A Contextual-Bandit Approach to Online Learning to Rank for Relevance and Diversity

Li, C.; Feng, H.; de Rijke, M.

Publication date
2019

Document Version
Submitted manuscript

License
Unspecified

Link to publication

Citation for published version (APA):
Li, C., Feng, H., & de Rijke, M. (2019). *A Contextual-Bandit Approach to Online Learning to Rank for Relevance and Diversity*. ArXiv. https://arxiv.org/abs/1912.00508v1

A Contextual-Bandit Approach to Online Learning to Rank for Relevance and Diversity

Chang Li[1] and **Haoyun Feng**[2] and **Maarten de Rijke**[3]

Abstract. Online learning to rank (LTR) focuses on learning a policy from user interactions that builds a list of items sorted in decreasing order of the item utility. It is a core area in modern interactive systems, such as search engines, recommender systems, or conversational assistants. Previous online LTR approaches either assume the relevance of an item in the list to be independent of other items in the list or the relevance of an item to be a submodular function of the utility of the list. The former type of approach may result in a list of low diversity that has relevant items covering the same aspects, while the latter approaches may lead to a highly diversified list but with some non-relevant items.

In this paper, we study an online LTR problem that considers both item relevance and topical diversity. We assume cascading user behavior, where a user browses the displayed list of items from top to bottom and clicks the first attractive item and stops browsing the rest. We propose a hybrid contextual bandit approach, called CascadeHybrid, for solving this problem. CascadeHybrid models item relevance and topical diversity using two independent functions and simultaneously learns those functions from user click feedback. We derive a gap-free bound on the n-step regret of CascadeHybrid. We conduct experiments to evaluate CascadeHybrid on the MovieLens and Yahoo music datasets. Our experimental results show that CascadeHybrid outperforms the baselines on both datasets.

1 INTRODUCTION

With the growing volume of information available online, there is an increasing need to improve the effectiveness of interactive systems. Ranking is at the heart of modern interactive systems and LTR addresses the challenge of improving retrieval effectiveness by learning an effective ranker [25]. Two essential properties that help determine the quality of a ranked list of items returned by a search engine or recommender systems are the *relevance* of individual items included in the list and the *diversity* of the list as a whole. Here, *relevance* refers to given a query, document pair and how well the document covers the information need expressed by query [5, 32]. And *diversity* refers to a list of items and the degree to which redundancy of aspects addressed by the items is avoided and the degree to which the items in the list manage to cover a broad range of aspects [12, 31].

Traditionally, rankers have been learned offline from labeled data [25]. But there is a growing realization of the limitations of this supervised setup based on manually created training material [14]. In response, learning to rank from user interactions has been seen rapid growth in online, offline, and counterfactual fashions [15–17, 36]. In this paper, we focus on online LTR. While online LTR has developed rapidly in recent years, with progress in areas such as safety [22], non-standard interfaces [28], and generalizability [27], diversity has received relatively less feedback in online LTR.

Previous work that focuses on online diverse LTR targets topical diversity and tends to ignore relevance of individual results. Yue and Guestrin [35] develop an online diverse LTR algorithm by optimizing submodular utility models [35]. Hiranandani et al. [13] improve online diverse LTR by bringing the cascading assumption into the objective function. More specifically, previous work assumes that items can be represented using a set of topics. The task of a diversified ranking algorithm is then taken to be to ensure broad coverage of the topics.

In this work, we focus on a novel online LTR problem that considers both item relevance and topical diversity. We propose CascadeHybrid to solve this problem, which simultaneously optimizes both relevance of individual documents and topical diversity of the ranked list.

More specifically, this paper makes the following contributions: (1) We focus on a novel online learning to rank problem that concerns target item relevance and result diversity, and formalize it as a variant of cascading click bandits (CB) (Section 3.1). (2) We propose CascadeHybrid, which utilizes a hybrid model, to solve this problem, Section 3.3. (3) We analyze CascadeHybrid and show the upper bound on the n-step regret to be $O(\sqrt{n})$ (Section 4). (4) We evaluate CascadeHybrid on two real-world recommendation datasets: MovieLens and Yahoo (Section 5).

2 BACKGROUND

In this section, we recapitulate the cascade model (CM), CB, and the submodular coverage model. Throughout the paper, we consider the ranking problem of L item candidates and K positions with $K \leq L$. We denote $1, \ldots, n$ by $[n]$ and for the collection of items we write $\mathcal{D} = [L]$. A ranked list contains $K \leq L$ items and is denoted by $\mathcal{R} \in \Pi_K(\mathcal{D})$, where $\Pi_K(\mathcal{D})$ is the set of the permutations of K distinct items from the collection $\mathcal{D}$. The item at the k-th position of the list is denoted by $\mathcal{R}(k)$ and, if $\mathcal{R}$ contains an item i, the position of this item in $\mathcal{R}$ is denoted by $\mathcal{R}^{-1}(i)$. All vectors are column vectors. We use bold font to indicate a vector and bold font with a capital letter to indicate a matrix. We write $\mathbf{I}_d$ to denote the $d \times d$ identity matrix.

2.1 Cascade model

Click models have been widely used to interpret user's click interaction behavior; cf. [7] for an overview. Briefly, a user is shown a ranked list $\mathcal{R}$, and then browses the list and leaves click feedback. Every click model makes unique assumptions and models a type of user interaction behavior. In this paper, we consider a simple but widely

[1] University of Amsterdam, c.li@uva.nl
[2] Bloomberg, hfeng19@bloomberg.net
[3] University of Amsterdam, derijke@uva.nl

used click model, the *cascade model* [8, 19, 20, 38], which makes the cascade assumption about user behavior. Under the cascade assumption, a user browses a ranked list $\mathcal{R}$ from the first item to the last one by one and clicks the first attractive item. After the click, the user leaves the list and stops browsing the remaining items. The click on an examined item $\mathcal{R}(i)$ can be modeled as a Bernoulli random variable with a probability of $\alpha(\mathcal{R}(i))$, which is also called the *attraction probability*. Here, the CM assumes that each item attracts the user independent of other items in $\mathcal{R}$. Thus, a CM is parametrized by a set of attraction probabilities $\boldsymbol{\alpha} \in [0,1]^L$. The examination probability of item $\mathcal{R}(i)$ is 1 if $i = 1$, otherwise $1 - \prod_{j=1}^{i-1}(1-\alpha(\mathcal{R}(j)))$.

A vanilla CM is only able to capture the first click in a session, and there are various extensions of CM to model multi-click scenarios [6, 7, 10]. However, we still focus on CM, because it has been shown in multiple publications [7, 19, 22] that CM achieves good performance in both online and offline setups.

