Unbiased Learning-to-Rank with Biased Feedback

Thorsten Joachims	Adith Swaminathan	Tobias Schnabel
Cornell University, Ithaca, NY	Cornell University, Ithaca, NY	Cornell University, Ithaca, NY
tj@cs.cornell.edu	adith@cs.cornell.edu	tbs49@cornell.edu

ABSTRACT

Implicit feedback (e.g., clicks, dwell times, etc.) is an abundant source of data in human-interactive systems. While implicit feedback has many advantages (e.g., it is inexpensive to collect, user centric, and timely), its inherent biases are a key obstacle to its effective use. For example, position bias in search rankings strongly influences how many clicks a result receives, so that directly using click data as a training signal in Learning-to-Rank (LTR) methods yields sub-optimal results. To overcome this bias problem, we present a counterfactual inference framework that provides the theoretical basis for unbiased LTR via Empirical Risk Minimization despite biased data. Using this framework, we derive a Propensity-Weighted Ranking SVM for discriminative learning from implicit feedback, where click models take the role of the propensity estimator. In contrast to most conventional approaches to de-biasing the data using click models, this allows training of ranking functions even in settings where queries do not repeat. Beyond the theoretical support, we show empirically that the proposed learning method is highly effective in dealing with biases, that it is robust to noise and propensity model misspecification, and that it scales efficiently. We also demonstrate the real-world applicability of our approach on an operational search engine, where it substantially improves retrieval performance.

1. INTRODUCTION

Batch training of retrieval systems requires annotated test collections that take substantial effort and cost to amass. While economically feasible for Web Search, eliciting relevance annotations from experts is infeasible or impossible for most other ranking applications (e.g., personal collection search, intranet search). For these applications, implicit feedback from user behavior is an attractive source of data. Unfortunately, existing approaches for Learning-to-Rank (LTR) from implicit feedback – and clicks on search results in particular – have several limitations or drawbacks.

First, the naïve approach of treating a click/no-click as a positive/negative relevance judgment is severely biased. In particular, the order of presentation has a strong influence on where users click [11]. This presentation bias leads to an incomplete and skewed sample of relevance judgments that is far from uniform, thus leading to biased learning-to-rank.

Second, treating clicks as preferences between clicked and skipped documents has been found to be accurate [9, 11], but it can only infer preferences that oppose the presented order. This again leads to severely biased data, and learning algorithms trained with these preferences tend to reverse the presented order unless additional heuristics are used [9].

Third, probabilistic click models (see [4]) have been used to model how users produce clicks, and they can take position and context biases into account. By estimating latent parameters of these generative click models, one can infer the relevance of a given document for a given query. However, inferring reliable relevance judgments typically requires that the same query is seen multiple times, which is unrealistic in many retrieval settings (e.g., personal collection search) and for tail queries.

Fourth, allowing the LTR algorithm to randomize what is presented to the user, like in online learning algorithms [17, 6] and batch learning from bandit feedback (BLBF) [25] can overcome the problem of bias in click data in a principled manner. However, requiring that rankings be actively perturbed during system operation whenever we collect training data decreases ranking quality and, therefore, incurs a cost compared to observational data collection.

In this paper we present a theoretically principled and empirically effective approach for learning from observational implicit feedback that can overcome the limitations outlined above. By drawing on counterfactual estimation techniques from causal inference [8], we first develop a provably unbiased estimator for evaluating ranking performance using biased feedback data. Based on this estimator, we propose a Propensity-Weighted Empirical Risk Minimization (ERM) approach to LTR, which we implement efficiently in a new learning method we call Propensity SVM-Rank. While our approach uses a click model, the click model is merely used to assign propensities to clicked results in hindsight, not to extract aggregate relevance judgments. This means that our Propensity SVM-Rank does not require queries to repeat, making it applicable to a large range of ranking scenarios. Finally, our methods can use observational data and we do not require that the system randomizes rankings during data collection, except for a small pilot experiment to estimate the propensity model.

When deriving our approach, we provide theoretical justification for each step, leading to a rigorous end-to-end approach that does not make unspecified assumptions or employs heuristics. This provides a principled basis for further improving components of the approach (e.g., the click propensity model, the ranking performance measure, the learning algorithm). We present an extensive empirical evaluation testing the limits of the approach on synthetic click data, finding that it performs robustly over a large range of bias, noise, and misspecification levels. Furthermore, we field our method in a real-world application on an operational search engine, finding that it is robust in practice and manages to substantially improve retrieval performance.

2. RELATED WORK

There are two groups of approaches for handling biases in implicit feedback for learning-to-rank. The first group assumes the feedback collection step is fixed, and tries to interpret the observationally collected data so as to minimize bias effects. Approaches in the second group intervene during feedback collection, trying to present rankings that will lead to less biased feedback data overall.

Approaches in the first group commonly assume some model of user behavior in order to explain bias effects. For example, in a cascade model [5], users are assumed to sequentially go down a ranking and click on a document if it is relevant. Clicks, under this model, let us learn preferences between skipped and clicked documents. Learning from these relative preferences lowers the impact of some biases [9]. Other click models ([5, 3, 1], also see [4]) have been proposed, and are trained to maximize log-likelihood of observed clicks. In these click modeling approaches, performance on downstream learning-to-rank algorithms is merely an afterthought. In contrast, we separate click propensity estimation and learning-to-rank in a principled way, and optimize for ranking performance directly. Our framework allows us to plug-and-play more sophisticated user models in place of the simple click models we use in this work.

The key technique used by approaches in the second group to obtain more reliable click data are randomized experiments. For instance, randomizing documents across all ranks lets us learn unbiased relevances for each document, and swapping neighboring pairs of documents [16] lets us learn reliable pairwise preferences. Similarly, randomized interleaving can detect preferences between different rankers reliably [2]. Different from online learning via bandit algorithms and interleaving [30, 22], batch learning from bandit feedback (BLBF) [25] still uses randomization during feedback collection, and then performs offline learning. Our problem formulation can be interpreted as being half way between the BLBF setting (loss function is unknown and no assumptions on loss function) and learning-to-rank from editorial judgments (components of ranking are fully labeled and loss function is given) since we know the form of the loss function but labels for only some parts of the ranking are revealed. All approaches that use randomization suffer from two limitations. First, randomization typically degrades ranking quality during data collection; second, deploying non-deterministic ranking functions introduces bookkeeping overhead. In this paper, the system can be deterministic and we merely exploit and model stochasticity in user behavior. Moreover, our framework also allows (but does not require)

the use of randomized data collection in order to mitigate the effect of biases and to lower estimator variance.

Our approach uses inverse propensity scoring (IPS), originally employed in survey sampling [7] and causal inference from observational studies [19], but more recently also in whole page optimization [29], IR evaluation with manual judgments [20], and recommender evaluation [13, 21]. We use randomized interventions similar to [5, 12, 28] to estimate propensities in a position discount model. Unlike the uniform ranking randomization of [28] (with its high performance impact) or swapping adjacent pairs as in [5], we swap documents in different ranks to the top position randomly as in [12]. See Section 5.3 for details.

