Learning Diverse Rankings
with Multi-Armed Bandits

Robert Kleinberg
Dept. of Computer Science
Cornell University
Ithaca, NY 14853
rdk@cs.cornell.edu

Filip Radlinski
Dept. of Computer Science
Cornell University
Ithaca, NY 14853
filip@cs.cornell.edu

Thorsten Joachims
Dept. of Computer Science
Cornell University
Ithaca, NY 14853
tj@cs.cornell.edu

Abstract

The probabilistic ranking principle advocates ranking documents in order of decreasing probability of relevance to a query, independent of how other documents are ranked. The result is that similar documents are often ranked at similar positions. In contrast, empirical studies have shown that a diverse set of results is often preferable over one containing redundant results, as typical web queries often have different meanings for different users (such as *jaguar*). We present a new multi-armed bandit learning algorithm that directly learns a diverse ranking of results based on users' clicking behavior. In particular, it maximizes the probability that a relevant document is found in the top k positions of a ranking. After T presentations of n documents ranked for a fixed query, our algorithm achieves a total payoff of at least $(1 - 1/e)\,OPT - O(k\sqrt{Tn\log n})$ where OPT is the payoff of the optimal ordering if we knew the information needs of all users and $(1 - 1/e)\,OPT$ is the best obtainable polynomial time approximation.

1 Introduction

Web search has become an essential component of the Internet infrastructure, and has hence attracted significant interest from the machine learning community as an application of learning algorithms. We consider the problem of minimizing abandonment in search engines via online learning and interactive experimentation using clickthrough data. Abandonment, which measures the fraction of users who do not click any search results after entering a query, is an important measure of user satisfaction because it indicates that users were presented with search results of no potential interest.

Our learning setting is fundamentally different from the conventional learning to rank problem (e.g. [1, 2, 3, 4, 5, 6, 7, 8]) in two ways. First, conventional algorithms require expert labeled data, and distinguish between a training phase and an evaluation phase. With only few exceptions (e.g. [9]), previous approaches do not conduct online experiments to optimize learning speed. The algorithm we design minimizes the total number of poor rankings displayed over *all* time. Second, conventional algorithms do not optimize abandonment, but information retrieval measures that assume the availability of expert labelled data such as Precision, Recall, Mean Average Precision (MAP) [10] and Normalized Discounted Cumulative Gain (NDCG) [7].

Unlike traditional retrieval approaches, minimizing abandonment requires accounting for dependencies between documents. Early search ranking algorithms weighted each word in each document using functions such as TFIDF [11] and BM25 [12], and then computed the cosine similarity between the query and document as a measure of relevance. As additional features proved useful for improving ranking quality, numerous techniques for learning to rank have been developed (e.g. [1, 2, 3, 4]). These algorithms use training data in the form of judgments assessing the relevance of individual documents to a query (either in absolute terms or comparing the relevance of two documents) to learn ranking function parameters θ. Given a new query q, the ranking function then *independently*

computes a score $f(q, d_i, \theta)$ for each document d_i and ranks documents by decreasing score. This also applies to recent algorithms that learn θ to maximize performance measures such as MAP [5, 6] and NDCG [7, 8] that do not treat documents independently.

The theoretical justification for ranking documents independently is the probabilistic ranking principle (PRP) [13]. It suggests that when ranking documents, we should assume that the relevance of a document is independent of the relevance of any other documents that may be ranked above or below it. The PRP permits document relevance judgments, typically provided by trained experts who are given queries and must deduce the actual user indent and document relevance, to be used as training data.

However, empirical studies have shown that given a fixed query, the same document can have different relevance to different users [14]. This undermines the assumption that each document has a single relevance score that can be provided as training data to the learning algorithm. Moreover, as users are usually satisfied with finding just one relevant document, the usefulness and relevance of a document does depend on other documents ranked higher. In fact most search engines today attempt to eliminate redundant results and produce *diverse* rankings that include documents that are potentially relevant to the query for different reasons. Yet doing this optimally using expert judgments would require document relevance to be measured for different possible meanings of a query.

Several researchers have presented algorithms for obtaining diverse rankings of documents from a non-diverse ranking. One common technique is Maximal Marginal Relevance (MMR) [15]. Given a similarity (relevance) measure between a document and a query $sim_1(d, q)$ and a similarity measure between two documents $sim_2(d_i, d_j)$, MMR iteratively selects documents by repeatedly finding $d_i = \operatorname{argmax}_{d \in \mathcal{D}} \lambda sim_1(d, q) - (1 - \lambda) \max_{d_j \in S} sim_2(d, d_j)$ where S is the set of documents already selected and λ is a tuning parameter. In this way the algorithm selects the most relevant documents that are also different from any documents selected earlier and ranked higher.

