Online Learning to Rank in Stochastic Click Models

Zoghi, M., Tunys, T., Ghavamzadeh, M., Kveton, B., Szepesvari, C., & Wen, Z.
ICML 2017

Kuhwan Jeong [1]

[1] Department of Statistics, Seoul National University, South Korea

July, 2018

Introduction

- *Click model* is a probabilistic model of how users examine and click on a list of documents.
- Click models is designed to explain the so-called position bias in click data.
- Goal of learning to rank (LTR) is to provide a list of K documents out of L that maximizes the number of clicks made by users.
- In this paper, an online LTR algorithm, which can be used in a broad class of click models, is proposed.

Click Models

- All documents (items) are represented by $\mathcal{D} = \{1, \ldots, L\}$.
- User is presented an ordered list of K documents out of L, denoted by $\mathcal{R} = (d_1, \ldots, d_K)$.
- Click models are parametrized by *attraction function* α, where $\alpha(d)$ is the probability that item d is attractive, and *examination function* χ, where $\chi(\alpha, \mathcal{R}, k)$ is the probability that the k-th item in $\mathcal{R}$ is examined.
- The probability of clicking k-th item in $\mathcal{R}$ is $\chi(\alpha, \mathcal{R}, k)\alpha(d_k)$.
- The expected number of click on list $\mathcal{R}$ is

$$r(\mathcal{R}) = \sum_{k=1}^{K} \chi(\alpha, \mathcal{R}, k)\alpha(d_k).$$

- Without loss of generality, let $\alpha(1) \geq \cdots \geq \alpha(L)$.

Click Models - (1) Position-Based Model

- Position-based model (PBM) assumes the examination function depends only on the position
$$\chi(\alpha, \mathcal{R}, k) = \chi(k).$$
- The expected number of clicks on $\mathcal{R}$ is
$$r(\mathcal{R}) = \sum_{k=1}^{K} \chi(k)\alpha(d_k).$$
- Under the assumption $\chi(1) \geq \cdots \geq \chi(K)$, $r(\mathcal{R})$ is maximized by the list of K most attractive items
$$\mathcal{R}^* = (1, \ldots, K)$$
where the k-th most attractive item is placed at position k.

Click Models - (2) Cascade Model

- In the cascade model (CM), the user scans a list of items $\mathcal{R} = (d_1, \ldots, d_K)$ from d_1 to d_K.

- If item d_k is *attractive*, the user *clicks* on it and does not examine the remaining items.

- If item d_k is not attractive, the user *examines* item d_{k+1}.

- The examine function χ becomes

$$\chi(\alpha, \mathcal{R}, k) = \prod_{i=1}^{k-1} (1 - \alpha(d_i)).$$

- The expected number of clicks on $\mathcal{R}$ is

$$r(\mathcal{R}) = \sum_{k=1}^{K} \chi(k)\alpha(d_k) = 1 - \prod_{k=1}^{K} (1 - \alpha(d_k)).$$

- Any permutation of $\{1, \ldots, K\}$ is optimal in the CM.

Online Learning to Rank

Algorithm 1 BatchRank

// Initialization
for $b = 1, \dots, 2K$ **do**
  for $l = 0, \dots, T-1$ **do**
    for all $d \in \mathcal{D}$ **do**
      $c_{b,l}(d) \leftarrow 0, \; n_{b,l}(d) \leftarrow 0$
    end for
  end for
end for
$\mathcal{B} \leftarrow \{1\}, \; b_{\max} \leftarrow 1,$
$I_1 \leftarrow (1, K), \; B_{1,0} \leftarrow \mathcal{D}, \; l_1 \leftarrow 0$
// BatchRank
for $t = 1, \dots, T$ **do**
  for all $b \in \mathcal{B}$ **do**
    DisplayBatch(b, t)
  end for
  for all $b \in \mathcal{B}$ **do**
    CollectClicks(b, t)
  end for
  for all $b \in \mathcal{B}$ **do**
    UpdateBatch(b, t)
  end for
end for

- Two key ideas are
  1. randomizing the placement of items to avoid position biases and
  2. recursively dividing the batches of items into more and less attractive items.
- $\mathcal{B}$: the set indices of batches
- $B_{b,l}$: the set of items in batch b at stage l
- I_b : the range of positions of items in batch b

Online Learning to Rank

Algorithm 2 DisplayBatch

$l \leftarrow l_b$
Let $d_1, \ldots, d_{|B_{b,l}|}$ be a permutation of items in $B_{b,l}$ such that $n_{b,l}(d_1) \leq \cdots \leq n_{b,l}(d_{|B_{b,l}|})$.
Let π be a random permutation of $\{I_b(1), \ldots, I_b(2)\}$.
for $k = I_b(1), \ldots, I_b(2)$ **do**
    $d_k^t \leftarrow d_{\pi(k - I_b(1) + 1)}$
end for