Finally, our approach is similar in spirit to [28], where propensity-weighting is used to correct for selection bias when discarding queries without clicks during learning-to-rank. The key insight of our work is to recognize that inverse propensity scoring can be employed much more powerfully, to account for position bias, trust bias, contextual effects, document popularity etc. using appropriate click models to estimate the propensity of each click rather than the propensity for a query to receive a click as in [28].

3. FULL-INFO LEARNING TO RANK

Before we derive our approach for LTR from biased implicit feedback, we first review the conventional problem of LTR from editorial judgments. In conventional LTR, we are given a sample X of i.i.d. queries $x_i \sim P(x)$ for which we assume the relevances $\mathrm{rel}(x, y)$ of all documents y are known. Since all relevances are assumed to be known, we call this the Full-Information Setting. The relevances can be used to compute the *loss* $\Delta(y|x)$ (e.g., negative DCG) of any ranking y for query x. Aggregating the losses of individual rankings by taking the expectation over the query distribution, we can define the overall *risk* of a ranking system S that returns rankings $S(x)$ as

$$R(S) \quad = \quad \int \Delta(S(x)|x) \, d\,P(x). \qquad (1)$$

The goal of learning is to find a ranking function $S \in \mathcal{S}$ that minimizes $R(S)$ for the query distribution $P(x)$. Since $R(S)$ cannot be computed directly, it is typically estimated via the *empirical risk*

$$\hat{R}(S) \quad = \quad \frac{1}{|X|} \sum_{x_i \in X} \Delta(S(x_i)|x_i).$$

A common learning strategy is *Empirical Risk Minimization (ERM)* [26], which corresponds to picking the system $\hat{S} \in \mathcal{S}$ that optimizes the empirical risk

$$\hat{S} \quad = \quad \mathrm{argmin}_{S \in \mathcal{S}} \left\{ \hat{R}(S) \right\},$$

possibly subject to some regularization in order to control overfitting. There are several LTR algorithms that follow this approach (see [15]), and we use SVM-Rank [9] as a representative algorithm in this paper.

The relevances $\mathrm{rel}(x, y)$ are typically elicited via expert judgments. Apart from being expensive and often infeasible (e.g., in personal collection search), expert judgments come with at least two other limitations. First, since it is clearly impossible to get explicit judgments for all documents, pooling techniques [23] are used such that only the most promising documents are judged. While cutting down

on judging effort, this introduces an undesired *pooling bias* because all unjudged documents are typically assumed to be irrelevant. The second limitation is that expert judgments $\mathrm{rel}(\boldsymbol{x}, y)$ have to be aggregated over all intents that underlie the same query string, and it can be challenging for a judge to properly conjecture the distribution of query intents to assign an appropriate $\mathrm{rel}(\boldsymbol{x}, y)$.

4. PARTIAL-INFO LEARNING TO RANK

Learning from implicit feedback has the potential to overcome the above-mentioned limitations of full-information LTR. By drawing the training signal directly from the user, it naturally reflects the user's intent, since each user acts upon their own relevance judgement subject to their specific context and information need. It is therefore more appropriate to talk about query instances $\boldsymbol{x}_i$ that include contextual information about the user, instead of query strings $\boldsymbol{x}$. For a given query instance $\boldsymbol{x}_i$, we denote with $\mathrm{r}_i(y)$ the user-specific relevance of result y for query instance $\boldsymbol{x}_i$. One may argue that what expert assessors try to capture with $\mathrm{rel}(\boldsymbol{x}, y)$ is the mean of the relevances $\mathrm{r}_i(y)$ over all query instances that share the query string, so, using implicit feedback for learning is able to remove a lot of guesswork about what the distribution of users meant by a query.

However, when using implicit feedback as a relevance signal, unobserved feedback is an even greater problem than missing judgments in the pooling setting. In particular, implicit feedback is distorted by presentation bias, and it is not missing completely at random [14]. To nevertheless derive well-founded learning algorithms, we adopt the following counterfactual model. It closely follows [20], which unifies several prior works on evaluating information retrieval systems.

For concreteness and simplicity, assume that relevances are binary, $\mathrm{r}_i(y) \in \{0, 1\}$, and our performance measure of interest is the sum of the ranks of the relevant results

$$\Delta(\boldsymbol{y}|\boldsymbol{x}_i, \mathrm{r}_i) = \sum_{y \in \boldsymbol{y}} \mathrm{rank}(y|\boldsymbol{y}) \cdot \mathrm{r}_i(y). \quad (2)$$

Analogous to (1), we can define the risk of a system as

$$R(S) = \int \Delta(S(\boldsymbol{x})|\boldsymbol{x}, \mathrm{r}) \, d\,\mathrm{P}(\boldsymbol{x}, \mathrm{r}). \quad (3)$$

In our counterfactual model, there exists a true vector of relevances r_i for each incoming query instance $(\boldsymbol{x}_i, \mathrm{r}_i) \sim \mathrm{P}(\boldsymbol{x}, \mathrm{r})$. However, only a part of these relevances is observed for each query instance, while typically most remain unobserved. In particular, given a presented ranking $\bar{\boldsymbol{y}}_i$ we are more likely to observe the relevance signals (e.g., clicks) for the top-ranked results than for results ranked lower in the list. Let o_i denote the 0/1 vector indicating which relevance values were revealed, $o_i \sim \mathrm{P}(o|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)$. For each element of o_i, denote with $Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)$ the marginal probability of observing the relevance $\mathrm{r}_i(y)$ of result y for query $\boldsymbol{x}_i$, if the user was presented the ranking $\bar{\boldsymbol{y}}_i$. We refer to this probability value as the *propensity* of the observation. We will discuss how o_i and Q can be obtained in Section 5.

