Pessimistic Off-Policy Optimization for Learning to Rank

Matej Cieľ, Branislav Kveton and Michal Kompan

aBrno University of Technology
bKempelen Institute of Intelligent Technologies
cAmazon

Abstract. Off-policy learning is a framework for optimizing policies without deploying them, using data collected by another policy. In recommender systems, this is especially challenging due to the imbalance in logged data: some items are recommended and thus logged more frequently than others. This is further perpetuated when recommending a list of items, as the action space is combinatorial. To address this challenge, we study pessimistic off-policy optimization for learning to rank. The key idea is to compute lower confidence bounds on parameters of click models and then return the list with the highest pessimistic estimate of its value. This approach is computationally efficient, and we analyze it. We study its Bayesian and frequentist variants and overcome the limitation of unknown prior by incorporating empirical Bayes. To show the empirical effectiveness of our approach, we compare it to off-policy optimizers that use inverse propensity scores or neglect uncertainty. Our approach outperforms all baselines and is both robust and general.

1 Introduction

Off-policy optimization can be used to learn better policies when the deployment and testing of sub-optimal solutions is costly, such as in recommender systems [12]. Despite the obvious benefits, off-policy optimization is often impeded by the feedback loop, where the currently deployed policy influences future training data [16]. This bias in data is one of the main challenges in off-policy optimization.

Several unbiased approaches exist to learn from biased data. Inverse propensity scoring (IPS), which re-weights observations with importance weights [14] to estimate a policy value, is popular. This so-called off-policy evaluation is often used in off-policy optimization, finding the policy with the highest estimated value [16]. While IPS is popular in practice [2], it has variance issues that compound at scale, which may prevent a successful deployment [8]. An important scenario where IPS has a high variance is recommending a ranked list of items. In this case, the action space is combinatorial, as the number of ranked lists, which represent actions, is exponential in the length of the lists.

Therefore, in real-world ranking problems (for example, news, web search, and e-commerce), model-based methods often outperform IPS methods [17]. The model-based methods rely on an explicit model of the reward conditioned on a context-action pair, such as the probability of a user clicking on a recommendation [10]. A prevalent approach to fitting model parameters, maximum likelihood estimation (MLE), is impacted by non-uniform data collection. As an example, consider choosing between two restaurants where the first has an average rating of 4.8, estimated from a thousand reviews, and the second has an average rating of 5, estimated from five reviews, and the second choice is more robust.

In our work, we account for the uncertainty caused by an unevenly explored action space by applying pessimism to reward models of action-context pairs for learning to rank. The challenge is to design lower confidence bounds that hold jointly for all lists as the number of unique lists grows exponentially with the list length. A naive application of existing pessimistic methods to each unique list is sample inefficient. Also, user behavior signals are often biased for higher-ranked items and can only be collected on items that users actually see. The main contributions of our paper are:

- We propose lower confidence bounds (LCBs) on parameters of model-based approaches in learning to rank and derive error bounds for acting on them in off-policy optimization.
- We study both Bayesian and frequentist approaches to estimating LCBs, including an empirical estimation of the prior, as it is often unknown in practice.
- We conduct extensive experiments that show the superiority of the proposed methods compared to IPS and MLE policies on four real-world learning to rank datasets with a large action space.

2 Related Work

Off-Policy Optimization: One popular approach to learning from bandit feedback is to employ the empirical risk minimization principle with IPS estimators [22, 2, 36]. Another popular choice is model-based methods [7]. These approaches learn a reward model for specific context-action pairs, which is then used to derive an optimal policy. Due to model misspecification, model-based methods are typically biased but have more favorable variance properties than IPS [16]. Variance issues of IPS estimators are further perpetuated in learning to rank since the action space is exponentially large.

Counterfactual Learning to Rank: Learning to rank models are often trained from user behavior data [19]. However, implicit feedback, such as user clicks, is noisy and affected by various biases [20]. Many studies have explored how to extract unbiased relevance signals from biased clicks. One approach is modeling how the user examines a list using click models [3, 4]. While IPS estimators for various click models have been studied in the past [28], the key assumption was that the value of a list is linear in the contributions of individual items in it. IPS estimators are unbiased, and their variance can be controlled in various ways [23, 36, 37], such as clipping of the
propensity weights [15]. Nevertheless, they do not model non-linear reward models.

Model-based methods can model non-linearity. Despite this, prior works on counterfactual learning to rank focused mainly on linear estimators [28, 38, 22]. More recently, a non-linear doubly-robust method for the cascade model has been proposed [25]. These methods suffer from biases in unexplored parts of the action space. They can be used for optimization but suffer from an overly optimistic estimation, a phenomenon known as the optimiser’s curse [34]. Our proposed method works with both linear and non-linear click models while alleviating the optimiser’s curse.

Pessimistic Off-Policy Optimization: While off-policy methods learn from data collected by a different policy, on-policy methods learn from the data that they collect. In online learning, the policy needs to balance the immediate rewards of actions and their information gain [30]. A common approach is being optimistic with respect to the rewards of actions, and upper confidence bounds (UCBs) have proved to be effective [27].

In the offline setting, new data cannot be collected, and a common approach is to act pessimistically to be robust. Prior works [16, 13] derived pessimistic Bayesian LCBS for reward models and showed that this can lead to major gains. Principled Bayesian methods can be used to obtain closed-form expressions, but this requires knowing the prior and is restricted to specific model classes [16, 3, 27].

This is the first work that applies pessimism to learning reward functions for ranking. While pessimism has been popular in offline reinforcement learning [41], it was applied only to unstructured action spaces before [16]. We extend this work from pointwise to listwise pessimism and study multiple approaches for constructing pessimistic estimates.

3 Setting
We start by introducing our setting. Specifically, we formally define a ranked list, how a user interacts with it, and how the data for off-policy optimization are collected.

We consider the following general model of a user interacting with a ranked list of items. Let \( \mathcal{E} \) be a ground set of items, such as all web pages or movies that can be recommended. Let \( \Pi_K(\mathcal{E}) \) be the set of all lists of length \( K \) over items \( \mathcal{E} \). A user is recommended a ranked list of items. We denote a ranked list with \( K \) items by \( A = (a_1, \ldots, a_K) \in \Pi_K(\mathcal{E}) \), where \( a_k \in \mathcal{E} \) is the item at position \( k \).