2.2 Cascading click bandits

Cascade click bandits (CB) is a type of online variant of CM and has been widely studied in literature [13, 19, 20, 38]. A CB is represented by a tuple $(\mathcal{D}, K, P)$, where P is a binary distribution over $\{0,1\}^L$. The learning agent interacts with CB and learns from the feedback. At each step t, the agent builds a ranked list $\mathcal{R}_t \in \Pi_K(\mathcal{D})$ depending on observations in the previous $t-1$ steps and shows it to the user. The user browses the list with cascading behavior and leaves click feedback. Since a CM accepts at most one click, we write $c_t \in [K+1]$ as the click indicator, where c_t indicates the position of the click and $c_t = K+1$ indicates no click. Let $A_t \in \{0,1\}^L$ be the attraction indicator, where A_t is drawn from P and $A_t(\mathcal{R}_t(i)) = 1$ indicates that item $\mathcal{R}_t(i)$ attracts the user at step t. The number of clicks at step t is considered as the reward and computed as follows:

$$r(\mathcal{R}_t, A_t) = 1 - \prod_{i=1}^{K}(1 - A_t(\mathcal{R}_t(i))). \tag{1}$$

Then, we assume that the attraction indicators of items are distributed independently as Bernoulli variables:

$$P(A) = \prod_{i \in \mathcal{D}} P_{\alpha(i)}(A(i)), \tag{2}$$

where $P_{\alpha(i)}(\cdot)$ is the Bernoulli distribution with mean $\alpha(i)$. The expected number of clicks at step t is computed as $\mathbb{E}\left[r(\mathcal{R}_t, A_t)\right] = r(\mathcal{R}_t, \boldsymbol{\alpha})$. The goal of the agent is to maximize the expected number of clicks in n steps or minimize the expected n-step regret:

$$R(n) = \sum_{t=1}^{n} \mathbb{E}\left[\max_{\mathcal{R} \in \Pi_K(\mathcal{D})} r(\mathcal{R}, \boldsymbol{\alpha}) - r(\mathcal{R}_t, A_t)\right]. \tag{3}$$

The CB has several variants depending on assumptions on the attraction probability $\boldsymbol{\alpha}$. Briefly, the cascade linear bandits model [38] assumes that an item a is represented by a feature vector $\mathbf{z}_a \in \mathbb{R}^m$ and that the attraction probability of an item a to a user is a linear combination of features: $\alpha(a) \approx \mathbf{z}_a^T \beta^*$, where $\beta^* \in \mathbb{R}^m$ is an unknown parameter. Meanwhile, the cascade linear submodular bandits model [13] assumes that the attraction probability is a submodular function of the higher ranked items and focuses on the diverse ranking.

2.3 Submodular coverage model

Let $g(\cdot)$ be a set function that maps a set to a real value, and is monotone and submodular. That is, given an item set $\mathcal{A}$, an item set $\mathcal{B} \subseteq \mathcal{A}$ and an item a, $g(\cdot)$ has two properties:

monotonicity: $g(\mathcal{A} \cup \{a\}) \geq g(\mathcal{A})$,
submodularity: $g(\mathcal{B} \cup \{a\}) - g(\mathcal{B}) \geq g(\mathcal{A} \cup \{a\}) - g(\mathcal{A})$.

Briefly, monotonicity and submodularity together provide the framework to capture the diminishing gain in the utility of a list: the gain in utility of adding an item to a small set $\mathcal{B}$ is larger than or equal to that of adding an item to a large set $\mathcal{A}$. One widely used submodular function for result diversification is the *probabilistic coverage model* [2, 3, 13, 29, 35].

Suppose that an item $a \in \mathcal{D}$ is represented by a d-dimensional vector $\mathbf{x}_a \in [0,1]^d$. Each entry of the vector $\mathbf{x}_a(j)$ describes the probability of item a covering topic j. Given a list $\mathcal{A}$, the probability of $\mathcal{A}$ covering topic j is

$$g_j(\mathcal{A}) = 1 - \prod_{a \in \mathcal{A}}(1 - \mathbf{x}_a(j)). \tag{4}$$

The gain in topic coverage of adding an item a to $\mathcal{A}$ is:

$$\Delta(a \mid \mathcal{A}) = (\Delta_1(a \mid \mathcal{A}), \ldots, \Delta_d(a \mid \mathcal{A})), \tag{5}$$

where $\Delta_j(a \mid \mathcal{A}) = g_j(\mathcal{A} \cup \{a\}) - g_1(\mathcal{A})$. With this model, the attraction probability of the i-th item in a ranked list $\mathcal{R}$ is defined as:

$$\alpha(\mathcal{R}(i)) = \boldsymbol{\omega}_{\mathcal{R}(i)}^T \boldsymbol{\theta}^*, \tag{6}$$

where $\boldsymbol{\omega}_{\mathcal{R}(i)} = \Delta(\mathcal{R}(i) \mid (\mathcal{R}_t(1), \ldots, \mathcal{R}_t(i-1))$ and $\boldsymbol{\theta}^*$ is the unknown user preference to different topics [13]. In Eq. (6), the attraction probability of an item depends on the items ranked above it; $\alpha(\mathcal{R}(i))$ is small if $\mathcal{R}(i)$ covers similar topics as higher ranked items. CascadeLSB has been proposed to solve cascading bandits with this type of attraction probability and aims at building diverse ranked lists.

3 ALGORITHM

This paper focuses on an online LTR problem that considers both the relevance of items and the diversity of the ranked list. In this section, we first formulate our online learning to rank problem, and then propose CascadeHybrid to solve it.

3.1 Problem formulation

We study a variant of cascading bandits, where the attraction probability of an item in a ranked list depends on two parts: item relevance and item novelty. Item relevance is independent on other items in the list. Novelty of an item depends on the topics covered by higher ranked items; a novel item brings a large gain in the topic coverage of the list, i.e., a large value in Eq. (6). Thus, given a ranked list $\mathcal{R}$, the attraction probability of item $\mathcal{R}(i)$ is defined as follows:

$$\alpha(\mathcal{R}(i)) = \mathbf{z}_a^T \boldsymbol{\beta}^* + \boldsymbol{\omega}_a^T \boldsymbol{\theta}^*, \tag{7}$$

where we write $a = \mathcal{R}(i)$ for simplicity, $\boldsymbol{\omega}_a = \Delta(a \mid (\mathcal{R}(1), \ldots, \mathcal{R}(i-1))$ is the topic coverage gain discussed in Section 2.3, $\mathbf{z}_a \in \mathbb{R}^m$ is the relevance feature, $\boldsymbol{\theta}^* \in \mathbb{R}^d$ and $\boldsymbol{\beta}^* \in \mathbb{R}^m$ are two unknown parameters that characterize the user preference.

Now, we define the learning problem as a tuple $(\mathcal{D}, \boldsymbol{\theta}^*, \boldsymbol{\beta}^*, K)$. Here, $\mathcal{D} = [L]$ is the item candidate and each item a can be represented by a feature vector $[\mathbf{x}_a^T, \mathbf{z}_a^T]^T$, where $\mathbf{x}_a \in [0,1]^d$ is the topic coverage of item a discussed in Section 2.3. K is the number of positions. The action space for the problem are all permutations of K individual items from $\mathcal{D}$, $\Pi_K(\mathcal{D})$. The reward of an action at step t is the number of clicks, defined in Eq. (1). Together with Eqs. (1), (2) and (7), the expectation of reward at step t is computed as follows:

$$\mathbb{E}\left[r(\mathcal{R}_t, A_t)\right] = 1 - \prod_{a \in \mathcal{R}_t} (1 - \mathbf{z}_a^T \boldsymbol{\beta}^* - \boldsymbol{\omega}_a^T \boldsymbol{\theta}^*). \tag{8}$$

In the rest of the paper, we write $r(\mathcal{R}_t) = \mathbb{E}\left[r(\mathcal{R}_t, A_t)\right]$ for short. And the goal of the learning agent is to maximize the reward or, equivalently, to minimize the n-step regret defined in Eq. (3).

3.2 Competing with a greedy benchmark

Finding the optimal set that maximizes the utility of a submodular function is an NP-hard problem [26]. In our setup, the attraction probability of each item also depends on the order in the list. To the best of our knowledge, we cannot find the optimal ranking

$$\mathcal{R}^* = \underset{\mathcal{R} \in \Pi_K(\mathcal{D})}{\arg\max}\ r(\mathcal{R}) \tag{9}$$

efficiently. Thus, we compete with a greedy benchmark that approximates the optimal ranking $\mathcal{R}^*$. The greedy benchmark chooses the items that has the highest attraction probability given the higher ranked items: for any positions $k \in [K]$,

$$\tilde{\mathcal{R}}(k) = \underset{a \in \mathcal{D} \setminus \{\tilde{\mathcal{R}}(1), \ldots, \tilde{\mathcal{R}}(k-1)\}}{\arg\max} \mathbf{z}_a^T \boldsymbol{\beta}^* + \boldsymbol{\omega}_a^T \boldsymbol{\theta}^*, \tag{10}$$

where $\tilde{\mathcal{R}}(k)$ is the ranked list output of the benchmark.