Using this counterfactual modeling setup, we can get an unbiased estimate of $\Delta(\boldsymbol{y}|\boldsymbol{x}_i, \mathrm{r}_i)$ for any new ranking $\boldsymbol{y}$ (typically different from the presented ranking $\bar{\boldsymbol{y}}_i$) via the inverse propensity scoring (IPS) estimator [7, 19, 8]

$$\hat{\Delta}_{IPS}(\boldsymbol{y}|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, o_i) = \sum_{y:o_i(y)=1} \frac{\mathrm{rank}(y|\boldsymbol{y}) \cdot \mathrm{r}_i(y)}{Q(o_i(y)=1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)}$$

$$= \sum_{\substack{y:o_i(y)=1 \\ \wedge\,\mathrm{r}_i(y)=1}} \frac{\mathrm{rank}(y|\boldsymbol{y})}{Q(o_i(y)=1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)}.$$

This is an unbiased estimate of $\Delta(\boldsymbol{y}|\boldsymbol{x}_i, \mathrm{r}_i)$ for any $\boldsymbol{y}$, if $Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i) > 0$ for all y that are relevant $\mathrm{r}_i(y) = 1$ (but not necessarily for the irrelevant y).

$$\mathbb{E}_{o_i}[\hat{\Delta}_{IPS}(\boldsymbol{y}|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, o_i)]$$

$$= \mathbb{E}_{o_i}\left[\sum_{y:o_i(y)=1} \frac{\mathrm{rank}(y|\boldsymbol{y}) \cdot \mathrm{r}_i(y)}{Q(o_i(y)=1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)} \right]$$

$$= \sum_{y \in \boldsymbol{y}} \mathbb{E}_{o_i}\left[\frac{o_i(y) \cdot \mathrm{rank}(y|\boldsymbol{y}) \cdot \mathrm{r}_i(y)}{Q(o_i(y)=1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i))} \right]$$

$$= \sum_{y \in \boldsymbol{y}} \frac{Q(o_i(y)=1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i) \cdot \mathrm{rank}(y|\boldsymbol{y}) \cdot \mathrm{r}_i(y)}{Q(o_i(y)=1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)}$$

$$= \sum_{y \in \boldsymbol{y}} \mathrm{rank}(y|\boldsymbol{y}) \, \mathrm{r}_i(y)$$

$$= \Delta(\boldsymbol{y}|\boldsymbol{x}_i, \mathrm{r}_i).$$

The second step uses linearity of expectation, and the fourth step uses $Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i) > 0$.

An interesting property of $\hat{\Delta}_{IPS}(\boldsymbol{y}|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, o_i)$ is that only those results y with $[o_i(y) = 1 \wedge \mathrm{r}_i(y) = 1]$ (i.e. clicked results, as we will see later) contribute to the estimate. We therefore only need the propensities $Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)$ for relevant results. Since we will eventually need to estimate the propensities $Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)$, an additional requirement for making $\hat{\Delta}_{IPS}(\boldsymbol{y}|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, o_i)$ computable while remaining unbiased is that the propensities only depend on observable information (i.e., unconfoundedness, see [8]).

To define the empirical risk to optimize during learning, we begin by collecting a sample of N query instances $\boldsymbol{x}_i$, recording the partially-revealed relevances r_i as indicated by o_i, and the propensities $Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)$ for the observed relevant results in the ranking $\bar{\boldsymbol{y}}_i$ presented by the system. Then, the empirical risk of a system is simply the IPS estimates averaged over query instances:

$$\hat{R}_{IPS}(S) = \frac{1}{N} \sum_{i=1}^{N} \sum_{\substack{y:o_i(y)=1 \\ \wedge\,\mathrm{r}_i(y)=1}} \frac{\mathrm{rank}(y|S(\boldsymbol{x}_i))}{Q(o_i(y)=1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \mathrm{r}_i)}. \quad (4)$$

Since $\hat{\Delta}_{IPS}(\boldsymbol{y}|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, o_i)$ is unbiased for each query instance, the aggregate $\hat{R}_{IPS}(S)$ is also unbiased for $R(S)$ from (3),

$$\mathbb{E}[\hat{R}_{IPS}(S)] = R(S).$$

Furthermore, it is easy to verify that $\hat{R}_{IPS}(S)$ converges to the true $R(S)$ under mild additional conditions (i.e., propensities bounded away from 0) as we increase the sample size N of query instances. So, we can perform ERM using this propensity-weighted empirical risk,

$$\hat{S} = \mathrm{argmin}_{S \in \mathcal{S}} \left\{ \hat{R}_{IPS}(S) \right\}.$$

Finally, using standard results from statistical learning theory [26], consistency of the empirical risk paired with capacity control implies consistency also for ERM. In intuitive

terms, this means that given enough training data, the learning algorithm is guaranteed to find the best system in $\mathcal{S}$.

5. FEEDBACK PROPENSITY MODELS

In Section 4, we showed that the relevance signal r_i, the observation pattern o_i, and the propensities of the observations $Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, r_i)$ are the key components for unbiased LTR from biased observational feedback. We now outline how these quantities can be elicited and modeled in a typical search-engine application. However, the general framework of Section 4 extends beyond this particular application, and beyond the particular feedback model below.

5.1 Position-Based Propensity Model

Search engine click logs provide a sample of query instances $\boldsymbol{x}_i$, the presented ranking $\bar{\boldsymbol{y}}_i$ and a (sparse) click-vector where each $c_i(y) \in \{0, 1\}$ indicates whether result y was clicked or not. To derive propensities of observed clicks, we will employ a click propensity model. For simplicity, we consider a straightforward examination model analogous to [18], where a click on a search result depends on the probability that a user examines a result (i.e., $e_i(y)$) and then decides to click on it (i.e., $c_i(y)$) in the following way:

$$P(e_i(y) = 1|\operatorname{rank}(y|\bar{\boldsymbol{y}})) \cdot P(c_i(y) = 1|r_i(y), e_i(y) = 1).$$

In this model, examination depends only on the rank of y in $\bar{\boldsymbol{y}}$. So, $P(e_i(y) = 1|\operatorname{rank}(y|\bar{\boldsymbol{y}}_i))$ can be represented by a vector of examination probabilities p_r, one for each rank r. These examination probabilities can model presentation bias documented in eye-tracking studies [11], where users are more likely to see results at the top of the ranking than those further down.

For the probability of click on an examined result $P(c_i(y) = 1|r_i(y), e_i(y) = 1)$, we first consider the simplest model where clicking is a deterministic noise-free function of the users private relevance assessment $r_i(y)$. Under this model, users click if and only if the result is examined and relevant ($c_i(y) = 1 \leftrightarrow [e_i(y) = 1 \wedge r_i(y) = 1]$). This means that for examined results (i.e., $e_i(y) = 1$) clicking is synonymous with relevance ($e_i(y) = 1 \to [c_i(y) = r_i(y)]$). Furthermore, it means that we observe the value of $r_i(y)$ perfectly when $e_i(y) = 1$ ($e_i(y) = 1 \to o_i(y) = 1$), and that we gain no knowledge of the true $r_i(y)$ when a result is not examined ($e_i(y) = 0 \to o_i(y) = 0$). Therefore, examination equals observation and $Q(o_i(y)|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, r_i) \equiv P(e_i(y)|\operatorname{rank}(y|\bar{\boldsymbol{y}}_i))$.