Critically, MMR requires the relevance of a document, determined by $sim_1(d, q)$, and the similarity of two documents, determined by $sim_2(d_i, d_j)$ to be known. It is usual to obtain sim_1 and sim_2 using algorithms such as those discussed above. The goal of MMR is rerank an already learned ranking of documents (that of ranking documents by decreasing sim_1 score) to improve diversity. All previous approaches of which we are aware that optimize for a diverse ranking similarly require a relevance function to be learned prior to performing a diversification step [16, 17, 18]. In our approach, we directly minimize abandonment based on interactive experimentation using the clicking behavior of users as training data.

2   Problem Formalization

We propose an approach for directly learning a diverse ranking of documents that does not require relevance judgments from experts as training data, and naturally accounts for dependencies between documents. At a high level, the algorithm learns a utility value for each document at each rank, maximizing the probability that a new user of the search system would find at least one relevant document within the top k positions. This is equivalent to minimizing the abandonment rate.

We model the optimization problem as follows. We wish to learn to rank a set of documents $\mathcal{D} = \{d_1, \ldots, d_n\}$ for one fixed query. Each user u_i in our population of users considers some subset of documents $A_i \subset \mathcal{D}$ as relevant to the query, and the remainder of the documents as non-relevant. Intuitively, users with different interpretations for the query would have different relevant sets, while users with similar interpretations would have identical or similar relevant sets.

At time t, we interact with a user represented by a relevant set of documents A_t. We present an ordered set of k documents, $B_t = (b_1(t), \ldots, b_k(t))$. The user considers the results in order and clicks on the highest ranked (i.e. first) relevant document. If none of the k documents are relevant, the user does not click on anything. We get payoff 1 if the user clicks, 0 if not.

This model corresponds to each user observing the top k documents until they find any relevant one, and then clicking and stopping. The total payoff summing for $t = 1$ to T is the number of users who were satisfied by the ranking they were presented with. The goal is to maximize the total payoff. In particular, we treat this problem as an online learning problem with no distinction between training and testing phases.

3 Ranked Bandits Algorithm

This section presents an online learning algorithm whose combined payoff after T time steps is at least $(1 - 1/e)OPT - O(k\sqrt{Tn \log n})$. OPT denotes the maximal payoff that could be obtained if the set of relevant documents for all users were known, and we could always present the ranking with maximum expected payoff.

Algorithm 1 Ranked Bandits Algorithm

1: initialize $\mathsf{MAB}_1(n), \ldots, \mathsf{MAB}_k(n)$ Initialize k Multi-Armed Bandit instances
2: **for** t = 1 ... T **do**
3:     **for** i = 1 ... k **do**   Sequentially select the documents to show
4:        $\hat{b}_i(t) \leftarrow$ select-arm (MAB_i)
5:        **if** $\hat{b}_i(t) \in \{b_1(t), \ldots, b_{i-1}(t)\}$ **then**   Replace repeated documents arbitrarily
6:           $b_i(t) \leftarrow$ arbitrary unselected document
7:        **else**
8:           $b_i(t) \leftarrow \hat{b}_i(t)$
9:        **end if**
10:     **end for**
11:     display $\{b_1(t), \ldots, b_k(t)\}$ to user; record rank of any click
12:     **for** i = 1 ... k **do**
13:        **if** user clicked at rank i **and** $\hat{b}_i(t) = b_i(t)$ **then**   Determine feedback value for MAB_i
14:           $f_{it} = 1$
15:        **else**
16:           $f_{it} = 0$
17:        **end if**
18:        update $(\mathsf{MAB}_i, arm = \hat{b}_i(t), reward = f_{it})$
19:     **end for**
20: **end for**

This algorithm works by running an instance of a multi-armed bandit MAB_i for *each rank i*. Each of the k copies of the multi-armed bandit algorithm maintains a value (or index) for every document in the collection. When selecting the ranking to display to users, the algorithm MAB_1 is responsible for choosing which document is shown at rank 1. Next, the algorithm MAB_2 determines which document is shown at rank 2, unless this is the same document as shown at the highest rank. In this case, the second document is picked arbitrarily. This process is repeated to select all top k documents.

Next, after a user considers up to the top k documents in order and selects the first relevant one, we need to update the indices. If the user clicks on a document selected by an MAB instance, the reward for the arm corresponding to that document for the multi-armed bandit at that rank is 1. The reward for the arms corresponding to all other selected documents is 0. In particular, note that the Ranked Bandits Algorithm treats the bandits corresponding to each rank independently.

