Learning optimally diverse rankings over large document collections

Aleksandrs Slivkins SLIVKINS@MICROSOFT.COM
Microsoft Research, 1065 La Avenida, Mountain View CA, USA

Filip Radlinski FILIPRAD@MICROSOFT.COM
Microsoft Research, 7 J.J. Thomson Ave., Cambridge UK

Sreenivas Gollapudi SREENIG@MICROSOFT.COM
Microsoft Research, 1065 La Avenida, Mountain View CA, USA

Abstract

Most learning to rank research has assumed that the utility of different documents is independent, which results in learned ranking functions that return redundant results. The few approaches that avoid this have rather unsatisfyingly lacked theoretical foundations, or do not scale. We present a learning-to-rank formulation that optimizes the fraction of satisfied users, with a scalable algorithm that explicitly takes document similarity and ranking context into account. We present theoretical justifications for this approach, as well as a near-optimal algorithm. Our evaluation adds optimizations that improve empirical performance, and shows that our algorithms learn orders of magnitude more quickly than previous approaches.

1. Introduction

Identifying the most relevant results to a query is a central problem in web search, hence learning ranking functions has received a lot of attention (e.g., Burges et al., 2005; Chu and Ghahramani, 2005; Taylor et al., 2008). One increasingly important goal is to learn from user interactions with search engines, such as clicks. We address the task of learning a ranking function to minimizes the likelihood of query abandonment (i.e. no click). This objective is particularly interesting as query abandonment is a major challenge in today's search engines, and is also sensitive to the diversity and redundancy among documents presented.

We consider the Multi-Armed Bandit (MAB) setting (e.g. Cesa-Bianchi and Lugosi, 2006), which captures many online learning problems wherein an algorithm chooses sequentially among a fixed set of alternatives ("arms" or "strategies"). MAB algorithms are ideal for online settings with exploration/exploitation tradeoffs. While most MAB literature corresponds to learning a single best alternative (*single-slot* MAB), MAB algorithms can also be extended to *multiple slots*, e.g. to learn a ranking of documents that minimizes query abandonment (Radlinski et al., 2008; Streeter and Golovin, 2009). However, most MAB algorithms are impractical at web scales.

Prior work on MAB algorithms has considered exploiting structure in the strategy space to improve convergence rates. One particular approach, articulated by Kleinberg et al. (2008) is well suited to our scenario: when the strategies (in our case, documents) form a metric space and the payoff function satisfies a Lipschitz condition with respect to the metric. The metric space allows the algorithm to make inferences about similar documents without exploring them. Further, they propose a "zooming algorithm" that learns to adaptively refine explored regions of the strategy space where there is likelihood of higher payoff, and provide strong provable guarantees about its performance.

In web search, a metric space directly models similarity between documents.[1] Further, one can use additional signals. A search user typically scans results top down, and clicks on more relevant documents. One can therefore infer the *context* in which a click happened: the skipped documents at higher ranks. To fully exploit the context we factor in both *conditional clickthrough rates* and *correlated clicks*. The former conditions on the event that the user skipped a set of documents (as suggested by Chen and Karger, 2006), and the latter refers to the probability that two documents are both relevant (or irrelevant) to a given user.

Our contributions. This paper initiates the study of online learning-to-rank in metric spaces. We propose a

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

[1] In fact, most offline learning-to-rank approaches also rely on similarity between documents, at least implicitly.

simple learning model that explicitly considers correlation of clicks and similarity between documents, and admits efficient bandit algorithms that, unlike those in prior work on bandit-based learning-to-rank, scale to large document collections. We study this model both theoretically and empirically. First, we validate the expressiveness of our model by providing an explicit construction for a wide family of plausible user distributions which fit the model. Second, we design several algorithms for our model, joining and extending ideas from "ranked bandits" (Radlinski et al., 2008), bandits on metric spaces (Kleinberg et al., 2008) and contextual bandits (Slivkins, 2009).[2] Third, we provide provable scalability guarantees. Finally, we empirically study their performance using the above-mentioned construction with realistic parameters.

In a more abstract view, we tackle the problem of using side information on document similarity in the online learning-to-rank setting. We focus on the case of "ideally clean" similarity data, with a two-pronged high-level question: how to model such data, and how to use it algorithmically. We believe that studying the "clean" case is useful (and perhaps necessary) to inform and guide the corresponding data-driven work.

Outline. We define the model in Section 2, validate its expressiveness in Section 3, design algorithms in Section 4, prove scalability guarantees in Section 5, and discuss simulations in Section 6. It is worth noting that much of the theoretical (provable) contribution concerns setting up the model in Sections 2,

the higher slots. x has a conditional click probability $\mu(x|Z_S)$. The function $\mu(\cdot\,|Z_S)$ satisfies the Lipschitz condition (1), which will allow us to use the machinery from MAB problems on metric spaces.