The user clicks on items in the list, and we observe click indicators on all positions \( Y = (Y_1, \ldots, Y_K) \), where \( Y_k \in \{0, 1\} \) is the click indicator on position \( k \). The list is chosen as a function of context \( X \in \mathcal{X} \), where \( X \) can be a user profile or a search query coming from a set of contexts \( \mathcal{X} \).

A ranking policy \( \pi(\cdot \mid X) \) is a conditional probability distribution over lists given context \( X \). It interacts with users for \( n \) rounds indexed by \( t \in [n] \). In round \( t \), it selects a list \( A_t \sim \pi(\cdot \mid X_t) \), where \( A_t = (a_{t1}, \ldots, a_{tK}) \in \Pi_K(\mathcal{E}) \) and \( X_t \) is the context in round \( t \). After that, it observes clicks \( Y_t = (Y_{t1}, \ldots, Y_{tK}) \) on all recommended items in the list. All interactions are recorded in a logged dataset \( D = \{(X_t, A_t, Y_t)\}_{t=1}^n \). The policy that collects \( D \) is called the logging policy, and we denote it by \( \pi_0 \).

Our goal is to find a policy that recommends the optimal list in every context. The optimal list in context \( X \) is defined as
\[
A_{\star, X} = \arg\max_{A \in \Pi_K(\mathcal{E})} V(A, X),
\]
where \( V(A, X) \) is the value of list \( A \) in context \( X \). This can be the expected number of clicks or the probability of observing a click.

Algorithm 1 Conservative off-policy optimization.

| Inputs: Logged dataset \( D \) |
| for \( X \in \mathcal{X} \) do |
| \( \hat{A}_X \leftarrow \arg\max_{A \in \Pi_K(\mathcal{E})} L(A, X) \) |
| end for |
| Output: \( \hat{A} = (\hat{A}_X)_{X \in \mathcal{X}} \) |

The definition of \( V \) depends on the chosen user interaction model. We present several of them in Section 5.

4 Pessimistic Optimization
Suppose that we want to find list \( A_{\star, X} \) in (1) but \( V(A, X) \) is unknown. Then, the most straightforward approach is to estimate it and choose the best list according to the estimate. As an example, let \( \hat{V}(A, X) \) be the maximum likelihood estimate (MLE) of \( V(A, X) \). Then the best empirically-estimated list in context \( X \) is
\[
\hat{A}_X = \arg\max_{A \in \Pi_K(\mathcal{E})} \hat{V}(A, X).
\]

This solution is problematic when \( \hat{V} \) estimates \( V \) poorly. The reason is that we may choose a list with a high estimated value \( \hat{V}(\hat{A}_X, X) \) but low actual value \( V(\hat{A}_X, X) \).

To account for uncertainty, prior works in bandits and reinforcement learning designed pessimistic lower confidence bounds (LCB) and acted on them [18]. We adopt the same design principle in our proposed method (Algorithm 1). At a high level, the algorithm first computes LCBS \( L(X, A) \) for all action-context pairs \( (A, X) \). The lower confidence bound satisfies \( L(A, X) \leq V(A, X) \) with a high probability. Then it takes an action \( \hat{A}_X \) with the highest lower confidence bound \( L(\cdot, X) \) in each context \( X \). In Sections 5 and 6, we show how to design LCBS for entire lists of items efficiently. These LCBS, and our subsequent analysis in Section 7, are our main technical contributions.

Lower confidence bounds are beneficial when \( \hat{V} \) does not approximate \( V \) uniformly well. Specifically, suppose that \( \hat{V} \) approximates \( V \) better around optimal solutions \( A_{\star, X} \). This is common in practice because deployed logging policies \( \pi_0 \) are already optimized to select high-value items. Then, low-value items can only be chosen if the LCBS of high-value items are low. This cannot happen because the high-value items are logged frequently; and thus their estimated mean values are high and their confidence intervals are tight.

As a concrete example, consider two lists of recommended items. The first list contains items with an estimated click-through rate (CTR) of 1, but all of them were recommended only once. The other list contains items with an estimated CTR of 0.5, but those items are popular and were recommended a thousand times. Off-policy optimization with the MLE estimator would choose the first list, whose estimated value is high, but the actual value may be low. Off-policy optimization with LCBS would choose the other list since its estimated value is reasonably high but more certain.

5 Structured Pessimism
In this section, we construct lower confidence bounds for lists. The main challenge is how to establish useful LCBS for all lists jointly since there can be exponentially many lists. To do that, we rely on user-interaction models with ranked lists, the so-called click models [4]. The models allow us to construct LCBS for the whole list by decomposing them into LCBS of items in it.
To illustrate the generality of our approach, we study three popular click models. To simplify notation, we assume that the context $X$ is fixed in this section and drop it from all terms. In each click model, the relevance of item $a \in \mathcal{E}$ is represented by its attraction probability $\theta_a \in [0, 1]$. This is the probability that the item is clicked, given that it is examined. In each model, we show that when we have LCBs for each $\theta_a$, we have LCBs for all lists $A$, trivially by the union bound.

### 5.1 Cascade Model

The cascade model (CM) [32, 5] assumes that a user examines items in a list from top to bottom until they find a relevant item and click on it [4]. Item $a_k$ at position $k$ is examined if and only if item $a_{k-1}$ is examined but not clicked. The first position is always examined. Specifically, the item is clicked only if its position is examined and the item is attractive.

Since at most one item is clicked, a natural choice for the value of list $A$ is the probability of a click defined as

$$V_{\text{CM}}(A) = 1 - \prod_{k=1}^{K} (1 - \theta_{a_k}), \tag{3}$$

where $\theta_a \in [0, 1]$ denotes the attraction probability of item $a \in \mathcal{E}$. To stress that the above value is for a specific model, the CM, in this case, we write $V_{\text{CM}}$. The optimal list $A^*$ contains $K$ items with the highest attraction probabilities [26].

