Online Learning to Rank with Features

Shuai Li [1] **Tor Lattimore** [2] **Csaba Szepesvári** [2]

Abstract

We introduce a new model for online ranking in which the click probability factors into an examination and attractiveness function and the attractiveness function is a linear function of a feature vector and an unknown parameter. Only relatively mild assumptions are made on the examination function. A novel algorithm for this setup is analysed, showing that the dependence on the number of items is replaced by a dependence on the dimension, allowing the new algorithm to handle a large number of items. When reduced to the orthogonal case, the regret of the algorithm improves on the state-of-the-art.

1. Introduction

Let $\mathcal{L}$ be a large set of items to be ranked. For example, a database of movies, news articles or search results. We consider a sequential version of the ranking problem where in each round the learner chooses an ordered list of K distinct items from $\mathcal{L}$ to show the user. We assume the feedback comes in the form of clicks and the learner's objective is to maximise the expected number of clicks over T rounds. Our focus is on the case where $\mathcal{L}$ is large (perhaps millions) and K is relatively small (fifty or so). There are two main challenges that arise in online ranking problems:

(a) The number of rankings grows exponentially in K, which makes learning one parameter for each ranking a fruitless endeavour. Click models may be used to reduce the dimensionality of the learning problem, but balancing generality of the model with learnability is a serious challenge. The majority of previous works on online learning to rank have used unstructured models, which are not well suited to our setting where $\mathcal{L}$ is large.

(b) Most click models depend on an unknown attractiveness function that endows the item set with an order. This yields a model with at least $|\mathcal{L}|$ parameters, which is prohibitively large in the applications we have in mind.

The first challenge is tackled by adapting the flexible click models introduced in (Zoghi et al., 2017; Lattimore et al., 2018) to our setting. For the second we follow previous works on bandits with large action sets by assuming the attractiveness function can be written as a linear function of a relatively small number of features.

Contribution We make several contributions:

- A new model for ranking problems with features is proposed that generalises previous work (Li et al., 2016; Zong et al., 2016; Liu et al., 2018) by relaxing the relatively restrictive assumptions on the probability that a user clicks on an item. The new model is strictly more robust than previous works focusing on regret analysis for large item sets.
- We introduce a novel polynomial-time algorithm called `RecurRank`. The algorithm operates recursively over an increasingly fine set of partitions of $[K]$. Within each partition the algorithm balances exploration and exploitation, subdividing the partition once it becomes sufficiently certain about the suboptimality of a subset of items.
- A regret analysis shows that the cumulative regret of `RecurRank` is at most $R_T = O(K\sqrt{dT\log(LT)})$, where K is the number of positions, L is the number of items and d is the dimension of the feature space. Even in the non-feature case where $L = d$ this improves on the state-of-the-art by a factor of $\sqrt{K}$.

A comparison with most related work is shown in Table 1.

Related work Online learning to rank has seen an explosion of research in the last decade and there are multiple ways of measuring the performance of an algorithm. One view is that the clicks themselves should be maximised, which we take in this article. An alternative is to assume an underlying relevance of all items in a ranking that is never directly observed, but can be inferred in some way from the observed clicks. In all generality this latter setting falls into the partial monitoring framework (Rustichini, 1999),

[1]The Chinese University of Hong Kong [2]DeepMind. Correspondence to: Shuai Li <shuaili@cse.cuhk.edu.hk>, Tor Lattimore <lattimore@google.com>, Csaba Szepesvári <szepi@google.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

Table 1. This table compares settings and regret bounds of most related works on online learning to rank. T is the number of total rounds, K is the number of positions, L is the number of items and d is the feature space dimension. Δ is the minimal gap between the expected click rate of the best items and the expected click rate of the suboptimal items.

	Context	Click Model	Regret
Kveton et al. (2015)	-	Cascade Model (CM)	$\Theta\left(\frac{L}{\Delta}\log(T)\right)$
Li et al. (2016) Zong et al. (2016) Li & Zhang (2018)	(Generalised) Linear Form	CM	$O\left(d\sqrt{TK}\log(T)\right)$
Katariya et al. (2016)	-	Dependent Click Model (DCM)	$\Theta\left(\frac{L}{\Delta}\log(T)\right)$
Liu et al. (2018)	Generalised Linear Form	DCM	$O\left(dK\sqrt{TK}\log(T)\right)$
Lagree et al. (2016)	-	Position-Based Model (PBM) with known position bias	$O\left(\frac{L}{\Delta}\log(T)\right)$
Zoghi et al. (2017)	-	General Click Model	$O\left(\frac{K^3L}{\Delta}\log(T)\right)$
Lattimore et al. (2018)	-	General Click Model	$O\left(\frac{KL}{\Delta}\log(T)\right)$ $O\left(\sqrt{K^3LT\log(T)}\right)$ $\Omega\left(\sqrt{KLT}\right)$
Ours	Linear Form	General Click Model	$O\left(K\sqrt{dT\log(LT)}\right)$

but has been studied in specific ranking settings (Chaudhuri, 2016, and references therein). See the article by Hofmann et al. (2011) for more discussion on various objectives.

Maximising clicks directly is a more straightforward objective because clicks are an observed quantity. Early work was empirically focused. For example, Li et al. (2010) propose a modification of LinUCB for contextual ranking and Chen & Hofmann (2015) modify the optimistic algorithms for linear bandits. These algorithms do not come with theoretical guarantees, however. There has recently been significant effort towards designing theoretically justified algorithms in settings of increasing complexity (Kveton et al., 2015; Combes et al., 2015; Zong et al., 2016; Katariya et al., 2016; Lagree et al., 2016). These works assume the user's clicks follow a click model that connects properties of the shown ranking to the probability that a user clicks on an item placed in a given position. For example, in the document-based model it is assumed that the probability that the user clicks on a shown item only depends on the unknown attractiveness of that item and not its position in the ranking or the other items. Other simple models include the position-based, cascade and dependent click models. For a survey of click models see (Chuklin et al., 2015).

