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 \((1-1/e) \cdot \text{OPT} - o(T)\) after facing \(T\) users.
  - \(\text{OPT}\) 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 $g_t : S \rightarrow [0, 1]$.
  - 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.

\[ \text{Regret}(T) = \max_{x \in S} \mathbb{E} \left[ \sum_{t=1}^{T} g_t(x) \right] - \mathbb{E} \left[ \sum_{t=1}^{T} g_t(x_t) \right]. \]
## Multi-armed Bandit Problems

<table>
<thead>
<tr>
<th>0.3</th>
<th>0.7</th>
<th>0.4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.1</td>
<td>0.6</td>
</tr>
<tr>
<td>0.2</td>
<td>0.2</td>
<td>0.7</td>
</tr>
<tr>
<td>0.3</td>
<td>0.8</td>
<td>0.5</td>
</tr>
<tr>
<td>0.6</td>
<td>0.1</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Regret = 0.4

2.2 vs 2.6
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, $\text{Regret}(T) \leq O(ns + \left(\log(nT)/s\right)^{1/2}T)$.

- **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, $\text{Regret}(T) \leq O((nT \log T)^{1/2})$ and $\text{Regret}(T) \leq C^*\log(T)$ 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 $q_t = (1-\gamma) p_t + \gamma \cdot \text{Uniform}$.
  - Increment $w(x)$ by $g_t(x)/q_t(x)$, leave $w(y)$ unchanged for $y \neq x$.
  - Against adaptive adversary, $\text{Regret}(T) = O(\gamma T + n \log(n)/\gamma)$.
- **Summary**: The three algorithms have different benefits.
  - Explore & Commit $\Rightarrow$ simplicity; UCB1 $\Rightarrow$ fast convergence; Exp3 $\Rightarrow$ 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 $NP \subseteq \text{DTIME}(n^{O(\log \log n)})$. 
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 $U = OPT - \text{(number of elements covered so far)}$.
- Initially $U = OPT$.
- Each iteration of greedy algorithm reduces $U$ by a factor of $1-1/k$ or better.
- At termination, $U \leq (1-1/k)^k \cdot OPT < (1/e) \cdot OPT$.
- Hence number of covered elements is $> (1-1/e) \cdot OPT$. 
Abandonment Minimization as a Bandit Problem

• Abandonment minimization is a multi-armed bandit problem:
  • Strategy set $S = \{\text{rankings}\}$
  • 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.
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 $1 - \delta$, the payoff is at least

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

1: **input:** Documents $(d_1, \ldots, d_n)$, parameters $\epsilon, \delta, k$.
2: $s \leftarrow \lceil 2k^2 / \epsilon^2 \log(2k/\delta) \rceil$
3: $(b_1, \ldots, b_k) \leftarrow k$ arbitrary documents.
4: **for** $i = 1 \ldots k$ **do**  // At every rank
5:   $\forall j, p_j \leftarrow 0$
6:   **for** counter = 1 \ldots $s$ **do**  // Loop $s$ times
7:     **for** $j = 1 \ldots n$ **do**  // over every document $d_j$
8:       $b_i \leftarrow d_j$
9:     **display** $\{b_1, \ldots, b_k\}$ to user; record clicks
10:    **if** user clicked on $b_i$ **then** $p_j \leftarrow p_j + 1$
11:   **end for**
12: **end for**
13: $j^* \leftarrow \text{argmax}_j p_j$  // Commit to best document at this rank
14: $b_i \leftarrow d_j^*$
15: **end for**
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.

```
1: initialize MAB\(_1\)(n),...,MAB\(_k\)(n)  // Initialize MABs
2: for t = 1 \ldots T do
3:   for i = 1 \ldots k do // Sequentially select documents
4:     \(\hat{b}_i(t) \leftarrow\) select-arm (MAB\(_i\))
5:     if \(\hat{b}_i(t) \in \{b_1(t),...,b_{i-1}(t)\}\) then // Replace repeats
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 \(\{b_1(t),...,b_k(t)\}\) to user; record clicks
12: for i = 1 \ldots k do // Determine feedback for MAB\(_i\)
13:   if user clicked \(b_i(t)\) and \(\hat{b}_i(t) = b_i(t)\) then
14:     \(f_{it} = 1\)
15:   else
16:     \(f_{it} = 0\)
17:   end if
18: update (MAB\(_i\), arm = \(\hat{b}_i(t)\), reward = \(f_{it}\))
19: end for
20: end for
```
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 \( MAB = \text{Exp3} \), we have \( R(T) = O((n T \log n)^{1/2}) \).
A probabilistic user $i$ is specified by probabilities $p(i,d)$ for each document $d$.

User scans list of documents from rank $l$ to rank $d$.

Given that it encounters document $d$ while scanning, with probability $p(i,d)$ it clicks this document 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.
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.