Bernoulli Rank-1 Bandits for Click Feedback

Sumeet Katariya [1], Branislav Kveton [2], Csaba Szepesvári [3], Claire Vernade [4], Zheng Wen [2]
[1]University of Wisconsin-Madison, [2]Adobe Research, [3]University of Alberta, [4]Telecom ParisTech
katariya@wisc.edu, kveton@adobe.com, szepesva@cs.ualberta.ca
claire.vernade@telecom-paristech.fr, zwen@adobe.com

Abstract

The probability that a user will click a search result depends both on its relevance and its position on the results page. The *position based model* explains this behavior by ascribing to every item an *attraction* probability, and to every position an *examination* probability. To be clicked, a result must be both attractive and examined. The probabilities of an item-position pair being clicked thus form the entries of a rank-1 matrix. We propose the learning problem of a *Bernoulli rank-1 bandit* where at each step, the learning agent chooses a pair of row and column arms, and receives the product of their Bernoulli-distributed values as a reward. This is a special case of the stochastic rank-1 bandit problem considered in recent work that proposed an elimination based algorithm `Rank1Elim`, and showed that `Rank1Elim`'s regret scales linearly with the number of rows and columns on "benign" instances. These are the instances where the minimum of the average row and column rewards μ is bounded away from zero. The issue with `Rank1Elim` is that it fails to be competitive with straightforward bandit strategies as $\mu \to 0$. In this paper we propose `Rank1ElimKL`, which replaces the crude confidence intervals of `Rank1Elim` with confidence intervals based on Kullback-Leibler (KL) divergences. With the help of a novel result concerning the scaling of KL divergences we prove that with this change, our algorithm will be competitive no matter the value of μ. Experiments with synthetic data confirm that on benign instances the performance of `Rank1ElimKL` is significantly better than that of even `Rank1Elim`. Similarly, experiments with models derived from real-data confirm that the improvements are significant across the board, regardless of whether the data is benign or not.

1 Introduction

When deciding which search results to present, click logs are of particular interest. A fundamental problem in click data is position bias. The probability of an element being clicked depends not only on its relevance, but also on its position on the results page. The position-based model (PBM), first proposed by Richardson *et al.* [2007] and then formalized by Craswell *et al.* [2008], models this behavior by associating with each item a probability of being *attractive*, and with each position a probability of being *examined*. To be clicked, a result must be both attractive and examined. Given click logs, the attraction and examination probabilities can be learned using the maximum-likelihood estimation (MLE) or the expectation-maximization (EM) algorithms [Chuklin *et al.*, 2015].

An online learning model for this problem is proposed in Katariya *et al.* [2017], called *stochastic rank-1 bandit*. The objective of the learning agent is to learn the most rewarding item and position, which is the maximum entry of a rank-1 matrix. At time t, the agent chooses a pair of row and column arms, and receives the product of their values as a reward. The goal of the agent is to maximize its expected cumulative reward, or equivalently to minimize its expected cumulative regret with respect to the optimal solution, the most rewarding pair of row and column arms. This learning problem is challenging because when the agent receives the reward of zero, it could mean either that the item was unattractive, or the position was not examined, or both.

Katariya *et al.* [2017] also proposed an elimination algorithm, `Rank1Elim`, whose regret is $\mathcal{O}((K + L)\,\mu^{-2}\Delta^{-1}\log n)$, where K is the number of rows, L is the number of columns, Δ is the minimum of the row and column gaps, and μ is the minimum of the average row and column rewards. When μ is bounded away from zero, the regret scales linearly with $K + L$, while it scales inversely with Δ. This is a significant improvement over using a standard bandit algorithm that (disregarding the problem structure) would treat item-position pairs as unrelated arms and would achieve a regret of $O(KL\Delta^{-1}\log n)$. The issue is that as μ gets small, the regret bound worsens significantly. As we verify in Section 5, this indeed happens on models derived from some real-world problems. To illustrate the severity of this problem, consider as an example where $K = L$, and the row and column rewards are Bernoulli distributed. Let the mean reward of row 1 and column 1 be Δ, and the mean reward of all other rows and columns be 0. We refer to this setting as a "needle in a haystack", because there is a single rewarding entry out of K^2 entries. For this setting, $\mu = \Delta/K$, and consequently the regret of `Rank1Elim` is $\mathcal{O}(\mu^{-2}\Delta^{-1}K\log n) = \mathcal{O}(K^3\log n)$. However, a naive

bandit algorithm that ignores the rank-1 structure and treats each row-column pair as unrelated arms has $\mathcal{O}(K^2 \log n)$ regret.[1] While a naive bandit algorithm is unable to exploit the rank-1 structure when μ is large, `Rank1Elim` is unable to keep up with a naive algorithm when μ is small. Our goal in this paper is to design an algorithm that performs well across all rank-1 problem instances regardless of their parameters.

In this paper we propose that this improvement can be achieved by replacing the "`UCB1` confidence intervals" used by `Rank1Elim` by strictly tighter confidence intervals based on Kullback-Leibler (KL) divergences. This leads to our algorithm that we call `Rank1ElimKL`. Based on the work of Garivier and Cappe [2011], we expect this change to lead to an improved behavior, especially for extreme instances, as $\mu \to 0$. Indeed, in this paper we show that KL divergences enjoy a peculiar "scaling". In particular, thanks to this improvement, for the "needle in a haystack" problem discussed above the regret of `Rank1ElimKL` becomes $\mathcal{O}(K^2 \log n)$.