This greedy benchmark has been used in previous literature [13, 35]. As shown by Hiranandani et al. [13], in the CM, the greedy benchmark is at least a η-approximation of $\mathcal{R}^*$. That is, $r(\tilde{\mathcal{R}}) \geq \eta r(\mathcal{R}^*)$. where $\eta = (1 - \frac{1}{e}) \max\{\frac{1}{K}, 1 - \frac{K-1}{2}\alpha_{max}\}$, with $\alpha_{max} = \max_{a \in \mathcal{D}} \mathbf{z}_a^T \boldsymbol{\beta}^* + \mathbf{x}_a^T \boldsymbol{\theta}^*$. In the rest of the paper, we focus on competing with this greedy benchmark.

Algorithm 1 CascadeHybrid

Input: γ

1: // Initialization
2: $\mathbf{H}_1 \leftarrow \mathbf{I}_d, \mathbf{u}_1 \leftarrow \mathbf{0}_d, \mathbf{M}_1 \leftarrow \mathbf{I}_m, \mathbf{y}_1 \leftarrow \mathbf{0}_m, \mathbf{B}_1 = \mathbf{0}_{d \times m}$
3: **for** $t = 1, 2, \ldots, n$ **do**
4: // Estimate parameters
5: $\hat{\boldsymbol{\theta}}_t \leftarrow \mathbf{H}_t^{-1}\mathbf{u}_t, \hat{\boldsymbol{\beta}}_t \leftarrow \mathbf{M}_t^{-1}(\mathbf{y}_t - \mathbf{B}_t^T \hat{\boldsymbol{\theta}}_t)$
6: // Build ranked list
7: $\mathcal{S} \leftarrow \emptyset$
8: **for** $k = 1, 2, \ldots K$ **do**
9: **for** $a \in \mathcal{D} \setminus \mathcal{S}$ **do**
10: $\boldsymbol{\omega}_a \leftarrow \Delta(\mathbf{x}_a | \mathcal{S})$
11: $s_a \leftarrow$ Eq. (14)
12: $\mu_a \leftarrow$ Eq. (13)
13: **end for**
14: $a_k^t \leftarrow \arg\max_{a \in \mathcal{D} \setminus \mathcal{S}} \mu_a$
15: $\mathcal{S} \leftarrow \mathcal{S} + a_k^t$
16: **end for**
17: $\mathcal{R}_t = (a_1^t, \ldots, a_K^t)$ // Ranked list
18: Display $\mathcal{R}_t$ and observe click feedback $c_t \in \{1, \ldots, K+1\}$
19: $k_t \leftarrow \min(K, c_t)$
20: // Update statistics
21: $\mathbf{H}_t \leftarrow \mathbf{H}_t + \mathbf{B}_t \mathbf{M}_t^{-1} \mathbf{B}_t^T, \mathbf{u}_t \leftarrow \mathbf{u}_t + \mathbf{B}_t \mathbf{M}_t^{-1} \mathbf{y}_t$
22: **for** $a \in \mathcal{R}_t(1 : k_t)$ **do**
23: $\mathbf{M}_{t+t} \leftarrow \mathbf{M}_t + \mathbf{z}_a \mathbf{z}_a^T, \mathbf{B}_{t+1} \leftarrow \mathbf{B}_t + \boldsymbol{\omega}_a \mathbf{z}_a^T, \mathbf{H}_t \leftarrow \mathbf{H}_t + \boldsymbol{\omega}_a \boldsymbol{\omega}_a^T$
24: **end for**
25: **if** $c_t \leq K$ **then**
26: $\mathbf{y}_{t+1} \leftarrow \mathbf{y}_t + \mathbf{z}_{\mathcal{R}_t(c_t)}, \mathbf{u}_t \leftarrow \mathbf{u}_t + \boldsymbol{\omega}_{\mathcal{R}_t(c_t)}$
27: **end if**
28: $\mathbf{H}_{t+1} \leftarrow \mathbf{H}_t - \mathbf{B}_{t+1} \mathbf{M}_{t+1}^{-1} \mathbf{B}_{t+1}^T, \mathbf{u}_{t+1} \leftarrow \mathbf{u}_t - \mathbf{B}_{t+1} \mathbf{M}_{t+1}^{-1} \mathbf{y}_{t+1}$
29: **end for**

3.3 CascadeHybrid

The attraction probability of an item is a hybrid of a modular function and a submodular function. Thus, we propose a hybrid model based algorithm, called CascadeHybrid, to solve the aforementioned online learning problem. The algorithm has access to item features and uses the probabilistic coverage model to compute the gains in topic coverage of each item. Initially, the user preference $\boldsymbol{\theta}^*$ and $\boldsymbol{\beta}^*$ are unknown to CascadeHybrid. They are estimated from interactions with the users. The only tunable hyperparameter for CascadeHybrid is $\gamma \in \mathbb{R}_+$, which controls the exploration: a larger value of γ means more exploration.

The details of CascadeHybrid are summarized in Algorithm 1. At the beginning of each step t (line 5), CascadeHybrid estimates the user preference as $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{\beta}}_t$ based on the previous $t-1$ step observations. $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{\beta}}_t$ can be viewed as maximum likelihood estimators on the rewards and computed as follows:[4]

$$\hat{\boldsymbol{\theta}}_t = \mathbf{H}_t^{-1}\mathbf{u}_t, \quad \hat{\boldsymbol{\beta}}_t = \mathbf{M}_t^{-1}(\mathbf{y}_t - \mathbf{B}_t^T \hat{\boldsymbol{\theta}}_t), \tag{11}$$

where $\mathbf{M}_t, \mathbf{H}_t, \mathbf{B}_t$ and $\mathbf{y}_t, \mathbf{u}_t$ summarize the features and click feedback of all observed items in the previous $t-1$ steps, and are computed as follows:

$$\begin{aligned}
\mathbf{M}_t &= \mathbf{I}_m + \sum_{i=1}^{t-1} \sum_{a \in \mathcal{O}_i} \mathbf{z}_a \mathbf{z}_a^T \\
\mathbf{B}_t &= \mathbf{0}_{d \times m} + \sum_{i=1}^{t-1} \sum_{a \in \mathcal{O}_i} \boldsymbol{\omega}_a \mathbf{z}_a^T \\
\mathbf{H}_t &= \mathbf{I}_d + \sum_{i=1}^{t-1} \sum_{a \in \mathcal{O}_i} \boldsymbol{\omega}_a \boldsymbol{\omega}_a^T - \mathbf{B}_t \mathbf{M}_t^{-1} \mathbf{B}_t^T \\
\mathbf{y}_t &= \mathbf{0}_m + \sum_{i=1}^{t-1} \sum_{a \in \mathcal{O}_i} \mathbf{z}_a \mathbb{1}(\mathcal{R}_i^{-1}(a) = c_i) \\
\mathbf{u}_t &= \mathbf{0}_d + \sum_{i=1}^{t-1} \sum_{a \in \mathcal{O}_i} \boldsymbol{\omega}_a \mathbb{1}(\mathcal{R}_i^{-1}(a) = c_i) - \mathbf{B}_t \mathbf{M}_t^{-1} \mathbf{y}_t,
\end{aligned} \tag{12}$$

where $\mathbf{0}_{d \times m}$ is the $d \times m$ zero matrix, $\mathbb{1}(\cdot)$ is the indicator function and $\mathcal{O}_t$ is the set of observed items in step t.