To establish LCBs for all lists, we need LCBs for all model parameters. In the CM, the value of a list depends only on the attraction probabilities of its items. Let $L(a)$ be the LCB on the attraction probability of item $a$, where $\theta_a \geq L(a)$ holds with probability at least $1 - \delta$. Then, for all lists $A$ jointly, the LCB

$$L_{\text{CM}}(A) = 1 - \prod_{k=1}^{K} (1 - L(a_k)) \leq 1 - \prod_{k=1}^{K} (1 - \theta_{a_k})$$

holds with probability at least $1 - \delta |E|$, by the union bound over all items. The above inequality holds because we have a lower bound on each term in the product.

### 5.2 Dependent-Click Model

The dependent-click model (DCM) [9] extends the CM to multiple clicks. This model assumes that after a user clicks on an item, they may continue examining items at lower positions in the list. Specifically, the probability that the user continues to explore after a click at position $k \in [K]$ is $\lambda_k \in [0, 1]$. A natural choice for the value of list $A$ in the DCM is the probability of a satisfactory click, a click upon which the user leaves satisfied. This can be formally written as

$$V_{\text{DCM}}(A) = 1 - \prod_{k=1}^{K} (1 - (1 - \lambda_k)\theta_{a_k}), \tag{4}$$

where $\theta_a \in [0, 1]$ denotes the attraction probability of item $a \in \mathcal{E}$, identically to Section 5.1. The optimal list $A^*$ contains $K$ items with the highest attraction probabilities, where the $k$-th most attractive item is placed at the position with the $k$-th highest $p_k$. Let $L(a)$ be defined as in Section 5.1. Then, for all lists $A$ jointly, the LCB

$$L_{\text{DCM}}(A) = 1 - \prod_{k=1}^{K} (1 - (1 - \lambda_k)L(a_k)) \leq 1 - \prod_{k=1}^{K} (1 - \lambda_k)\theta_{a_k}$$

holds with probability at least $1 - \delta |E|$, by the union bound over all items. We assume that the position examination probabilities $p_k$ are known, similarly to the DCM (Section 5.2).

### 6 Lower Confidence Bounds on Attraction Probabilities

In this section, we construct LCBs for attraction probabilities $\theta_a$ of individual items in Section 5. Note that they are means of Bernoulli random variables, which we use in our derivations. We consider two types of LCBs: Bayesian and frequentist. The Bayesian bounds assume that the attraction probabilities are drawn i.i.d. from a known prior distribution. The frequentist bounds make no assumption on the distribution of the attraction probabilities. The Bayesian bounds are more practical when the prior is available, while the frequentist bounds are more robust due to fewer modeling assumptions. Our bounds are derived independently for each context $X$. To simplify notation, we drop it in the derivations in this section.

#### 6.1 Bayesian Lower Confidence Bounds

Let $\theta_a \in [0, 1]$ be the mean of a Bernoulli random variable representing the attraction probability of item $a$. Let $Y_1, \ldots, Y_n$ be $n$ i.i.d. observations of $\theta_a$. Let the number of positive and negative observations be $n_{a,+}$ and $n_{a,-}$, respectively. We estimate $n_{a,+}$ and $n_{a,-}$ for each click model in Appendix A. In the Bayesian setting, we make an assumption that $\theta_a \sim \text{Beta}(\alpha, \beta)$. Thus $\theta_a \mid Y_1, \ldots, Y_n \sim \text{Beta}(\alpha + n_{a,+}, \beta + n_{a,-})$ [1] and a lower confidence bound on $\theta_a$ that holds with probability at least $1 - \delta$ is $L(a) = \max \left\{ \ell \in [0, 1] : \int_{z=0}^{\ell} \text{Beta}(z; \alpha + n_{a,+}, \beta + n_{a,-}) \, dz \leq \frac{\delta}{2} \right\}$.\tag{6}

According to (6), $L(a)$ is the largest value such that the probability of $\theta_a \leq L(a)$ is at most $\frac{\delta}{2}$ [1]. We note that $L(a)$ is a quantile of a probability density.
6.2 Frequentist Lower Confidence Bounds

If the prior of \( \theta_a \) is unknown or poorly estimated, Bayesian estimates could be biased. In this case, Hoeffding’s inequality [40] would be preferred, as it provides a confidence interval for any random variable on \([0, 1]\). Specifically, let \( \theta_a \) be any value in \([0, 1]\), and all other quantities be defined as in the Bayesian estimator. Then a \( 1 - \delta \) lower confidence bound on \( \theta_a \) is

\[
L(a) = n_{a,+}/n_a - \sqrt{\log(1/\delta)/(2n_a)},
\]

where \( n_{a,+}/n_a \) is the MLE, \( n_a = n_{a,+} + n_{a,-} \), and event \( \theta_a \leq L(a) \) occurs with probability at most \( \delta \) [40].

6.3 Prior Estimation

One shortcoming of Bayesian methods is that the prior is often unknown. To address this, we show how to estimate it using empirical Bayes [29], which can be implemented for attraction probabilities as follows. For any \( a \in E \), let \( \theta_a \sim \text{Beta}(\alpha, \beta) \) be the mean of a Bernoulli random variable, which is drawn i.i.d. from the unknown prior \( \text{Beta}(\alpha, \beta) \). Let \( N_{a,+} \) and \( N_{a,-} \) be the random variables that denote the number of positive and negative observations, respectively, of \( \theta_a \). Let \( n_{a,+} \) and \( n_{a,-} \) be their observed values, and \( n_a = n_{a,+} + n_{a,-} \). Then, the likelihood of the observations for any fixed \( \alpha \) and \( \beta \) is

\[
L(\alpha, \beta) = \prod_{a \in E} (N_{a,+} = n_{a,+}, N_{a,-} = n_{a,-} | \alpha, \beta)
= \prod_{a \in E} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_0^1 z^{\alpha+n_{a,+}-1}(1-z)^{\beta+n_{a,-}-1} \, dz
= \prod_{a \in E} \frac{\Gamma(\alpha + \beta + n_a)}{\Gamma(\alpha)\Gamma(\beta)\Gamma(\alpha + \beta + n_a)}.\]

The last equality follows from the fact that

\[
\int_0^1 \frac{\Gamma(\alpha + \beta + n_a)}{\Gamma(\alpha + n_{a,+})\Gamma(\beta + n_{a,-})} \, dz = 1.
\]

Empirical Bayes [29] is a statistical procedure for finding \((\alpha, \beta)\) that maximize \(L(\alpha, \beta)\). To find them, we search on a grid. Specifically, let \( G = [m] \) for some integer \( m > 0 \). Then we search over all \((\alpha, \beta)\) in \( G^2 \). The grid search is feasible since the parameter space has only 2 dimensions.