As usual, however, algorithms designed for specific models are brittle when the modelling assumptions are not met. Recent work has started to relax the strong assumptions by making the observation that in all of the above click models the probability of a user clicking on an item can be written as the product of the item's inherent attractiveness and the probability that the user examines its position in the list. Zoghi et al. (2017) use a click model where this decomposition is kept, but the assumption on how the examination probability of a position depends on the list is significantly relaxed. This is relaxed still further by Lattimore et al. (2018) who avoid the factorisation assumption by making assumptions directly on the click probabilities, but the existence of an attractiveness function remains.

The models mentioned in the last paragraph do not make assumptions on the attractiveness function, which means the regret depends badly on the size of $\mathcal{L}$. Certain simple click models have assumed the attractiveness function is a linear function of an item's features and the resulting algorithms are suitable for large action sets. This has been done for the cascade model (Li et al., 2016) and the dependent-click model (Liu et al., 2018). While these works are welcomed, the strong assumptions leave a lingering doubt that perhaps the models may not be a good fit for practical problems. Of course, our work is closely related to stochastic linear bandits, first studied by Abe & Long (1999) and refined by Auer (2002); Abbasi-Yadkori et al. (2011); Valko et al. (2014) and many others.

Ranking has also been examined in an adversarial frame-

work by Radlinski et al. (2008). These settings are most similar to the stochastic position-based and document-based models, but with the additional robustness bought by the adversarial framework. Another related setup is the rank-1 bandit problem in which the learner should choose just one of L items to place in one of K positions. For example, the location of a billboard with the budget to place only one. These setups have a lot in common with the present one, but cannot be directly applied to ranking problems. For more details see (Katariya et al., 2017a;b).

Finally, we note that some authors do not assume an ordering of the item set provided by an attractiveness function. The reader is referred to the work by Slivkins et al. (2013) (which is a follow-up work to Radlinski et al. (2008)) where the learner's objective is to maximise the probability that a user clicks on *any* item, rather than rewarding multiple clicks. This model encourages diversity and provides an interesting alternative approach.

2. Preliminaries

Notation Let $[n] = \{1, 2, \ldots, n\}$ denote the first n natural numbers. Given a set X the indicator function is $\mathbb{1}_X$. For vector $x \in \mathbb{R}^d$ and positive definite matrix $V \in \mathbb{R}^{d \times d}$ we let $\|x\|_V^2 = x^\top V x$. The Moore-Penrose pseudoinverse of a matrix V is $V^\dagger$.

Problem setup Let $\mathcal{L} \subset \mathbb{R}^d$ be a finite set of items, $L = |\mathcal{L}|$ and $K > 0$ a natural number, denoting the number of positions. A ranking is an injective function from $[K]$, the set of positions, to $\mathcal{L}$ and the set of all rankings is denoted by Σ. We use uppercase letters like A to denote rankings in Σ and lowercase letters a, b to denote items in $\mathcal{L}$. The game proceeds over T rounds. In each round $t \in [T]$ the learner chooses a ranking $A_t \in \Sigma$ and subsequently receives feedback in the form of a vector $C_t \in \{0,1\}^K$ where $C_{tk} = 1$ if the user clicked on the kth position. We assume that the conditional distribution of C_t only depends on A_t, which means there exists an unknown function $v : \Sigma \times [K] \to [0,1]$ such that for all $A \in \Sigma$ and $k \in [K]$,

$$\mathbb{P}\left(C_{tk} = 1 \mid A_t = A\right) = v(A, k)\,. \tag{1}$$

Remark 1. *We do not assume conditional independence of* $(C_{tk})_{k=1}^K$.

In all generality the function v has $K|\Sigma|$ parameters, which is usually impractically large to learn in any reasonable timeframe. A click model corresponds to making assumptions on v that reduces the statistical complexity of the learning problem. We assume a factored model:

$$v(A, k) = \chi(A, k)\alpha(A(k))\,, \tag{2}$$

where $\chi : \Sigma \times [K] \to [0,1]$ is called the examination probability and $\alpha : \mathcal{L} \to [0,1]$ is the attractiveness function. We assume that attractiveness is linear in the action, which means there exists an unknown $\theta_* \in \mathbb{R}^d$ such that

$$\alpha(a) = \langle a, \theta_* \rangle \quad \text{for all } a \in \mathcal{L}\,. \tag{3}$$

Let a_k^* be the k-th best item sorted in order of decreasing attractiveness. Then let $A^* = (a_1^*, \ldots, a_K^*)$. In case of ties the choice of A^* may not be unique. All of the results that follow hold for any choice.

The examination function satisfies three additional assumptions. The first says the examination probability of position k only depends on the identity of the first $k-1$ items and not their order:

Assumption 1. $\chi(A, k) = \chi(A', k)$ *for any* $A, A' \in \Sigma$ *with* $A([k-1]) = A'([k-1])$.

The second assumption is that the examination probability on any ranking is monotone decreasing in k:

Assumption 2. $\chi(A, k+1) \le \chi(A, k)$ *for all* $A \in \Sigma$ *and* $k \in [K-1]$.

The third assumption is that the examination probability on ranking A^* is minimal:

Assumption 3. $\chi(A, k) \ge \chi(A^*, k) =: \chi_k^*$ *for all* $A \in \Sigma$ *and* $k \in [K]$.

All of these assumptions are satisfied by many standard click models, including the document-based, position-based and cascade models. These assumptions are strictly weaker than those made by Zoghi et al. (2017) and orthogonal to those by Lattimore et al. (2018) as we discuss it in Section 6.

The learning objective We measure the performance of our algorithm in terms of the cumulative regret, which is

$$R_T = T \sum_{k=1}^K v(A^*, k) - \mathbb{E}\left[\sum_{t=1}^T \sum_{k=1}^K v(A_t, k)\right]\,.$$

Remark 2. *The regret is defined relative to* A^**, but our assumptions do not imply that*

$$A^* \in \arg\max_{A \in \Sigma} \sum_{k=1}^K v(A, k)\,. \tag{4}$$

The assumptions in all prior work in Table 1 either directly or indirectly ensure that (4) holds. Our regret analysis does not rely on this, so we do not assume it. Note, however, that the definition of regret is most meaningful when (Eq. (4)) approximately holds.