Document model. Web documents are often classified into hierarchies, where closer pairs are more similar.[6] For evaluation, we assume the documents X fall in such a tree, with each document $x \in X$ a leaf in the tree. On this tree, we consider a very natural metric: the ϵ-*exponential tree metric*: the distance between any two nodes is exponential in the depth of their least common ancestor, with base $\epsilon \in (0,1)$. However, our algorithms and analyses extend to arbitrary metric spaces; see full version for further discussion.

Observation. An alternative notion of document similarity focuses on correlation between clicks. Call $\mathcal{P}$ *conditionally L-correlated* w.r.t. $(X,\mathcal{D})$ if

$$\Pr_{\pi\sim(\mathcal{P}|Z_S)}[\pi(x)\neq\pi(y)] \le \mathcal{D}(x,y) \quad \forall x,y\in X, S\subset X.$$

It is easy to see that conditional L-correlation implies conditional L-continuity. Moreover, in the full version we show that conditional L-continuity w.r.t. $(X,\mathcal{D})$ implies conditional L-correlation w.r.t. $(X,2\mathcal{D})$. Thus, the two notions are essentially equivalent.

3. Model Expressiveness

Our approach relies on the conditional L-continuity of the user distribution, which is a non-trivial property about correlated clicks. We now argue that this property is plausible in a realistic setting, and provide a family of user distributions to be used in experiments in Section 6. We accomplish both by defining a natural (albeit highly stylized) generative model for the user distribution. Given a tree metric space $(X,\mathcal{D})$ and the desired pointwise mean μ, this model provides a rich family of user distributions that are conditionally L-continuous w.r.t. $(X, c\,\mathcal{D})$, for some small c.

The generative model is a tree-shaped Bayesian network in which leaves correspond to the click events on documents. The tree is essentially a topical taxonomy on documents such that the click event on each sub-topic is obtained from that on the parent topic via a low-probability mutation. It is fairly easy to see that the mutation probabilities need to be bounded in terms of the distance between the child and the parent, and derive a necessary and sufficient condition to obtain a given pointwise mean μ. The hard part is to prove that the low-probability mutations are *sufficient* to guarantee conditional L-correlation.

Assume documents are leaves of a finite rooted edge-weighted tree with node set V and leaf set $X \subset V$. Let $\mathcal{D}$ be the (weighted) shortest-paths metric on V.

[6] E.g., the Open Directory Project http://dmoz.org/

Algorithm 1 User distribution for tree metrics

Input: Tree (root r, node set V); $\mu(r)\in[0,1]$
mutation probabilities $q_0, q_1 : V \to [0,1]$
Output: random click vector $\pi : V \to \{0,1\}$

function AssignClicks(tree node v)
  $b \leftarrow \pi(v)$
  for each child u of v **do**
    $\pi(u) \leftarrow \begin{cases} 1-b & \text{w/prob } q_b(u) \\ b & \text{otherwise} \end{cases}$
    AssignClicks(u)

Pick $\pi(r)\in\{0,1\}$ at random with expectation $\mu(r)$
AssignClicks(r)

The high-level construction is very intuitive. We start with a function $\mu : X \to [\alpha, \frac{1}{2}]$, $\alpha > 0$, that is L-continuous w.r.t. $(X,\mathcal{D})$. We can show (proof omitted) that μ can be extended from X to V so that $\mu : V \to [\alpha, \frac{1}{2}]$ is L-continuous w.r.t. $(V,\mathcal{D})$. We pick $\pi(\text{root}) \in \{0,1\}$ at random with a suitable expectation, and then proceed top-down so that the child's click is obtained from the parent's click via a low-probability mutation. The mutation is parameterized by functions $q_0, q_1 : V \to [0,1]$, as described in Algorithm 1. These parameters let us vary the degree of independence between each child and its parent, resulting in a rich family of user distributions.

To ensure that $\mathbb{E}[\pi(v)] = \mu(v)$ for all $v \in V$, we posit

$$\mu(u) = (1-\mu(v))\,q_0(u) + \mu(v)(1-q_1(u)) \qquad (2)$$

whenever u is a child of v. Further, we assume that

$$q_0(u) + q_1(u) \le \mathcal{D}(u,v)/\min(\mu(u),\,\mu(v)). \qquad (3)$$

For a concrete example, one could define

$$(q_0(u),\,q_1(u)) = \begin{cases} \left(0,\,\frac{\mu(v)-\mu(u)}{\mu(v)}\right) & \text{if } \mu(v)\ge\mu(u) \\ \left(\frac{\mu(u)-\mu(v)}{1-\mu(v)},\,0\right) & \text{otherwise.} \end{cases} \qquad (4)$$

We show that the user distribution π constructed by Algorithm 1 has pointwise mean μ, it is L-correlated w.r.t. $\mathcal{D}_\mu(x,y) \triangleq \mathcal{D}(x,y)\min\left(\frac{1}{\alpha}, \frac{3}{\mu(x)+\mu(y)}\right)$, and conditionally L-correlated w.r.t. $\mathcal{D}_\mu(x,y)\,\frac{2}{1-\mathcal{D}_\mu(x,y)}$.