Our contributions are as follows. First, we propose a *Bernoulli rank-*1 *bandit*, which is a special class of a *stochastic rank-*1 *bandit* where the rewards are Bernoulli distributed. Second, we modify `Rank1Elim` for solving the Bernoulli rank-1 bandit, which we call `Rank1ElimKL`, to use `KL-UCB` intervals. Third, we derive a $O((K + L)\,(\mu\gamma\Delta)^{-1} \log n)$ gap-dependent upper bound on the n-step regret of `Rank1ElimKL`, where K, L, Δ and μ are as above, while $\gamma = \max\{\mu, 1 - p_{\max}\}$ with $p_{\max}$ being the maximum of the row and column rewards; effectively replacing the μ^{-2} term of the previous regret bound of `Rank1Elim` with $(\mu\gamma)^{-1}$. It follows that the new bound is an unilateral improvement over the previous one and is a strict improvement when $\mu < 1 - p_{\max}$, which is expected to happen quite often in practical problems. For the "needle in a haystack" problem, the new bound essentially matches that of the naive bandit algorithm, while never worsening the bound of `Rank1Elim`. Our final contribution is the experimental validation of `Rank1ElimKL`, on both synthetic and real-world problems. The experiments indicate that `Rank1ElimKL` outperforms several baselines across almost all problem instances.

We denote random variables by boldface letters and define $[n] = \{1, \dots, n\}$. For any sets A and B, we denote by A^B the set of all vectors whose entries are indexed by B and take values from A. We let $d(p, q) = p \log \frac{p}{q} + (1 - p) \log \frac{1-p}{1-q}$ denote the KL divergence between the Bernoulli distributions with means $p, q \in [0, 1]$. As usual, the formula for $d(p, q)$ is defined through its continuous extension as p, q approach the boundaries of $[0, 1]$.

2 Setting

The setting of the *Bernoulli rank-*1 *bandit* is the same as that of the stochastic rank-1 bandit [Katariya *et al.*, 2017], with the additional requirement that the row and column rewards are Bernoulli distributed. We state the setting for completeness, and borrow the notation from Katariya *et al.* [2017] for the ease of comparison.

An instance of our learning problem is defined by a tuple $(K, L, P_{\mathrm{U}}, P_{\mathrm{V}})$, where K is the number of rows, L is the number of columns, P_{U} is a distribution over $\{0, 1\}^K$ from which the row rewards are drawn, and P_{V} is a distribution over $\{0, 1\}^L$ from which the column rewards are drawn.

Let the row and column rewards be

$$(\mathbf{u}_t, \mathbf{v}_t) \overset{\text{i.i.d}}{\sim} P_{\mathrm{U}} \otimes P_{\mathrm{V}}\,, \qquad t = 1, \dots, n\,.$$

In particular, $\mathbf{u}_t$ and $\mathbf{v}_t$ are drawn independently at any time t. At time t, the learning agent chooses a row index $\mathbf{i}_t \in [K]$ and a column index $\mathbf{j}_t \in [L]$, and observes $\mathbf{u}_t(\mathbf{i}_t)\mathbf{v}(\mathbf{j}_t)$ as its reward. The indices $\mathbf{i}_t$ and $\mathbf{j}_t$ chosen by the learning agent are allowed to depend only on the history of the agent up to time t.

Let the time horizon be n. The goal of the agent is to maximize its expected cumulative reward in n steps. This is equivalent to minimizing the *expected cumulative regret* in n steps

$$R(n) = \mathbb{E}\left[\sum_{t=1}^{n} R(\mathbf{i}_t, \mathbf{j}_t, \mathbf{u}_t, \mathbf{v}_t)\right]\,,$$

where $R(\mathbf{i}_t, \mathbf{j}_t, \mathbf{u}_t, \mathbf{v}_t) = \mathbf{u}_t(i^*)\mathbf{v}_t(j^*) - \mathbf{u}_t(\mathbf{i}_t)\mathbf{v}_t(\mathbf{j}_t)$ is the *instantaneous stochastic regret* of the agent at time t, and

$$(i^*, j^*) = \arg\max_{(i,j) \in [K] \times [L]} \mathbb{E}\left[\mathbf{u}(i)\mathbf{v}(j)\right]$$

is the *optimal solution* in hindsight of knowing P_{U} and P_{V}.

3 `Rank1ElimKL` Algorithm

The pseudocode of our algorithm, `Rank1ElimKL`, is in Algorithm 1. As noted earlier this algorithm is based on `Rank1Elim` [Katariya *et al.*, 2017] with the difference that we replace their confidence intervals with KL-based confidence intervals. For the reader's benefit, we explain the full algorithm.

`Rank1ElimKL` is an elimination algorithm that operates in stages, where the elimination is conducted with `KL-UCB` confidence intervals. The lengths of the stages quadruple from one stage to the next, and the algorithm is designed such that at the end of stage ℓ, it eliminates with high probability any row and column whose gap scaled by a problem dependent constant is at least $\tilde{\Delta}_\ell = 2^{-\ell}$. We denote the *remaining rows and columns* in stage ℓ by $\mathbf{I}_\ell$ and $\mathbf{J}_\ell$, respectively.

Every stage has an exploration phase and an exploitation phase. During row-exploration in stage ℓ (lines 12–16), every remaining row is played with a randomly chosen remaining column,

In the exploitation phase, we construct high-probability `KL-UCB` [Garivier and Cappe, 2011] confidence intervals $[\mathbf{L}^{\mathrm{U}}_\ell(i), \mathbf{U}^{\mathrm{U}}_\ell(i)]$ for row $i \in \mathbf{I}_\ell$, and confidence intervals $[\mathbf{L}^{\mathrm{V}}_\ell(j), \mathbf{U}^{\mathrm{V}}_\ell(j)]$ for column $j \in \mathbf{J}_\ell$. As noted earlier, this is where we depart from `Rank1Elim`. The elimination uses row $\mathbf{i}_\ell$ and column $\mathbf{j}_\ell$, where

$$\mathbf{i}_\ell = \arg\max_{i \in \mathbf{I}_\ell} \mathbf{L}^{\mathrm{U}}_\ell(i)\,, \qquad \mathbf{j}_\ell = \arg\max_{j \in \mathbf{J}_\ell} \mathbf{L}^{\mathrm{V}}_\ell(j)\,.$$

[1] Alternatively, the worst-case regret bound for `Rank1Elim` becomes $O(Kn^{2/3} \log n)$, while that of for a naive bandit algorithm with a naive bound is $O(Kn^{1/2} \log n)$.