Experimental design Our algorithm makes use of an exploration 'spanner' that approximately minimises the covariance of the least-squares estimator. Given an arbitrary

finite set of vectors $X = \{x_1, \dots, x_n\} \subset \mathbb{R}^d$ and distribution $\pi : X \to [0,1]$ let $Q(\pi) = \sum_{x \in X} \pi(x) x x^\top$. By the Kiefer–Wolfowitz theorem (Kiefer & Wolfowitz, 1960) there exists a π called the G-optimal design such that

$$\max_{x \in X} \|x\|^2_{Q(\pi)^\dagger} \leq d\,. \tag{5}$$

As explained in Chap. 21 of (Lattimore & Szepesvári, 2018), π may be chosen so that $|\{x : \pi(x) > 0\}| \leq d(d+1)/2$. A G-optimal design π for X has the property that if each element $x \in X$ is observed $n\pi(x)$ times for some n large enough, the value estimate obtained via least-squares will have its maximum uncertainty over the items minimised. Given a finite (multi-)set of vectors $X \subset \mathbb{R}^d$ we let $\textsc{Gopt}(X)$ denote a G-optimal design distribution. Methods from experimental design have been used for pure exploration in linear bandits (Soare et al., 2014; Xu et al., 2017) and also finite-armed linear bandits (Lattimore & Szepesvári, 2018, Chap. 22) as well as adversarial linear bandits (Bubeck et al., 2012).

3. Algorithm

The new algorithm is called `RecurRank` ('recursive ranker'). The algorithm maintains a partition of the K positions into intervals. Associated with each interval is an integer-valued 'phase number' and an ordered set of items, which has the same size as the interval for all but the last interval (containing position K). Initially the partition only contains one interval that is associated with all the items and phase number $\ell = 1$.

At any point in time, `RecurRank` works in parallel on all intervals. Within an interval associated with phase number ℓ, the algorithm balances exploration and exploitation while determining the relative attractiveness of the items to accuracy $\Delta_\ell = 2^{-\ell}$. To do this, items are placed in the first position of the interval in proportion to an experimental design. The remaining items are placed in order in the remaining positions. Once sufficient data is collected, the interval is divided into a collection of subintervals and the algorithm is restarted on each subinterval with the phase number increased.

The natural implementation of the algorithm maintains a list of partitions and associated items. In each round it iterates over the partitions and makes assignments of the items within each partition. The assignments are based on round-robin idea using an experimental design, which means the algorithm needs to keep track of how often each item has been placed in the first position. This is not a problem from an implementation perspective, but stateful code is hard to interpret in pseudocode. We provide a recursive implementation that describes the assignments made within each interval and the rules for creating a new partition. A flow chart depicting the operation of the algorithm is given in Fig. 1. The code is provided in the supplementary material.

Algorithm 1 `RecurRank`

1: **Input:** Phase number ℓ and $\mathcal{A} = (a_1, a_2, \dots)$ and $\mathcal{K} = (k, k+1, \dots, k+m-1)$

2: Find a G-optimal design $\pi = \textsc{Gopt}(\mathcal{A})$

3: Let $\Delta_\ell = 2^{-\ell}$ and

$$T(a) = \left\lceil \frac{d\,\pi(a)}{2\Delta_\ell^2} \log\left(\frac{|\mathcal{A}|}{\delta_\ell}\right) \right\rceil \tag{6}$$

This instance will run $\sum_{a \in \mathcal{A}} T(a)$ times

4: Select each item $a \in \mathcal{A}$ exactly $T(a)$ times at position k and put available items in $\{a_1, \dots, a_m\}$ sequentially in positions $\{k+1, \dots, k+m-1\}$ and receive feedbacks (synchronized by a global clock).

5: Let $\mathcal{D} = \{(\beta_1, \zeta_1), \dots\}$ be the multiset of item/clicks from position k and compute

$$\hat{\theta} = V^\dagger S \quad \text{with} \quad V = \sum_{(\beta,\zeta) \in \mathcal{D}} \beta\beta^\top \text{ and } S = \sum_{(\beta,\zeta) \in \mathcal{D}} \beta\zeta \tag{7}$$

6: Let $a^{(1)}, a^{(2)}, \dots, a^{(|\mathcal{A}|)}$ be an ordering $\mathcal{A}$ such that

$$\varepsilon_i = \langle \hat{\theta}, a^{(i)} - a^{(i+1)} \rangle \geq 0 \text{ for all } 1 \leq i < |\mathcal{A}|$$

and set $\varepsilon_{|\mathcal{A}|} = 2\Delta_\ell$

7: Let $(u_1, \dots, u_p) = (i \in [|\mathcal{A}|] : \varepsilon_i \geq 2\Delta_\ell)$ and $u_0 = 0$

$$\mathcal{A}_i = (a^{(u_{i-1}+1)}, \dots, a^{(u_i)})$$
$$\mathcal{K}_i = (k + u_{i-1}, \dots, k + \min(m, u_i) - 1)$$

8: For each $i \in [p]$ such that $k + u_{i-1} \leq k + m - 1$ call `RecurRank` $(\ell+1, \mathcal{A}_i, \mathcal{K}_i)$ on separate threads

The pseudocode of the core subroutine on each interval is given in Algorithm 1. The subroutine accepts as input (1) the phase number ℓ, (2) the positions of the interval $\mathcal{K} \subseteq [K]$ and (3) an ordered list of items, $\mathcal{A}$. The phase number determines the length of the experiment and the target precision. The ordering of the items in $\mathcal{A}$ is arbitrary in the initial partition (when $\ell = 1$). When $\ell > 1$ the ordering is determined by the empirical estimate of attractiveness in the previous experiment, which is crucial for the analysis. The whole algorithm is started by calling `RecurRank`$(1, \mathcal{L}, (1, 2, \dots, K))$ where the order of $\mathcal{L}$ is random. The algorithm is always instantiated with parameters that satisfy $|\mathcal{A}| \geq |\mathcal{K}| = m$. Furthermore, $|\mathcal{A}| > |\mathcal{K}|$ is only possible when $K \in \mathcal{K}$.