The proof of this result, omitted due to space constraints, is a key theoretical contribution of this work, and by far the most technical one. It can be found in the full version of this paper. One could strengthen this result by replacing $\mathcal{D}_\mu$ with the shortest-paths metric induced by $\mathcal{D}_\mu$. Further, the result can be extended to arbitrary metric spaces.

4. Algorithms

In this section we present algorithms for the problem as defined in Section 2. We start by describing prior work and simple adaptations of existing algorithms, following with a presentation of two new algorithms that explicitly take context into account.

In what follows, the "metric-aware" algorithms are well-defined for arbitrary metric spaces, but for simplicity we present them for a special case: documents are leaves in a *document tree* τ_{d} with an ϵ-exponential tree metric. In all these algorithms, a *subtree* is chosen in each round. Then a document in this subtree is sampled at random, choosing uniformly at each branch.

Ranked Bandits. Letting `Bandit` be some algorithm for the MAB problem, the "ranked" bandit algorithm `RankBandit` for the multi-slot MAB problem is defined as follows (Radlinski et al., 2008). We have k slots (i.e., ranks) for which we wish to find the best documents to present. In each slot i, a separate copy $\mathcal{A}_i$ of `Bandit` is instantiated. In each round, if a user clicks on slot i, then this slot receives a payoff of 1, and all higher (i.e., skipped) slots $j < i$ receive a payoff of 0. For slots $j > i$, the state is rolled back as if this round had never happened (as if the user never considered these documents). If no slot is clicked, then all slots receive a payoff of 0.

In (Radlinski et al., 2008), this approach gives rise to algorithms `RankUCB1` and `RankEXP3`, based on MAB algorithms `UCB1` and `EXP3` (Auer et al., 2002a, 2002b). `EXP3` is designed for the *adversarial* setting with no assumptions on how the clicks are generated, which translates into concrete provable guarantees for `RankEXP3`. `UCB1` is geared towards the *stochastic* setting with i.i.d. payoffs on each arm, although the per-slot i.i.d. assumption breaks for slots $i > 1$ because of the influence of the higher slots. Nevertheless, in small-scale experiments `RankUCB1` performs much better than `RankEXP3` (Radlinski et al., 2008).

In `UCB1`-style algorithms, including the zooming algorithm, one can damp exploration by replacing the $4\log(T)$ factor in (5) with 1. Such change effectively makes the algorithm more *optimistic*; it was found beneficial for `RankUCB1` by Radlinski et al. (2008). We will denote this version by appending '+' to the algorithm's name, e.g. `RankUCB1+`. We will see that it can also greatly improve average performance here.

Using the metric space. Both above algorithms are impractical when there are too many documents to explore them all. To avoid this challenge, we can exploit the similarity information provided by the metric space in our setting. Since the payoff function $\mu(\cdot)$ is an L-continuous function on $(X, \mathcal{D})$, we can use the (single-slot) bandit algorithms that are designed for the Lipschitz MAB problem to improve the speed of convergence of the RankedBandit approach.

Algorithm 2 "Zooming algorithm" in trees

initialize (document tree τ_{d})**:**
$\mathcal{A} \leftarrow \emptyset$; activate($\texttt{root}(\tau_{\mathrm{d}})$)

activate($u \in \texttt{nodes}(\tau_{\mathrm{d}})$)**:**
$\mathcal{A} \leftarrow \mathcal{A} \cup \{u\}$; $n(u) \leftarrow 0$; $r(u) \leftarrow 0$

Main loop:
$u \leftarrow \operatorname{argmax}_{u \in \mathcal{A}} \frac{r(u)}{n(u)} + 2\,\texttt{rad}(u)$
"Play" a random document from $\texttt{subtree}(u)$
$r(u) \leftarrow r(u) + \{\text{reward}\}$; $n(u) \leftarrow n(u) + 1$
if $\texttt{rad}(u) < \texttt{W}(u)$ **then**
deactivate u: remove u from $\mathcal{A}$
activate all children of u

The meta-algorithm `GridBandit` (Kleinberg, 2004) is one such algorithm. It proceeds in phases: In phase i, the depth-i subtrees are treated as "arms", and a fresh copy of `Bandit` is run on these arms.[7] Phase i lasts for $k\epsilon^{-2i}$ rounds, where k is the number of depth-i subtrees. This meta-algorithm (coupled with an adversarial MAB algorithm such as `EXP3`) is the only prior algorithm that takes advantage of the metric space in the adversarial setting. Following (Radlinski et al., 2008), we expect `GridEXP3` to be overly pessimistic for our problem, trumped by the corresponding stochastic MAB approaches such as `GridUCB1`.