Algorithm 1 `Rank1ElimKL` for Bernoulli rank-1 bandits.

1: // Initialization
2: $t \leftarrow 1,\ \tilde{\Delta}_0 \leftarrow 1, n_{-1} \leftarrow 0$
3: $\mathbf{C}_0^{\mathrm{U}} \leftarrow 0_{K,L}$, $\mathbf{C}_0^{\mathrm{V}} \leftarrow 0_{K,L}$ // Zero matrix with K rows and L columns
4: $\mathbf{h}_0^{\mathrm{U}} \leftarrow (1, \dots, K),\ \mathbf{h}_0^{\mathrm{V}} \leftarrow (1, \dots, L)$
5:
6: **for** $\ell = 0, 1, \dots$ **do**
7: $\quad n_\ell \leftarrow \left\lceil 16 \tilde{\Delta}_\ell^{-2} \log n \right\rceil$
8: $\quad \mathbf{I}_\ell \leftarrow \bigcup_{i \in [K]} \{\mathbf{h}_\ell^{\mathrm{U}}(i)\},\ \mathbf{J}_\ell \leftarrow \bigcup_{j \in [L]} \{\mathbf{h}_\ell^{\mathrm{V}}(j)\}$
9:
10: $\quad$ // Row and column exploration
11: $\quad$ **for** $n_\ell - n_{\ell-1}$ times **do**
12: $\quad\quad$ Choose uniformly at random column $j \in [L]$
13: $\quad\quad j \leftarrow \mathbf{h}_\ell^{\mathrm{V}}(j)$
14: $\quad\quad$ **for all** $i \in \mathbf{I}_\ell$ **do**
15: $\quad\quad\quad \mathbf{C}_\ell^{\mathrm{U}}(i,j) \leftarrow \mathbf{C}_\ell^{\mathrm{U}}(i,j) + \mathbf{u}_t(i)\mathbf{v}_t(j)$
16: $\quad\quad\quad t \leftarrow t + 1$
17: $\quad\quad$ Choose uniformly at random row $i \in [K]$
18: $\quad\quad i \leftarrow \mathbf{h}_\ell^{\mathrm{U}}(i)$
19: $\quad\quad$ **for all** $j \in \mathbf{J}_\ell$ **do**
20: $\quad\quad\quad \mathbf{C}_\ell^{\mathrm{V}}(i,j) \leftarrow \mathbf{C}_\ell^{\mathrm{V}}(i,j) + \mathbf{u}_t(i)\mathbf{v}_t(j)$
21: $\quad\quad\quad t \leftarrow t + 1$
22:
23: $\quad$ // UCBs and LCBs on the expected rewards of all remaining rows and columns with divergence constraint $\delta_\ell \leftarrow \log n + 3 \log\log n$
24:
25: $\quad$ **for all** $i \in \mathbf{I}_\ell$ **do**
26: $\quad\quad \hat{\mathbf{u}}_\ell(i) \leftarrow n_\ell^{-1} \sum_{j=1}^{L} \mathbf{C}_\ell^{\mathrm{U}}(i,j)$
27: $\quad\quad \mathbf{U}_\ell^{\mathrm{U}}(i) \leftarrow \arg\max_{q \in [\hat{\mathbf{u}}_\ell(i), 1]} \{n_\ell d(\hat{\mathbf{u}}_\ell(i), q) \le \delta_\ell\}$
28: $\quad\quad \mathbf{L}_\ell^{\mathrm{U}}(i) \leftarrow \arg\min_{q \in [0, \hat{\mathbf{u}}_\ell(i)]} \{n_\ell d(\hat{\mathbf{u}}_\ell(i), q) \le \delta_\ell\}$
29: $\quad$ **for all** $j \in \mathbf{J}_\ell$ **do**
30: $\quad\quad \hat{\mathbf{v}}_\ell(j) \leftarrow n_\ell^{-1} \sum_{i=1}^{K} \mathbf{C}_\ell^{\mathrm{V}}(i,j)$
31: $\quad\quad \mathbf{U}_\ell^{\mathrm{V}}(j) \leftarrow \arg\max_{q \in [\hat{\mathbf{v}}_\ell(j), 1]} \{n_\ell d(\hat{\mathbf{v}}_\ell(j), q) \le \delta_\ell\}$
32: $\quad\quad \mathbf{L}_\ell^{\mathrm{V}}(j) \leftarrow \arg\min_{q \in [0, \hat{\mathbf{v}}_\ell(j)]} \{n_\ell d(\hat{\mathbf{v}}_\ell(j), q) \le \delta_\ell\}$
33:
34: $\quad$ // Row and column elimination
35: $\quad \mathbf{i}_\ell \leftarrow \arg\max_{i \in \mathbf{I}_\ell} \mathbf{L}_\ell^{\mathrm{U}}(i)$
36: $\quad \mathbf{h}_{\ell+1}^{\mathrm{U}} \leftarrow \mathbf{h}_\ell^{\mathrm{U}}$
37: $\quad$ **for** $i = 1, \dots, K$ **do**
38: $\quad\quad$ **if** $\mathbf{U}_\ell^{\mathrm{U}}(\mathbf{h}_\ell^{\mathrm{U}}(i)) \le \mathbf{L}_\ell^{\mathrm{U}}(\mathbf{i}_\ell)$ **then**
39: $\quad\quad\quad \mathbf{h}_{\ell+1}^{\mathrm{U}}(i) \leftarrow \mathbf{i}_\ell$
40:
41: $\quad \mathbf{j}_\ell \leftarrow \arg\max_{j \in \mathbf{J}_\ell} \mathbf{L}_\ell^{\mathrm{V}}(j)$
42: $\quad \mathbf{h}_{\ell+1}^{\mathrm{V}} \leftarrow \mathbf{h}_\ell^{\mathrm{V}}$
43: $\quad$ **for** $j = 1, \dots, L$ **do**
44: $\quad\quad$ **if** $\mathbf{U}_\ell^{\mathrm{V}}(\mathbf{h}_\ell^{\mathrm{V}}(j)) \le \mathbf{L}_\ell^{\mathrm{V}}(\mathbf{j}_\ell)$ **then**
45: $\quad\quad\quad \mathbf{h}_{\ell+1}^{\mathrm{V}}(j) \leftarrow \mathbf{j}_\ell$
46:
47: $\quad \tilde{\Delta}_{\ell+1} \leftarrow \tilde{\Delta}_\ell / 2,\ \mathbf{C}_{\ell+1}^{\mathrm{U}} \leftarrow \mathbf{C}_\ell^{\mathrm{U}},\ \mathbf{C}_{\ell+1}^{\mathrm{V}} \leftarrow \mathbf{C}_\ell^{\mathrm{V}}$