The subroutine learns about the common unknown parameter vector by placing items in the first position of the interval in proportion to a G-optimal design for the available items. The remaining items in $\mathcal{A}$ are placed in order into

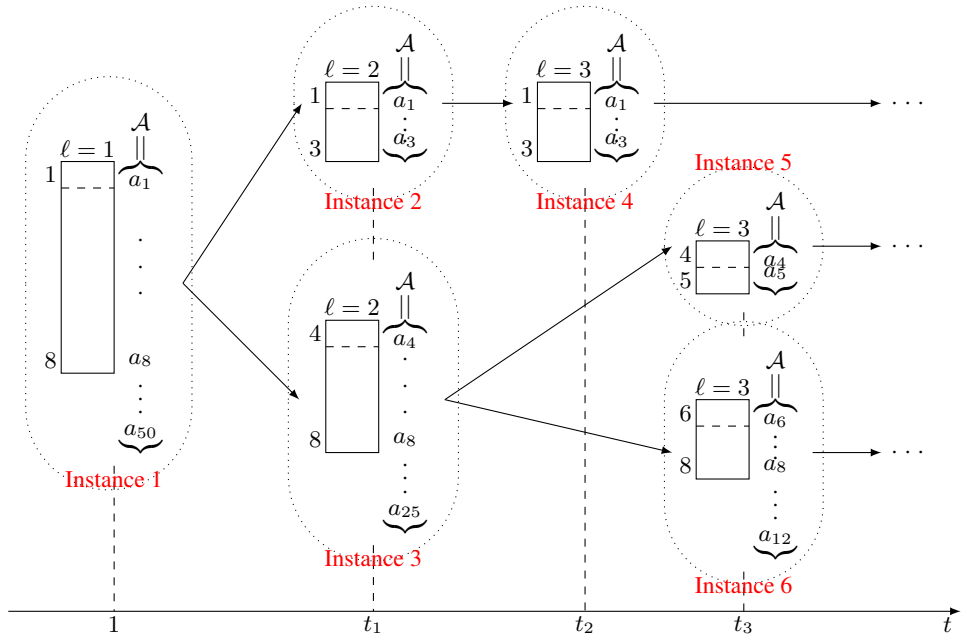

Figure 1. A flow chart demonstration for the algorithm. Each dotted circle represents a subinterval and runs an instance of Algorithm 1. The dashed line denotes the first position for each interval.

the remaining positions (Line 4). This means that each item $a \in \mathcal{A}$ is placed exactly $T(a)$ times in the first position k of the interval. The choice of $T(a)$ is based on the phase number ℓ and the G-optimal design π over $\mathcal{A}$ (Line 2). Note $T(a) = 0$ if $\pi(a) = 0$. For example, if $\mathcal{A} = (a_1, \ldots, a_m)$ and a_3 is placed at the first position, then the rest positions are filled in as $a_1, a_2, a_4, a_5, \ldots, a_m$. The subroutine runs for $\sum_{a \in \mathcal{A}} T(a)$ rounds. The G-optimal design means that the number of rounds required to estimate the value of each item to a fixed precision depends only logarithmically on the number of items. Higher phase number means longer experiment and also higher target precision.

Once all arms $a \in \mathcal{A}$ have been placed in the first position of the interval $T(a)$ times, `RecurRank` estimates the attractiveness of the items in $\mathcal{A}$ using a least-squares estimator based on the data collected from the first position (Line 5). The items are then ordered based on their estimated attractiveness. The subroutine then partitions the ordered items when the difference between estimated attractiveness of consecutive items is sufficiently large (Line 7). Finally the subroutine recursively calls `RecurRank` on each partition for which there are positions available with an increased phase number with items sorted according to their empirical attractiveness (Line 8).

Remark 3. *Items are eliminated entirely if at the end of a subroutine a partition is formed for which there are no available positions. For example, consider the first instantiation of* `RecurRank` *with $\ell = 1$ and $\mathcal{K} = [K]$ and $\mathcal{A} = \mathcal{L}$. Suppose the observed data is such that $p = 2$ and $u_1 \geq K$, then items $a^{(u_1+1)}, a^{(u_1+2)}, \ldots, a^{(u_2)}$ will be discarded because the starting position of the second partition would be larger than K.*

Remark 4. *The least-squares estimator $\hat{\theta}$ defined in Eq. (7) actually does not have expectation θ, which means the algorithm is not really estimating attractiveness. Our assumptions ensure that the expectation of $\hat{\theta}$ is proportional to θ, however, which is sufficient for our analysis. This is the reason for only using the first position within an interval for estimation.*

Remark 5. *The subroutine only uses data collected during its own run. Not doing this would introduce bias that may be hard to control.*

In Fig. 1, the algorithm starts with Instance 1 of phase number $\ell = 1$, all items and all positions. At time t_1, Instance 1 splits into two, each with an increased phase number $\ell = 2$. Instance 2 contains 3 items and 3 positions and Instance 3 contains 5 positions but 22 items. The remaining items have been eliminated. At time t_2, Instance 2 finishes running but has no split, so it calls Instance 4 with the same items, same positions but increased phase number $\ell = 3$. During time t_1 to t_2, Instance 2 and Instance 3 run in parallel and recommend lists together; during time t_2 to t_3, Instances 3 and 4 run in parallel and recommend lists together. At time t_3, Instance 3 finishes and splits into another two threads, both with increased phase number $\ell = 3$. Instance 5 contains exactly 2 items and 2 positions and Instance 6 contains 3 positions but 7 items. Note that the involved items become even less. Right after time t_3, Instance $4, 5, 6$ run in parallel and recommend lists together.

RecurRank has two aspects that one may think can lead to an unjustifiable increase of regret: *(i)* each subroutine only uses data from the first position to estimate attractiveness, and *(ii)* data collected by one subroutine is not re-used subsequently. The second of these is relatively minor. Like many elimination algorithms, the halving of the precision means that at most a constant factor is lost by discarding the data. The first issue is more delicate. On the one hand, it seems distasteful not to use all available data. But the assumptions do not make it easy to use data collected in later positions. And actually the harm may not be so great. Intuitively the cost of only using data from the first position is greatest when the interval is large and the attractiveness varies greatly within the interval. In this case, however, a split will happen relatively fast.

