Learning Diverse Rankings with Multi-Armed Bandits

Filip Radlinski Robert Kleinberg Thorsten Joachims
Cornell University

The Importance of Being Diverse

- Typical method for learning to rank: learn a scoring function and output the *k* documents with the top scores.
- Justification: the *probabilistic ranking principle*. (Documents should be ranked according to their probability of relevance.)
- This overlooks the diversity of users' information needs.
- Example: query term is *bandits*. Do you mean...
  - A class of online learning problems featuring tradeoffs between exploration and exploitation?
  - The indoor lacrosse team from Buffalo, NY?
  - A film made by Barry Levinson in 2001?

Using Implicit Feedback

- Our approach: learn from usage data (clickthrough logs) rather than manually labeled relevance judgments.
  - Usage data is more plentiful and is obtained at low cost.
  - Adjusts over time to changes in users' needs and in the documents available.
  - Obtaining expert judgments is not even feasible in some settings, e.g. searching small document collections.
- Learning from usage data is an online learning problem: choose which *k* documents to present to the current user, based on past performance (exploitation) and on utility for training future decisions (exploration).

Prior Work

- Manually labeled training data, ranking by relevance
  - large body of work, too many to list here.
- Manually labeled training data, accounting for diversity
  - Learn a relevance function, then use a technique such as maximum marginal relevance (MMR) (Carbonell & Goldstein, '98) to re-rank in a subsequent diversification step (Zhu et al. '07, Zhang et al. '05, Zhai et al. '03).
  - Assume given a model estimating probability of relevance given a query and other non-relevant documents (Chen & Karger '06).
  - Train structural SVM's to maximize topic coverage (Yue & Joachims '08).

Prior Work (cont'd)

- Train using usage data, rank by relevance
  - Passive methods, using clickthrough logs to generate features. (e.g. Joachims '02, Radlinski & Joachims '05, Agichtein et al. '06)
  - Active methods: choose top two documents to maximize the benefit of information gained from experimentation. (Radlinski & Joachims '07)
- Train using usage data, accounting for diversity
  - This work. (Radlinski et al. '08)
  - Not just a trivial combination of techniques discussed earlier: the online learning problem has $O(n^k)$ strategies, rendering existing algorithms impractical. Instead we exploit problem structure to design a special-purpose online learning algorithm.

Our Results

- Theoretical
  - Two algorithms, each achieving a payoff (number of clicks) at least (1-1/e)*OPT - o(T) after facing T users.
  - OPT is the number of clicks received by the best ranking.
  - The factor of 1-1/e is unavoidable for complexity reasons.
- Simulation results
  - Under realistic simulation of user behavior, both algorithms significantly outperform both the 1-1/e bound and the static popularity-based ranking.
  - A third algorithm, using the same ideas but without the theoretical guarantees, exhibits faster convergence.

Outline of This Talk

1. Introduction
2. Review of multi-armed bandit algorithms
3. Ranked bandit algorithms
4. Simulation results
5. Extensions and conclusions

Multi-armed Bandit Problems

- In a multi-armed bandit problem one has:
  - A set S of n *strategies*;
  - Unknown sequence of *payoff functions* $g_t: S \to [0,1]$.
  - An *adaptive adversary* chooses g_t arbitrarily based on past history.
  - An *iid adversary* samples g_t from a fixed, unknown distribution.
- In each time step *t*...
  - Algorithm picks x_t in S based on past history.
  - Adversary picks payoff function g_t.
  - $g_t(x_t)$ is revealed.
- *Regret* = diff. between algorithm's expected payoff and best strategy.

$$\mathrm{Regret}(T) = \max_{x \in S} \mathbf{E}\left[\sum_{t=1}^{T} g_t(x)\right] - \mathbf{E}\left[\sum_{t=1}^{T} g_t(x_t)\right].$$

Multi-armed Bandit Problems

0.3	0.7	0.4	
0.5	0.1	0.6	
0.2	0.2	0.7	
0.3	0.8	0.5	
0.6	0.1	0.4	
2.2	vs	2.6	Regret = 0.4