Using these equivalences, we can simplify the IPS estimator from (4) by substituting p_r as the propensities and by using $c_i(y) = 1 \leftrightarrow [o_i(y) = 1 \wedge r_i(y) = 1]$

$$\hat{R}_{IPS}(S) = \frac{1}{n}\sum_{i=1}^{n}\sum_{y:c_i(y)=1}\frac{\operatorname{rank}(y|S(\boldsymbol{x}_i))}{p_{\operatorname{rank}(y|\bar{\boldsymbol{y}}_i)}}. \quad (5)$$

$\hat{R}_{IPS}(S)$ is an unbiased estimate of $R(S)$ under the position-based propensity model if $p_r > 0$ for all ranks. While absence of a click does not imply that the result is not relevant (i.e., $c_i(y) = 0 \not\to r_i(y) = 0$), the IPS estimator has the nice property that such explicit negative judgments are not needed to compute an unbiased estimate of $R(S)$ for the loss in (2). Similarly, while absence of a click leaves us unsure about whether the result was examined (i.e., $e_i(y) = ?$), the IPS estimator only needs to know the indicators $o_i(y) = 1$ for results that are also relevant (i.e., clicked results).

Finally, note the conceptual difference in how we use this standard examination model compared to most prior work. We do not try to estimate an average relevance rating $\operatorname{rel}(\boldsymbol{x}, y)$ by taking repeat instances of the same query $\boldsymbol{x}$, but we use the model as a propensity estimator to de-bias individual observed user judgments $r_i(y)$ to be used directly in ERM.

5.2 Incorporating Click Noise

In Section 5.1, we assumed that clicks reveal the user's true r_i in a noise-free way. This is clearly unrealistic. In addition to the stochasticity in the examination distribution $P(e_i(y) = 1|\operatorname{rank}(y|\boldsymbol{y}))$, we now also consider noise in the distribution that generates the clicks. In particular, we no longer require that a relevant result is clicked with probability 1 and an irrelevant result is clicked with probability 0, but instead, for $1 \geq \epsilon_+ > \epsilon_- \geq 0$,

$$P(c_i(y) = 1|r_i(y) = 1, o_i(y) = 1) = \epsilon_+,$$
$$P(c_i(y) = 1|r_i(y) = 0, o_i(y) = 1) = \epsilon_-.$$

The first line means that users click on a relevant result only with probability ϵ_+, while the second line means that users may erroneously click on an irrelevant result with probability ϵ_-. An alternative and equivalent way of thinking about click noise is that users still click deterministically as in the previous section, but based on a noisily corrupted version $\tilde{r}_i$ of r_i. This means that all reasoning regarding observation (examination) events o_i and their propensities p_r still holds, and that we still have that $c_i(y) = 1 \to o_i(y) = 1$. What does change, though, is that we no longer observe the "correct" $r_i(y)$ but instead get feedback according to the noise-corrupted version $\tilde{r}_i(y)$. What happens to our learning process if we estimate risk using (5), but now with $\tilde{r}_i$?

Fortunately, the noise does not affect ERM's ability to find the best ranking system given enough data. While using noisy clicks leads to biased empirical risk estimates w.r.t. the true r_i (i.e., $\mathbb{E}[\hat{R}_{IPS}(S)] \neq R(S)$), in expectation this bias is order preserving for $R(S)$ such that the risk minimizer remains the same.

$$\mathbb{E}[\hat{R}_{IPS}(S_1)] > \mathbb{E}[\hat{R}_{IPS}(S_2)]$$

$$\Leftrightarrow \mathbb{E}_{\boldsymbol{x}, r, \bar{\boldsymbol{y}}}\left[\mathbb{E}_o\mathbb{E}_{c|o}\left[\sum_{y:c(y)=1}\frac{\operatorname{rank}(y|S_1(\boldsymbol{x})) - \operatorname{rank}(y|S_2(\boldsymbol{x}))}{p_{\operatorname{rank}(y|\bar{\boldsymbol{y}})}}\right]\right] > 0$$

$$\Leftrightarrow \mathbb{E}_{\boldsymbol{x}, r}\left[\sum_y P(c(y) = 1|o(y) = 1, r(y))\delta\operatorname{rank}(y|\boldsymbol{x})\right] > 0$$

$$\Leftrightarrow \mathbb{E}_{\boldsymbol{x}, r}\left[\sum_y \delta\operatorname{rank}(y|\boldsymbol{x}) \cdot (\epsilon_+ r(y) + \epsilon_-(1 - r(y)))\right] > 0$$

$$\Leftrightarrow \mathbb{E}_{\boldsymbol{x}, r}\left[\sum_y \delta\operatorname{rank}(y|\boldsymbol{x}) \cdot ((\epsilon_+ - \epsilon_-) r(y) + \epsilon_-)\right] > 0$$

$$*\Leftrightarrow \mathbb{E}_{\boldsymbol{x}, r}\left[\sum_y \delta\operatorname{rank}(y|\boldsymbol{x}) \cdot (\epsilon_+ - \epsilon_-) r(y)\right] > 0$$

$$\Leftrightarrow \mathbb{E}_{\boldsymbol{x}, r}\left[\sum_y \delta\operatorname{rank}(y|\boldsymbol{x}) \cdot r(y)\right] > 0$$

$$\Leftrightarrow R(S_1) > R(S_2),$$

where $\delta\operatorname{rank}(y|\boldsymbol{x})$ is short for $\operatorname{rank}(y|S_1(\boldsymbol{x})) - \operatorname{rank}(y|S_2(\boldsymbol{x}))$ and we use the fact that $\epsilon_- \sum_{y \in \bar{\boldsymbol{y}}} \delta\operatorname{rank}(y|\boldsymbol{x}) = 0$ in the step marked $*$. This implies that our propensity-weighted ERM

is a consistent approach for finding a ranking function with the best true $R(S)$,

$$\hat{S} = \text{argmin}_{S \in \mathcal{S}} \{R(S)\}$$
$$= \text{argmin}_{S \in \mathcal{S}} \left\{ \mathbb{E}[\hat{R}_{IPS}(S)] \right\}, \quad (6)$$

even when the objective is corrupted by click noise as specified above.

5.3 Propensity Estimation

As the last step of defining the click propensity model, we need to address the question of how to estimate its parameters (i.e. the vector of examination probabilities p_r) for a particular search engine [12]. The following shows that we can get estimates using data from a simple intervention similar to [28], but without the strong negative impact of presenting uniformly random results to some users. This also relates to the Click@1 metric proposed by [3].

First, note that it suffices to estimate the p_r up to some positive multiplicative constant, since any such constant does not change how the IPS estimator (5) orders different systems. We therefore merely need to estimate how much p_r changes relative to p_k for some "landmark" rank k. This suggests the following experimental intervention for estimating p_r: before presenting the ranking to the user, swap the result at rank k with the result at rank r. If we denote with y' the results originally in rank k, our click model before and after the intervention indicates that

$$P(c_i(y') = 1|\text{no-swap}) = p_k \cdot P(c_i(y') = 1|e_i(y') = 1)$$
$$P(c_i(y') = 1|\text{swap-k-and-r}) = p_r \cdot P(c_i(y') = 1|e_i(y') = 1)$$

where

$$P(c_i(y') = 1|e_i(y') = 1)$$
$$= \sum_{v \in \{0,1\}} P(c_i(y')=1|r_i(y')=v, e_i(y')=1) \cdot P(r_i(y')=v)$$

is constant regardless of the intervention. This means that the clickthrough rates $P(c_i(y') = 1|\text{swap-k-and-r})$, which we can estimate from the intervention data, are proportional to the parameters p_r for any r. By performing the swapping intervention between rank k and all other ranks r, we can estimate all the p_r parameters.