Running time The most expensive component is computing the G-optimal design. This is a convex optimisation problem and has been studied extensively (see, Boyd & Vandenberghe 2004, §7.5 and Todd 2016). It is not necessary to solve the optimisation problem exactly. Suppose instead we find a distribution π on $\mathcal{A}$ with support at most $D(D+1)/2$ and for which $\max_{a\in\mathcal{A}} \|a\|^2_{Q(\pi)^\dagger} \leq D$. Then our bounds continue to hold with d replaced by D. Such approximations are generally easy to find. For example, π may be chosen to be a uniform distribution on a volumetric spanner of $\mathcal{A}$ of size D. See the supplementary material for a summary of volumetric spanners. Hazan & Karnin (2016) provide a randomized algorithm that returns a volumetric spanner of size at most $O(d\log(d)\log(|\mathcal{A}|))$ with an expected running time of $O(|\mathcal{A}|\, d^2)$. For the remaining parts of the algorithm, the least-squares estimation is at most $O(d^3)$. The elimination and partitioning run in $O(|\mathcal{A}|\, d)$. Note these computations happen only once for each instantiation. The update for each partition in each round is $O(d^2)$. The total running time is $O(Ld^2\log(T) + Kd^2T)$.

4. Regret Analysis

Our main theorem bounds the regret of Algorithm 1.

Theorem 1. *There exists a universal constant $C > 0$ such that the regret bound for Algorithm 1 with $\delta = 1/\sqrt{T}$ satisfies*

$$R_T \leq CK\sqrt{dT\log(LT)}\,.$$

Let I_ℓ be the number of calls to RecurRank with phase number ℓ. Hence each $i \in [I_\ell]$ corresponds to a call of RecurRank with phase number ℓ and the arguments are denoted by $\mathcal{A}_{\ell i}$ and $\mathcal{K}_{\ell i}$. Abbreviate $K_{\ell i} = \min \mathcal{K}_{\ell i}$ for the first position of $\mathcal{K}_{\ell i}$, $M_{\ell i} = |\mathcal{K}_{\ell i}|$ for the number of positions and $\mathcal{K}^+_{\ell i} = \mathcal{K}_{\ell i} \setminus \{K_{\ell i}\}$. We also let $K_{\ell,I_\ell+1} = K+1$ and assume that the calls $i \in [I_\ell]$ are ordered so that

$$1 = K_{\ell 1} < K_{\ell 2} < \cdots < K_{\ell I_\ell} \leq K < K+1 = K_{\ell,I_\ell+1}\,.$$

The reader is reminded that $\chi^*_k = \chi(A^*, k)$ is the examination probability of the kth position under the optimal list. Let $\chi_{\ell i} = \chi^*_{K_{\ell i}}$ be the shorthand for the optimal examination probability of the first position in call (ℓ, i). We let $\hat\theta_{\ell i}$ be the least-squares estimator computed in Eq. (7) in Algorithm 1. The maximum phase number during the entire operation of the algorithm is $\ell_{\max}$.

Definition 1. *Let F be the failure event that there exists an $\ell \in [\ell_{\max}]$, $i \in [I_\ell]$ and $a \in \mathcal{A}_{\ell i}$ such that*

$$\left|\langle\hat\theta_{\ell i}, a\rangle - \chi_{\ell i}\langle\theta_*, a\rangle\right| \geq \Delta_\ell$$

or there exists an $\ell \in [\ell_{\max}]$, $i \in [I_\ell]$ and $k \in \mathcal{K}_{\ell i}$ such that $a^*_k \notin \mathcal{A}_{\ell i}$.

The first lemma shows that the failure event occurs with low probability. The proof follows the analysis in (Lattimore & Szepesvári, 2018, Chap. 22) and is summarised in the supplementary material.

Lemma 1. $\mathbb{P}(F) \leq \delta$.

The proofs of the following lemmas are also provided in the supplementary material.

Lemma 2. *On the event F^c it holds for any $\ell \in [\ell_{\max}]$, $i \in [I_\ell]$ and positions $k, k+1 \in \mathcal{K}_{\ell i}$ that $\chi_{\ell i}(\alpha(a^*_k) - \alpha(a^*_{k+1})) \leq 8\Delta_\ell$.*

Lemma 3. *On the event F^c it holds for any $\ell \in [\ell_{\max}]$ and $a \in \mathcal{A}_{\ell I_\ell}$ that $\chi_{\ell I_\ell}(\alpha(a^*_K) - \alpha(a)) \leq 8\Delta_\ell$.*

Lemma 4. *Suppose that in its (ℓ, i)th call RecurRank places item a in position $k = K_{\ell i}$. Then, provided F^c holds, $\chi_{\ell i}(\alpha(a^*_k) - \alpha(a)) \leq 8M_{\ell i}$.*

Lemma 5. *Suppose that in its (ℓ, i)th call RecurRank places item a in position $k \in \mathcal{K}^+_{\ell i}$. Then provided F^c holds, $\chi_{\ell i}(\alpha(a^*_k) - \alpha(a)) \leq 4\Delta_\ell$.*

Proof of Theorem 1. The first step is to decompose the regret using the failure event:

$$R_T \leq \mathbb{P}(F)\, TK + \mathbb{E}\left[\mathbb{1}_{F^c}\sum_{t=1}^{T}\sum_{k=1}^{K}(v(A^*,k) - v(A_t,k))\right].$$

From now on we assume that F^c holds and bound the term inside the expectation. Given ℓ and $i \in [I_\ell]$ let $\mathcal{T}_{\ell i}$ be the set of rounds when algorithm (ℓ, i) is active. Then

$$\sum_{t=1}^{T}\sum_{k=1}^{K}(v(A^*,k) - v(A_t,k)) = \sum_{\ell=1}^{\ell_{\max}}\sum_{i=1}^{I_\ell} R_{\ell i}\,, \qquad (8)$$

where $R_{\ell i}$ is the regret incurred during call (ℓ, i):

$$R_{\ell i} = \sum_{t\in\mathcal{T}_{\ell i}}\sum_{k\in\mathcal{K}_{\ell i}}(v(A^*,k) - v(A_t,k))\,.$$