Then, CascadeHybrid builds the ranked list $\mathcal{R}_t$, sequentially (line 7–17). Particularly, for each position k, CascadeHybrid makes an optimistic estimate of the attraction probability of each item (line 9–13) and chooses the one with the highest estimated attraction probability (line 14). This is known as the principle of optimism in the face of uncertainty [4], and the estimator for an item a is called upper

[4] The derivation is based on the matrix block-wise inversion. Due to the space limitation, we omit the derivation process.

confidence bound (UCB):

$$\mu_a = \boldsymbol{\omega}_a^T \hat{\boldsymbol{\theta}}_t + \mathbf{z}_a^T \hat{\boldsymbol{\beta}}_t + \gamma\sqrt{s_a}, \tag{13}$$

with

$$s_a = \boldsymbol{\omega}_a^T \mathbf{H}_t^{-1} \boldsymbol{\omega}_a - 2\boldsymbol{\omega}_a^T \mathbf{H}_t^{-1} \mathbf{B}_t \mathbf{M}_t^{-1} \mathbf{z}_a + \mathbf{z}_a \mathbf{M}_t^{-1} \mathbf{z}_a + \mathbf{z}_a \mathbf{M}_t^{-1} \mathbf{B}_t^T \mathbf{H}_t^{-1} \mathbf{B}_t \mathbf{M}_t^{-1} \mathbf{z}_a. \tag{14}$$

Finally, CascadeHybrid displays the ranked list $\mathcal{R}_t$ to the user and collects click feedback (line 18–28). Since CascadeHybrid only accepts one click, we use $c_t \in [1, K+1]$ to indicate the position of the click. $c_t = K+1$ indicates that no item in $\mathcal{R}_t$ is clicked.

3.4 Computational complexity

Computing matrix inverse is the main computational cost of Algorithm 1. Practically, we can use the Woodbury matrix identity [9] to update $\mathbf{H}_t^{-1}$ and $\mathbf{M}_t^{-1}$ instead of $\mathbf{H}_t$ and $\mathbf{M}_t$, which is $O(m^2 + d^2)$. At each step, CascadeHybrid chooses K items out of L, greedily. For each item, computing UCB requires $O(m^2 + d^2 + md)$. In total, the per-step computational complexity of CascadeHybrid is $O(LK(m^2 + d^2 + md))$.

4 ANALYSIS

Since CascadeHybrid competes with the greedy benchmark, we focus on the η-scaled expected n-step regret which is defined as:

$$R_\eta(n) = \sum_{t=1}^{n} \mathbb{E}\left[\eta r(\mathcal{R}^*, \boldsymbol{\alpha}) - r(\mathcal{R}_t, A_t)\right], \tag{15}$$

where $\eta = (1 - \frac{1}{e}) \max\{\frac{1}{K}, 1 - \frac{K-1}{2}\alpha_{max}\}$. This is a reasonable metric, since computing the optimal $\mathcal{R}^*$ is computationally inefficient. The similar scaled regret has previously been used in diversity problems [13, 30, 35]. For simplicity, we write $\mathbf{w}^* = [\boldsymbol{\theta}^{*T}, \boldsymbol{\beta}^{*T}]^T$. Then, we bound the η-scaled regret of CascadeHybrid as follows:

Theorem 1. *For $\|\mathbf{w}^*\|_2 \leq 1$ and any*

$$\gamma \geq \sqrt{(m+d)\log(1 + \frac{nk}{m+d}) + 2\log(n)} + \|\mathbf{w}^*\|_2, \tag{16}$$

we have

$$R_\eta(n) \leq 2\gamma\sqrt{2nK(m+d)\log\left(1 + \frac{nK}{m+d}\right)} + 1. \tag{17}$$

Combining Eqs. (16) and (17), we have $R_\eta(n) = \tilde{O}((m+d)\sqrt{Kn})$, where $\tilde{O}$ notation ignores logarithmic factors. Our bound has three characteristics: (1) Theorem 1 states a gap-free bound, where the factor $\sqrt{n}$ is considered near optimal; (2) This bound is linear to the number of features, which is a common dependence in learner bandit algorithms [1]; (3) Our bound is $\tilde{O}(\sqrt{K})$ lower than other linear bandit bounds in CB [13, 38]. We include a proof of Theorem 1 in Appendix A.

5 EXPERIMENTS

This section starts with the experimental setup, where we introduce the datasets, click simulator and baselines. After that we report our experimental results.

5.1 Experimental setup

Off-policy evaluation [23] is one approach to evaluate interaction algorithms without live experiments. However, in our problem, the action space is exponential in K, which is too large for commonly used off-policy evaluation methods. As an alternative, we evaluate the CascadeHybrid in a simulated interaction environment, where the simulator is built based on offline datasets. This is a widely used evaluation setup in literature [13, 14, 19, 38].

Datasets. We evaluate CascadeHybrid on two datasets: MovieLens 20M [11] and Yahoo.[5] The MovieLens dataset contains 20M ratings on 27k movies by 138k users and 20 genres.[6] Each movie belongs to at least one genre. The Yahoo dataset contains over 700M ratings of 136k songs given by 1.8M users and genre attributes of each music; we consider the top level attribute, which has 20 different genres; each song belongs to a single genre. All the ratings in two datasets are in the 5-start scale. All movies and songs are considered as items and genres are considered as topics.

Data preprocessing. We follow a widely used data preprocessing approach [13, 24, 38]. First, we extract the 1k most active users and the 1k most rated items. Let $\mathcal{U} = [1000]$ be the user set, and $\mathcal{D} = [1000]$ be the item set. Then, the ratings are mapped in binary scale: rating 5 is converted to 1 and others to 0. After this mapping, in the MovieLens dataset, about 7% of the user-item pairs are attractive and, in the Yahoo dataset, about 11% of user-item pairs are attractive. Then, we use the matrix $\mathbf{F} \in \{0,1\}^{|\mathcal{U}|\times|\mathcal{D}|}$ to capture the converted ratings and use the matrix $\mathbf{G} \in \{0,1\}^{|\mathcal{D}|\times d}$ to record the items and topics, where d is the number of topics and each entry $\mathbf{G}_{jk} = 1$ indicates that item j belongs to topic k.

Click simulator. The click simulator that is used to simulate click feedback on a ranked list considers both the item relevance and the diversity of the list. To design such a simulator, we combine the simulators used in [24] and [13]. Because of the cascading assumption, we only need to define the way of computing attraction probability of an item in a list.

We first divide users into two groups evenly, and get $\mathbf{F}_{train}$ and $\mathbf{F}_{test}$. $\mathbf{F}_{train}$ and $\mathbf{G}$ are used to estimate the topic coverage of items used by the online algorithms, while $\mathbf{F}_{test}$ and $\mathbf{G}$ are used to define the click simulator. In this setup, for an item a, the topic coverage used by the online algorithms is slightly different from those used by the click simulator. This is to mimic the real-world scenario that the online algorithms estimate user preferences without knowing the perfect topic coverage of items.