7 Analysis

The analysis is structured as follows. First, we derive confidence intervals for items and lists. Then, we show how the error of acting with respect to LCBs is bounded. Finally, we discuss how different choices of \( \alpha_0 \) affect the error in Theorem 3. All proofs are presented in Appendix B. We conduct a frequentist analysis based on the confidence intervals in Section 6.2. A similar analysis could be done for the Bayesian setting (Section 6.1). The analysis is exact for the CM and DCM. For the PBM, it is approximate, and we provide details in Appendix A.

Let \( \theta_{a,X} \in [0, 1] \) be the attraction probability of item \( a \in E \) in context \( X \in X \). Let \( \hat{\theta}_{a,X} = \frac{n_{a,X} + 1}{n_{a,X} + n_{a,X}} \) be its empirical estimate in (7), where \( n_{a,X} \) is the number of clicks on item \( a \) in context \( X \) and \( n_{a,X} \) is the number of times that item \( a \) is observed in context \( X \). Then, we get the following concentration bound for all items.

**Lemma 1 (Concentration of item estimates).** Let

\[
c(a, X) = \sqrt{\log(1/\delta)/(2n_{a,X})}.
\]

Then for any item \( a \in E \) and context \( X \in X \), \( |\hat{\theta}_{a,X} - \theta_{a,X}| \leq c(a, X) \) holds with probability at least \( 1 - \delta \).

Let \( V(A, X) \) be the value of list \( A \in \Pi_K(E) \) in context \( X \in X \), using the unknown attraction probabilities \( \theta_{a,X} \). Let \( \hat{V}(A, X) \) be its estimate using \( \hat{\theta}_{a,X} \), for any click model in Section 5. Then, we get the following concentration bound for all lists.

**Lemma 2 (Concentration of list estimates).** Let

\[
c(A, X) = \sum_{a \in A} \sqrt{\log(|E| |X| / \delta) / 2n_{a,X}}.
\]

Then for any list \( A \in \Pi_K(E) \) and context \( X \in X \), and any click model in Section 5, \(|\hat{V}(A, X) - V(A, X)| \leq c(A, X) \) holds with probability at least \( 1 - \delta \), jointly over all \( A \) and \( X \).

Now we show how the error due to acting pessimistically does not depend on the uncertainty of the chosen list but on the confidence interval width of optimal list \( c(A^*, X) \). This is desirable when the logging policy already performs well (Section 4).

**Theorem 3 (Error of acting pessimistically).** Let \( A_{a,X} \) and \( \tilde{A} \) be defined as in (1) and Algorithm 1, respectively. Let \( L(A, X) = \hat{V}(A, X) - c(A, X) \) be a high-probability lower bound on the value of list \( A \) in context \( X \), where all quantities are defined as in Lemma 2. Then, for any context \( X \), the error of acting with respect to a lower confidence bound satisfies

\[
V(A_{a,X}, X) - V(\tilde{A}, X) \leq 2c(A_{a,X}, X)
\]

with probability at least \( 1 - \delta \), jointly over all \( X \).

Theorem 3 shows that our error depends on the number of observations of items in the optimal list \( a \in A_{a,X} \). To illustrate how it depends on the logging policy \( \pi_0 \), fix context \( X \in X \) and let \( n_X \) be the number of logged lists in context \( X \). We discuss two cases.

Suppose that \( \pi_0 \) is uniform, and thus each item \( a \in E \) is placed at the first position with probability \( 1/|E| \). Moreover, suppose that the first position is examined with a high probability. This happens with probability 1 in the CM (Section 5.1) and DCM (Section 5.2), and with a high probability in the PBM (Section 5.3) when \( p_1 \) is high. Then \( n_{a,X} = \Omega(n_X / |E|) \) as \( n_X \to \infty \) and the error bound in Theorem 3 becomes \( \tilde{O}(K \sqrt{|E|}/n_X) \), where \( \tilde{O} \) and \( \Omega \) are big O notations up to logarithmic factors. The bound is independent of the number of lists \( |\Pi_K(E)| \), which is exponentially large.

Now, suppose that the logging policy is near optimal. One way of formalizing it is that each item \( a \in A_{a,X} \) is placed at the first position with probability \( 1/K \). Then, under the same assumptions as in the earlier discussion, \( n_{a,X} = \tilde{O}(n_X / K) \) as \( n_X \to \infty \) and the error bound in Theorem 3 becomes \( \tilde{O}(K \sqrt{n_X}/n_X) \). This improves by a factor of \(|E|/K \) upon the earlier discussed bound.

8 Experiments

We conduct extensive experiments where we compare our policies to four baselines: MLE, IPS [33], structured item-position IPS [28], and pseudo-inverse estimator [38]. We call our method LCB because it optimizes lower confidence bounds.
8.1 Experimental Setup

We use the Yandex dataset [43] for the first three experiments. We treat each query as a different context \(X\), perform the computations separately, and then average the results. Due to a huge position bias in the dataset, most clicks occur at the first positions; we only keep the first 4 positions in each list and discard the rest, as done in prior works [28]. We observe improvements without this preprocessing step but they are less pronounced.

All methods are evaluated as follows. We first fit a click model (Section 5) to a dataset. Because of that, we know the optimal list \(A^*_X\) in each context \(X\) under that model. We select a specific set of queries from the dataset for each experiment. Then, for each query, we select uniformly at random items from that query and generate clicks based on the fitted click model. We repeat this \(n\) times and obtain a logged dataset \(D = \{(X_t, A_t, Y_t)\}_{t=1}^n\), where \(X_t\) and \(A_t\) are from the original dataset, and \(Y_t\) is generated by the click model. Unless stated, \(n = 1000\). We apply off-policy methods to \(D\) to find the most valuable lists \(\hat{A}_X\). We evaluate these lists against the true optimal lists \(A^*_X\) using error \(E_{X_t}[V(A^*_X, X) - V(\hat{A}_X, X)]\). We estimate the logging policy \(p_X\) from \(D\). We repeat each experiment 500 times and report all mean values with standard errors (shaded areas around lines in the plots).