This swap-intervention experiment is of much lower impact than the uniform randomization proposed in [28] for a different propensity estimation problem, and careful consideration of which rank k to choose can further reduce impact of the swap experiment. From a practical perspective, it may also be unnecessary to separately estimate p_r for each rank. Instead, one may want to interpolate between estimates at well-chosen ranks and/or employ smoothing. Finally, note that the intervention only needs to be applied on a small subset of the data used for fitting the click propensity model, while the actual data used for training the ERM learning algorithm does not require any interventions.

5.4 Alternative Feedback Propensity Models

The click propensity model we define above is arguably one of the simplest models one can employ for propensity modeling in LTR, and there is broad scope for extensions.

First, one could extend the model by incorporating other biases, for example, trust bias [11] which affects perceived relevance of a result based on its position in the ranking.

This can be captured by conditioning click probabilities also on the position $P(c_i(y') = 1|r_i(y'), e_i(y') = 1, \text{rank}(y|\bar{y}_i))$. We have already explored that the model can be extended to include trust bias, but it is omitted due to space constraints. Furthermore, it is possible to model saliency biases [31] by replacing the p_r with a regression function.

Second, we conjecture that a wide range of other click models (e.g., cascade model [5] and others [5, 3, 1, 4]) can be adapted as propensity models. The main requirement is that we can compute marginal click probabilities for the clicked documents in hindsight, which may be feasible for other existing models.

Third, we may be able to define and train new types of click models. In particular, for our propensity ERM approach we only need the propensities $Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, r_i)$ for observed and relevant documents to evaluate the IPS estimator, but not for irrelevant documents. This can be substantially easier than a full generative model of how people reveal relevance judgments through implicit feedback. In particular, this model can condition on all the revealed relevances $r_i(y_j)$ in hindsight, and it does not need to treat them as latent variables.

Finally, the ERM learning approach is not limited to binary click feedback, but applies to a large range of feedback settings. For example, the feedback may be explicit star ratings in a movie recommendation system, and the propensities may be the results of self-selection by the users as in [21]. In such an explicit feedback setting, o_i is fully known, which simplifies propensity estimation substantially.

6. PROPENSITY-WEIGHTED SVM-RANK

We now derive a concrete learning method that implements propensity-weighted LTR. It is based on SVM-Rank [9, 10], but we conjecture that propensity-weighted versions of other LTR methods can be derived as well.

Consider a dataset of n examples of the following form. For each query-result pair $(\boldsymbol{x}_j, y_j)$ that is clicked, we compute the propensity $q_i = Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, r_i)$ of the click according to our click propensity model. We also record the candidate set Y_j of all results for query $\boldsymbol{x}_j$. Typically, Y_j contains a few hundred documents – selected by a stage-one ranker [27] – that we aim to rerank. Note that each click generates a separate training example, even if multiple clicks occur for the same query.

Given this propensity-scored click data, we define Propensity SVM-Rank as a generalization of conventional SVM-Rank. Propensity SVM-Rank learns a linear scoring function $f(\boldsymbol{x}, y) = w \cdot \phi(\boldsymbol{x}, y)$ that can be used for ranking results, where w is a weight vector and $\phi(\boldsymbol{x}, y)$ is a feature vector that describes the match between query $\boldsymbol{x}$ and result y.

Propensity SVM-Rank optimizes the following objective,

$$\hat{w} = \text{argmin}_{w,\xi} \ \frac{1}{2} w \cdot w + \frac{C}{n} \sum_{j=1}^{n} \frac{1}{q_j} \sum_{y \in Y_j} \xi_{jy}$$

$$s.t. \quad \forall y \in Y_1 \backslash \{y_1\} : w \cdot [\phi(\boldsymbol{x}_1, y_1) - \phi(\boldsymbol{x}_1, y)] \geq 1 - \xi_{1y}$$
$$\vdots$$
$$\forall y \in Y_n \backslash \{y_n\} : w \cdot [\phi(\boldsymbol{x}_n, y_n) - \phi(\boldsymbol{x}_n, y)] \geq 1 - \xi_{ny}$$
$$\forall j \forall y : \xi_{jy} \geq 0.$$

C is a regularization parameter that is typically selected via cross-validation. The training objective optimizes an

upper bound on the regularized IPS estimated empirical risk of (5), since each line of constraints corresponds to the rank of a relevant document (minus 1). In particular, for any feasible (w, ξ)

$$\text{rank}(y_i|\boldsymbol{y}) - 1 = \sum_{y \neq y_i} \mathbb{1}_{w \cdot [\phi(\boldsymbol{x}_i, y) - \phi(\boldsymbol{x}_i, y_i)] > 0}$$

$$\leq \sum_{y \neq y_i} \max(1 - w \cdot [\phi(\boldsymbol{x}_i, y_i) - \phi(\boldsymbol{x}_i, y)], 0)$$

$$\leq \sum_{y \neq y_i} \xi_{iy}.$$

We can solve this type of Quadratic Program efficiently via a one-slack formulation [10], and we are using SVM-Rank with appropriate modifications to include IPS weights $1/q_j$. The modifications are integrated into the latest version of SVM-Rank, and the code is available at `http://www.joachims.org/svm_light/svm_rank.html`.

In the empirical evaluation, we compare against the naive application of SVM-Rank, which minimizes the rank of the clicked documents while ignoring presentation bias. In particular, Naive SVM-Rank sets all the q_i uniformly to the same constant (e.g., 1).

7. EMPIRICAL EVALUATION

We take a two-pronged approach to evaluating our approach empirically. First, we use synthetically generated click data to explore the behavior of our methods over the whole spectrum of presentation bias severity, click noise, and propensity misspecification. Second, we explore the real-world applicability of our approach by evaluating on an operational search engine using real click-logs from live traffic.

7.1 Synthetic Data Experiments

To be able to explore the full spectrum of biases and noise, we conducted experiments using click data derived from the Yahoo Learning to Rank Challenge corpus (set 1). This corpus contains a large number of manually judged queries, where we binarized relevance by assigning $r_i(y) = 1$ to all documents that got rated 3 or 4, and $r_i(y) = 0$ for ratings $0, 1, 2$. We adopt the train, validation, test splits in the corpus. This means that queries in the three sets are disjoint, and we never train on any data from queries in the test set. To have a gold standard for reporting test-set performance, we measure performance on the binarized full-information ratings using (2).