First, we follow the process in [13]. If item a belongs to topic j, we compute the topic coverage of item a to topic j as the quotient of the number of users rating item a to be attractive to the number of users who rate at least one item in topic j to be attractive:

$$\mathbf{x}_{a,j} = \frac{\sum_{u\in\mathcal{U}} F_{u,a} G_{a,j}}{\sum_{u\in\mathcal{U}} \mathbb{1}\{\exists a' \in \mathcal{D} : F_{u,a'} G_{a',j} > 0\}}. \tag{18}$$

Given user u, the preference for topic j is computed as the number of items rated to be attractive in topic j over the number of items in all topics rated by u to be attractive:

$$\boldsymbol{\theta}_j^* = \frac{\sum_{a\in\mathcal{D}} F_{u,a} G_{a,j}}{\sum_{j'\in[d]} \sum_{a'\in\mathcal{D}} F_{u,a'} G_{a',j'}}. \tag{19}$$

5 R2 - Yahoo! Music User Ratings of Songs with Artist, Album, and Genre Meta Information, v. 1.0 https://webscope.sandbox.yahoo.com/catalog.php?datatype=r

6 In both datasets, one of the 20 genres is called *unknown*.

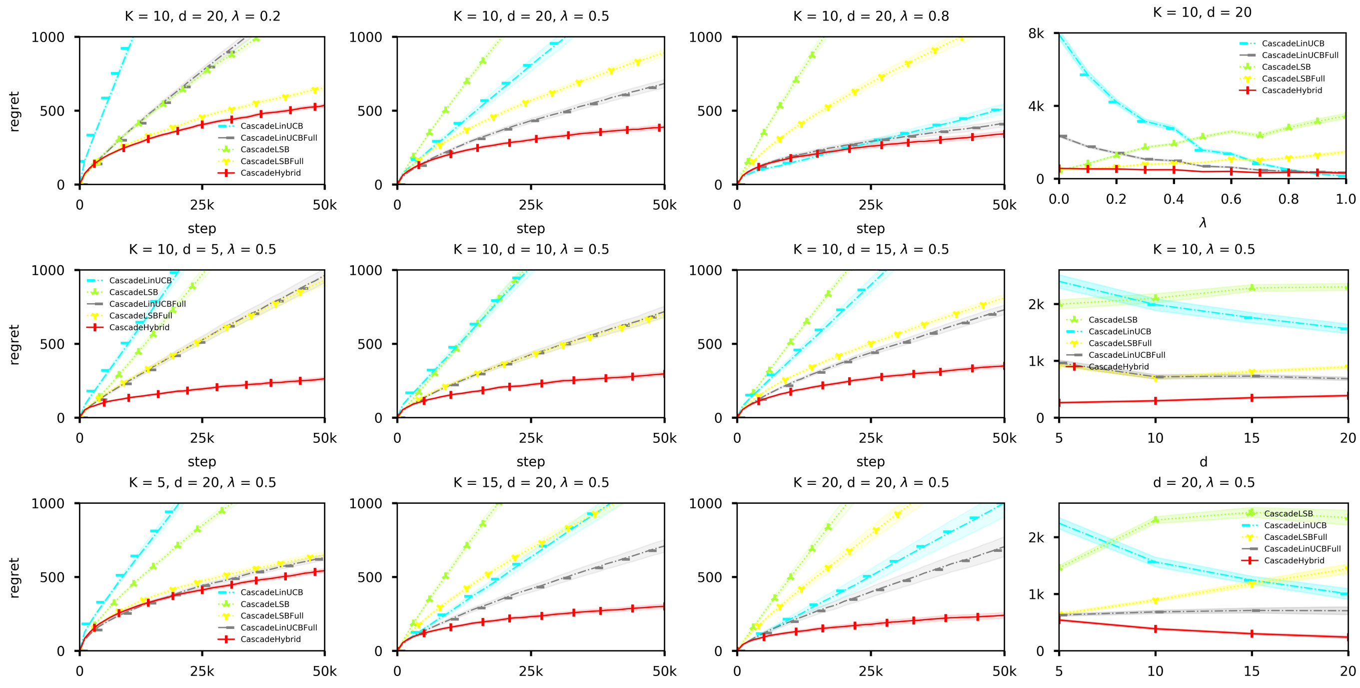

Figure 1: n-step regret on the MovieLens dataset. Results are averaged over 500 users with 2 repeats per user. Shaded areas are the standard errors of estimates.

For all items, we have $\mathbf{x}_a^T \boldsymbol{\theta}^* \in [0, 1]$, since $\forall a \in \mathcal{D} : \sum_{j \in [d]} \mathbf{x}_{a,j} \leq 1$ and $\sum_{j \in [d]} \boldsymbol{\theta}_j^* = 1$.

We then follow [24] to obtain the relevance part of the attraction probability. The relevance features $\mathbf{z}$ are obtained by conducting singular-value decomposition on $\mathbf{F}_{train}$. We pick the 10 largest singular values and thus the dimension of relevance features is $m = 10$. Then, we normalize each relevance feature by the transformation: $\mathbf{z}_a \leftarrow \frac{\mathbf{z}_a}{\|\mathbf{z}_a\|_2}$, where $\|\mathbf{z}_a\|_2$ is the L2 norm of $\mathbf{z}_a$. The user preference $\boldsymbol{\beta}^*$ is computed by solving the least square on $\mathbf{F}_{test}$ and then $\boldsymbol{\beta}^*$ is normalized by the same transformation. Note that $\forall a \in \mathcal{D} : \mathbf{z}_a^T \boldsymbol{\beta}^* \in [0, 1]$, since $\|\mathbf{z}_a\|_2 = 1$ and $\|\boldsymbol{\beta}^*\|_2 = 1$.

Finally, we combine the two parts and obtain our click simulator. Since $\mathbf{x}_a^T \boldsymbol{\theta}^* \in [0, 1]$ and $\mathbf{z}_a^T \boldsymbol{\beta}^* \in [0, 1]$, if we directly combine the two parts, the attraction probability may be larger than 1. Thus, we introduce a trade-off parameter $\lambda \in [0, 1]$ and compute the attraction probability as:

$$\alpha(a) = (1 - \lambda)\mathbf{x}_a^T \boldsymbol{\theta}^* + \lambda \mathbf{z}_a^T \boldsymbol{\beta}^*. \tag{20}$$

By changing the value of λ, we simulate different types of user preference: a larger value of λ means that the user prefers items to be relevant, while a smaller value of λ means that the user prefers the ranked list to be diverse.

Baselines. We compare CascadeHybrid to two online algorithms, each of which has two configurations. In total, we have four baselines, namely CascadeLinUCB and CascadeLinUCBFull [38], and CascadeLSB and CascadeLSBFull [13]. The first two only consider relevance ranking in the objective function. The differences are that CascadeLinUCB takes $\mathbf{z}$ as the features, while CascadeLinUCBFull takes $\{\mathbf{x}, \mathbf{z}\}$ as the features. CascadeLSB and CascadeLSBFull only consider diversity of the result list in the objective function, where CascadeLSB takes $\mathbf{x}$ as features, while CascadeLSBFull takes $\{\mathbf{x}, \mathbf{z}\}$ as features. We expect that CascadeLinUCB and CascadeLinUCBFull perform well when $\lambda \to 1$, and that CascadeLSB and CascadeLSBFull perform well when $\lambda \to 0$. For all baselines, we set the exploration parameter $\gamma = 1$ and the learning rate to 1. The parameter setup is used in [35], which leads to better empirical performance. We also set the $\gamma = 1$ for CascadeHybrid.

5.2 Experimental results

We report the cumulative regret, Eq. (3), within 50k steps, called n-step regret. We conduct the experiments with 500 users from the test set and 2 repeats per user. In total, the results are averaged over 1k repeats. We also include the standard errors of our estimates.

We first report the experiments on the MovieLens dataset. We choose $\lambda \in \{0.0, 0.1, \ldots, 1.0\}$, the number of positions $K \in \{5, 10, 15, 20\}$ and the number of topics $d \in \{5, 10, 15, 20\}$ to show the impact of different factors on the performance of the online LTR algorithms. The MovieLens dataset has 20 topics. For $d \in \{5, 10, 15\}$, we choose the topics with the maximum number of items.