This quantity is further decomposed into the first position in $\mathcal{K}_{\ell i}$, which is used for exploration, and the remaining positions:

$$R^{(1)}_{\ell i} = \sum_{t \in \mathcal{T}_{\ell i}} (v(A^*, K_{\ell i}) - v(A_t, K_{\ell i}))\,.$$
$$R^{(2)}_{\ell i} = \sum_{t \in \mathcal{T}_{\ell i}} \sum_{k \in \mathcal{K}^+_{\ell i}} (v(A^*, k) - v(A_t, k))\,.$$

Each of these terms is bounded separately. For the first term we have

$$\begin{aligned} R^{(1)}_{\ell i} &= \sum_{t \in \mathcal{T}_{\ell i}} (v(A^*, K_{\ell i}) - v(A_t, K_{\ell i})) \\ &= \sum_{t \in \mathcal{T}_{\ell i}} \chi(A^*, K_{\ell i})\alpha(a^*_{K_{\ell i}}) - \chi(A_t, K_{\ell i})\alpha(A_t(K_{\ell i})) \\ &= \sum_{t \in \mathcal{T}_{\ell i}} \chi_{\ell i} \left\{\alpha(a^*_{K_{\ell i}}) - \alpha(A_t(K_{\ell i}))\right\} \\ &\le 8 \sum_{t \in \mathcal{T}_{\ell i}} M_{\ell i}\Delta_\ell\,, \end{aligned} \tag{9}$$

where the first equality is the definition of $R^{(1)}_{\ell i}$, the second is the definition of v. The third inequality is true because event F^c ensures that

$$\{A_t(k) : k < K_{\ell i}\} = \{a^*_k : k < K_{\ell i}\}\,,$$

which combined with Assumption 1 shows that $\chi(A^*, K_{\ell i}) = \chi(A_t, K_{\ell i}) = \chi_{\ell i}$. The inequality in Eq. (9) follows from Lemma 4. Moving on to the second term,

$$\begin{aligned} R^{(2)}_{\ell i} &= \sum_{t \in \mathcal{T}_{\ell i}} \sum_{k \in \mathcal{K}^+_{\ell i}} (v(A^*, k) - v(A_t, k)) \\ &\le \sum_{t \in \mathcal{T}_{\ell i}} \sum_{k \in \mathcal{K}^+_{\ell i}} \chi^*_k(\alpha(a^*_k) - \alpha(A_t(k))) \\ &\le \sum_{t \in \mathcal{T}_{\ell i}} \sum_{k \in \mathcal{K}^+_{\ell i}} \chi_{\ell i}(\alpha(a^*_k) - \alpha(A_t(k))) \\ &\le 4 \sum_{t \in \mathcal{T}_{\ell i}} \sum_{k \in \mathcal{K}^+_{\ell i}} \Delta_\ell \\ &\le 4 \sum_{t \in \mathcal{T}_{\ell i}} M_{\ell i}\Delta_\ell\,, \end{aligned} \tag{10}$$

where the second inequality follows from Assumption 3 and the third inequality follows from Assumption 2 on ranking A^*. The inequality in Eq. (10) follows from Lemma 5 and the one after it from the definition of $M_{\ell i} = |\mathcal{K}_{\ell i}|$. Putting things together,

$$(8) = 12 \sum_{\ell=1}^{\ell_{\max}} \sum_{i \in I_\ell} |\mathcal{T}_{\ell i}| M_{\ell i}\Delta_\ell \le 12K \sum_{\ell=1}^{\ell_{\max}} \max_{i \in I_\ell} |\mathcal{T}_{\ell i}|\Delta_\ell\,, \tag{11}$$

where we used that $\sum_{i \in I_\ell} M_{\ell i} = K$. To bound $|\mathcal{T}_{\ell i}|$ note that, on the one hand, $|\mathcal{T}_{\ell i}| \le T$ (this will be useful when ℓ is large), while on the other hand, by the definition of the algorithm and the fact that the G-optimal design is supported on at most $d(d+1)/2$ points we have

$$\begin{aligned} |\mathcal{T}_{\ell i}| &\le \sum_{a \in \mathcal{A}_{\ell i}} \left\lceil \frac{2d\pi(a)\log(1/\delta_\ell)}{\Delta^2_\ell} \right\rceil \\ &\le \frac{d(d+1)}{2} + \frac{2d\log(1/\delta_\ell)}{\Delta^2_\ell}\,. \end{aligned}$$

We now split to sum in (11) into two. For $1 \le \ell_0 \le \ell_{\max}$ to be chosen later,

$$\sum_{\ell=1}^{\ell_0} \max_{i \in I_\ell} |\mathcal{T}_{\ell i}|\Delta_\ell \le \frac{d(d+1)}{2} + 4d\log(1/\delta_{\ell_0})2^{\ell_0}\,,$$

while

$$\sum_{\ell=\ell_0+1}^{\ell_{\max}} \max_{i \in I_\ell} |\mathcal{T}_{\ell i}|\Delta_\ell \le T \sum_{\ell=\ell_0+1}^{\ell_{\max}} \Delta_\ell \le T2^{-\ell_0}\,,$$

hence,

$$(8) \le 12K \left\{ \frac{d(d+1)}{2} + 4d\log(1/\delta_{\ell_0})2^{\ell_0} + T2^{-\ell_0} \right\}\,.$$

The result is completed by optimising ℓ_0. □

5. Experiments

We run experiments to compare `RecurRank` with `CascadeLinUCB` (Li et al., 2016; Zong et al., 2016) and `TopRank` (Lattimore et al., 2018).