We eliminate any row i and column j such that

$$\mathbf{U}_\ell^{\mathrm{U}}(i) \le \mathbf{L}_\ell^{\mathrm{U}}(\mathbf{i}_\ell)\,, \qquad \mathbf{U}_\ell^{\mathrm{V}}(j) \le \mathbf{L}_\ell^{\mathrm{V}}(\mathbf{j}_\ell)\,.$$

We also track the remaining rows and columns in stage ℓ by $\mathbf{h}_\ell^{\mathrm{U}}$ and $\mathbf{h}_\ell^{\mathrm{V}}$, respectively. When row i is eliminated by row $\mathbf{i}_\ell$, we set $\mathbf{h}_\ell^{\mathrm{U}}(i) = \mathbf{i}_\ell$. If row $\mathbf{i}_\ell$ is eliminated by row $\mathbf{i}_{\ell'}$ at a later stage $\ell' > \ell$, we update $\mathbf{h}_\ell^{\mathrm{U}}(i) = \mathbf{i}_{\ell'}$. This is analogous for columns. The remaining rows $\mathbf{I}_\ell$ and columns $\mathbf{J}_\ell$ can be then defined as the unique values in $\mathbf{h}_\ell^{\mathrm{U}}$ and $\mathbf{h}_\ell^{\mathrm{V}}$, respectively. The maps $\mathbf{h}_\ell^{\mathrm{U}}$ and $\mathbf{h}_\ell^{\mathrm{V}}$ help to guarantee that the row and column means are non-decreasing.

The `KL-UCB` confidence intervals in `Rank1ElimKL` can be found by solving a one-dimensional convex optimization problem for every row (lines 27–28) and column (lines 31–32). They can be found efficiently using binary search because the Kullback-Leibler divergence $d(x, q)$ is convex in q as q moves away from x in either direction. The `KL-UCB` confidence intervals need to be computed only once per stage. Hence, `Rank1ElimKL` has to solve at most $K + L$ convex optimization problems per stage, and hence $(K + L) \log n$ problems overall.

4 Analysis

In this section, we derive a gap-dependent upper bound on the n-step regret of `Rank1ElimKL`. The hardness of our learning problem is measured by two kinds of metrics. The first kind are gaps. The *gaps* of row $i \in [K]$ and column $j \in [L]$ are defined as

$$\Delta_i^{\mathrm{U}} = \bar{u}(i^*) - \bar{u}(i)\,, \quad \Delta_j^{\mathrm{V}} = \bar{v}(j^*) - \bar{v}(j)\,, \tag{1}$$

respectively; and the *minimum row and column gaps* are defined as

$$\Delta_{\min}^{\mathrm{U}} = \min_{i \in [K]: \Delta_i^{\mathrm{U}} > 0} \Delta_i^{\mathrm{U}}\,, \quad \Delta_{\min}^{\mathrm{V}} = \min_{j \in [L]: \Delta_j^{\mathrm{V}} > 0} \Delta_j^{\mathrm{V}}\,, \tag{2}$$

respectively. Roughly speaking, the smaller the gaps, the harder the problem. This inverse dependence on gaps is tight [Katariya *et al.*, 2017].

The second kind of metrics are the extremal parameters

$$\mu = \min\left\{ \frac{1}{K} \sum_{i=1}^{K} \bar{u}(i),\ \frac{1}{L} \sum_{j=1}^{L} \bar{v}(j) \right\}, \tag{3}$$

$$p_{\max} = \max\left\{ \max_{i \in [K]} \bar{u}(i),\ \max_{j \in [L]} \bar{v}(j) \right\}. \tag{4}$$

The first metric, μ, is the minimum of the average of entries of $\bar{u}$ and $\bar{v}$. This quantity appears in our analysis due to the averaging character of `Rank1ElimKL`. The smaller the value of μ, the larger the regret. The second metric, $p_{\max}$, is the maximum entry in $\bar{u}$ and $\bar{v}$. As we shall see the regret scales inversely with

$$\gamma = \max\{\mu, 1 - p_{\max}\}\,. \tag{5}$$

Note that if $\mu \to 0$ and $p_{\max} \to 1$ at the same time, then the row and columns gaps must also approach one. With this we are ready to state our main result.

Theorem 1. *Let $C = 6e + 82$ and $n \ge 5$. Then the expected n-step regret of* `Rank1ElimKL` *is bounded as*

$$R(n) \le \frac{160}{\mu\gamma} \left(\sum_{i=1}^{K} \frac{1}{\Delta_i^{\mathrm{U}}} + \sum_{j=1}^{L} \frac{1}{\Delta_j^{\mathrm{V}}} \right) \log n + C(K + L)\,,$$

where

$$\bar{\Delta}_i^{\mathrm{U}} = \Delta_i^{\mathrm{U}} + \mathbb{1}\{\Delta_i^{\mathrm{U}} = 0\}\, \Delta_{\min}^{\mathrm{V}}\,,$$
$$\bar{\Delta}_j^{\mathrm{V}} = \Delta_j^{\mathrm{V}} + \mathbb{1}\{\Delta_j^{\mathrm{V}} = 0\}\, \Delta_{\min}^{\mathrm{U}}\,.$$

The difference from the main result of Katariya *et al.* [2017] is that the first term in our bound scales with $1/(\mu\gamma)$ instead of $1/\mu^2$. Since $\mu \leq \gamma$ and in fact often $\mu \ll \gamma$, this is a significant improvement. We validate this empirically in the next section.