Multi-armed Bandit Algorithms

- Explore and Commit: Given positive integer parameter s,
  - Sample each strategy s times.
  - After this sampling period, select the one with highest average payoff and play only this strategy from then on.
  - Against iid adversary, Regret(T) ≤ $O(ns+(\log(nT)/s)^{1/2}T)$.
- UCB1:
  - Initially play each strategy once.
  - After that, always play the one that maximizes $\mu_t(x)+w_t(x)$ where $\mu_t(x)$=average payoff, $w_t(x)=(2 \log t \,/\, n_t(x))^{1/2}$.
  - Against iid adversary, Regret(T) ≤ $O((nT \log T)^{1/2})$ and Regret(T) ≤ C*log(T) for an adversary-dependent const. C.

Multi-armed Bandit Algorithms

- Exp3: Given parameter $\gamma>0$.
  - Maintain an unbiased estimator $u(x)$ of total payoff received by x during steps 1,2, ...,t.
  - Let p_t be a prob. distrib. such that $p_t(x) \propto \exp(-\gamma u(x)/n)$.
  - Sample i from distribution $q_t = (1-\gamma)*p_t + \gamma*Uniform$.
  - Increment $w(x)$ by $g_t(x)/q_t(x)$, leave $w(y)$ unchanged for $y \neq x$.
  - Against adaptive adversary, Regret(T) = $O(\gamma T + n\ log(n)/\gamma)$.
- Summary: The three algorithms have different benefits.
  - Explore & Commit ⇒ simplicity; UCB1 ⇒ fast convergence;

    Exp3 ⇒ robustness against adaptive adversary.

Abandonment Minimization

- Our problem: Learning a diverse ranking for a *single query.*
- Standing assumptions:
  - There is a set of *n* documents.
  - User *i* is represented by a set of relevant documents A_i.
  - User behavior (for now): when presented with list of *k* documents, click the first relevant one.
  - Later, will generalize user behavior model to allow probabilistic clicking.
- Payoff = number of clicks.
- Maximizing payoff = minimizing abandonment.

Offline Complexity of Abandonment Minimization

- Suppose, at time 0, we are told the set A_i for every user *i*.
- Equivalently, for every document *d* we are told the set B_d of users satisfied by *d*.
- Computing the optimal ranking (list of *k* documents) is thus equivalent to the following problem: choose *k* of the sets B_d so as to maximize the cardinality of their union.
- This is the NP-hard maximum coverage problem.
- The greedy algorithm (always pick the set with the most uncovered elements) has approximation ratio 1-1/e = 0.63...
- No better approximation ratio achievable in poly-time unless NP $\subseteq$ DTIME($n^{O(\log \log n)}$).

Analysis of Greedy Algorithm

- Let $B_1,...,B_k$ be the sets with largest union. Let B be their union and denote its cardinality by OPT.
- Let $U = OPT$ - (number of elements covered so far).
- Initially $U = OPT$.
- Each iteration of greedy algorithm reduces U by a factor of $1\text{-}1/k$ or better.
- At termination, $U \leq (1\text{-}1/k)^k \cdot OPT < (1/e) \cdot OPT$.
- Hence number of covered elements is $> (1\text{-}1/e) \cdot OPT$.

Abandonment Minimization as a Bandit Problem

- Abandonment minimization is a multi-armed bandit problem:
  - Strategy set S = {rankings}
  - Payoff $g_t(x) = 1$ if x contains an element of A_t, else 0.
- But $|S| = n(n-1)\cdots(n-k+1) \Rightarrow$ standard methods don't scale.
- Recent work combining online learning with approximation algorithms (Kakade, Kalai, Ligett STOC '07) only applies with linear payoff functions. We have submodular payoffs.
- Instead we design an online algorithm modeled on the offline greedy algorithm: instantiate one MAB algorithm for each position in the ranking; it tries to maximize marginal benefit.

Ranked Explore and Commit

Go through each position in the ranking, sequentially, running explore-and-commit to decide on a document for that position, given previous committed decisions.