To generate click data from this full-information dataset of ratings, we first trained a normal Ranking SVM using 1 percent of the full-information training data to get a ranking function S_0. We employ S_0 as the "Production Ranker", and it is used to "present" rankings $\bar{\boldsymbol{y}}$ when generating the click data. We generate clicks using the rankings $\bar{\boldsymbol{y}}$ and ground-truth binarized relevances from the Yahoo dataset according to the following process. Depending on whether we are generating a training or a validation sample of click data, we first randomly draw a query $\boldsymbol{x}$ from the respective full-information dataset. For this query we compute $\bar{\boldsymbol{y}} = S_0(\boldsymbol{x})$ and generate clicks based on the model from Section 5. Whenever a click is generated, we record a training example with its associated propensity $Q(o(y) = 1|\boldsymbol{x}, \bar{\boldsymbol{y}}, \text{r})$. For the
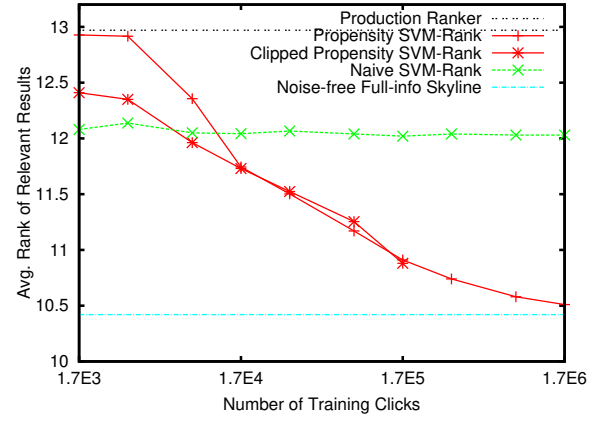

Figure 1: Test set performance in terms of (2) for Propensity SVM-Rank with and without clipping compared to SVM-Rank naively ignoring the bias in clicks ($\eta = 1$, $\epsilon_- = 0.1$). The skyline is a Ranking SVM trained on all data without noise in the full-information setting, and the baseline is the production ranker S_0.

experiments, we model presentation bias via

$$Q(o(y) = 1|\boldsymbol{x}, \bar{\boldsymbol{y}}, \text{r}) = p_{rank(y|\bar{\boldsymbol{y}})} = \left(\frac{1}{rank(y|\bar{\boldsymbol{y}})}\right)^{\eta}. \quad (7)$$

The parameter η lets us control the severity of the presentation bias. We also introduce noise into the clicks according to the model described in Section 5. When not mentioned otherwise, we use the parameters $\eta = 1$, $\epsilon_- = 0.1$, and $\epsilon_+ = 1$, which leads to click data where about 33% of the clicks are noisy clicks on irrelevant results and where the result at rank 10 has a 10% probability of being examined. We also explore other bias profiles and noise levels in the following experiments.

In all experiments, we select any parameters (e.g., C) of the learning methods via cross-validation on a validation set. The validation set is generated using the same click model as the training set, but using the queries in the validation-set portion of the Yahoo dataset. For Propensity SVM-Rank, we always use the (unclipped) IPS estimator (5) to estimate validation set performance. Keeping with the proportions of the original Yahoo data, the validation set size is always about 15% the size of the training set.

The primary baseline we compare against is a naive application of SVM-Rank that simply ignores the bias in the click data. We call this method *Naive SVM-Rank*. It is equivalent to a standard ranking SVM [9], but is most easily explained as equivalent to Propensity SVM-Rank with all q_j set to 1. Analogously, we use the corresponding naive version of (5) with propensities set to 1 to estimate validation set performance for Naive SVM-Rank.

7.2 How does ranking performance scale with training set size?

We first explore how the test-set ranking performance changes as the learning algorithm is given more and more click data. The resulting learning curves are given in Figure 1, and the performance of S_0 is given as a baseline. The click data has presentation bias according to (2) with $\eta = 1$
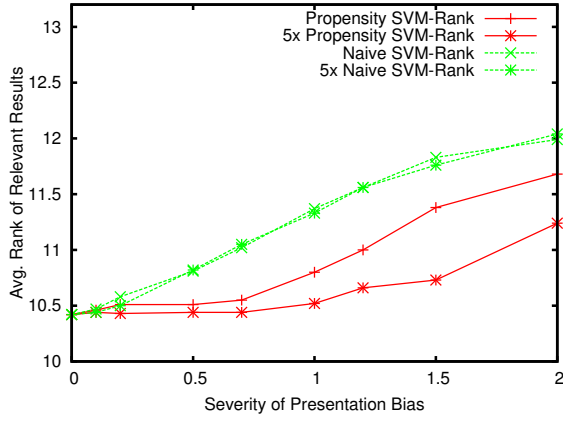
Figure 2: Test set performance for Propensity SVM-Rank and Naive SVM-Rank as presentation bias becomes more severe in terms of η ($n = 45K$ and $n = 225K$, $\epsilon_- = 0$).

Figure 3: Test set performance for Propensity SVM-Rank and Naive SVM-Rank as the noise level increases in terms of ϵ_- ($n = 170K$ and $n = 850K$, $\eta = 1$).

and noise $\epsilon_- = 0.1$. For small datasets, results are averaged over 5 draws of the click data.

With increasing amounts of click data, Propensity SVM-Rank approaches the skyline performance of the full-information SVM-Rank trained on the complete training set of manual ratings without noise. This is in stark contrast to Naive SVM-Rank which fails to account for the bias in the data and does not reach this level of performance. Furthermore, Naive SVM-Rank cannot make effective use of additional data and its learning curve is essentially flat. This is consistent with the theoretical insight that estimation error in Naive SVM-Rank's empirical risk $\hat{R}(S)$ is dominated by asymptotic bias due to biased clicks, which does not decrease with more data and leads to suboptimal learning. The unbiased risk estimate $\hat{R}_{IPS}(S)$ of Propensity SVM-Rank, however, has estimation error only due to finite sample variance, which is decreased by more data and leads to consistent learning.

While unbiasedness is an important property when click data is plenty, the increased variance of $\hat{R}_{IPS}(S)$ can be a drawback for small datasets. This can be seen in Figure 1, where Naive SVM-Rank outperforms Propensity SVM-Rank for small datasets. This can be remedied using techniques like "propensity clipping" [24], where small propensities are clipped to some threshold value τ to trade bias for variance.

$$\hat{R}_{CIPS}(S) = \frac{1}{n} \sum_{\boldsymbol{x}_i} \sum_{y \in S(\boldsymbol{x}_i)} \frac{\text{rank}(y|S(\boldsymbol{x}_i)) \cdot \text{r}_i(y)}{\max\{\tau, Q(o_i(y) = 1|\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i, \text{r}_i)\}}.$$

Figure 1 shows the learning curve of Propensity SVM-Rank with clipping, cross-validating both the clipping threshold τ and C. Clipping indeed improves performance for small datasets. While $\tau = 1$ is equivalent to Naive SVM-Rank, the validation set is too small (and hence, the finite sample error of the validation performance estimate too high) to reliably select this model in every run. In practice, however, we expect click data to be plentiful such that lack of training data is unlikely to be a persistent issue.