The "zooming algorithm" (Kleinberg et al., 2008, Algorithm 2) is a more efficient version of `GridUCB1`: instead of iteratively reducing the grid size in the entire metric space, it selectively refines the grid in promising areas. It maintains a set $\mathcal{A}$ of *active subtrees* which collectively partition the leaf set. In each round the active subtree with the maximal *index* is chosen. The index of a subtree is (assuming stochastic payoffs) the best available upper confidence bound on the click probabilities in this subtree. It is defined via the *confidence radius*[8] given (letting T be the time horizon) by

$$\texttt{rad}(\cdot) \triangleq \sqrt{4\log(T)/(1+\#\texttt{samples}(\cdot))}. \qquad (5)$$

The algorithm "zooms in" on a given active subtree u (de-activates u and activates all its children) when $\texttt{rad}(u)$ becomes smaller than its *width* $\texttt{W}(u) \triangleq \epsilon^{\text{depth}(u)} = \max_{x,x' \in u} \mathcal{D}(x, x')$. The "ranked zooming algorithm" will be denoted `RankZoom`.

Contextual bandits. Our subsequent algorithms leverage prior work on *contextual MAB*. The relevant

[7] As an empirical optimization, previous events can also be replayed to better initialize later phases.

[8] The meaning of $\texttt{rad}(\cdot)$ is that the sample average is within $\pm\texttt{rad}(\cdot)$ from the true mean with high probability.

Algorithm 3 `ContextZoom` in trees

initialize (document tree τ_{d}, context tree τ_{c}):
$\mathcal{A} \leftarrow \emptyset$; activate($\texttt{root}(\tau_{\mathrm{d}})$, $\texttt{root}(\tau_{\mathrm{c}})$)

activate ($u \in \texttt{nodes}(\tau_{\mathrm{d}})$, $u_{\mathrm{c}} \in \texttt{nodes}(\tau_{\mathrm{c}})$):
$\mathcal{A} \leftarrow \mathcal{A} \cup \{(u, u_{\mathrm{c}})\}$; $n(u, u_{\mathrm{c}}) \leftarrow 0$; $r(u, u_{\mathrm{c}}) \leftarrow 0$

Main loop:
Input a context $h \in \texttt{nodes}(\tau_{\mathrm{c}})$
$(u, u_{\mathrm{c}}) \leftarrow \operatorname{argmax}_{(u,u_{\mathrm{c}}) \in \mathcal{A}:\, h \in u_{\mathrm{c}}} \texttt{W}(u \times u_{\mathrm{c}}) + \frac{r(u,u_{\mathrm{c}})}{n(u,u_{\mathrm{c}})} + \texttt{rad}(u, u_{\mathrm{c}})$
"Play" a random document from $\texttt{subtree}(u)$
$r(u,u_{\mathrm{c}}) \leftarrow r(u,u_{\mathrm{c}}) + \{\text{reward}\}$; $n(u,u_{\mathrm{c}}) \leftarrow n(u,u_{\mathrm{c}}) + 1$
if $\texttt{rad}(u, u_{\mathrm{c}}) < \texttt{W}(u, u_{\mathrm{c}})$ **then**
deactivate (u, u_{c}): remove (u, u_{c}) from $\mathcal{A}$
activate all pairs (child(u), child(u_{c}))

contextual MAB setting is as follows: in each round nature reveals a *context* h, an algorithm chooses a document x, and the resulting payoff is an independent $\{0,1\}$ sample with expectation $\mu(x|h)$. Further, one is given *similarity information*: metrics $\mathcal{D}$ and $\mathcal{D}_{\mathrm{c}}$ on documents and contexts, resp., such that for any two documents x, x' and any two contexts h, h' we have

$$|\mu(x|h) - \mu(x'|h')| \leq \mathcal{D}(x, x') + \mathcal{D}_{\mathrm{c}}(h, h'). \quad (6)$$

We will use the "contextual zooming algorithm" (`ContextZoom`) from (Slivkins, 2009). This algorithm is well-defined for arbitrary metrics $\mathcal{D}, \mathcal{D}_{\mathrm{c}}$, but for simplicity we will state it for ϵ-exponential tree metrics.