We experiment with both Bayesian and frequentist lower confidence bounds in Section 6. They hold with probability \(1 - \delta\), where \(\delta \in [0.05, 1]\) is a tunable parameter representing the width of the confidence interval. The estimation of our model parameters is detailed in Appendix A. Since Bayesian methods depend on the prior, we also evaluate empirical Bayes for learning the prior (Section 6.3) with grid \(G = \{2^{i-1}\}_{i=1}^{10}\).

8.2 Baselines

The first baseline is the best list under the same click model as in our method but with MLE-estimated parameters. We also use the following baselines from prior works [15, 28, 38].

**IPS:** We implement an estimator using propensity weights, where the whole list is a unique action. We compute the propensity weights separately for each query. We also add a tunable clipping parameter \(M\) [15]. IPS optimizer then selects \(A_X\) that maximizes

\[
\hat{V}_{\text{IPS}}(A,X) = \min_{t \in T_X} \{M, \frac{\mathbb{1}(A_t = A)}{p_{A,X}} Y_t\},
\]

where we estimate propensities \(p_{A,X} = \frac{\sum_{t \in T_X} \mathbb{1}(A_t = A)}{|T_X|}\) as the frequency of recommending list \(A_t\) and \(T_X\) is the set of all indices \(t \in [n]\) such that \(X_t = X\). The maximization of \(\hat{V}_{\text{IPS}}(A,X)\) is a linear program, where we search over all \(A \in \Pi_K(\mathcal{E})\) [35].

**Item-Position IPS:** Similarly to the IPS estimator, we implement a structured IPS estimator using linearity of the item-position model [28], where the expected value of a list is the sum of attraction probabilities of its item-position entries. The list value is

\[
\hat{V}_{\text{IP-IPS}}(A,X) = \sum_{t \in T_X} \sum_{k=1}^K \min_{a \in A} \left\{M, \frac{\mathbb{1}(a_{t,k} = a)}{p_{a,k,X}} Y_{t,k}\right\},
\]

where \(p_{a,k,X} = \frac{\sum_{t \in T_X} \mathbb{1}(a_{t,k} = a)}{|T_X|}\) and \(T_X\) is the set of all \(t \in [n]\) such that \(X_t = X\). To maximize \(\hat{V}_{\text{IP-IPS}}(A,X)\), we select an item with the highest attraction probability for each position \(k \in [K]\).
Pseudo-Inverse Estimator (PI): We also compare to the pseudo-inverse estimator [38], which assumes that the value of a list is the sum of individual items in it. The context-specific weight vector $\phi_X$ can then be learned in a closed form as

$$\phi_X = \left( E_{\pi_0} \left[ I_A^T X \right] \right)^{-1} E_{\pi_0} \left[ Y I_A | X \right],$$

(11)

where $I_A \in \{0, 1\}^{K(F)}$ is a list indicator vector whose entries indicate which item is at which position and $E_{\pi_0}$ is an expectation over $A \sim \pi_0(\cdot | X)$. We denote by $Y$ the sum of logged clicks on the list $A$ and by $M$ the Moore-Penrose pseudo-inverse of a matrix $M$.

8.3 Yandex Results

The experiments are organized as follows. First, we compare LCBs to all baselines on frequent queries while we vary the confidence interval width represented by parameter $\delta$. Second, we compare LCBs to MLE while automatically learning parameter $\delta$ from data. Finally, we study the robustness of model-based LCB estimators to model misspecification. We refer readers to Appendix C for experiments on less frequent queries, where we show that LCBs work well even with less data. We also experiment with the hyperparameters of empirical Bayes from Section 6.3. Most plots are reported as a function of $\delta$ because it is a tunable parameter of our method. We map the clipping parameter $M$ to $\delta$ as follows.

| $\delta$ | .05 | .1 | .15 | .2 | .25 | .35 | .45 | .5 |
|---------|-----|----|-----|----|-----|-----|-----|----|
| $M$     | 1   | 5  | 10  | 50 | 100 | 300 | 500 | 600|
| $\delta$ | .55 | .65 | .75 | .8 | .85 | .9 | .95 | 1  |
| $M$     | 700 | 900 | 1100 | 1200 | 1300 | 1400 | 1500 | $\infty$ |

Top 10 Queries: We start with evaluating all estimators on 10 most frequent queries in the Yandex dataset. Results in Figure 1 show improvements when using LCBs for all models. Specifically, for almost any $\delta$ in all models, the error is lower than using MLE. When optimizing non-linear list values, such as those in CM and DCM, we outperform all the baselines that assume linearity. In PBM, where the list value is linear, the baselines can perform similarly to the LCB estimators. We observe that the empirical estimation of the prior improves upon an uninformative $\text{Beta}(1, 1)$ prior.

More Realistic Comparison to MLE: MLE is common in practice and does not have a hyper-parameter $\delta$ to tune, unlike our method. To show that our approach can beat the MLE in a realistic setting, we estimate $\delta$ on past data and then apply it to future data. We apply the evaluation protocol from the Top 10 Queries experiment on the first 5 days of data with fixed sample size $n = 1000$ for each query. We select $\delta$ that minimizes the Bayesian LCB error, which is $\delta = 0.2$. We fix $\delta = 0.2$ and apply the evaluation protocol from the Top 10 Queries experiment on the last 23 days. We report the difference between MLE and Bayesian LCB errors. This is repeated 500 times while varying logged sample size $n \in [50, 500, 000]$. Figure 2 shows that the largest improvements are at the sample size $100,000$. This implies that LCBs have a sweet spot where they work the best. We observe smaller improvements for smaller sample sizes because the uncertainty is too high to allow effective modeling. On the other hand, when the sample size is large, the uncertainty is low everywhere and it is not needed to model it.