Due to the lack of space, we only provide a sketch of the proof of Theorem 1. At a high level, it follows the steps of the proof of Katariya *et al.* [2017]. Focusing on the source of the improvement, we first state and prove a new lemma, which allows us to replace one $1/\mu$ in the regret bound with $1/\gamma$. Recall from Section 1 that d denotes the KL divergence between Bernoulli random variables with means $p, q \in [0, 1]$.

Lemma 1. *Let $c, p, q \in [0, 1]$. Then*

$$c(1 - \max\{p, q\})d(p, q) \leq d(cp, cq) \leq cd(p, q)\,. \quad (6)$$

In particular,

$$2c\max(c, 1 - \max\{p, q\})(p - q)^2 \leq d(cp, cq)\,. \quad (7)$$

Proof. The proof of (6) is based on differentiation. The first two derivatives of $d(cp, cq)$ with respect to q are

$$\frac{\partial}{\partial q} d(cp, cq) = \frac{c(q - p)}{q(1 - cq)}\,,$$
$$\frac{\partial^2}{\partial q^2} d(cp, cq) = \frac{c^2(q - p)^2 + cp(1 - cp)}{q^2(1 - cq)^2}\,;$$

and the first two derivatives of $cd(p, q)$ with respect to q are

$$\frac{\partial}{\partial q}[cd(p, q)] = \frac{c(q - p)}{q(1 - q)}\,,$$
$$\frac{\partial^2}{\partial q^2}[cd(p, q)] = \frac{c(q - p)^2 + cp(1 - p)}{q^2(1 - q)^2}\,.$$

The second derivatives show that both $d(cp, cq)$ and $cd(p, q)$ are convex in q for any p. The minima are at $q = p$.

We fix p and c, and prove (6) for any q. The upper bound is derived as follows. Since

$$d(cp, cx) = cd(p, x) = 0$$

when $x = p$, the upper bound holds if $cd(p, x)$ increases faster than $d(cp, cx)$ for any $p < x \leq q$, and if $cd(p, x)$ decreases faster than $d(cp, cx)$ for any $q \leq x < p$. This follows from the definitions of $\frac{\partial}{\partial x} d(cp, cx)$ and $\frac{\partial}{\partial x}[cd(p, x)]$. In particular, both derivatives have the same sign for any x, and $1/(1 - cx) \leq 1/(1 - x)$ for $x \in [\min\{p, q\}, \max\{p, q\}]$.

The lower bound is derived as follows. Note that the ratio of $\frac{\partial}{\partial x}[cd(p, x)]$ and $\frac{\partial}{\partial x} d(cp, cx)$ is bounded from above as

$$\frac{\frac{\partial}{\partial x}[cd(p, x)]}{\frac{\partial}{\partial x} d(cp, cx)} = \frac{1 - cx}{1 - x} \leq \frac{1}{1 - x} \leq \frac{1}{1 - \max\{p, q\}}$$

for any $x \in [\min\{p, q\}, \max\{p, q\}]$. Therefore, we get a lower bound on $d(cp, cq)$ when we multiply $cd(p, q)$ by $1 - \max\{p, q\}$.

To prove (7) note that by Pinsker's inequality, for any p, q, $d(p, q) \geq 2(p-q)^2$. Hence, on one hand, $d(cp, cq) \geq 2c^2(p-q)^2$. On the other hand, we have from (6) that $d(cp, cq) \geq 2c(1 - \max\{p, q\})(p - q)^2$. Taking the maximum of the right-hand sides in these two equations gives (7). ■

Proof sketch of Theorem 1. We proceed along the lines of Katariya *et al.* [2017]. The key step in their analysis is the upper bound on the expected n-step regret of any suboptimal row $i \in [K]$. This bound is proved as follows. First, Katariya *et al.* [2017] show that row i is eliminated with a high probability after $O((\mu\Delta_i^{\mathrm{U}})^{-2} \log n)$ observations, for any column elimination strategy. Then they argue that the amortized per-observation regret before the elimination is $O(\Delta_i^{\mathrm{U}})$. Therefore, the maximum regret due to row i is $O(\mu^{-2}(\Delta_i^{\mathrm{U}})^{-1} \log n)$. The expected n-step regret of any suboptimal column $j \in [L]$ is bounded analogously.

We modify the above argument as follows. Roughly speaking, due to the `KL-UCB` confidence interval, a suboptimal row i is eliminated with a high probability after

$$O\left(\frac{1}{d(\mu(\bar{u}(i^*) - \Delta_i^{\mathrm{U}}), \mu\bar{u}(i^*))} \log n\right)$$

observations. Therefore, the expected n-step regret due to exploring row i is

$$O\left(\frac{\Delta_i^{\mathrm{U}}}{d(\mu(\bar{u}(i^*) - \Delta_i^{\mathrm{U}}), \mu\bar{u}(i^*))} \log n\right)\,.$$

Now we apply (7) of Lemma 1 to get that the regret is

$$O\left(\frac{1}{\mu\gamma\Delta_i^{\mathrm{U}}} \log n\right)\,.$$

The regret of any suboptimal column $j \in [L]$ is bounded analogously. ■

5 Experiments

We conduct two experiments. In Section 5.1, we compare our algorithm to other algorithms in the literature on a synthetic problem. In Section 5.2, we evaluate the same algorithms on click models that are trained on a real-world dataset.

