldsymbol{X}_t^T)\end{aligned}$$

  The first inequality is due to $\mathbb{E}[\tilde{\boldsymbol{\epsilon}}_t\tilde{\boldsymbol{\epsilon}}_t^T] \le I$.

Finally, with $\tilde{\boldsymbol{\theta}}_* \approx \boldsymbol{\theta}_*$, we can finish the proof. □

Proof of Lemma 3:

Proof. In the following analysis, we take $\tilde{\boldsymbol{x}}_{k,\tau} = \frac{\sum_{a\in\mathcal{A}}\boldsymbol{w}_{a,k}\boldsymbol{x}_{a,\tau}}{\sum_{a\in\mathcal{A}}\boldsymbol{w}_{a,k}}$.

$$\begin{aligned}\tilde{\boldsymbol{\theta}}_t &= \tilde{\boldsymbol{M}}_t^{-1}\boldsymbol{b}_t\\ &= \tilde{\boldsymbol{\theta}}_* - \tilde{\lambda}\tilde{\boldsymbol{M}}_t^{-1}\tilde{\boldsymbol{\theta}}_* + \tilde{\boldsymbol{M}}_t^{-1}\left(\textstyle\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\tilde{\boldsymbol{x}}_{k,\tau}\tilde{\epsilon}_{k,\tau}\right)\end{aligned}$$

Thus, for a fixed x,

$$|\boldsymbol{x}^T\tilde{\boldsymbol{\theta}}_t - \boldsymbol{x}^T\tilde{\boldsymbol{\theta}}_*| \le \tilde{\lambda}|\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}\tilde{\boldsymbol{\theta}}_*| + |\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}\left(\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\tilde{\boldsymbol{x}}_{k,\tau}\tilde{\epsilon}_{k,\tau}\right)|.$$

Since $|\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}\tilde{\boldsymbol{\theta}}_*| \le \|\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}\|_2\|\tilde{\boldsymbol{\theta}}_*\|_2 \le \frac{\|\boldsymbol{x}\|_{\tilde{\boldsymbol{M}}_t^{-1}}}{\sqrt{\tilde{\lambda}}}\|\tilde{\boldsymbol{\theta}}_*\|_2$, we can bound the first term: $\tilde{\lambda}|\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}\tilde{\boldsymbol{\theta}}_*| \le \sqrt{\tilde{\lambda}}\|\boldsymbol{x}\|_{\tilde{\boldsymbol{M}}_t^{-1}}\|\tilde{\boldsymbol{\theta}}_*\|_2$. We next try to bound the second term. According to the key-term-selection strategy in Eq. (8), we can get

$$\mathbb{E}[\textstyle\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\tilde{\epsilon}_{k,\tau}] = \sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\mathbb{E}[\tilde{\epsilon}_{k,\tau}] = 0,$$

and thus by Azuma's inequality, for a fixed x at round t, with $\alpha = \sqrt{\frac{1}{2}\log\frac{2}{\sigma}}$, we have

$$\begin{aligned}&\mathbb{P}\left(|\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}(\textstyle\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\tilde{\boldsymbol{x}}_{k,\tau}\tilde{\epsilon}_{k,\tau})| \ge \alpha\|\boldsymbol{x}\|_{\tilde{\boldsymbol{M}}_t^{-1}}\right)\\ &\le 2\exp\left(-\frac{2\alpha^2\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}\boldsymbol{x}}{\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}(\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}\tilde{\boldsymbol{x}}_{k,\tau})^2}\right) \le 2\exp(-2\alpha^2)) = \sigma,\end{aligned}$$

since

$$\begin{aligned}\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}\boldsymbol{x} &= \boldsymbol{x}^T\tilde{\boldsymbol{M}}_t^{-1}\left(\tilde{\lambda}\boldsymbol{I} + \textstyle\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}\tilde{\boldsymbol{x}}_{k,\tau}\tilde{\boldsymbol{x}}_{k,\tau}^T\right)\tilde{\boldsymbol{M}}_t^{-1}\boldsymbol{x}\\ &\le \textstyle\sum_{\tau=1}^{t}\sum_{k\in\mathcal{K}_\tau}(\boldsymbol{x}^T\tilde{\boldsymbol{M}}_t\tilde{\boldsymbol{x}}_{k,\tau})^2.\end{aligned}$$