Theorem: Assume iid users. With probability 1-δ, the payoff is at least

$$\left(1 - \frac{1}{e} - \epsilon\right) \cdot OPT - O\left(k^3 n/\epsilon^2 \ln(k/\delta)\right)$$

input: Documents $(d_1, .., d_n)$, parameters ϵ, δ, k.
$s \leftarrow \lceil 2k^2/\epsilon^2 \log(2k/\delta) \rceil$
$(b_1, \ldots, b_k) \leftarrow k$ arbitrary documents.
for i=1 . . . k **do** *// At every rank*
  $\forall j.\ p_j \leftarrow 0$
  for counter=1 . . . s **do** *// Loop s times*
    for j=1 . . . n **do** *// over every document d_j*
      $b_i \leftarrow d_j$
      display $\{b_1, \ldots, b_k\}$ to user; record clicks
      if user clicked on b_i **then** $p_j \leftarrow p_j + 1$
    end for
  end for
  $j^* \leftarrow \mathrm{argmax}_j\, p_j$ *// Commit to best document at this rank*
  $b_i \leftarrow d_{j^*}$
end for

Ranked Bandits Algorithm

Run a separate MAB algorithm (e.g. Exp3) at each rank 1,...,k.

Its feedback is its marginal contribution to the payoff.

Thus, feedback to MAB_i is 1 if the user clicked the document that it chose, and this document was not displayed at any higher rank.

initialize $\text{MAB}_1(n), \ldots, \text{MAB}_k(n)$ *// Initialize MABs*
for t = 1 . . . T **do**
    for i = 1 . . . k **do** *// Sequentially select documents*
        $\hat{b}_i(t) \leftarrow$ select-arm (MAB_i)
        if $\hat{b}_i(t) \in \{b_1(t), .., b_{i-1}(t)\}$ **then** *// Replace repeats*
            $b_i(t) \leftarrow$ arbitrary unselected document
        else
            $b_i(t) \leftarrow \hat{b}_i(t)$
        end if
    end for
    display $\{b_1(t), \ldots, b_k(t)\}$ to user; record clicks
    for i = 1 . . . k **do** *// Determine feedback for* MAB_i
        if user clicked $b_i(t)$ **and** $\hat{b}_i(t) = b_i(t)$ **then**
            $f_{it} = 1$
        else
            $f_{it} = 0$
        end if
        update $(\text{MAB}_i, arm = \hat{b}_i(t), reward = f_{it})$
    end for
end for

Ranked Bandits Algorithm

Theorem: If each subroutine MAB_i has regret bounded by R(T) against adaptive adversary, then the payoff of the Ranked Bandits Algorithm is at least

$$\left(1 - \frac{1}{e}\right) \cdot OPT - kR(T).$$

Remarks:

1. Analysis closely parallels analysis of offline greedy algorithm.
2. The MAB subroutines must work against adaptive adversary even if users are iid. This is because MAB_i faces a changing payoff distribution due to the changing behavior of MAB_j, for j<i.
3. For MAB=Exp3, we have R(T) = $O((n\,T\log n)^{1/2})$.