5.1 Comparison to Alternative Algorithms

Following Katariya *et al.* [2017], we consider the "needle in a haystack" class of problems, where only one item is attractive and one position is examined. We recall the problem here. The i-th entry of $\mathbf{u}_t$, $\mathbf{u}_t(i)$, and the j-th entry of $\mathbf{v}_t$, $\mathbf{v}_t(j)$, are independent Bernoulli variables with means

$$\bar{u}(i) = p_{\mathrm{U}} + \Delta_{\mathrm{U}}\mathbb{1}\{i = 1\}\,,$$
$$\bar{v}(j) = p_{\mathrm{V}} + \Delta_{\mathrm{V}}\mathbb{1}\{j = 1\}\,, \quad (8)$$

for some $(p_{\mathrm{U}}, p_{\mathrm{V}}) \in [0, 1]^2$ and gaps $(\Delta_{\mathrm{U}}, \Delta_{\mathrm{V}}) \in (0, 1-p_{\mathrm{U}}] \times (0, 1 - p_{\mathrm{V}}]$. Note that arm $(1, 1)$ is optimal with an expected reward of $(p_{\mathrm{U}} + \Delta_{\mathrm{U}})(p_{\mathrm{V}} + \Delta_{\mathrm{V}})$.

The goal of this experiment is to compare `Rank1ElimKL` with five other algorithms from the literature and validate that its regret scales linearly with K and L, which implies that it exploits the problem structure. In this experiment, we set

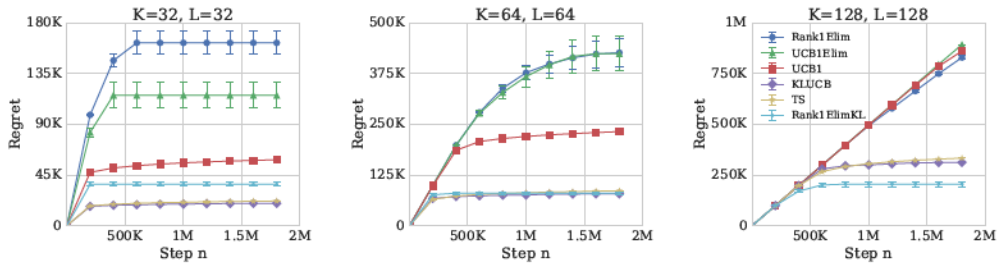

Figure 1: The n-step regret of `Rank1ElimKL`, `UCB1Elim`, `Rank1Elim` and `UCB1` on problem (8) for (a) $K = L = 32$ (b) $K = L = 64$ (c) $K = L = 128$. The results are averaged over 20 runs.

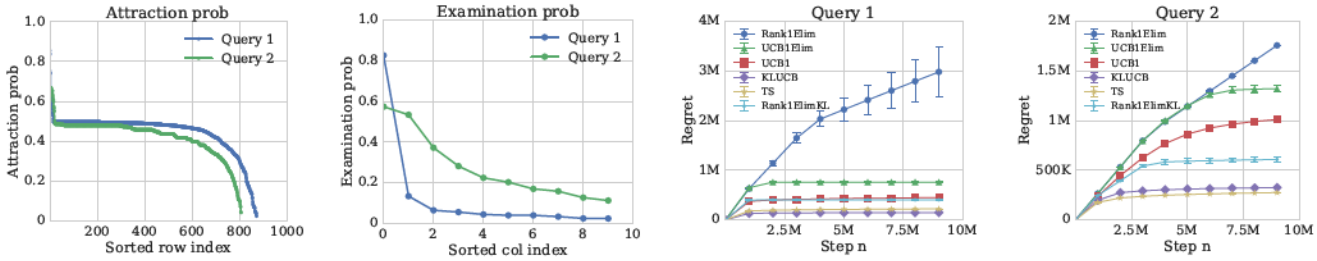

Figure 2: (a) The sorted attraction probabilities of the items from 2 queries from the Yandex dataset. (b) The sorted examination probabilities of the positions for the same 2 queries. (c) The n-step regret in Query 1. (d) The n-step regret in Query 2. The results are averaged over 5 runs.

$p_u = p_v = 0.25, \Delta_u = \Delta_v = 0.5$, and $K = L$, so that $\mu = (1-1/K)0.25+0.75/K = 0.25+0.5/K$, $1-p_{\max} = 0.25$, and $\gamma = \mu = 0.25 + 0.5/K$.

In addition to comparing to `Rank1Elim`, we also compare to `UCB1Elim` [Auer and Ortner, 2010], `UCB1` [Auer *et al.*, 2002], `KL-UCB` [Garivier and Cappe, 2011], and Thompson sampling [Thompson, 1933]. `UCB1` is chosen as a baseline as it has been used by Katariya *et al.* [2017] in their experiments. `UCB1Elim` uses an elimination approach similar to `Rank1Elim` and `Rank1ElimKL`. `KL-UCB` is similar to `UCB1`, but it uses `KL-UCB` confidence intervals. Thompson sampling (`TS`) is a Bayesian algorithm that maximizes the expected reward with respect to a randomly drawn belief.

Figure 1 shows the n-step regret of the algorithms described above as a function of time n for $K = L$, the latter of which doubles from one plot to the next. We observe that the regret of `Rank1ElimKL` flattens in all three problems, which indicates that `Rank1ElimKL` learns the optimal arm. We also see that the regret of `Rank1ElimKL` doubles as K and L double, indicating that our bound in Theorem 1 has the right scaling in $K + L$, and that the algorithm leverages the problem structure. On the other hand, the regret of `UCB1`, `UCB1Elim`, `KL-UCB` and `TS` quadruples when K and L double, confirming that their regret is $\Omega(KL)$. Next, we observe that while `KL-UCB` and `TS` have smaller regret than `Rank1ElimKL` when K and L are small, the $(K + L)$-scaling of `Rank1ElimKL` enables it to outperform these algorithms for large K and L (Figure 1c). Finally, note that `Rank1ElimKL` outperforms `Rank1Elim` in all three experiments, confirming the importance of tighter confidence intervals. It is worth noting that $\mu = \gamma$ for this problem, and hence $\mu^2 = \mu\gamma$. According to Theorem 1, `Rank1ElimKL` should not perform better than `Rank1Elim`. Yet it is 4 times better as seen in Figure 1a. This suggests that our upper bound is loose.