The precise multi-armed bandit algorithm used for each rank is not critical, and in fact any algorithm for the non-stochastic multi-armed bandit problem will suffice. The only facts we use are:

- The algorithm has a set S of n strategies.
- In each period t a payoff function $f_t : S \rightarrow [0, 1]$ is defined. This function is not revealed to the algorithm.
- In each period the algorithm chooses a (random) element $y_t \in S$ based on the feedback revealed in prior periods.
- The feedback revealed in period t is the number $f_t(y_t)$.
- The expected payoffs of the chosen strategies satisfy:

$$\sum_{t=1}^{T} \mathbf{E}[f_t(y_t)] \geq \max_{y \in S} \sum_{t=1}^{T} \mathbf{E}[f_t(y)] - R(T)$$

where $R(T)$ is an explicit function in $o(T)$ which depends on the particular multi-armed bandit algorithm chosen. We will use the **Exp3** algorithm in our analysis, where $R(T) = O(\sqrt{Tn \log n})$ [19].

4 Analysis of Ranked Bandits Algorithm

A first observation is that the problem of choosing the optimum set of k documents for a given user population is NP-hard, even if all the information about the user population (i.e. the set of relevant documents for each user) is given offline. This is because it is equivalent to the maximum coverage problem: given a positive integer k and a collection of subsets $S_1, S_2, \ldots, S_n$ of an m-element set, find k of the subsets whose union has the largest possible cardinality. It is well-known that the greedy algorithm is a $(1-1/e)$-approximation algorithm for this maximization problem, and that no better approximation ratio is achievable in polynomial time unless $NP \subseteq DTIME\left(n^{\log \log n}\right)$. (For the hardness result, see Khuller, Moss, and Naor [20])

The Ranked Bandits Algorithm (RBA) works by simulating the offline greedy algorithm, using a separate instance of the multi-armed bandit algorithm for each step of the greedy algorithm. Except for the sublinear regret term, the combined payoff is as high as possible without violating the hardness-of-approximation result stated in the preceding paragraph.

To analyze RBA, it will be useful to introduce some notation. For a set A and a sequence $B = (b_1, b_2, \ldots, b_k)$, let

$$G_i(A, B) = \begin{cases} 1 & \text{if } A \text{ intersects } \{b_1, \ldots, b_i\} \\ 0 & \text{otherwise} \end{cases}$$

$$g_i(A, B) = G_i(A, B) - G_{i-1}(A, B)$$

Note that $G_k(A_t, B)$ is the payoff of presenting B to the user u_t. Let

$$OPT = \max_B \sum_{t=1}^{T} G_k(A_t, B)$$

$$B^* = \operatorname*{argmax}_B \sum_{t=1}^{T} G_k(A_t, B).$$

Recall that $(\hat{b}_1(t), \ldots, \hat{b}_k(t))$ is the sequence of documents chosen by the algorithms $\mathsf{MAB}_1, \ldots, \mathsf{MAB}_k$ at time t, and that $(b_1(t), \ldots, b_k(t))$ is the sequence of documents presented to the user. Define the feedback function f_{it} for algorithm MAB_i at time t, as follows:

$$f_{it}(b) = \begin{cases} 1 & \text{if } G_{i-1}(A_t, B_t) = 0 \text{ and } b \in A_t \\ 0 & \text{otherwise} \end{cases}.$$

Note that the value of f_{it} defined in the pseudocode for the Ranked Bandits Algorithms is equal to $f_{it}(\hat{b}_i(t))$.

Lemma 1. *For all* i,

$$\mathbf{E}\left[\sum_{t=1}^{T} g_i(A_t, B_t)\right] \geq \frac{1}{k} \mathbf{E}\left[\sum_{t=1}^{T} \left(G_k(A_t, B^*) - G_{i-1}(A_t, B_t)\right)\right] - R(T).$$

Proof. First, note that

$$g_i(A_t, B_t) \geq f_{it}(\hat{b}_i(t)). \tag{1}$$

This is trivially true when $f_{it}(\hat{b}_i(t)) = 0$. When $f_{it}(\hat{b}_i(t)) = 1$, we have $G_{i-1}(A_t, B_t) = 0$ and $\hat{b}_i(t) \in A_t$. This implies that $b_i(t) = \hat{b}_i(t)$ and that $g_i(A_t, B_t) = 1$.

Now using the regret bound for MAB_i we obtain

$$\sum_{t=1}^{T} \mathbf{E}[f_{it}(\hat{b}_i(t))] \geq \max_b \sum_{t=1}^{T} \mathbf{E}[f_{it}(b)] - R(T)$$

$$\geq \frac{1}{k} \mathbf{E}\left[\sum_{b \in B^*} \sum_{t=1}^{T} f_{it}(b)\right] - R(T). \tag{2}$$