7.3 How much presentation bias can be tolerated?

We now vary the severity of the presentation bias via η to understand its impact on Propensity SVM-Rank. Fig-
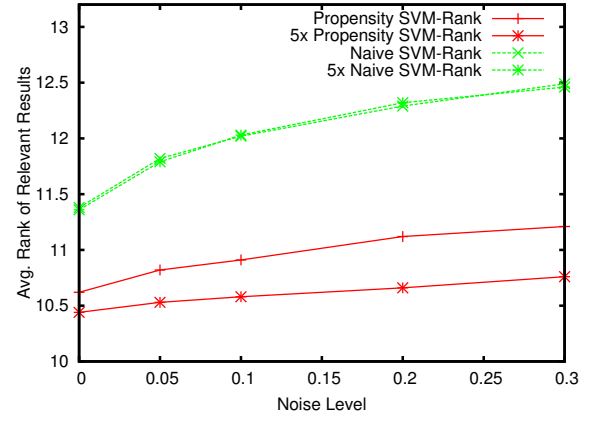
ure 2 shows that inverse propensity weighting is beneficial whenever substantial bias exists. Furthermore, increasing the amount of training data by a factor of 5 leads to further improvement for the Propensity SVM-Rank, while the added training data has no effect on Naive SVM-Rank. This is consistent with our arguments from Section 4 – more training data does not help when bias dominates estimation error, but it can reduce estimation error from variance in the unbiased risk estimate of Propensity SVM-Rank.

7.4 How robust are the methods to click noise?

Figure 3 shows that Propensity SVM-Rank also enjoys a substantial advantage when it comes to noise. When increasing the noise level in terms of ϵ_- from 0 up to 0.3 (resulting in click data where 59.8% of all clicks are on irrelevant documents), Propensity SVM-Rank increasingly outperforms Naive SVM-Rank. And, again, the unbiasedness of the empirical risk estimate allows Propensity SVM-Rank to benefit from more data.

7.5 How robust is Propensity SVM-Rank to misspecified propensities?

So far all experiments have assumed that Propensity SVM-Rank has access to accurate propensities. In practice, however, propensities need to be estimated and are subject to model assumptions. We now evaluate how robust Propensity SVM-Rank is to misspecified propensities. Figure 4 shows the performance of Propensity SVM-Rank when the training data is generated with $\eta = 1$, but the propensities used by Propensity SVM-Rank are misspecified using the η given in the x-axis of the plot. The plot shows that even misspecified propensities can give substantial improvement over naively ignoring the bias, as long as the misspecification is "conservative" – i.e., overestimating small propensities is tolerable (which happens when $\eta < 1$), but underestimating small propensities can be harmful (which happens when $\eta > 1$). This is consistent with theory, and clipping is one particular way of overestimating small propensities that can even improve performance. Overall, we conclude that even a mediocre propensity model can improve over the naive approach – after all, the naive approach can be thought of as a particularly poor propensity model that implicitly assumes no presentation bias and uniform propensities.
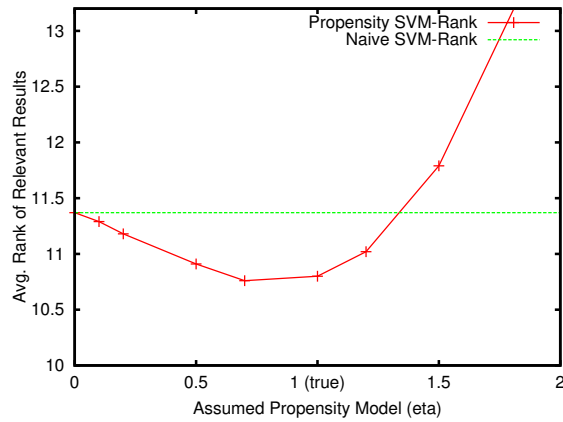
Figure 4: Test set performance for Propensity SVM-Rank and Naive SVM-Rank as propensities are misspecified (true $\eta = 1$, $n = 170K$, $\epsilon_- = 0.1$).

7.6 Real-World Experiment

We now examine the performance of Propensity SVM-rank when learning a new ranking function for the Arxiv Full-Text Search (http://search.arxiv.org:8081/) based on real-world click logs from this system. The search engine uses a linear scoring function as outlined in Section 6. Query-document features $\phi(\boldsymbol{x}, y)$ are represented by a $1000-$dimensional vector, and the production ranker used for collecting training clicks employs a hand-crafted weight vector w (denoted Prod). Observed clicks on rankings served by this ranker over a period of 21 days provide implicit feedback data for LTR as outlined in Section 6.

To estimate the propensity model, we consider the simple position-based model of Section 5.1 and we collect new click data via randomized interventions for 7 days as outlined in Section 5.3 with landmark rank $k = 1$. Before presenting the ranking, we take the top-ranked document and swap it with the document at a uniformly at random chosen rank $j \in \{1, \ldots 21\}$. The ratio of observed click-through rates (CTR) on the formerly top-ranked document now at position j vs. its CTR at position 1 gives a noisy estimate of p_j/p_1 in the position-based click model. We additionally smooth these estimates by interpolating with the overall observed CTR at position j (normalized so that $CTR@1 = 1$). This yields p_r that approximately decay with rank r with the smallest $p_r \simeq 0.12$. For $r > 21$, we impute $p_r = p_{21}$.

We partition the click-logs into a train-validation split: the first 16 days are the train set and provide 5437 click-events for SVM-rank, while the remaining 5 days are the validation set with 1755 click events. The hyper-parameter C is picked via cross validation. Analogous to Section 7.1, we use the IPS estimator for Propensity SVM-Rank, and naive estimator with $Q(o(y) = 1|\boldsymbol{x}, \bar{\boldsymbol{y}}, \mathrm{r}) = 1$ for Naive SVM-Rank. With the best hyper-parameter settings, we re-train on all 21 days worth of data to derive the final weight vectors for either method.

We fielded these learnt weight vectors in two online interleaving experiments [2], the first comparing Propensity SVM-Rank against Prod and the second comparing Propensity SVM-Rank against Naive SVM-Rank. The results are summarized in Table 1. We find that Propensity SVM-Rank significantly outperforms the hand-crafted production ranker that was used to collect the click data for training

Table 1: Per-query balanced interleaving results for detecting relative performance between the hand-crafted production ranker used for click data collection (Prod), Naive SVM-Rank and Propensity SVM-Rank.