To show the impact of λ, we consider the problem of recommending top 10 items with all topics, i.e., $K = 10$ and $d = 20$. The results are displayed at the top row in Fig. 1. CascadeHybrid outperforms baselines when $\lambda \in \{0.1, 0.2, \ldots 0.8\}$. To our surprise, in the two extreme cases, i.e., $\lambda \in \{0, 1\}$, CascadeHybrid performs close to CascadeLSB and CascadeLinUCB, respectively. Particularly, the regret of CascadeHybrid and CascadeLSB with $\lambda = 0$ is: 903.12 ± 21.77 and 750.87 ± 18.75, and the regret of CascadeHybrid and CascadeLinUCB with $\lambda = 1$ is: 260.63 ± 26.44 and 149.30 ± 15.44. Note that CascadeLSB and CascadeLinUCB are specifically designed for those cases and have fewer parameters to be estimated than CascadeHybrid. With other values of λ, CascadeLSB and CascadeLinUCB perform suboptimally. Because they have access to more features, CascadeLSBFull and CascadeLinUCBFull generalize better with respect to different λs, and CascadeLSBFull outperforms CascadeHybrid when $\lambda = 0$. However, in other cases,

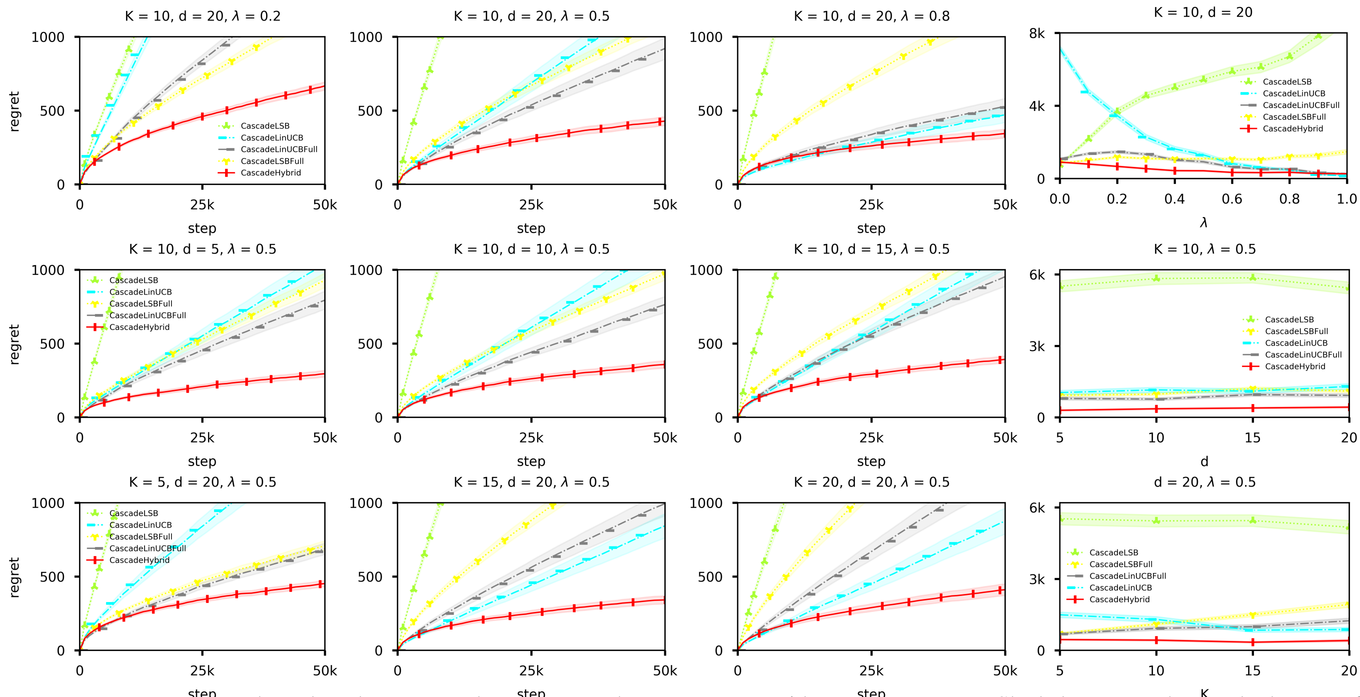

Figure 2: n-step regret on the Yahoo dataset. Results are averaged over 500 users with 2 repeats per user. Shaded areas are the standard errors of estimates.

CascadeLSBFull and CascadeLinUCBFull have higher regret than CascadeHybrid. This result shows that including more features in CascadeLSB and CascadeLinUCB is not sufficient to capture both item relevance and result diversity.

Then, we show the impact of different numbers of topics, where we fix $\lambda = 0.5$ and $K = 10$. From the middle row in Fig. 1, we see that CascadeHybrid outperforms all baselines with large gaps. In the last plot of the middle row, we see that the gap of regret between CascadeHybrid and CascadeLinUCBFull decreases with larger d. This is because, on the MovieLens dataset, when d is small, a user tends to prefer a diverse ranked list: when d is small, an item is more likely to belong to only one topic, and each entry of $\boldsymbol{\theta}_j^*$ becomes relatively larger since $\sum_{j\in[d]} \boldsymbol{\theta}_j^* = 1$. Thus, given an item a and a set $\mathcal{S}$, the difference between $\Delta(a|\mathcal{S})^T\boldsymbol{\theta}^*$ and $\Delta(a|\emptyset)^T\boldsymbol{\theta}^*$ is larger.

Finally, to show the impact of the number of positions, we fix $\lambda = 0.5$ and $d = 20$ and report the results in the last row in Fig. 1. Again, we see that CascadeHybrid outperforms baselines with large gaps. In the last plots, the regret of CascadeHybrid decreases when we increase the number of positions, which is not consistent with Theorem 1. Theorem 1 provides a gap-free bound on the regret, which is designed to bound the case of a large or infinite number of items. For a finite number of items, a gap-dependent bound may be more suitable. We speculate that such a bound might be $\tilde{O}(L-K)$.

We report the results on the Yahoo dataset in Fig. 2. We follow the same setup as used on the MovieLens dataset. CascadeHybrid has slightly higher regret than the best performing baselines in three cases: CascadeLSB when $\lambda = 0$ and CascadeLinUCB when $\lambda \in \{0.9, 1\}$. Note that these are relatively extreme cases, where the particularly designed baselines can benefit most. Meanwhile, CascadeLSB and CascadeLinUCB do not generalize well with different λs. In all setups, CascadeHybrid has lower regret than CascadeLSBFull and CascadeLinUCBFull, which confirms our hypothesis that the hybrid model has benefit in capturing both relevance and diversity.

6 RELATED WORK

Online LTR in a stochastic click models has been widely studied [18–20, 22, 24, 31, 33, 35, 37, 38]. Previous work can be categorized into two groups: feature-free models or feature-based models. Algorithms from the former group use a tabular representation on items and maintain an estimator for each item. They learn inefficiently and are limited to a small number of item candidates. Thus, we do not consider them in our experiments, where L, the number of items, is large.

Feature-based models learn efficiently in terms of L. However, to the best of our knowledge, these approaches either only focus on item relevance or on result diversification. They cannot handle the problem we considered. Among them, ranked bandits [31, 33] are an early approach to online LTR. In ranked bandits, each position is model as a multi-armed bandit (MAB) and diversity of results is addressed in the sense that items ranked at lower positions are less likely to be clicked than those at higher positions, which is different from the topical diversity as we study. Also, ranked bandits do not consider the position bias and are suboptimal in the CM [19].

LSBGreedy [35] and C^2UCB [30] use submodular functions to solve the online diverse LTR problem. They assume that the user browses all displayed items and, thus, does not consider the order of the displayed items, which can be viewed as an easier setup than ours. RecuRank [24] is a recently proposed algorithm that aims at learning the optimal list in term of item relevance in most click models. However, to achieve this task, RecuRank requires a lot of randomly shuffled lists and is outperformed by CascadeLinUCB in the CM [24].

Our work is closely related to CascadeLinUCB [38] and CascadeLSB [13], and can be viewed as a combination of both. CascadeLinUCB and CascadeLSB are designed for online LTR in the CM. The former focuses on relevance ranking, while the latter focuses on result diversification. They have demonstrated supreme performance for their respective tasks and we choose them as baselines in our experiments.