Synthetic experiments We construct environments using the cascade click model (CM) and the position-based click model (PBM) with $L = 10^4$ items in $d = 5$ dimension to be displayed in $K = 10$ positions. We first randomly draw item vectors $\mathcal{L}$ and weight vector θ_* in $d-1$ dimension with each entry a standard Gaussian variable, then normalise, add one more dimension with constant 1, and divide by $\sqrt{2}$. The transformation is as follows:

$$x \mapsto \left(\frac{x}{\sqrt{2}\,\|x\|}, \frac{1}{\sqrt{2}} \right)\,. \tag{12}$$

This transformation on both the item vector $x \in \mathcal{L} \subset \mathbb{R}^d$ and weight vector θ_* is to guarantee the attractiveness $\langle \theta_*, x \rangle$ of each item x lies in $[0, 1]$. The position bias for PBM is set as $\left(1, \frac{1}{2}, \frac{1}{3}, \ldots, \frac{1}{K}\right)$ which is often adopted in applications (Wang et al., 2018). The evolution of the regret as a function of time is shown in Fig. 2(a)(b). The regrets at the end and total running times are given in the supplementary material.

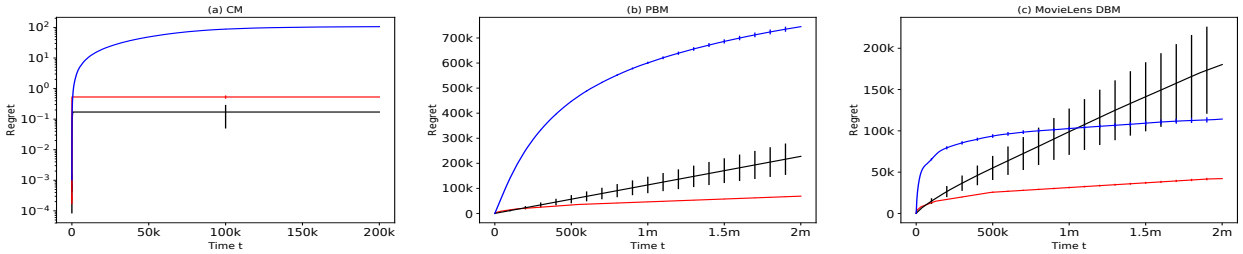

Figure 2. The figures compare `RecurRank` (red) with `CascadeLinUCB` (black) and `TopRank` (blue). Subfigure (a) shows results for an environment that follows the cascade click model (CM), while subfigure (b) does the same for the position-based click model (PBM). On these figures, regret over time is shown (smaller is better). In both models there are $L = 10^4$ items and $K = 10$ positions, and the feature space dimension is $d = 5$. Note the logarithmic scale of the y axis on subfigure (a). Subfigure (c) shows the regret over time on the MovieLens dataset with $L = 10^3$, $d = 5$, $K = 10$. All results are averaged over 10 random runs. The error bars are standard errors.

`CascadeLinUCB` is best in CM but worst in PBM because of its modelling bias. `TopRank` takes much longer time to converge than either `CascadeLinUCB` or `RecurRank` since it neither exploits the specifics of the click model, nor does it use the linear structure.

MovieLens dataset We use the $20m$ MovieLens dataset (Harper & Konstan, 2016) which contains 20 million ratings for 2.7×10^4 movies by 1.38×10^5 users. We extract $L = 10^3$ movies with most ratings and 1.1×10^3 users who rate most and randomly split the user set to two parts, U_1 and U_2 with $|U_1| = 100$ and $|U_2| = 10^3$. We then use the rating matrix of users in U_1 to derive feature vectors with $d = 5$ for all movies by singular-value decomposition (SVD). The resulting feature vectors $\mathcal{L}$ are also processed as (12). The true weight vector θ_* is computed by solving the linear system of $\mathcal{L}$ w.r.t. the rating matrix of U_2. The environment is the document-based click model (DBM) with $\mathcal{L}$ and θ_* and we set $K = 10$. The performances are measured in regret, as shown in Fig. 2(c). As can be seen, `RecurRank` learns faster than the other two algorithms. Of these two algorithms, the performance of `CascadeLinUCB` saturates: this is due to its incorrect bias.

6. Discussion

Assumptions Our assumptions are most closely related to the work by Lattimore et al. (2018) and Zoghi et al. (2017). The latter work also assumes a factored model where the probability of clicking on an item factors into an examination probability and an attractiveness function. None of these works make use of features to model the attractiveness of items: They are a special case of our model when we set the features of items to be orthogonal to each other (in particular, $d = L$). Our assumptions on the examination probability function are weaker than those by Zoghi et al. (2017). Despite this, our regret upper bound is better by a factor of K (when setting $d = L$) and the analysis is also simpler. The paper by Lattimore et al. (2018) does not assume a factored model, but instead places assumptions directly on v. They also assume a specific behaviour of the v function under pairwise exchanges that is not required here. Their assumptions are weaker in the sense that they do not assume the probability of clicking on position k only depends on the identities of the items in positions $[k-1]$ and the attractiveness of the item in position k. On the other hand, they do assume a specific behaviour of the v function under pairwise exchanges that is not required by our analysis. It is unclear which set of these assumptions is preferable.

Lower bounds In the orthogonal case where $d = L$ the lower bound in (Lattimore et al., 2018) provides an example where the regret is at least $\Omega(\sqrt{TKL})$. For $d \leq L$, the standard techniques for proving lower bounds for linear bandits can be used to prove the regret is at least $\Omega(\sqrt{dTK})$, which except for logarithmic terms means our upper bound is suboptimal by a factor of at most $\sqrt{K}$. We are not sure whether either the lower bound or the upper bound is tight.

Open questions Only using data from the first position seems suboptimal, but is hard to avoid without making additional assumptions. Nevertheless, we believe a small improvement should be possible here. Another natural question is how to deal with the situation when the set of available items is changing. In practice this happens in many applications, either because the features are changing or because new items are being added or removed. Other interesting directions are to use weighted least-squares estimators to exploit the low variance when the examination probability and attractiveness are small. Additionally one can use a generalised linear model instead of the linear model to model the attractiveness function, which may be analysed using techniques developed by Filippi et al. (2010) and Jun et al. (2017). Finally, it could be interesting to generalise to the setting where item vectors are sparse (see Abbasi-Yadkori et al. 2012 and Lattimore & Szepesvári 2018, Chap. 23).