Thus, for a fixed x and fixed t, with probability $1-\sigma$,

$$\left\langle\boldsymbol{x}^T, \tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\right\rangle \le \left(\sqrt{\tilde{\lambda}}\|\tilde{\boldsymbol{\theta}}_*\|_2 + \sqrt{\frac{1}{2}\log\frac{2}{\sigma}}\right)\|\boldsymbol{x}\|_{\tilde{\boldsymbol{M}}_t^{-1}}.$$

Next, using thee above bound, we can bound $\|\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\|_{\tilde{\boldsymbol{M}}_t}$, where:

$$\|\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\|_{\tilde{\boldsymbol{M}}_t} = < \tilde{\boldsymbol{M}}_t^{1/2}\boldsymbol{X}, \tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_* >, \boldsymbol{X} = \frac{\tilde{\boldsymbol{M}}_t^{1/2}(\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*)}{\|\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\|_{\tilde{\boldsymbol{M}}_t}}.$$

We follow the *covering argument* in Chapter 20 of [8] to prove. First, we identify a finite set $C_\epsilon \subset R^d$ such that whatever value $\boldsymbol{X}$ takes, there exists some $\boldsymbol{x} \in C_\epsilon$ that are ϵ-close to $\boldsymbol{X}$. By definition, we have $\|\boldsymbol{X}\|_2^2 = 1$, which means $\boldsymbol{X} \in S^{d-1} = \{\boldsymbol{x} \in R^d : \|\boldsymbol{x}\|_2 = 1\}$. Thus, it is sufficient to cover S^{d-1}. Tor-lattimore et. al. [8] has proven the following Lemma. □

Lemma 6. *There exists a set $C_\epsilon \subset R^d$ with $|C_\epsilon| \le (3/\epsilon)^d$ such that for all $x \in S^{d-1}$, there exist a $y \in C_\epsilon$ with $\|x - y\| \le \epsilon$.*

Then we apply a union bound for the elements in C_ϵ, we have:

$$\mathbb{P}\left(\exists\boldsymbol{x} \in C_\epsilon, \left\langle\tilde{\boldsymbol{M}}_t^{1/2}x, \tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\right\rangle \ge \left(\sqrt{\tilde{\lambda}}\|\tilde{\boldsymbol{\theta}}_*\|_2 + \sqrt{\tfrac{1}{2}\log\tfrac{2|C_\epsilon|}{\sigma}}\right)\right) \le \sigma.$$

Then

$$\begin{aligned}&\|\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\|_{\tilde{\boldsymbol{M}}_t} = \max_{\boldsymbol{x}\in S^{d-1}}\left\langle\tilde{\boldsymbol{M}}_t^{1/2}\boldsymbol{x}, \tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\right\rangle\\ &= \max_{\boldsymbol{x}\in S^{d-1}}\min_{\boldsymbol{y}\in C_\epsilon}\left[\left\langle\tilde{\boldsymbol{M}}_t^{1/2}(\boldsymbol{x}-\boldsymbol{y}), \tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\right\rangle + \left\langle\tilde{\boldsymbol{M}}_t^{1/2}\boldsymbol{y}, \tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\right\rangle\right]\\ &\le \max_{\boldsymbol{x}\in S^{d-1}}\min_{\boldsymbol{y}\in C_\epsilon}\left[\|\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\|_{\tilde{\boldsymbol{M}}_t}\|\boldsymbol{x}-\boldsymbol{y}\|_2 + \sqrt{\tilde{\lambda}}\|\tilde{\boldsymbol{\theta}}_*\|_2 + \sqrt{\tfrac{1}{2}\log\tfrac{2|C_\epsilon|}{\sigma}}\right]\\ &\le \epsilon\|\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\|_{\tilde{\boldsymbol{M}}_t} + \sqrt{\tilde{\lambda}}\|\tilde{\boldsymbol{\theta}}_*\|_2 + \sqrt{\tfrac{1}{2}\log\tfrac{2|C_\epsilon|}{\sigma}}.\end{aligned}$$