Let us assume that documents and contexts are leaves in a document tree τ_{d} and context tree τ_{c}, resp. The algorithm (see Algorithm 3 for pseudocode) maintains a set $\mathcal{A}$ of *active strategies* of the form (u, u_{c}), where u is a subtree in τ_{d} and u_{c} is a subtree in τ_{c}. At any given time the active strategies partition the space of all (document, context) pairs, henceforth the *DC-space*. In each round, a context h arrives, and one of the active strategies (u, u_{c}) with $h \in u_{\mathrm{c}}$ is chosen: namely the one with the maximal *index*, and then a document $x \in u$ is picked uniformly at random. The index of (u, u_{c}) is, essentially, the best available upper confidence bound on expected payoffs from choosing a document $x \in u$ given a context $h \in u_{\mathrm{c}}$: as per (Slivkins, 2009), with high probability it holds that

$$\texttt{index}(u, u_{\mathrm{c}}) \geq \mu(x|h), \quad \forall x \in u,\ h \in u_{\mathrm{c}}. \quad (7)$$

The index is defined via sample average, confidence radius (5), and "width" $\texttt{W}(u \times u_{\mathrm{c}})$, which is an upper bound on the diameter in the DC-space:

$$\texttt{W}(u, u_{\mathrm{c}}) \geq \max_{x,x' \in u,\ h,h' \in u_{\mathrm{c}}} \mathcal{D}(x, x') + \mathcal{D}_{\mathrm{c}}(h, h'). \quad (8)$$

The (de)activation rule ensures that the active strategies form a finer partition in the regions of the DC-space that correspond to higher payoffs and more frequently occurring contexts.

New approach: ranked contextual algorithms. We now present a new approach in which the upper slot selections are taken into account as a *context*.

The slot algorithms in the `RankBandit` setting can make their selections sequentially. Then w.l.o.g. each slot algorithm $\mathcal{A}_i$ knows the set S of documents in the upper slots. We propose to treat S as a "context" to $\mathcal{A}_i$. Specifically, $\mathcal{A}_i$ will assume that none of the documents in S is clicked, i.e. event Z_S happens (else the i-th slot is ignored by the user). For each such round, the click probabilities for $\mathcal{A}_i$ are given by $\mu(\cdot\,|Z_S)$, which is an L-continuous function on $(X, \mathcal{D})$.

Let us specify a suitable metric $\mathcal{D}_{\mathrm{c}}$ on contexts $S \subset X$ which (as we show in the full version) satisfies (6):

$$\mathcal{D}_{\mathrm{c}}(S, S') \triangleq 4 \inf \textstyle\sum_{j=1}^{n} \mathcal{D}(x_j, x'_j), \quad (9)$$

where the infimum is taken over all $n \in \mathbb{N}$ and over all n-element sequences $\{x_j\}$ and $\{x'_j\}$ that enumerate, possibly with repetitions, all documents in S and S'.

Having defined $\mathcal{D}_{\mathrm{c}}$, we can use any contextual MAB algorithm for $\mathcal{A}_i$. We will use `ContextZoom`.

Let us supply the missing details as they apply to the $(i+1)$-th slot. (For slot 1, `ContextZoom` reduces to Algorithm 2.) The contexts are unordered i-tuples of documents. Given a document tree τ_{d}, let us define *context tree* τ_{c} as follows. Depth-l nodes of τ_{c} are unordered i-tuples of depth-l nodes from τ_{d}, and leaves are contexts. The root of τ_{c} is $(r \dots r)$, where $r = \texttt{root}(\tau_{\mathrm{d}})$. For each internal node $u_{\mathrm{c}} = (u_1 \dots u_i)$ of τ_{c}, its children are all unordered tuples $(v_1 \dots v_i)$ such that each v_j is a child of u_j in τ_{d}. This completes the definition of τ_{c}. Letting u and u_{c} be level-l subtrees of τ_{d} and τ_{c}, resp., it follows from (9) that $\mathcal{D}_{\mathrm{c}}(S, S') \leq 4i\,\epsilon^l$ for any contexts $S, S' \in u_{\mathrm{c}}$. Thus setting $\texttt{W}(u \times u_{\mathrm{c}}) \triangleq \epsilon^l(4i+1)$ satisfies (8).

We will use `ContextZoom` (with τ_{c} and $\texttt{W}(u, u_{\mathrm{c}})$ as above) for slots $i \geq 2$; for slot 1, contexts are empty, so `ContextZoom` reduces to Algorithm 2. The resulting "ranked" algorithm is called `RankContextZoom`.

We can further exploit the conditional L-continuity. In the analysis of `ContextZoom`, the index can be decreased, improving performance, as long as (7) holds. Fix context $S \subset X$. Since $\mu(y|Z_S) = 0$ for any $y \in S$,

$$\mu(x|Z_S) = |\mu(x|Z_S) - \mu(y|Z_S)| \leq \mathcal{D}(x, y), \quad \forall y \in S$$
$$\text{hence} \quad \mu(x|Z_S) \leq \mathcal{D}(x, S) \triangleq \min_{y \in S} \mathcal{D}(x, y). \quad (10)$$

The intuition is that nearby documents being non-relevant limits how good a given document can be.

Using (10) we can decrease the index of many strategies while keeping it a valid upper confidence bound on payoffs. Thus, we "upgrade" `RankContextZoom` with the following *correlation rule*: for each active strategy (u, u_c), where u is a subtree of documents, and u_c is a set of contexts, cap the index of each (u, u_c) at $\mathcal{D}(u, S)$.