5.2 Models Based on Real-World Data

In this experiment, we compare `Rank1ElimKL` to other algorithms on click models that are trained on the *Yandex* dataset [Yandex, 2013], an anonymized search log of 35M search sessions. Each session contains a query, the list of displayed documents at positions 1 to 10, and the clicks on those documents. We select 20 most frequent queries from the dataset, and estimate the parameters of the PBM model using the EM algorithm [Markov, 2014; Chuklin *et al.*, 2015].

To illustrate our learned models, we plot the parameters of two queries, Queries 1 and 2. Figure 2a shows the sorted attraction probabilities of items in the queries, and Figure 2b shows the sorted examination probabilities of the positions. Query 1 has $L = 871$ items and Query 2 has $L = 807$ items. $K = 10$ is the number of documents displayed per query. We illustrate the performance on these queries because they differ notably in their μ (3) and $p_{\max}$ (4), so we can study the performance of our algorithm in different real-world settings. Figure 2c and d show the regret of all algorithms on Queries 1 and 2, respectively.

We first note that `KL-UCB` and `TS` do better than `Rank1ElimKL` on both queries. As seen in Section 5.1, `Rank1ElimKL` is expected to improve over these baselines for large K and L, which is not the case here. With respect to

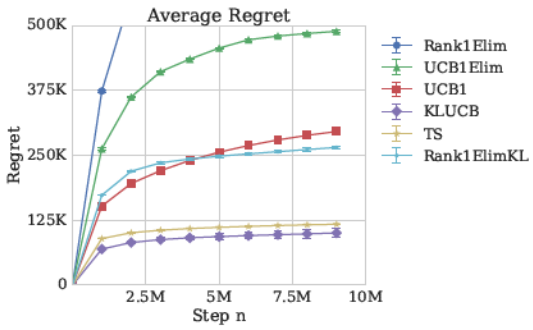

Figure 3: The average n-step regret over all 20 queries from the Yandex dataset, with 5 runs per query.

other algorithms, we see that `Rank1ElimKL` is significantly better than `Rank1Elim` and `UCB1Elim` and no worse than `UCB1` on Query 1, while for Query 2, `Rank1ElimKL` is superior to all of them. Note that $p_{\max} = 0.85$ in Query 1 is higher than $p_{\max} = 0.66$ in Query 2. Also, $\mu = 0.13$ in Query 1 is lower than $\mu = 0.28$ in Query 2. From (5), $\gamma = 0.15$ in Query 1, which is lower than $\gamma = 0.34$ in Query 2. Our upper bound (Theorem 1) on the regret of `Rank1ElimKL` scales as $\mathcal{O}((\mu\gamma)^{-1})$, and so we expect `Rank1ElimKL` to perform better on Query 2. Our results confirm this expectation.

In Figure 3, we plot the average regret over all 20 queries, where the standard error is computed by repeating this procedure 5 times. `Rank1ElimKL` has the lowest regret of all algorithms except for `KL-UCB` and `TS`. Its regret is 10.9 percent lower than that of `UCB1`, and 79 percent lower than that of `Rank1Elim`. This is expected. Some real-world instances have a benign rank-1 structure like Query 2, while others do not, like Query 1. Hence, we see a reduction in the average gains of `Rank1ElimKL` over `UCB1` in Figure 3 as compared to Figure 2d. The high regret of `Rank1Elim`, which is also designed to exploit the problem structure, shows that it is more sensitive to unfavorable rank-1 structures. Thus, the good news is that `Rank1ElimKL` improves on this limitation of `Rank1Elim`. However, `KL-UCB` and `TS` perform better on average, and we believe this is due to the fact 14 out of our 20 queries have $L < 200$, and hence $KL < 2000$. This is in line with the results of Section 5.1, which suggest that the advantage of `Rank1ElimKL` over `KL-UCB` and `TS` will "kick in" only for much larger values of K and L.

6 Related Work

Our algorithm is based on `Rank1Elim` of Katariya *et al.* [2017]. The main difference is that we replace the confidence intervals of `Rank1Elim`, which are based on subgaussian tail inequalities, with confidence intervals based on KL divergences. As discussed beforehand, this results in an unilateral improvement of their regret bound. The new algorithm is still able to exploit the problem structure of benign instances, while its regret is controlled on problem instances that are "hard" for `Rank1Elim`. As demonstrated in the previous section, the new algorithm is also a major practical improvement over `Rank1Elim`, while it remains competitive with alternatives on hard instances.

Several other papers studied bandits where the payoff is given by a low rank matrix. Zhao *et al.* [2013] proposed a bandit algorithm for low-rank matrix completion, which approximates the posterior over latent item features by a single point. The authors do not analyze this algorithm. Kawale *et al.* [2015] proposed a bandit algorithm for low-rank matrix completion using Thompson sampling with Rao-Blackwellization. They analyze a variant of their algorithm whose n-step regret for rank-1 matrices is $O((1/\Delta^2)\log n)$. This is suboptimal compared to our algorithm. Maillard *et al.* [2014] studied a bandit problem where the arms are partitioned into latent groups. In this work, we do not make any such assumptions, but our results are limited to rank 1. Gentile *et al.* [2014] proposed an algorithm that clusters users based on their preferences, under the assumption that the features of items are known. Sen *et al.* [2017] proposed an algorithm for contextual bandits with latent confounders, which reduces to a multi-armed bandit problem where the reward matrix is low-rank. They use an NMF-based approach and require that the reward matrix obeys a variant of the restricted isometry property. We make no such assumptions. Also, our learning agent controls both the row and column while in the above papers, the rows are controlled by the environment.