7 CONCLUSION

In real world interactive systems, both relevance of individual items and topical diversity of result lists are critical factors in user satisfaction. In order to better meet users' information needs, we propose a novel online LTR algorithm that optimizes both factors in a hybrid fashion. We formulate the problem as a variant of cascading click bandits, where the attraction probability is a hybrid function that combines a function of relevance features and a submodular function of topic features. CascadeHybrid utilizes a hybrid model as a scoring function and the UCB policy for exploration. We provide a gap-free bound on the η-scaled n-step regret of CascadeHybrid, and conduct experiments on two real-world datasets. Our empirical study shows that CascadeHybrid outperforms two existing online LTR algorithms that exclusively consider either item relevance or topic diversity.

In future work, we intend to conduct experiments on live systems, where feedback is obtained from multiple users so as to test whether CascadeHybrid can learn across users. Another direction is to adapt Thompson sampling [34] to our hybrid model, since Thompson sampling generally outperforms UCB-based algorithms [21, 38].

A PROOF OF THEOREM 1

We write $\mathbf{w}^* = [\boldsymbol{\theta}^{*T}, \boldsymbol{\beta}^{*T}]^T$. Given a ranked list $\mathcal{R}$ and $a = \mathcal{R}(i)$, we write $\boldsymbol{\phi}_a = [\boldsymbol{\omega}_a^T, \mathbf{z}_a^T]^T$. We write $\mathbf{O}_t = \mathbf{I}_{m+d} + \sum_{i=1}^{t-1}\sum_{a\in\mathcal{O}_i}\boldsymbol{\phi}_a\boldsymbol{\phi}_a^T$ as the collected features in t steps, and $\mathcal{H}_t$ as the collected features and clicks up to step t. We write $\mathcal{R}^i = (\mathcal{R}(1), \ldots, \mathcal{R}(i))$. Then, the confidence bound in Eq. (14) on the i-th item in $\mathcal{R}$ can be re-written as:

$$s(\mathcal{R}^i) = \boldsymbol{\phi}_{\mathcal{R}(i)}^T \mathbf{O}_t^{-1} \boldsymbol{\phi}_{\mathcal{R}(i)}. \tag{21}$$

Let $\Pi(\mathcal{D}) = \cup_{i=1}^{L}\Pi_i(\mathcal{D})$ be the set of all ranked lists of $\mathcal{D}$ with length $[L]$, and $\kappa : \Pi(\mathcal{D}) \to [0,1]$ be an arbitrary list function. For any $\mathcal{R} \in \Pi(\mathcal{D})$ and any κ, we define

$$f(\mathcal{R}, \kappa) = 1 - \prod_{i=1}^{|\mathcal{R}|}(1 - \kappa(\mathcal{R}^i)). \tag{22}$$

We define upper and lower confidence bound, and κ as:

$$\begin{aligned} u_t(\mathcal{R}) &= \mathrm{F}_{[0,1]}[\boldsymbol{\phi}_{\mathcal{R}(l)}^T \hat{\mathbf{w}}_t + s(\mathcal{R}^l)] \\ l_t(\mathcal{R}) &= \mathrm{F}_{[0,1]}[\boldsymbol{\phi}_{\mathcal{R}(l)}^T \hat{\mathbf{w}}_t - s(\mathcal{R}^l)] \\ \kappa(R) &= \boldsymbol{\phi}_{\mathcal{R}(l)}^T \mathbf{w}^*, \end{aligned} \tag{23}$$

where $l = |\mathcal{R}|$ and $\mathrm{F}_{[0,1]}[\cdot] = \max(0, \min(1, \cdot))$. With the definitions in Eq. (23), $f(\mathcal{R}, \kappa) = r(\mathcal{R}, \boldsymbol{\alpha})$ is the reward of list $\mathcal{R}$.

Proof. Let $g_t = \{l_t(\mathcal{R}) \le \kappa(\mathcal{R}) \le u_t(\mathcal{R}), \forall \mathcal{R} \in \Pi(\mathcal{D})\}$ be the event that the attraction probabilities are bounded by the lower and upper confidence bound, and $\bar{g}_t$ be the complement of g_t. We have

$$\begin{aligned} \mathbb{E}\left[\eta r(\mathcal{R}^*, \boldsymbol{\alpha}) - r(\mathcal{R}_t, \mathbf{A}_t)\right] &= \mathbb{E}\left[\eta f(\mathcal{R}^*, \kappa) - f(\mathcal{R}_t, \kappa)\right] \\ &\overset{(a)}{\le} P(g_t)\mathbb{E}\left[\eta f(\mathcal{R}^*, \kappa) - f(\mathcal{R}_t, \kappa)\right] + P(\bar{g}_t) \\ &\overset{(b)}{\le} P(g_t)\mathbb{E}\left[\eta f(\mathcal{R}^*, u_t) - f(\mathcal{R}_t, \kappa)\right] + P(\bar{g}_t) \\ &\overset{(c)}{\le} P(g_t)\mathbb{E}\left[f(\mathcal{R}_t, u_t) - f(\mathcal{R}_t, \kappa)\right] + P(\bar{g}_t), \end{aligned} \tag{24}$$

where inequality (a) is because $\mathbb{E}\left[\eta f(\mathcal{R}^*, \kappa) - f(\mathcal{R}_t, \kappa)\right] \le 1$, inequality (b) is because under event g_t we have $f(\mathcal{R}, l_t) \le f(\mathcal{R}, \kappa) \le f(\mathcal{R}, u_t), \forall \mathcal{R} \in \Pi(\mathcal{D})$, and inequality (c) is by the definition of the η-approximation, where we have

$$\eta f(\mathcal{R}^*, u_t) \le \max_{\mathcal{R}\in\Pi_K(\mathcal{D})} \eta f(\mathcal{R}, u_t) \le f(\mathcal{R}_t, u_t). \tag{25}$$

By the definition of the list function $f(\cdot,\cdot)$ in Eq. (22), we have

$$\begin{aligned} f(\mathcal{R}_t, u_t) - f(\mathcal{R}_t, \kappa) &= \prod_{k=1}^{K}(1 - \kappa(\mathcal{R}_t^k)) - \prod_{k=1}^{K}(1 - u_t(\mathcal{R}_t^k)) \\ &\overset{(a)}{=} \sum_{k=1}^{K}\left[\prod_{i=1}^{k-1}(1 - \kappa(\mathcal{R}_t^i))\right](u_t(\mathcal{R}_t^k) - \kappa(\mathcal{R}_t^k))\left[\prod_{j=k+1}^{K}(1 - u(\mathcal{R}_t^j))\right] \\ &\overset{(b)}{\le} \sum_{k=1}^{K}\left[\prod_{i=1}^{k-1}(1 - \kappa(\mathcal{R}_t^i))\right](u_t(\mathcal{R}_t^k) - \kappa(\mathcal{R}_t^k)), \end{aligned} \tag{26}$$

where equality (a) follows from Lemma 1 in [38] and inequality (b) is because of the fact that $0 \le \kappa(\mathcal{R}_t) \le u_t(\mathcal{R}_t) \le 1$. We then define the event $h_{ti} = \{\text{item } \mathcal{R}_t(i) \text{ is observed}\}$, where we have $\mathbb{E}\left[\mathbb{1}(h_{ti})\right] = \prod_{k=1}^{i-1}(1 - \kappa(\mathcal{R}_t^k))$. For any $\mathcal{H}_t$ such that g_t holds, we have