We also define `RankCorrZoom`, a version of `RankZoom` which uses a similar "correlation rule": for each slot, cap the index of each active subtree u at $\mathcal{D}(u, S)$. We view it as a light-weight version of `RankContextZoom`.

5. Provable scalability guarantees

Here we summarize the relevant provable guarantees from prior work, and apply them to our multi-slot algorithms. The purpose is two-fold: to provide intuition behind these algorithms, and to prove their scalability. We also provide an improved convergence result.

Single-slot bandits. Provable guarantees for single-slot MAB algorithms are usually expressed via *regret* w.r.t. a benchmark: the best arm in hindsight. Regret $R(T)$ of an algorithm is the expected payoff of the benchmark in T rounds minus that of the algorithm.

For MAB with n arms, `EXP3` (Auer et al., 2002b) achieves regret $R(T) = \tilde{O}(\sqrt{nT})$ against an oblivious (non-adaptive) adversary. In the stochastic setting, `UCB1` (Auer et al., 2002a) performs much better, with *logarithmic* regret:[9] letting $\Delta(x) = \max \mu(\cdot) - \mu(x)$,

$$R(T) = \min_{r>0} \left(rT + \sum_{x \in X:\, \Delta(x)>r} \frac{O(\log T)}{\Delta(x)} \right). \qquad (11)$$

For the Lipschitz MAB problem (Kleinberg, 2004; Kleinberg et al., 2008), regret guarantees are independent of the number of arms. Against an oblivious adversary, `GridEXP3` has regret

$$R(T) = \tilde{O}(T^{(d+1)/(d+2)}), \qquad (12)$$

where d is the covering dimension of the metric space. For the stochastic setting, `GridUCB1` has the same regret guarantee, whereas the zooming algorithm has regret (12) w.r.t. a different, smaller d ("zooming dimension") which is much smaller (e.g., $d = 0$) for "benign" problem instances, see (Kleinberg et al., 2008).[10]

For the contextual MAB problem, regret is w.r.t. a much stronger benchmark: the best arm in hindsight *for every given context*. `ContextZoom` has regret (12) with d equal to the "contextual zooming dimension" of the problem instance, which for the i-th slot in our setting is at most $i \times$ CoveringDim, but is much smaller with "benign" context arrivals and "benign" click probabilities, see (Slivkins, 2009) for details.

Multi-slot bandits. Letting T be the time horizon and `OPT` be the probability of clicking on the optimal ranking, algorithm `RankBandit` achieves

$$\mathbb{E}[\#\texttt{clicks}] \geq (1 - \tfrac{1}{e})\, T \times \texttt{OPT} - k\, R(T), \qquad (13)$$

where $R(T)$ is any upper bound on regret for `Bandit` in each slot (Radlinski et al., 2008).

In the multi-slot setting, *performance* of an algorithm is defined as the time-averaged expected number of clicks. If $R(T) = o(T)$, performance of `RankBandit` converges with time to $(1-\frac{1}{e})$`OPT` (or exceeds it), which is proved worst-case optimal. Thus, as long as $R(T)$ scales well with #documents (e.g., as in (12)), Radlinski et al. (2008) interpret (13) as a proof of an algorithm's scalability in the multi-slot MAB setting.

`RankBandit` is an online version of the *greedy algorithm*: an offline fully informed algorithm that selects documents greedily slot by slot from top to bottom. The performance of this algorithm is called the *greedy optimum*,[11] which is equal $(1-\frac{1}{e})$ `OPT` in the worst case but can be as good as `OPT`. Thus, greedy optimum is a natural benchmark for `RankBandit`. Surprisingly, results w.r.t. this benchmark are absent.

Our results. We show that `RankGridEXP3` and `RankContextZoom` scale to large document collections, in the sense that they achieve (13) with $R(T)$ that does not degenerate with #documents. (We simply note that their regret bounds plug into (13).)

Further, we obtain an improved convergence result: the performance of `RankContextZoom` converges with time to (or exceeds) the greedy optimum.[12]

Discussion. It is not clear whether, and under which assumptions, this convergence result can be extended to the "ranked" versions of non-contextual bandit algorithms such as `RankUCB1`. One assumption that appears essential is the uniqueness of the "greedy ranking". Indeed, if the pointwise mean $\mu(\cdot)$ has two peaks with equal value, then a "reasonable" slot 1 algorithm will alternate between (the vicinities of) these peaks, thus distorting the statistics for the slot 2 algorithm and causing it to converge on a suboptimal document; see the full version for a specific example.

Discussion. We believe that the above guarantees do

[9] Regret in (11) is logarithmic for every fixed μ. For a worst-case μ it is $\tilde{\Theta}(\sqrt{nT})$, matching `EXP3`.

[10] For both algorithms there is a "better" μ-specific guarantee in the style of (11), of which (12) is the worst case.

[11] If due to ties there are multiple "greedy rankings", define the greedy optimum via the *worst* of them.