Extension: probabilistic users

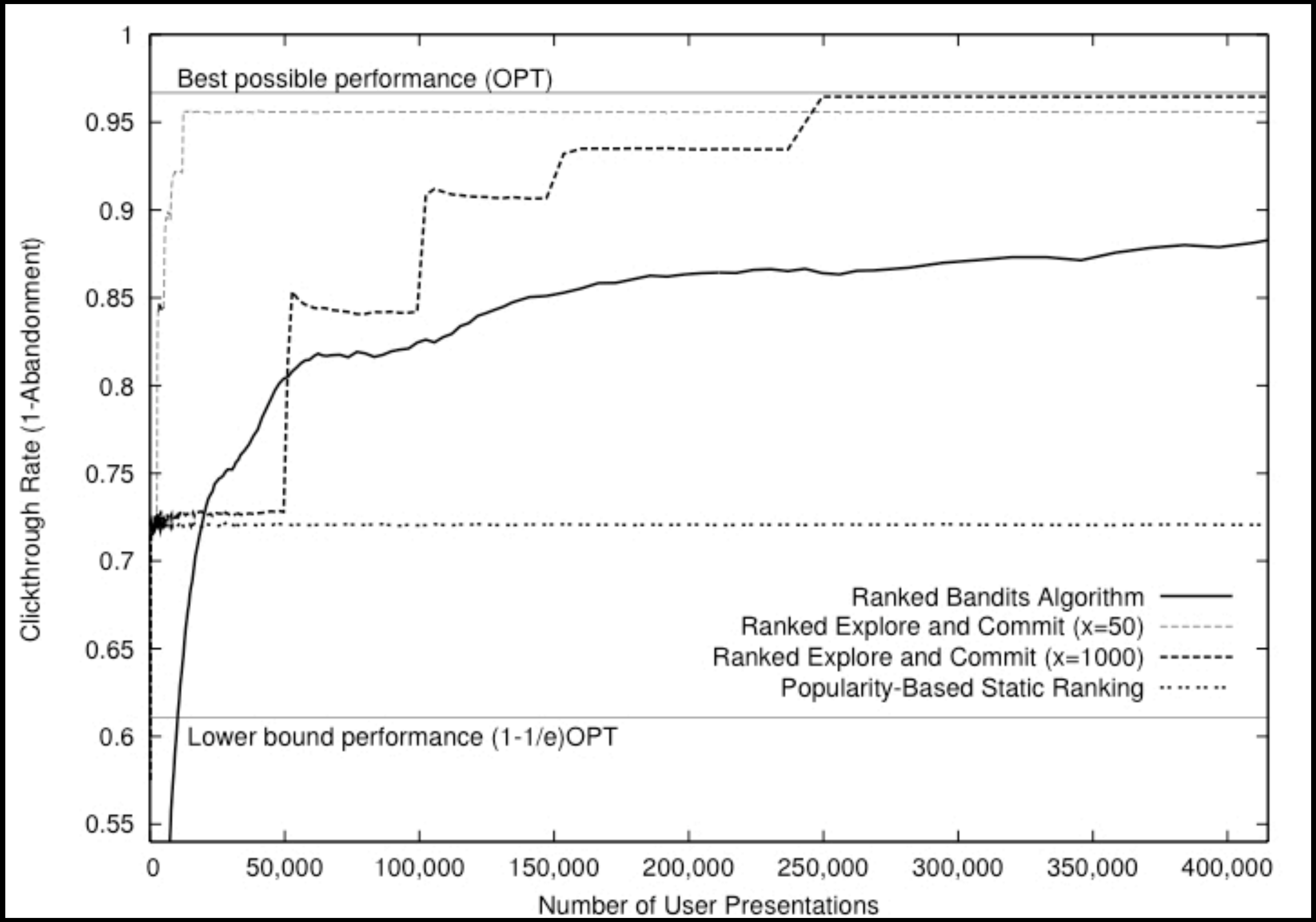
- A probabilistic user i is specified by probabilities $p(i,d)$ for each document d.
- User scans list of documents from rank 1 to rank d.
- Given that it encounters document d while scanning, with probability $p(i,d)$ it clicks this documents and stops scanning.
- All of our results carry over to probabilistic users since a probabilistic user is nothing more than a probability distribution over deterministic users. (Sample independent Bernoulli random variables $X(i,d)$ with expectations $p(i,d)$ and let A_i be the set of all d such that $X(i,d)=1$.)

Simulation Environment

- 20 simulated users assigned to topics of interest using Chinese Restaurant Process. (Generates power-law distribution of topic popularity. Mean number of topics was 6.5.)
- 50 documents assigned to topics in proportion to popularity.
- A user deems all documents in its assigned topic relevant, all others irrelevant.
- Draw uniformly-random users with replacement, run algorithm with k=5 on this user sequence.
- Report averages over 1000 algorithm runs.