Robustness to Model Misspecification: Now, we examine how the estimators behave when the model is misspecified. In the Top 10 Queries experiment, we observe that the baselines with linearity assumptions do not perform well in non-linear models, such as CM or DCM, but they perform well when the value of a list is the sum of clicks, such as in PBM. We fit PBM and use it to generate a logged dataset. Then, we use DCM to learn the attraction probabilities for MLE and LCB methods. We also examine the opposite scenario, using DCM as a ground truth model and estimating attraction probabilities with PBM. This does not impact IPS and PI baselines. Our results are reported in Figure 3. In the left plot, we use a non-linear model to fit linear rewards. As a result, MLE and LCB methods are misspecified. Other baselines that assume linearity perform better in this setup. Nevertheless, Bayesian LCBs still have a 50% lower error than MLE. In the right plot, the reward is non-linear, and all methods (except IPS) assume linearity in item-level rewards. Here, MLE is comparable to other baselines, and LCBs consistently outperform all baselines. In summary, we show that LCBs are relatively robust to model misspecification, and definitely much more than MLE.

8.4 Results on Other Datasets

We validated the results on other popular datasets, namely Yahoo! Websope [42], Istella [6], and MSLR-WEB [31]. These datasets do not contain clicks, only human-labeled query-document relevance scores, with $\text{score}(a) \in \{0, 1, 2, 3, 4\}$ for item $a$ ranked from 0 (not relevant) to 4 (highly relevant). We follow a standard procedure to generate clicks by mapping relevance scores to attraction probabilities based on the navigational user model [Table 2, 11].

| $\text{score}(a)$ | 0 | 1 | 2 | 3 | 4 |
|-------------------|---|---|---|---|---|
| $\theta_a$        | 0.05 | 0.1 | 0.2 | 0.3 | 0.4 | 0.8 |

In the PBM, we define position examination probabilities based on an eye-tracking experiment [21]. In the DCM, at position $k \in [K]$, \(\lambda_k = 1 - \exp(-k + 0.5)/0.5\). We randomly select 5000 queries $q$. Each query has its own set of labeled documents denoted by $E_q$. To get a logged dataset, we sample 100 lists of length $K = 4$ from labeled documents for each query from a Dirichlet distribution with parameters $\alpha = (\theta_a)_{a \in E_q}$. We use the same evaluation protocol as in Section 8.1. Figures 4 to 6 validate our earlier findings. In summary, LCBs outperform MLE and other baselines for most $\delta$ values and provide major improvements. We observed similar results for other sample sizes and list lengths.

9 Conclusions

We study pessimistic off-policy optimization in learning to rank. The key idea is to design lower confidence bounds (LCBs) for values of lists through LCBs of items in them. We prove that the error due to choosing the best list in our model is polynomial in the number of items in the list, as opposed to polynomial in the number of lists, which is exponential in the length of the list. We also apply LCBs to non-linear objectives, such as the CM in (3) and DCM in (4), based on their linearization. This is the first paper in off-policy learning that approximates the CM when its approximation was applied and analyzed. Our approach outperforms MLE and IPS methods experimentally. Furthermore, we show that our LCBs are robust to model misspecification and perform better with almost any confidence interval width. We also show how to estimate the prior with empirical Bayes when it is unknown.

In the PBM, we define position examination probabilities based on an eye-tracking experiment [21]. In the DCM, at position $k \in [K]$, \(\lambda_k = 1 - \exp(-k + 0.5)/0.5\). We randomly select 5000 queries $q$. Each query has its own set of labeled documents denoted by $E_q$. To get a logged dataset, we sample 100 lists of length $K = 4$ from labeled documents for each query from a Dirichlet distribution with parameters $\alpha = (\theta_a)_{a \in E_q}$. We use the same evaluation protocol as in Section 8.1. Figures 4 to 6 validate our earlier findings. In summary, LCBs outperform MLE and other baselines for most $\delta$ values and provide major improvements. We observed similar results for other sample sizes and list lengths.

9 Conclusions

We study pessimistic off-policy optimization in learning to rank. The key idea is to design lower confidence bounds (LCBs) for values of lists through LCBs of items in them. We prove that the error due to choosing the best list in our model is polynomial in the number of items in the list, as opposed to polynomial in the number of lists, which is exponential in the length of the list. We also apply LCBs to non-linear objectives, such as the CM in (3) and DCM in (4), based on their linearization. This is the first paper in off-policy learning where this approximation was applied and analyzed. Our approach outperforms MLE and IPS methods experimentally. Furthermore, we show that our LCBs are robust to model misspecification and perform better with almost any confidence interval width. We also show how to estimate the prior with empirical Bayes when it is unknown.
One natural future direction is to extend our work to any click model. This can be generally achieved with an ensemble of models, each trained on a different bootstrapped dataset. This, however, presents computational challenges, and further research is needed to address them. We also do not use theory-suggested confidence intervals in experiments because they are too conservative. Therefore, more data-dependent confidence intervals are needed. Finally, the focus of our work is on a large action space of all lists. We wanted this contribution to stand out and thus focus on tabular contexts. An extension to large context spaces should be possible, though.

Acknowledgements

This research was partially supported by DisAI, a project funded by the European Union under the Horizon Europe, GA No. 101079164, https://doi.org/10.3030/101079164.
[30] J. McInerney, B. Lacker, S. Hansen, K. Higley, H. Bouchard, A. Gruson, and R. Mehrotra. Explore, exploit, and explain: personalizing explainable recommendations with bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys ’18, pages 31–39, New York, NY, USA, Sept. 2018. Association for Computing Machinery. ISBN 978-1-4503-5901-6. doi: 10.1145/3240323.3240354. URL: https://doi.org/10.1145/3240323.3240354.

[31] T. Qin and T.-Y. Liu. Introducing LETOR 4.0 Datasets, June 2013. URL: http://arxiv.org/abs/1306.2597. arXiv:1306.2597 [cs].

[32] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th international conference on World Wide Web*, WWW ’07, pages 521–530, New York, NY, USA, May 2007. Association for Computing Machinery. ISBN 978-1-59593-654-7. doi: 10.1145/1242572.1242643. URL: https://doi.org/10.1145/1242572.1242643.

[33] J. M. Robins, A. Rotnitzky, and L. P. Zhao. Estimation of Regression Coefficients When Some Regressors are not Always Observed. *Journal of the American Statistical Association*, 89(427):846–866, Sept. 1994. ISSN 0162-1459. doi: 10.1080/01621459.1994.10476818. URL: https://doi.org/10.1080/01621459.1994.10476818. Publisher: Taylor & Francis. eprint: https://doi.org/10.1080/01621459.1994.10476818.