Algorithm 3 CollectClicks

$l \leftarrow l_b, \; n_{\min} \leftarrow \min_{d \in B_{b,l}} n_{b,l}(d)$
for $k = I_b(1), \ldots, I_b(2)$ **do**
    if $n_{b,l}(d_k^t) = n_{\min}$ **then**
        $c_{b,l}(d_k^t) \leftarrow c_{b,l}(d_k^t) + c_t(k)$
        $n_{b,l}(d_k^t) \leftarrow n_{b,l}(d_k^t) + 1$
    end if
end for

Online Learning to Rank

Algorithm 4 UpdateBatch

$l \leftarrow l_b,\ \mathrm{len}(b) = I_b(2) - I_b(1) + 1$
if $\min_{d \in B_{b,l}} n_{b,l}(d) = n_l$ **then**
  for all $d \in B_{b,l}$ **do**
    Compute $U_{b,l}(d)$ and $L_{b,l}(d)$.
  end for
  Let $d_1, \ldots, d_{|B_{b,l}|}$ be a permutation of items in $B_{b,l}$ such that $L_{b,l}(d_1) \geq \cdots \geq L_{b,l}(d_{|B_{b,l}|})$.
  // Find a split
  $s \leftarrow 0$
  for $k = 1, \ldots, \mathrm{len}(b) - 1$ **do**
    $B_k^+ \leftarrow \{d_1, \ldots, d_k\},\ B_k^- \leftarrow B_{b,l} \setminus B_k^+$
    if $L_{b,l}(d_k) > \max_{d \in B_k^-} U_{b,l}(d)$ **then**
      $s \leftarrow k$
    end if
  end for
  if $s = 0,\ |B_{b,l}| > \mathrm{len}(b)$ **then**
    $B_{b,l+1} \leftarrow \{d \in B_{b,l} : U_{b,l}(d) \geq L_{b,l}(d_{\mathrm{len}(b)})\}$
    $l_b \leftarrow l_b + 1$
  else if $s > 0$ **then**
    // Split
    $\mathcal{B} \leftarrow \{b_{\max} + 1, b_{\max} + 2\} \setminus \{b\},\ b_{\max} \leftarrow b_{\max} + 2$
    $I_{b_{\max}+1} \leftarrow (I_b(1), I_b(1) + s - 1),\ B_{b_{\max}+1,0} \leftarrow B_s^+,\ l_{b_{\max}+1} \leftarrow 0$
    $I_{b_{\max}+2} \leftarrow (I_b(1) + s, I_b(2)),\ B_{b_{\max}+2,0} \leftarrow B_s^-,\ l_{b_{\max}+2} \leftarrow 0$
  end if
end if

Online Learning to Rank

- Any item $d \in B_{b,l}$ in stage l is explored n_l times where

$$n_l = \lceil 2^{2l+4} \log T \rceil .$$

- At the end of the stage, the probability of clicking on item d is estimated as

$$\hat{c}_{b,l}(d) = c_{b,l}(d)/n_l .$$

- KL-UCB upper and lower confidence bounds are

$$U_{b,l}(d) \leftarrow \underset{q \in [\hat{c}_{b,l}(d),1]}{\operatorname{argmax}} \left\{ D_{\mathrm{KL}}(\hat{c}_{b,l}(d) || q) \leq \frac{\log T + 2 \log \log T}{n_l} \right\},$$

$$L_{b,l}(d) \leftarrow \underset{q \in [0,\hat{c}_{b,l}(d)]}{\operatorname{argmin}} \left\{ D_{\mathrm{KL}}(\hat{c}_{b,l}(d) || q) \leq \frac{\log T + 2 \log \log T}{n_l} \right\},$$

where $D_{\mathrm{KL}}(p||q)$ denotes the *Kullback-Leibler divergence* between Bernoulli random variables with means p and q.

Experiments

- *Yandex* dataset
  - A dataset of 35M search sessions, each of which may contain multiple search queries
  - Each query is associated with displayed documents at positions 1 to 10 and their clicks.
- 60 frequent search queries are selected.
- For each query, CM and PBM are learned using PyClick.
- For each query, the goal is to rerank $L = 10$ most attrative items with the objective of maximizing the expected number of clicks at the first $K = 5$ positions.

Experiments

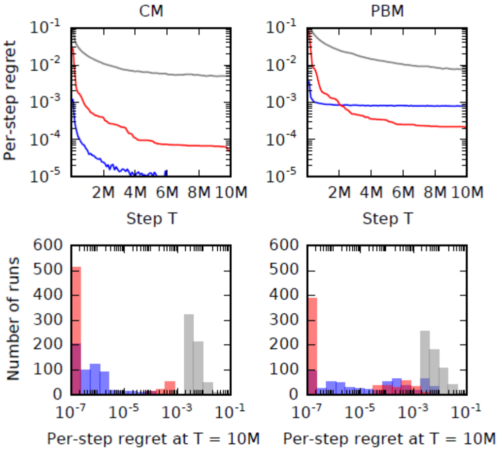

Figure 2. The comparison of `BatchRank` (red), `CascadeKL-UCB` (blue), and `RankedExp3` (gray) in the CM and PBM. In the top plots, we report the per-step regret as a function of time T, averaged over 60 queries and 10 runs per query. In the bottom plots, we show the distribution of the regret at $T = 10\text{M}$.