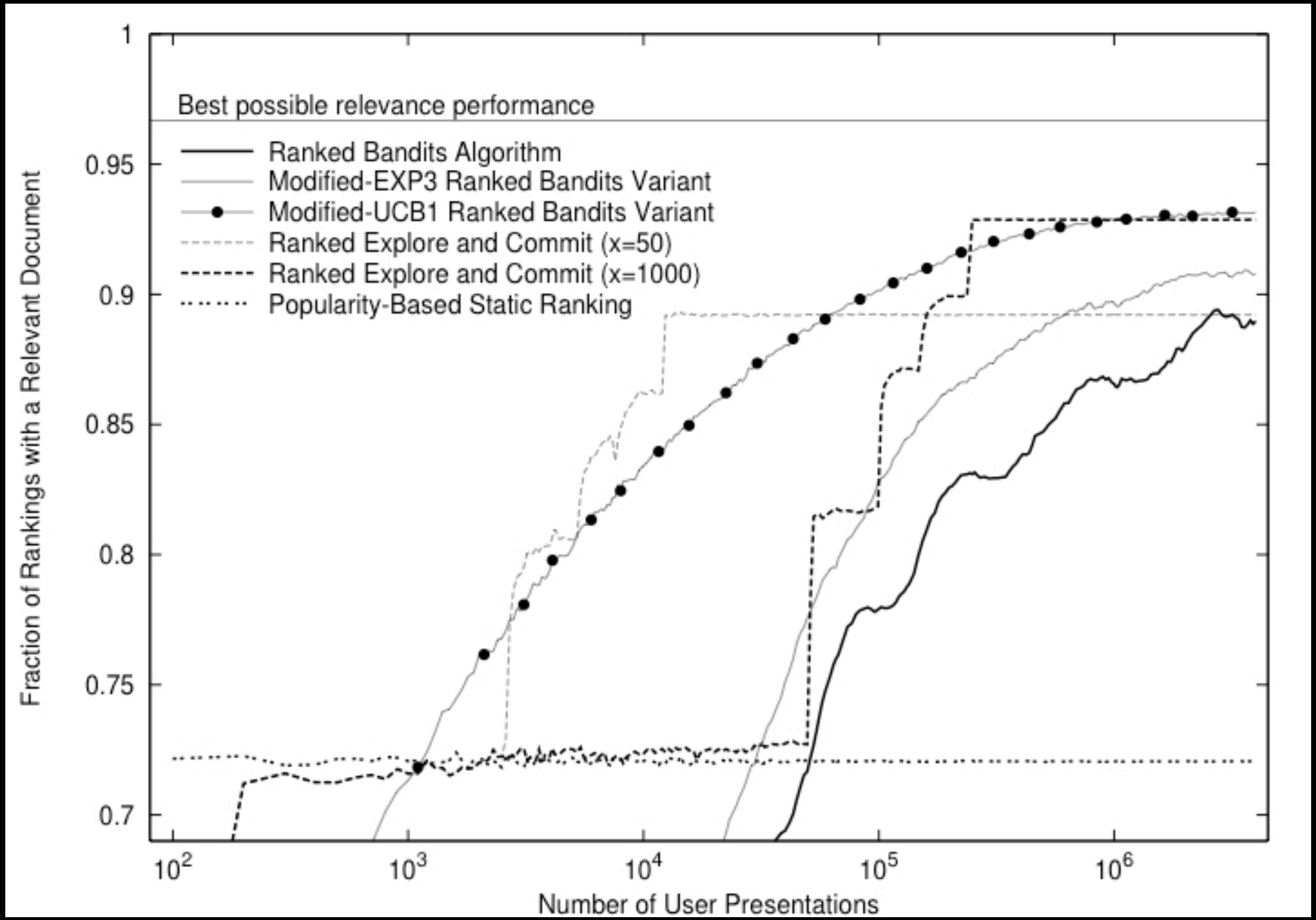
Evaluation

REC and RBA outperform (1-1/e)*OPT and they outperform popularity-based static ranking. REC payoff is near OPT.



Evaluation

RBA using UCB1 performs best of all, despite lacking any theoretical guarantees!



Conclusions

- Main contributions of this work
  1. Formulated the problem of minimizing abandonment as an online learning problem.
  2. Proposed the Ranked Bandits Algorithm, which achieves optimal worst-case performance given polynomial-time computation.
  3. Identified a "tweaked" version of this algorithm which outperforms all others that we tested on synthetic data.
- Main open problem: an online learning algorithm, driven by usage data, that can generalize across multiple queries.