[34] J. E. Smith and R. L. Winkler. The Optimizer’s Curse: Skepticism and Postdecision Surprise in Decision Analysis. *Management Science*, 52 (3):311–322, Mar. 2006. ISSN 0025-1909. doi: 10.1287/mnsc.1050.0451. URL: https://pubsonline.informs.org/doi/abs/10.1287/mnsc.1050.0451. Publisher: INFORMS.

[35] A. Strehl, J. Langford, L. Li, and S. M. Kakade. Learning from Logged Implicit Exploration Data. In *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. URL: https://proceedings.neurips.cc/paper_files/paper/2010/hash/ef016e8c900fa5e5e57789ec22f2c9d0-Abstract.html.

[36] A. Swaminathan. *Counterfactual Evaluation And Learning From Logged User Feedback*. PhD thesis, Cornell University, Ithaca, NY, United States, May 2017. URL: https://ecommons.cornell.edu/handle/1813/51557.

[37] A. Swaminathan and T. Joachims. The Self-Normalized Estimator for Counterfactual Learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/hash/5352696a3397beb79f116f3a33991-Abstract.html.

[38] A. Takács and D. Tikk. Alternating least squares for personalized ranking. In *Proceedings of the sixth ACM conference on Recommender systems*, RecSys ’12, pages 83–90, New York, NY, USA, Sept. 2012. Association for Computing Machinery. ISBN 978-1-4503-1270-7. doi: 10.1145/2365952.2365972. URL: https://doi.org/10.1145/2365952.2365972.

[39] R. Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge University Press, Sept. 2018. ISBN 978-1-108-24454-1. Google-Books-ID: TalxDwAAQBAJ.

[40] T. Xie, C.-A. Cheng, N. Jiang, P. Mineiro, and A. Agarwal. Bellman-consistent Pessimism for Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 6683–6694. Curran Associates, Inc., 2021. URL: https://proceedings.neurips.cc/paper_files/paper/2021/hash/34f96c7c5d7063181da850c8ad25265a-Abstract.html.

[41] Yahoo! C14 - Yahoo! Learning to Rank Challenge, 2010. URL: http://webscope.sandbox.yahoo.com/.

[42] Yandex. *Yandex Personalized Web Search Challenge, 2013*. URL: https://www.kaggle.com/c/yandex-personalized-web-search-challenge.
A Learning Click Model Parameters

To simplify notation, we only consider a small finite number of contexts, such as the day of the week or user characteristics. For each such context, we estimate the attraction probabilities separately. In this section, we show the calculation of model parameters with respective applications of LCB to the three click models mentioned in Section 5. We assume that the context is fixed, and computations are done over each context separately; therefore, we define $T_X$ to be the set of all indices of $|n|$ such that $X_t = X, \forall t \in T_X$.

Cascade Model: The cascade model has only one type of parameters, attraction probabilities $\theta_a$, and these can be estimated from clicks, as the number of clicks over the number of examinations [26]. According to the cascade model assumptions, items are examined from the top until the final observed click at the lowest position and disregarding all items below. Observed impressions on each item $a$ in context $X$ and calculate either Bayesian, frequentist LCBs, and prior according to (6), (7), and Section 6.3.

Dependent-Click Model: When fitting the dependent-click model, we process the logged data according to the following assumption: examining items from the top until the final observed click at the lowest position and disregarding all items below. Observed impressions on each item $a$ in context $X$ are sampled from its unknown Bernoulli distribution $\text{Ber}(\theta_a)$. To model $\theta_a$, we collect the number of positive impressions $n_{a,X,+} = \sum_{t \in T_X} \sum_{k=1}^{K} \mathbb{1}\{a_{t,k} = a \land Y_{t,k} = 1\}$ (user examined and clicked) and the number of negative impressions $n_{a,X,-} = \sum_{t \in T_X} \sum_{k=1}^{K} \mathbb{1}\{a_{t,k} = a\} \{1-Y_{t,k}\}$ (examined, but did not click) for item $a$ in context $X$ and calculate either Bayesian, frequentist LCBs, and prior according to (6), (7), and Section 6.3.

Position-Based Model: To prove the bound for CM and DCM, we first show how the difference of two products with $K$ variables is bounded by the difference of their sums.

Lemma 4. Let $(a_k)_{k=1}^{K} \in [0, 1]^K$ and $(b_k)_{k=1}^{K} \in [0, 1]^K$. Then

$$\left| \prod_{k=1}^{K} a_k - \prod_{k=1}^{K} b_k \right| \leq \sum_{k=1}^{K} |a_k - b_k|.$$
Proof. We start with
\[
\prod_{k=1}^{K} a_k - \prod_{k=1}^{K} b_k = K \prod_{k=1}^{K} a_k - a_1 \prod_{k=2}^{K} b_k + a_1 \prod_{k=2}^{K} b_k - \prod_{k=1}^{K} b_k = a_1 \left( \prod_{k=2}^{K} a_k - \prod_{k=2}^{K} b_k \right) + (a_1 - b_1) \prod_{k=1}^{K} b_k
\]
\[
= \left( \sum_{k=1}^{K} a_k \right) \left( \prod_{k=1}^{K} b_k \right) - \left( \sum_{k=1}^{K} b_k \right) \left( \prod_{k=1}^{K} a_k \right),
\]
where (a) is by recursively applying the same argument to \( \prod_{k=1}^{K} a_k - \prod_{k=1}^{K} b_k \). Now, we apply the absolute value and get
\[
\left| \prod_{k=1}^{K} a_k - \prod_{k=1}^{K} b_k \right| = \left| \left( \sum_{k=1}^{K} a_k \right) \left( \prod_{k=1}^{K} b_k \right) - \left( \sum_{k=1}^{K} b_k \right) \left( \prod_{k=1}^{K} a_k \right) \right| \leq \sum_{k=1}^{K} |a_k - b_k|,
\]
since \( \prod_{k=1}^{K} a_k \in [0, 1] \) and \( \prod_{k=1}^{K} b_k \in [0, 1] \).