`Rank1ElimKL` is motivated by the structure of the PBM [Richardson *et al.*, 2007]. Lagree *et al.* [2016] proposed a bandit algorithm for this model but they assume that the examination probabilities are known. `Rank1ElimKL` can be used to solve this problem without this assumption. The cascade model [Craswell *et al.*, 2008] is an alternative way of explaining the position bias in click data [Chuklin *et al.*, 2015]. Bandit algorithms for this class of models have been proposed in several recent papers [Kveton *et al.*, 2015a; Combes *et al.*, 2015; Kveton *et al.*, 2015b; Katariya *et al.*, 2016; Zong *et al.*, 2016; Li *et al.*, 2016].

7 Conclusions

In this work, we proposed `Rank1ElimKL`, an elimination algorithm that uses `KL-UCB` confidence intervals to find the maximum entry of a stochastic rank-1 matrix with Bernoulli rewards. The algorithm is a modification of `Rank1Elim` [Katariya *et al.*, 2017], where the subgaussian confidence intervals are replaced by the ones with KL divergences. As we demonstrate both empirically and analytically, this change results in a significant improvement. As a result, we obtain the first algorithm that is able to exploit the rank-1 structure without paying a significant penalty on instances where the rank-1 structure cannot be exploited.

We note that `Rank1ElimKL` uses the rank-1 structure of the problem and that there are no guarantees beyond rank-1. While the dependence of the regret of `Rank1ElimKL` on Δ is known to be tight [Katariya *et al.*, 2017], the question about the optimal dependence on μ is still open. Finally, we point out that `TS` and `KL-UCB` perform better than `Rank1ElimKL` in our experiments, especially for small L and K. This is because `Rank1ElimKL` is an elimination algorithm. Elimination algorithms tend to have higher regret initially than UCB-style algorithms because they explore more aggressively. It is not inconceivable to have `TS` algorithms that leverage the rank-1 structure in the future.

References

[Auer and Ortner, 2010] Peter Auer and Ronald Ortner. UCB revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica*, 61(1-2):55–65, 2010.

[Auer *et al.*, 2002] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

[Chuklin *et al.*, 2015] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. *Click Models for Web Search*. Morgan & Claypool Publishers, 2015.

[Combes *et al.*, 2015] Richard Combes, Stefan Magureanu, Alexandre Proutiere, and Cyrille Laroche. 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*, 2015.

[Craswell *et al.*, 2008] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. An experimental comparison of click position-bias models. In *Proceedings of the 1st ACM International Conference on Web Search and Data Mining*, pages 87–94, 2008.

[Garivier and Cappe, 2011] Aurelien Garivier and Olivier Cappe. The KL-UCB algorithm for bounded stochastic bandits and beyond. In *Proceeding of the 24th Annual Conference on Learning Theory*, pages 359–376, 2011.

[Gentile *et al.*, 2014] Claudio Gentile, Shuai Li, and Giovanni Zappella. Online clustering of bandits. In *Proceedings of the 31st International Conference on Machine Learning*, pages 757–765, 2014.

[Katariya *et al.*, 2016] Sumeet Katariya, Branislav Kveton, Csaba Szepesvari, and Zheng Wen. DCM bandits: Learning to rank with multiple clicks. In *Proceedings of the 33rd International Conference on Machine Learning*, 2016.

[Katariya *et al.*, 2017] Sumeet Katariya, Branislav Kveton, Csaba Szepesvari, Claire Vernade, and Zheng Wen. Stochastic rank-1 bandits. In *AISTATS*, 2017.

[Kawale *et al.*, 2015] Jaya Kawale, Hung Bui, Branislav Kveton, Long Tran-Thanh, and Sanjay Chawla. Efficient Thompson sampling for online matrix-factorization recommendation. In *Advances in Neural Information Processing Systems 28*, pages 1297–1305, 2015.

[Kveton *et al.*, 2015a] Branislav Kveton, Csaba Szepesvari, Zheng Wen, and Azin Ashkan. Cascading bandits: Learning to rank in the cascade model. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[Kveton *et al.*, 2015b] Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvari. Combinatorial cascading bandits. In *Advances in Neural Information Processing Systems 28*, pages 1450–1458, 2015.

[Lagree *et al.*, 2016] Paul Lagree, Claire Vernade, and Olivier Cappe. Multiple-play bandits in the position-based model. *CoRR*, abs/1606.02448, 2016.

[Li *et al.*, 2016] Shuai Li, Baoxiang Wang, Shengyu Zhang, and Wei Chen. Contextual combinatorial cascading bandits. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 1245–1253, 2016.

[Maillard and Mannor, 2014] Odalric-Ambrym Maillard and Shie Mannor. Latent bandits. In *Proceedings of the 31st International Conference on Machine Learning*, pages 136–144, 2014.

[Markov, 2014] Ilya Markov. Pyclick - click models for web search. https://github.com/markovi/PyClick, 2014.

[Richardson *et al.*, 2007] Matthew Richardson, Ewa Dominowska, and Robert Ragno. Predicting clicks: Estimating the click-through rate for new ads. In *Proceedings of the 16th International Conference on World Wide Web*, pages 521–530, 2007.

[Sen *et al.*, 2017] Rajat Sen, Karthikeyan Shanmugam, Murat Kocaoglu, Alexandros G Dimakis, and Sanjay Shakkottai. Contextual bandits with latent confounders: An nmf approach. In *AISTATS*, 2017.

[Thompson, 1933] William. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

[Yandex, 2013] Yandex personalized web search challenge. https://www.kaggle.com/c/yandex-personalized-web-search-challenge, 2013.

[Zhao *et al.*, 2013] Xiaoxue Zhao, Weinan Zhang, and Jun Wang. Interactive collaborative filtering. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management*, pages 1411–1420, 2013.

[Zong *et al.*, 2016] Shi Zong, Hao Ni, Kenny Sung, Nan Rosemary Ke, Zheng Wen, and Branislav Kveton. Cascading bandits for large-scale recommendation problems. In *Proceedings of the 32nd Conference on Uncertainty in Artificial Intelligence*, 2016.