We set $\epsilon = \frac{1}{2}$. Up to round t, we only update $\tilde{\boldsymbol{\theta}}_t$ at most $b(t)$ times, thus by the union bound, at each round t, with probability $1-\sigma$:

$$\|\tilde{\boldsymbol{\theta}}_t - \tilde{\boldsymbol{\theta}}_*\|_{\tilde{\boldsymbol{M}}_t} \le \sqrt{2\left(d\log 6 + \log(\tfrac{2b(t)}{\sigma})\right)} + 2\sqrt{\tilde{\lambda}}\|\tilde{\boldsymbol{\theta}}_*\|_2.$$

This proof is then complete.

Proof of Theorem 4:

Proof. Let a_t^* denote the best arm at round t, and $c_t(\boldsymbol{x}_{a,t}) = \alpha_t\|\boldsymbol{x}_{a,t}\|_{\boldsymbol{M}_t^{-1}}$, and $\tilde{c}_t(\boldsymbol{x}_{a,t}) = \tilde{\alpha}_t\|\boldsymbol{x}_{a,t}^T\boldsymbol{M}_t^{-1}\|_{\tilde{\boldsymbol{M}}_t^{-1}}$. Then the regret at round t is:

$$\begin{aligned}R_t &= \boldsymbol{\theta}_*^T\boldsymbol{x}_{a_t^*,t} - \boldsymbol{\theta}_*^T\boldsymbol{x}_{a_t,t}\\ &= \boldsymbol{\theta}_*^T\boldsymbol{x}_{a_t^*,t} - \boldsymbol{\theta}_t^T\boldsymbol{x}_{a_t^*,t} + \boldsymbol{\theta}_t^T\boldsymbol{x}_{a_t^*,t} + c_t(\boldsymbol{x}_{a_t^*,t}) + \tilde{c}_t(\boldsymbol{x}_{a_t^*,t})\\ &\quad - c_t(\boldsymbol{x}_{a_t^*,t}) - \tilde{c}_t(\boldsymbol{x}_{a_t^*,t}) - \boldsymbol{\theta}_*^T\boldsymbol{x}_{a_t,t}\\ &\le (\boldsymbol{\theta}_* - \boldsymbol{\theta}_t)^T\boldsymbol{x}_{a_t^*,t} + \boldsymbol{\theta}_t^T\boldsymbol{x}_{a_t,t} + c_t(\boldsymbol{x}_{a_t,t}) + \tilde{c}_t(\boldsymbol{x}_{a_t,t})\\ &\quad - c_t(\boldsymbol{x}_{a_t^*,t}) - \tilde{c}_t(\boldsymbol{x}_{a_t^*,t}) - \boldsymbol{\theta}_*^T\boldsymbol{x}_{a_t,t} \qquad (1)\\ &\le 2c_t(\boldsymbol{x}_{a_t,t}) + 2\tilde{c}_t(\boldsymbol{x}_{a_t,t}) \qquad (2)\end{aligned}$$

Inequality (1) is due to the arm selection strategy, i.e.,

$$a_t = \arg\max_{a\in\mathcal{A}_t}\boldsymbol{x}_{a,t}^T\boldsymbol{\theta}_t + c_t(\boldsymbol{x}_{a,t}) + \tilde{c}_t(\boldsymbol{x}_{a,t}).$$