[12] Proof outline: Since the number of arms is finite, by induction on k we can prove that for a large enough time T, in all but $o(T)$ rounds we have a greedy ranking.

not reflect the full power of our algorithms and our setting. First, we conjecture that (13) can be strengthened to use the greedy optimum as a benchmark, and (in some sense) be extended to "ranked" versions of non-contextual stochastic MAB algorithms. Second, the guarantees for `RankContextZoom` seem unsatisfying, as they only use a weak form of the regret bound for `ContextZoom`.[13] Finally, one should be able to express and prove the empirically observed performance gains of the "correlation rule".

6. Evaluation

Let us evaluate the performance of the algorithms presented in Section 4: "metric-oblivious" `RankUCB1` and `RankEXP3`, "metric-aware" non-contextual `RankGridUCB1`, `RankGridEXP3` and `RankZoom`, and contextual `RankContextZoom` and `RankCorrZoom`.

Experimental setup. Using the generative model from Section 3 (Algorithm 1 with (4)), we created a document collection of size $|X| = 2^{15} \approx 32,000$ in a binary ϵ-exponential tree metric space with $\epsilon = 0.837$. This is a realistic number of documents that may be considered in detail for a typical web search query after pruning very unlikely documents. The value for ϵ was chosen so that the most dissimilar documents in the collection still have a non-trivial similarity, as may be expected for web documents. Each document's expected relevance $\mu(x)$ was set by first identifying a small number of "peaks" $y_i \in X$, choosing $\mu(\cdot)$ for these documents, and then defining the relevance of other documents as the minimum allowed while obeying L-continuity and a background relevance rate μ_0:

$$\mu(x) \triangleq \max(\mu_0,\ \tfrac{1}{2} - \min_i \mathcal{D}(x, y_i)).$$

For internal nodes in the tree, μ is defined bottom-up (from leaves to the root) as the mean value of all children nodes.

Our simulation was run over a 5-slot ranked bandit setting, learning the best 5 documents. We considered 300,000 user visits sampled from $\mathcal{P}$ per Algorithm 1. Performance within 50,000 impressions, typical for the number of times relatively frequent queries are seen by commercial search engines in a month, is essential for any practical applicability of this approach, although a longer evaluation allows for a deeper understanding of the convergence properties of the algorithms.

We consider two models for $\mu(\cdot)$. In the first model, two "peaks" $\{y_1, y_2\}$ are selected at random with $\mu(\cdot) = \frac{1}{2}$, and μ_0 set to 0.05. The second model is less "rigid" (and thus more realistic): the relevant documents y_i and their expected relevance rates $\mu(\cdot)$ are selected according to a Chinese Restaurant Process with parameters $n = 20$ and $\theta = 2$, see (Radlinski et al., 2008) for details, and with $\mu_0 = 0.01$.

[13] Namely, we ignore the effect that for a given slot, contexts $S \subset X$ may gradually converge and become "low-dimensional"; see the full version for more discussion.

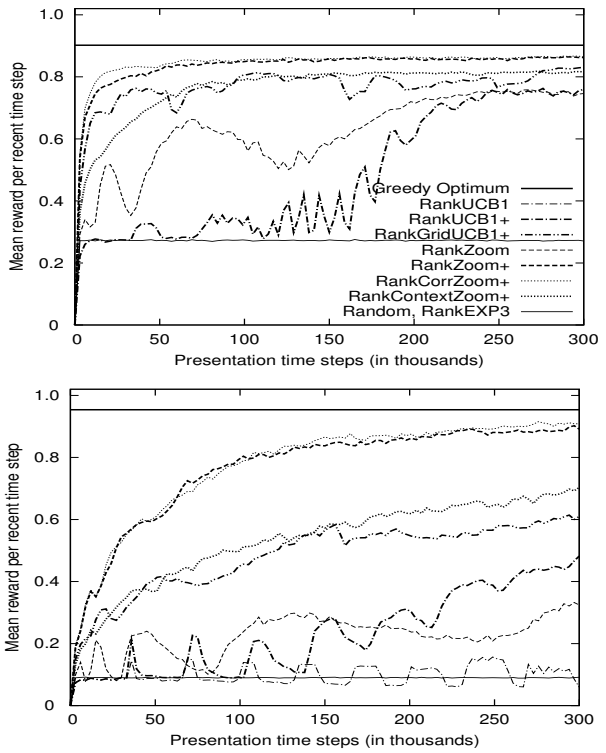

Figure 1. Various learning algorithms on 5-slot problem instances with two relevance peaks (above) or random (Chinese Restaurant Process) relevance (below).

As baselines we use selecting documents at random and the offline greedy ranking (see Section 5).

Experimental results (see Figure 1).

`RankEXP3` and `RankUCB1` perform as poorly as picking documents randomly: the three curves are indistinguishable. This is due to the large number of available documents and slow convergence rates of these algorithms. It is consistent with results reported by (Radlinski et al., 2008) on just 50 documents.