References

Abbasi-Yadkori, Y., Pál, D., and Szepesvári, C. Improved algorithms for linear stochastic bandits. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 24*, NIPS, pp. 2312–2320. Curran Associates, Inc., 2011.

Abbasi-Yadkori, Y., Pal, D., and Szepesvári, C. Online-to-confidence-set conversions and application to sparse stochastic bandits. In Lawrence, N. D. and Girolami, M. (eds.), *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pp. 1–9, La Palma, Canary Islands, 21–23 Apr 2012. PMLR.

Abe, N. and Long, P. M. Associative reinforcement learning using linear probabilistic concepts. In *Proceedings of the 16th International Conference on Machine Learning*, ICML, pp. 3–11, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

Auer, P. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Bubeck, S., Cesa-Bianchi, N., and Kakade, S. Towards minimax policies for online linear optimization with bandit feedback. In *Annual Conference on Learning Theory*, volume 23, pp. 41–1. Microtome, 2012.

Chaudhuri, S. *Learning to Rank: Online Learning, Statistical Theory and Applications.* PhD thesis, 2016.

Chen, Y. and Hofmann, K. Online learning to rank: Absolute vs. relative. In *Proceedings of the 24th International Conference on World Wide Web*, pp. 19–20. ACM, 2015.

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

Combes, R., Magureanu, S., Proutiere, A., and Laroche, C. Learning to rank: Regret lower bounds and efficient algorithms. In *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pp. 231–244. ACM, 2015. ISBN 978-1-4503-3486-0.

Filippi, S., Cappe, O., Garivier, A., and Szepesvári, C. Parametric bandits: The generalized linear case. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A. (eds.), *Advances in Neural Information Processing Systems 23*, NIPS, pp. 586–594. Curran Associates, Inc., 2010.

Harper, F. M. and Konstan, J. A. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):19, 2016. URL https://grouplens.org/datasets/movielens/20m/.

Hazan, E. and Karnin, Z. Volumetric spanners: an efficient exploration basis for learning. *The Journal of Machine Learning Research*, 17(1):4062–4095, 2016.

Hofmann, K., Whiteson, S., and De Rijke, M. A probabilistic method for inferring preferences from clicks. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pp. 249–258. ACM, 2011.

Jun, K., Bhargava, A., Nowak, R., and Willett, R. Scalable generalized linear bandits: Online computation and hashing. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 99–109. Curran Associates, Inc., 2017.

Katariya, S., Kveton, B., Szepesvári, C., and Wen, Z. DCM bandits: Learning to rank with multiple clicks. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 1215–1224, 2016.

Katariya, S., Kveton, B., Szepesvári, C., Vernade, C., and Wen, Z. Bernoulli rank-1 bandits for click feedback. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017a.

Katariya, S., Kveton, B., Szepesvári, C., Vernade, C., and Wen, Z. Stochastic rank-1 bandits. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017b.

Kiefer, J. and Wolfowitz, J. The equivalence of two extremum problems. *Canadian Journal of Mathematics*, 12 (5):363–365, 1960.

Kveton, B., Szepesvári, C., Wen, Z., and Ashkan, A. Cascading bandits: Learning to rank in the cascade model. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, pp. 767–776. JMLR.org, 2015.

Lagree, P., Vernade, C., and Cappé, O. Multiple-play bandits in the position-based model. In *Advances in Neural Information Processing Systems 29*, NIPS, pp. 1597–1605. Curran Associates Inc., 2016.

Lattimore, T. and Szepesvári, C. *Bandit Algorithms*. preprint, 2018.

Lattimore, T., Kveton, B., Li, S., and Szepesvári, C. Toprank: A practical algorithm for online stochastic ranking. In *Proceedings of the 31st Conference on Neural Information Processing Systems*. 2018.

Li, L., Chu, W., Langford, J., and Schapire, R. E. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on world wide web*, pp. 661–670. ACM, 2010.

Li, S. and Zhang, S. Online clustering of contextual cascading bandits. In *The 32nd AAAI Conference on Artificial Intelligence*, pp. 3554–3561, 2018.

Li, S., Wang, B., Zhang, S., and Chen, W. Contextual combinatorial cascading bandits. In *Proceedings of the 33rd International Conference on Machine Learning*, pp. 1245–1253, 2016.

Liu, W., Li, S., and Zhang, S. Contextual dependent click bandit algorithm for web recommendation. In *International Computing and Combinatorics Conference*, pp. 39–50. Springer, 2018.

Radlinski, F., Kleinberg, R., and Joachims, T. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 784–791. ACM, 2008.

Rustichini, A. Minimizing regret: The general case. *Games and Economic Behavior*, 29(1):224–243, 1999.

Slivkins, A., Radlinski, F., and Gollapudi, S. Ranked bandits in metric spaces: learning diverse rankings over large document collections. *Journal of Machine Learning Research*, 14(Feb):399–436, 2013.

Soare, M., Lazaric, A., and Munos, R. Best-arm identification in linear bandits. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27*, NIPS, pp. 828–836. Curran Associates, Inc., 2014.

Todd, M. J. *Minimum-volume ellipsoids: Theory and algorithms*. SIAM, 2016.

Valko, M., Munos, R., Kveton, B., and Kocák, T. Spectral bandits for smooth graph functions. In Xing, E. P. and Jebara, T. (eds.), *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 46–54, Bejing, China, 22–24 Jun 2014. PMLR.

Wang, X., Golbandi, N., Bendersky, M., Metzler, D., and Najork, M. Position bias estimation for unbiased learning to rank in personal search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 610–618. ACM, 2018.

Xu, L., Honda, J., and Sugiyama, M. Fully adaptive algorithm for pure exploration in linear bandits. *arXiv preprint arXiv:1710.05552*, 2017.

Zoghi, M., Tunys, T., Ghavamzadeh, M., Kveton, B., Szepesvári, C., and Wen, Z. Online learning to rank in stochastic click models. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *PMLR*, pp. 4199–4208, 2017.

Zong, S., Ni, H., Sung, K., Ke, R. N., Wen, Z., and Kveton, B. Cascading bandits for large-scale recommendation problems. In *Proceedings of the 32nd Conference on Uncertainty in Artificial Intelligence*, UAI, 2016.