Inequality (2) is from $|\boldsymbol{x}_{a,t}^T(\boldsymbol{\theta}_t - \boldsymbol{\theta}_*)| \leq c_t(\boldsymbol{x}_{a,t}) + \tilde{c}_t(\boldsymbol{x}_{a,t}), \forall a \in \mathcal{A}$. For the following analysis, we abbreviate $\boldsymbol{x}_{a_t,t}$ as $\boldsymbol{x}_t$. Thus, the cumulative regret until T is:

$$R(T) = \sum_{t=1}^T R_t \leq 2\sum_{t=1}^T \left(c_t(\boldsymbol{x}_{a_t,t}) + \tilde{c}_t(\boldsymbol{x}_{a_t,t})\right)$$
$$\leq 2\lambda\alpha_T\sqrt{T\sum_{t=1}^T \|\boldsymbol{x}_t\|^2_{\boldsymbol{M}_t^{-1}}} + 2(1-\lambda)\tilde{\alpha}_T\sqrt{T\sum_{t=1}^T \|\boldsymbol{x}_t^T\boldsymbol{M}_t^{-1}\|^2_{\tilde{\boldsymbol{M}}_t^{-1}}}.$$

The last inequality is from the Cauchy-Schwarz inequality, and $\alpha_t, \tilde{\alpha}_t$ are non-decreasing. Moreover,

$$\|\boldsymbol{x}_t^T\boldsymbol{M}_t^{-1}\|^2_{\tilde{\boldsymbol{M}}_t^{-1}} = \boldsymbol{x}_t^T\boldsymbol{M}_t^{-1}\tilde{\boldsymbol{M}}_t^{-1}\boldsymbol{M}_t^{-1}\boldsymbol{x}_t \leq \frac{1}{\tilde{\lambda}(1-\lambda)}\|\boldsymbol{x}_t\|^2_{\boldsymbol{M}_t^{-1}}, \quad (11)$$

Thus, the cumulative regret is bounded by:

$$R(T) \leq 2\left(\lambda\alpha_T + \sqrt{\tfrac{1-\lambda}{\tilde{\lambda}}}\tilde{\alpha}_T\right)\sqrt{T\sum_{t=1}^T \|\boldsymbol{x}_t\|^2_{\boldsymbol{M}_t^{-1}}}. \quad (12)$$

Together with α_t in Lemma 1, and $\tilde{\alpha}_t$ in Lemma 3, as well as Lemma 11 in [1], we can finish the proof. □

REFERENCES

[1] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. 2011. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*. 2312–2320.

[2] Shipra Agrawal and Navin Goyal. 2013. Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*. 127–135.

[3] Yuheng Bu and Kevin Small. 2018. Active Learning in Recommendation Systems with Multi-level User Preferences. *arXiv preprint arXiv:1811.12591* (2018).

[4] Nicolo Cesa-Bianchi, Claudio Gentile, and Giovanni Zappella. 2013. A gang of bandits. In *Advances in Neural Information Processing Systems*. 737–745.

[5] Konstantina Christakopoulou, Alex Beutel, Rui Li, Sagar Jain, and Ed H Chi. 2018. Q&R: A two-stage approach toward interactive recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 139–148.

[6] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. 2016. Towards conversational recommender systems. In *KDD*. ACM, 815–824.

[7] Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. 2010. Parametric bandits: The generalized linear case. In *NIPS*. 586–594.

[8] Tor Lattimore and Csaba Szepesvári. [n.d.]. Bandit algorithms. ([n. d.]).

[9] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 661–670.

[10] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 297–306.

[11] Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. 2016. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 539–548.

[12] Yueming Sun and Yi Zhang. 2018. Conversational Recommender System. *arXiv preprint arXiv:1806.03277* (2018).

[13] Huazheng Wang, Qingyun Wu, and Hongning Wang. 2016. Learning hidden features for contextual bandits. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1633–1642.

[14] Qingyun Wu, Huazheng Wang, Quanquan Gu, and Hongning Wang. 2016. Contextual bandits in a collaborative environment. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 529–538.

[15] Tong Yu, Yilin Shen, and Hongxia Jin. 2019. An Visual Dialog Augmented Interactive Recommender System. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 157–165.

[16] Yisong Yue, Sue Ann Hong, and Carlos Guestrin. 2012. Hierarchical exploration for accelerating contextual bandits. *arXiv preprint arXiv:1206.6454* (2012).

[17] Chunqiu Zeng, Qing Wang, Shekoofeh Mokhtari, and Tao Li. 2016. Online context-aware recommendation with time varying multi-armed bandit. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2025–2034.

[18] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of CIKM*. ACM, 177–186.