Interleaving Experiment	Propensity SVM-Rank		
	wins	loses	ties
against Prod	87	48	83
against Naive SVM-Rank	95	60	102

(two-tailed binomial sign test $p = 0.001$ with relative risk 0.71 compared to null hypothesis). Furthermore, Propensity SVM-Rank similarly outperforms Naive SVM-Rank, demonstrating that even a simple propensity model provides benefits on real-world data (two-tailed binomial sign test $p = 0.006$ with relative risk 0.77 compared to null hypothesis). Note that Propensity SVM-Rank not only significantly, but also substantially outperforms both other rankers in terms of effect size – and the synthetic data experiments suggest that additional training data will further increase its advantage.

8. CONCLUSIONS AND FUTURE

This paper introduced a principled approach for learning-to-rank under biased feedback data. Drawing on counterfactual modeling techniques from causal inference, we present a theoretically sound Empirical Risk Minimization framework for LTR. We instantiate this framework with a Propensity-Weighted Ranking SVM, and provide extensive empirical evidence that the resulting learning method is robust to selection biases, noise, and model misspecification. Furthermore, our real-world experiments on a live search engine show that the approach leads to substantial retrieval improvements, without any heuristic or manual interventions in the learning process.

Beyond the specific learning methods and propensity models we propose, this paper may have even bigger impact for its theoretical contribution of developing the general counterfactual model for LTR, thus articulating the key components necessary for LTR under biased feedback. First, the insight that propensity estimates are crucial for ERM learning opens a wide area of research on designing better propensity models. Second, the theory demonstrates that LTR methods should optimize propensity-weighted ERM objectives, raising the question of which other learning methods beyond the Ranking SVM can be adapted to the Propensity ERM approach. Third, we conjecture that Propensity ERM approaches can be developed also for pointwise and listwise LTR methods using techniques from [20].

Beyond learning from implicit feedback, propensity-weighted ERM techniques may prove useful even for optimizing offline IR metrics on manually annotated test collections. First, they can eliminate pooling bias, since the use of sampling during judgment elicitation puts us in a controlled setting where propensities are known (and can be optimized [20]) by design. Second, propensities estimated via click models can enable click-based IR metrics like click-DCG to better correlate with test set DCG.

9. REFERENCES

[1] A. Borisov, I. Markov, M. de Rijke, and P. Serdyukov. A neural click model for web search. In *Proceedings of the 25th International Conference on World Wide Web*, pages 531–541, 2016.

[2] O. Chapelle, T. Joachims, F. Radlinski, and Y. Yue. Large-scale validation and analysis of interleaved search evaluation. *ACM Transactions on Information Systems (TOIS)*, 30(1):6:1–6:41, 2012.

[3] O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. In *International Conference on World Wide Web (WWW)*, pages 1–10. ACM, 2009.

[4] A. Chuklin, I. Markov, and M. de Rijke. *Click Models for Web Search*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, 2015.

[5] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *International Conference on Web Search and Data Mining (WSDM)*, pages 87–94. ACM, 2008.

[6] K. Hofmann, A. Schuth, S. Whiteson, and M. de Rijke. Reusing historical interaction data for faster online learning to rank for ir. In *International Conference on Web Search and Data Mining (WSDM)*, pages 183–192, 2013.

[7] D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260):663–685, 1952.

[8] G. Imbens and D. Rubin. *Causal Inference for Statistics, Social, and Biomedical Sciences*. Cambridge University Press, 2015.

[9] T. Joachims. Optimizing search engines using clickthrough data. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 133–142, 2002.

[10] T. Joachims. Training linear SVMs in linear time. In *ACM SIGKDD International Conference On Knowledge Discovery and Data Mining (KDD)*, pages 217–226, 2006.

[11] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)*, 25(2), April 2007.

[12] J. Langford, A. Strehl, and J. Wortman. Exploration scavenging. In *Proceedings of the 25th International Conference on Machine Learning*, pages 528–535, 2008.

[13] L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *International Conference on Web Search and Data Mining (WSDM)*, pages 297–306, 2011.

[14] R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. John Wiley, 2002.

[15] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, Mar. 2009.

[16] K. Raman and T. Joachims. Learning socially optimal information systems from egoistic users. In *European Conference on Machine Learning (ECML)*, pages 128–144, 2013.

[17] K. Raman, T. Joachims, P. Shivaswamy, and T. Schnabel. Stable coactive learning via perturbation. In *International Conference on Machine Learning (ICML)*, pages 837–845, 2013.

[18] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: Estimating the click-through rate for new ads. In *International Conference on World Wide Web (WWW)*, pages 521–530. ACM, 2007.

[19] P. R. Rosenbaum and D. B. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

[20] T. Schnabel, A. Swaminathan, P. Frazier, and T. Joachims. Unbiased comparative evaluation of ranking functions. In *ACM International Conference on the Theory of Information Retrieval (ICTIR)*, 2016.

[21] T. Schnabel, A. Swaminathan, A. Singh, N. Chandak, and T. Joachims. Recommendations as treatments: Debiasing learning and evaluation. In *International Conference on Machine Learning (ICML)*, 2016.

[22] A. Schuth, H. Oosterhuis, S. Whiteson, and M. de Rijke. Multileave gradient descent for fast online learning to rank. In *International Conference on Web Search and Data Mining (WSDM)*, pages 457–466, 2016.

[23] K. Sparck-Jones and C. J. V. Rijsbergen. Report on the need for and provision of an "ideal" information retrieval test collection. Technical report, University of Cambridge, 1975.

[24] A. L. Strehl, J. Langford, L. Li, and S. Kakade. Learning from logged implicit exploration data. In *Conference on Neural Information Processing Systems (NIPS)*, pages 2217–2225, 2010.

[25] A. Swaminathan and T. Joachims. Batch learning from logged bandit feedback through counterfactual risk minimization. *Journal of Machine Learning Research (JMLR)*, 16:1731–1755, Sep 2015.

[26] V. Vapnik. *Statistical Learning Theory*. Wiley, Chichester, GB, 1998.

[27] L. Wang, J. J. Lin, and D. Metzler. A cascade ranking model for efficient ranked retrieval. In *ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 105–114, 2011.

[28] X. Wang, M. Bendersky, D. Metzler, and M. Najork. Learning to rank with selection bias in personal search. In *ACM Conference on Research and Development in Information Retrieval (SIGIR)*. ACM, 2016.

[29] Y. Wang, D. Yin, L. Jie, P. Wang, M. Yamada, Y. Chang, and Q. Mei. Beyond ranking: Optimizing whole-page presentation. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, pages 103–112, 2016.

[30] Y. Yue and T. Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *International Conference on Machine Learning (ICML)*, pages 151–159, 2009.

[31] Y. Yue, R. Patel, and H. Roehrig. Beyond position bias: examining result attractiveness as a source of presentation bias in clickthrough data. In *International Conference on World Wide Web (WWW)*, pages 1011–1018. ACM, 2010.