Now, we prove our main claim. For any context \( X \) and list \( A = (a_k)_{k \in [K]} \), we have from the definition of the CM and from the bound in Lemma 4 that
\[
\left| V_{cm}(A, X) - \hat{V}_{cm}(A, X) \right| = \left| 1 - \prod_{k=1}^{K} (1 - \theta_{a_k, X}) - 1 + \prod_{k=1}^{K} (1 - \theta_{a_k, X}) \right| = \left| \prod_{k=1}^{K} (1 - \theta_{a_k, X}) - \prod_{k=1}^{K} (1 - \theta_{a_k, X}) \right| \leq \sum_{k=1}^{K} \left| \theta_{a_k, X} - \theta_{a_k, X} \right| \leq \sum_{a \in A} \sqrt{\log(|X|)} |E| |X| / (2n_{a, X})
\]
The last inequality is under the assumption that (12) holds, which holds with probability at least \( 1 - \delta \).

**DCM:** For any context \( X \) and list \( A = (a_k)_{k \in [K]} \), we have from the definition of the DCM and from the bound in Lemma 4 that
\[
\left| V_{dcm}(A, X) - \hat{V}_{dcm}(A, X) \right| = \left| 1 - \prod_{k=1}^{K} (1 - (1 - \lambda_{a_k, X}) \theta_{a_k, X}) - 1 + \prod_{k=1}^{K} (1 - (1 - \lambda_{a_k, X}) \theta_{a_k, X}) \right| = \left| \prod_{k=1}^{K} (1 - (1 - \lambda_{a_k, X}) \theta_{a_k, X}) - \prod_{k=1}^{K} (1 - (1 - \lambda_{a_k, X}) \theta_{a_k, X}) \right| \leq \sum_{k=1}^{K} \left| \theta_{a_k, X} - \theta_{a_k, X} \right| \leq \sum_{a \in A} \sqrt{\log(|X|)} |E| |X| / (2n_{a, X})
\]
The first inequality holds as \( \lambda_{a_k, X} \in [0, 1] \), and the last inequality is under the assumption that (12) holds, which holds with probability at least \( 1 - \delta \). This completes the proof.

**Proof of Theorem 3:**

Proof. Let \( L(A, X) = \hat{V}(A, X) - c(A, X) \). Suppose that \( \hat{V}(A, X) - V(A, X) \leq c(A, X) \) holds for all \( A \) and \( X \). Note that this also implies \( L(A, X) \leq V(A, X) \). Then, for any \( X \),
\[
V(A_{+}, X) - V(\hat{A}_{X}, X) = V(A_{+}, X) - L(A_{+}, X) + L(A_{+}, X) - V(\hat{A}_{X}, X)
\]
\[
\leq V(A_{+}, X) - L(A_{+}, X) + L(A_{+}, X) - V(\hat{A}_{X}, X)
\]
\[
\leq V(A_{+}, X) - L(A_{+}, X) = V(A_{+}, X) - \hat{V}(A_{+}, X) + c(A_{+}, X)
\]
\[
\leq 2c(A_{+}, X).
\]
The first inequality holds because \( \hat{A}_{X} \) maximizes \( L(\cdot, X) \). The second inequality holds because \( L(\hat{A}_{X}, X) \leq V(\hat{A}_{X}, X) \). The last inequality holds because \( V(A_{+}, X) - \hat{V}(A_{+}, X) \leq c(A_{+}, X) \).

It remains to prove that \( \hat{V}(A, X) - V(A, X) \leq c(A, X) \) holds for all \( A \) and \( X \). By Lemma 2, this occurs with probability at least \( 1 - \delta \), jointly over all \( A \) and \( X \), for \( c(A, X) \) in Lemma 2. This completes the proof. \( \square \)
C Additional Experiments

Most Frequent Query: In Figure 7, we show the simplest case when using only the most frequent query to observe the effect of LCBs without possible skewing due to averaging over multiple queries. Results found in the most frequent query support those already discussed in Top 10 Queries.

Less Frequent Queries: We study how less frequent queries impact the performance of LCBs. We use the same setup as in the Top 10 Queries experiment and observe how the error changes as the number of queries increases. In Figure 8, we show comparable improvements to those of DCM in Figure 1, showing that LCBs perform well even in less frequent queries. We performed this experiment with CM and PBM as well and observed a similar behavior.

Cross-Validation Setting: In all of the previous experiments, we fit a model on logged data $D$, use that model to generate new clicks, and then learn a new model using those clicks. One can argue that data generated under the right model yields better results, and under the wrong model, LCBs do not hold. Therefore, we split logged data $D$ on the 23rd day to train and test the set and use the first 23 days to train model parameters. We randomly select 1000 samples for each query from the remaining 4 days to fit models using LCB while varying $\delta$. The rest of the setting is the same as in the Top 10 Queries experiment. The reason for this split is that using the remaining four days provides enough data in the top 10 queries to sample from. We tested other split ratios and observed that the results did not differ significantly. In Figure 9, we observe the Bayesian LCBs outperform all other baselines, and reasoning under uncertainty is still better than using MLE. Similarly to the previous experiments, we omit other linear baselines in CM and DCM plots as they perform considerably worse.

Figure 7: Error of the estimated lists $\hat{A}$ compared to optimal lists $A$, on the most frequent query.

Figure 8: Comparison of our methods to baselines on DCM and less frequent queries.

Figure 9: Comparison of our methods against other baselines in a setting where the fitted ground truth model does not generate clicks.
**Finding Prior:** When we estimated prior in the previous experiments, we did so on a grid \((\alpha, \beta) \in G^2, G = \{2^i\}_{i=1}^{10}\) and we observed significant improvements over fixed prior Beta(1, 1) mostly when \(\delta = 1\). In this experiment, the goal is to measure an improvement with the increasing grid size. We evaluate DCM using Bayesian LCBs and keep the rest of the setting the same as in the Top 10 Queries experiment. We then estimate prior from these queries by searching over grid \(G^2, G = \{2^i-1\}_{i=1}^{m}, m \in \{1, 2, 5, 10, 20\}\). In Figure 10, we see how the results get more robust against the \(\delta\) hyperparameter until \(|G| = 10\), and after that, the method still finds the same optimal prior. Therefore, in our case, it is sufficient to search over \(G = \{2^i\}_{i=1}^{10}\). The optimal values found on \(G\) are \(\theta \sim \text{Beta}(1, 8)\) and \(\lambda \sim \text{Beta}(1, 64)\).

![Figure 10: Empirical Bayes showing increasing performance with larger grid size. Red and purple lines are overlapping.](image-url)