Making the `UCB1`-style algorithms "optimistic" ("+", see Section 4) improved performance dramatically. In particular, `GridUCB1+` performed best of the non-contextual algorithms. `RankZoom` performs comparably to `RankUCB1+`, and becomes extremely effective if made optimistic. For the two contextual algorithms, we saw a similar increase in performance, hence we only show the performance of the optimistic versions.

`RankCorrZoom+` achieves the best empirical performance, converging rapidly to near-optimal rankings.

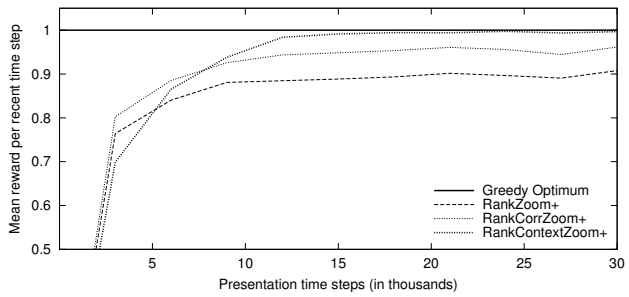

Figure 2. Comparison of zooming variants in a two-slot setting over a small document collection.

`RankZoom+` is a close second. Interestingly, the theoretically preferred `RankContextZoom` does not perform as well in simulations. This appears to be due to the much larger branching factor in the strategies activated by `RankContextZoom` slowing down the convergence.

Secondary experiment. As discussed in Section 5, some `RankBandit`-style algorithms may converge to a suboptimal ranking if μ has multiple peaks with similar values. To investigate this, we designed a small-scale experiment presented in Figure 2. We generated a small collection of 128 documents using the same setup with two "peaks", and assumed 2 slots. Each peak corresponds to a half of the user population, with peak value $\mu = \frac{1}{2}$ and background value $\mu_0 = 0.05$.

We see that `RankContextZoom+` converges more slowly than the other zooming variants, but eventually outperforms them. This confirms our intuition, and suggests that `RankContextZoom+` may eventually outperform the other algorithms on a larger collection.

7. Further directions

This paper initiates the study of online learning to rank in metric spaces, focusing on the "clean" similarity model (conditional L-continuity). As discussed in Section 5, we conjecture that provable guarantees for the algorithms can be improved significantly. On the experimental side, future work will include evaluating the model on web search data, and designing sufficiently memory- and time-efficient implementations to allow experiments on real users. An interesting challenge in such endeavor would be to come up with effective similarity measures. An natural next step would be to also exploit the similarity between search queries.

References

Agrawal, R. (1995). The continuum-armed bandit problem. *SIAM J. Control and Optimization*, 33(6):1926–1951.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002a). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256.

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. (2002b). The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77.

Auer, P., Ortner, R., and Szepesvári, C. (2007). Improved Rates for the Stochastic Continuum-Armed Bandit Problem. In *COLT*, pages 454–478.

Bubeck, S., Munos, R., Stoltz, G., and Szepesvari, C. (2008). Online Optimization in X-Armed Bandits. In *NIPS*, pages 201–208.

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

Carbonell, J. and Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*, pages 335–336.

Cesa-Bianchi, N. and Lugosi, G. (2006). *Prediction, learning, and games*. Cambridge Univ. Press.

Chen, H. and Karger, D. R. (2006). Less is more: Probabilistic models for retrieving fewer relevant documents. In *SIGIR*, pages 429–436.

Chu, W. and Ghahramani, Z. (2005). Gaussian processes for ordinal regression. *J. of Machine Learning Research*, 6:1019–1041.

Hazan, E. and Megiddo, N. (2007). Online Learning with Prior Knowledge. In *COLT*, pages 499–513.

Kleinberg, R. (2004). Nearly tight bounds for the continuum-armed bandit problem. In *NIPS*, pages 697–704.

Kleinberg, R. and Slivkins, A. (2010). Sharp Dichotomies for Regret Minimization in Metric Spaces. In *SODA*, pages 827–846.

Kleinberg, R., Slivkins, A., and Upfal, E. (2008). Multi-Armed Bandits in Metric Spaces. In *STOC*, pages 681–690.

Pandey, S., Agarwal, D., Chakrabarti, D., and Josifovski, V. (2007). Bandits for Taxonomies: A Model-based Approach. In *SIAM Intl. Conf. on Data Mining (SDM)*.

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

Slivkins, A. (2009). Contextual Bandits with Similarity Information. http://arxiv.org/abs/0907.3986.

Streeter, M. and Golovin, D. (2009). An online algorithm for maximizing submodular functions. In *NIPS*, pages 1577–1584.

Taylor, M., Guiver, J., Robertson, S., and Minka, T. (2008). Softrank: optimizing non-smooth rank metrics. In *WSDM*, pages 77–86.