4

To complete the proof of the lemma, we will prove that

$$\sum_{b \in B^*} f_{it}(b) \geq G_k(A_t, B^*) - G_{i-1}(A_t, B_t). \tag{3}$$

The lemma follows immediately by combining (1)-(3). Observe that the left side of (3) is a non-negative integer, while the right side takes one of the values $\{-1, 0, 1\}$. Thus, to prove (3) it suffices to show that the left side is greater than or equal to 1 whenever the right side is equal to 1. The right side equals 1 only when $G_{i-1}(A_t, B_t) = 0$ and A_t intersects B^*. In this case it is clear that there exists at least one $b \in B^*$ such that $f_{it}(b) = 1$, hence the left side is greater than or equal to 1. $\quad\square$

Theorem 1. *The algorithm's combined payoff after T rounds satisfies:*

$$\mathbf{E}\left[\sum_{t=1}^{T} G_k(A_t, B_t)\right] \geq \left(1 - \frac{1}{e}\right) OPT - kR(T).$$

Proof. We will prove, by induction on i, that

$$OPT - \mathbf{E}\left[\sum_{t=1}^{T} G_i(A_t, B_t)\right] \leq \left(1 - \frac{1}{k}\right)^i OPT + kR(T). \tag{4}$$

The theorem follows by taking $i = k$ and using the inequality $\left(1 - \frac{1}{k}\right)^k < \frac{1}{e}$.

In the base case $i = 0$, inequality (4) is trivial. For the induction step, let

$$Z_i = OPT - \mathbf{E}\left[\sum_{t=1}^{T} G_i(A_t, B_t)\right].$$

We have

$$Z_i = Z_{i-1} - \mathbf{E}\left[\sum_{t=1}^{T} g_i(A_t, B_t)\right], \tag{5}$$

and Lemma 1 says that

$$\mathbf{E}\left[\sum_{t=1}^{T} g_i(A_t, B_t)\right] \geq \frac{1}{k} Z_{i-1} - R(T). \tag{6}$$

Combining (5) with (6), we obtain

$$Z_i \leq \left(1 - \frac{1}{k}\right) Z_{i-1} + R(T).$$

Combining this with the induction hypothesis proves (4). $\quad\square$

Note now that B^* is defined as the optimal subset of k documents, and OPT is the payoff of presenting B^*, without specifying the order documents are presented in. However, the Ranked Bandits Algorithm learns an order for the documents in addition to identifying a set of documents. In particular, given $k' < k$, RBA(k') would receive exactly the same feedback as the first k' instances of MAB receive when running RBA(k). Hence any k' sized prefix of the learned ranking also has the same performance bound with respect the appropriate smaller set B'^*.

5 Conclusions

We have presented the Ranked Bandits Algorithm that directly optimizes a ranking of documents for diversity as well as relevance. In contrast to previous approaches, we do not require document relevance to be learned prior to a diversification step. We have shown this algorithm to be optimal within a sub-linear regret term relative to the best obtainable ranking in polynomial time even given complete information. We plan to extend this algorithm to non-binary document relevance settings, and perform empirical evaluations of its performance.

References

[1] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*, pages 115–132, 2000.

[2] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.

[3] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2005.

[4] Wei Chu and Zoubin Ghahramani. Gaussian processes for ordinal regression. *Journal of Machine Learning Research*, 6:1019–1041, 2005.

[5] Donald Metzler and W. Bruce Croft. A markov random field model for term dependencies. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 472–479, 2005.

[6] Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. A support vector method for optimizing average precision. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 2007.

[7] Christopher J. C. Burges, Robert Ragno, and Quoc Viet Le. Learning to rank with nonsmooth cost functions. In *Proceedings of the Internation Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 193–200. MIT Press, 2006.

[8] Michael J. Taylor, John Guiver, Stephen E. Robertson, and Tom Minka. Softrank: Optimizing non-smooth ranking metrics. In *Proceedings of ACM International Conference on Web Search and Data Mining*, to appear 2008.

[9] Filip Radlinski and Thorsten Joachims. Active exploration for learning rankings from clickthrough data. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD)*, 2007.

[10] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, New York, NY, 1999.

[11] Gerry Salton and Chris Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[12] Stephen E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *Proceedings of TREC-3*, 1994.

[13] Stephen E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304, 1977.

[14] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Characterizing the value of personalizing search. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 2007.

[15] Jamie Carbonell and Jade Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 335–336, 1998.

[16] Xiaojin Zhu, Andrew B Goldberg, Jurgen Van Gael, and David Andrzejewski. Improving diversity in ranking using absorbing random walks. *HLT/NAACL*, 2007.

[17] Benyu Zhang, Hua Li, Yi Liu, Lei Ji, Wensi Xi, Weiguo Fan, Zheng Chen, and Wei-Ying Ma. Improving web search results using affinity graph. In *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM)*, 2005.

[18] Cheng Zhai, William W. Cohen, and John Lafferty. Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 2003.

[19] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The non-stochastic multi-armed bandit problem. *SIAM Journal of Computing*, 32(1):48–77, 2002.

[20] Samir Khuller, Anna Moss, and Joseph Naor. The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39–45, April 1997.