$$\begin{aligned} &\mathbb{E}\left[f(\mathcal{R}_t, u_t) - f(\mathcal{R}_t, \kappa) \mid \mathcal{H}_t\right] \\ &\le \sum_{i=1}^{K}\mathbb{E}\left[\mathbb{1}(h_{ti}) \mid \mathcal{H}_t\right](u_t(\mathcal{R}_t^i) - l_t(\mathcal{R}_t^i)) \\ &\overset{(b)}{\le} 2\gamma\mathbb{E}\left[\sum_{i=1}^{\min(K,c_t)}\sqrt{s(\mathcal{R}_t^i)} \mid \mathcal{H}_t\right], \end{aligned} \tag{27}$$

where inequality (b) follows from the definition of u_t and l_t in Eq. (23), and the definition of h_{ti}. Now, together with Eqs. (15), (24) and (27), we have

$$\begin{aligned} R_\eta(n) &= \sum_{t=1}^{n}\mathbb{E}\left[\eta r(\mathcal{R}^*, \boldsymbol{\alpha}) - r(\mathcal{R}_t, \mathbf{A}_t)\right] \\ &\le \sum_{t=1}^{n}\left[2\gamma\mathbb{E}\left[\sum_{i=1}^{\min(K,c_t)}\sqrt{s(\mathcal{R}_t^i)} \mid g_t\right]P(g_t) + P(\bar{g}_t)\right] \\ &\le 2\gamma\mathbb{E}\left[\sum_{t=1}^{n}\sum_{i=1}^{K}\sqrt{s(\mathcal{R}_t^i)}\right] + \sum_{t=1}^{n}p(\bar{g}_t). \end{aligned} \tag{28}$$

For the first term in Eq. (28), we have

$$\begin{aligned} &\sum_{t=1}^{n}\sum_{i=1}^{K}\sqrt{s(\mathcal{R}_t^i)} \\ &\overset{(a)}{\le} \sqrt{nK\sum_{t=1}^{n}\sum_{i=1}^{K}s(\mathcal{R}_t^i)} \overset{(b)}{\le} \sqrt{nK 2\log det(\mathbf{O}_t)}, \end{aligned} \tag{29}$$

where inequality (a) follows from the Cauchy-Schwarz inequality and (b) follows from Lemma 5 in [35]. Note that $\log det(\mathbf{O}_t) \le (m+d)\log(K(1 + n/(m+d)))$ and together with Eq. (29):

$$\sum_{t=1}^{n}\sum_{i=1}^{K}\sqrt{s(\mathcal{R}_t^i)} \le \sqrt{2nK(m+d)\log(K(1+\frac{n}{m+d}))}. \tag{30}$$

For the second term in Eq. (28), by Lemma 3 in [13], we have $P(\bar{g}_t) \le 1/n$ for any γ satisfies Eq. (16). Thus, together with Eqs. (28) to (30), we have

$$R_\eta(n) \le 2\gamma\sqrt{2nK(m+d)\log(1+\frac{nK}{m+d}) + 2\log(n)} + 1.$$

That concludes the proof of Theorem 1. □

REFERENCES

[1] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. In *NIPS*, pages 2312–2320, 2011.

[2] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *WSDM*, pages 5–14, 2009.

[3] A. Ashkan, B. Kveton, S. Berkovsky, and Z. Wen. Optimal greedy diversity for recommendation. In *IJCAI*, pages 1742–1748, 2015.

[4] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

[5] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, H. N., and G. Hullender. Learning to rank using gradient descent. In *ICML*, pages 89–96, 2005.

[6] O. Chapelle and Y. Zhang. A dynamic Bayesian network click model for web search ranking. In *WWW*, pages 1–10, 2009.

[7] A. Chuklin, I. Markov, and M. de Rijke. *Click Models for Web Search*. Morgan & Claypool, 2015.

[8] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *WSDM*, pages 87–94, 2008.

[9] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins, Baltimore, MD, 3 edition, 1996.

[10] F. Guo, C. Liu, A. Kannan, T. Minka, M. Taylor, Y. M. Wang, and C. Faloutsos. Click chain model in web search. In *WWW*, pages 11–20, 2009.

[11] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, 2015.

[12] J. He, E. Meij, and M. de Rijke. Result diversification based on query-specific cluster ranking. *J. Am. Soc. Inform. Sci. and Tech.*, 62(3):550–571, 2011.

[13] G. Hiranandani, H. Singh, P. Gupta, I. A. Burhanuddin, Z. Wen, and B. Kveton. Cascading linear submodular bandits: Accounting for position bias and diversity in online learning to rank. In *UAI*, 2019.

[14] K. Hofmann, S. Whiteson, and M. de Rijke. Balancing exploration and exploitation in learning to rank online. In *ECIR*, pages 251–263, 2011.

[15] K. Hofmann, A. Schuth, S. Whiteson, and M. de Rijke. Reusing historical interaction data for faster online learning to rank for IR. In *WSDM*, pages 183–192, 2013.

[16] R. Jagerman, H. Oosterhuis, and M. de Rijke. To model or to intervene: A comparison of counterfactual and online learning to rank from user interactions. In *SIGIR*, pages 15–24, 2019.

[17] T. Joachims. Optimizing search engines using clickthrough data. In *SIGKDD*, pages 133–142, 2002.

[18] S. Katariya, B. Kveton, C. Szepesvari, and Z. Wen. DCM bandits: Learning to rank with multiple clicks. In *ICML*, pages 1215–1224, 2016.

[19] B. Kveton, C. Szepesvari, Z. Wen, and A. Ashkan. Cascading bandits: Learning to rank in the cascade model. In *ICML*, pages 767–776, 2015.

[20] C. Li and M. de Rijke. Cascading non-stationary bandits: Online learning to rank in the non-stationary cascade model. In *IJCAI*, pages 2859–2865, 2019.

[21] C. Li, I. Markov, M. de Rijke, and M. Zoghi. Merge double Thompson sampling for large scale online ranker evaluation. *arXiv preprint arXiv:1812.04412*, 2018.

[22] C. Li, B. Kveton, T. Lattimore, I. Markov, M. de Rijke, C. Szepesvári, and M. Zoghi. BubbleRank: Safe online learning to re-rank via implicit click feedback. In *UAI*, 2019.

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

[24] S. Li, T. Lattimore, and C. Szepesvári. Online learning to rank with features. In *ICML*, pages 3856–3865, 2019.

[25] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[26] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978.

[27] H. Oosterhuis and M. de Rijke. Differentiable unbiased online learning to rank. In *CIKM*, pages 1293–1302, 2018.

[28] H. Oosterhuis and M. de Rijke. Ranking for relevance and display preferences in complex presentation layouts. In *SIGIR*, pages 845–854, 2018.

[29] L. Qin and X. Zhu. Promoting diversity in recommendation by entropy regularizer. In *IJCAI*, pages 2698–2704, 2013.

[30] L. Qin, S. Chen, and X. Zhu. Contextual combinatorial bandit and its application on diversified online recommendation. In *SDM*, pages 461–469, 2014.

[31] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *ICML*, pages 784–791, 2008.

[32] T. Saracevic. Relevance: A review of and a fframework for the thinking on the notion in information science. *J. Am. Soc. Inform. Sci.*, 26(6):321–343, 1975.

[33] A. Slivkins, F. Radlinski, and S. Gollapudi. Learning optimally diverse rankings over large document collections. In *ICML*, pages 983–990, 2010.

[34] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25:285–294, 1933.

[35] Y. Yue and C. Guestrin. Linear submodular bandits and their application to diversified retrieval. In *NIPS*, pages 2483–2491, 2011.

[36] Y. Yue and T. Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML*, pages 1201–1208, 2009.

[37] M. Zoghi, T. Tunys, M. Ghavamzadeh, B. Kveton, C. Szepesvari, and Z. Wen. Online learning to rank in stochastic click models. In *ICML*, pages 4199–4208, 2017.

[38] S. Zong, H. Ni, K. Sung, N. R. Ke, Z. Wen, and B. Kveton. Cascading bandits for large-scale recommendation problems. In *UAI*, 2016.