Learning More Powerful Test Statistics for Click-Based Retrieval Evaluation

Yisong Yue
Cornell University
Ithaca, NY, USA
yyue@cs.cornell.edu

Yue Gao
Cornell University
Ithaca, NY, USA
ygao@cs.cornell.edu

Olivier Chapelle
Yahoo! Research
Santa Clara, CA, USA
chap@yahoo-inc.com

Ya Zhang
Shanghai Jiao Tong University
Shanghai, China
ya_zhang@sjtu.edu.cn

Thorsten Joachims
Cornell University
Ithaca, NY, USA
tj@cs.cornell.edu

ABSTRACT

Interleaving experiments are an attractive methodology for evaluating retrieval functions through implicit feedback. Designed as a blind and unbiased test for eliciting a preference between two retrieval functions, an interleaved ranking of the results of two retrieval functions is presented to the users. It is then observed whether the users click more on results from one retrieval function or the other. While it was shown that such interleaving experiments reliably identify the better of the two retrieval functions, the naive approach of counting all clicks equally leads to a suboptimal test. We present new methods for learning how to score different types of clicks so that the resulting test statistic optimizes the statistical power of the experiment. This can lead to substantial savings in the amount of data required for reaching a target confidence level. Our methods are evaluated on an operational search engine over a collection of scientific articles.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval.

General Terms: Measurement, Human Factors, Experimentation.

Keywords: Implicit feedback, retrieval evaluation, click-through data.

1. INTRODUCTION

Given the rapidly growing breadth and quantity of information needs and retrieval tasks, the need to develop scalable and reliable evaluation methodologies has likewise been gaining in importance. Towards this end, evaluating retrieval performance based on implicit feedback (e.g., clicks, reformulations and dwell time) is an attractive option for several reasons. First, the evaluation is done on the actual population of users in their natural usage contexts, which is difficult to replicate when using editorial judgments. Second, recording implicit feedback is inexpensive and much faster than obtaining editorial judgments. And, finally, implicit feedback is available even for collections where it would not be economically feasible to hire relevance judges.

One approach for deriving reliable judgments from implicit feedback is to focus on collecting *relative* as opposed to *absolute* feedback. For example, while it is difficult to interpret clicks on an absolute scale (e.g., clicked results are relevant, non-clicked results are not relevant), there is clear evidence that clicks provide reliable relative feedback (e.g., clicked results are better than skipped results) [2, 14, 18]. This property is exploited in Interleaving Experiments [13, 18] to compare the relative quality of two ranked retrieval functions h and h'. For every incoming query, the rankings of the two retrieval functions are presented to the user as a single interleaved ranking, and the user's clicks are observed. If the user clicks more on results from h than from h' in the interleaved ranking, it was shown that one can reliably conclude that h is preferred over h' [18, 17]. From an experiment design perspective, interleaving provides a blind paired test where presentation bias is eliminated through randomization under reasonable assumptions.

In this paper, we aim to make interleaving experiments more efficient – or scalable – by developing a more powerful test statistic. Our motivation comes from the intuition that not every click in the interleaved ranking is equally informative. For example, a click on the result at rank 1 in a query session immediately followed by a "back" (i.e., a quick return to the search results page) is probably less informative than the last click in the session (which satisfies the information need). As such, having more flexible weighting schemes on clicks can reduce the variance of the test statistic[1]. This improved experiment design will allow us to confidently tease apart the quality of two competing retrieval functions using substantially less data.

We present three learning methods for optimizing test statistics by using training data from pairs of retrieval functions of known relative retrieval quality (e.g., by gathering enough data so that the conventional test statistic is significant). The learned test statistic can then be used to more quickly identify the superior retrieval function in future interleaving experiments. Learning test statistics

[1]This is also known as the credit assignment problem [17].

SIGIR'10, July 19–23, 2010, Geneva, Switzerland.

Algorithm 1 Team-Draft Interleaving

Input: Rankings $A = (a_1, a_2, \dots)$ and $B = (b_1, b_2, \dots)$
Init: $I \leftarrow ()$; $TeamA \leftarrow \emptyset$; $TeamB \leftarrow \emptyset$;
while $(\exists i : A[i] \notin I) \wedge (\exists j : B[j] \notin I)$ **do**
  if $(|TeamA| < |TeamB|) \vee$
    $((|TeamA| = |TeamB|) \wedge (RandBit() = 1))$ **then**
    $k \leftarrow \min_i \{i : A[i] \notin I\}$ ..... *top result in A not yet in I*
    $I \leftarrow I + A[k]$; ..... *append it to I*
    $TeamA \leftarrow TeamA \cup \{A[k]\}$ ..... *clicks credited to A*
  else
    $k \leftarrow \min_i \{i : B[i] \notin I\}$ ..... *top result in B not yet in I*
    $I \leftarrow I + B[k]$ ..... *append it to I*
    $TeamB \leftarrow TeamB \cup \{B[k]\}$ ..... *clicks credited to B*
  end if
end while
Output: Interleaved ranking I, $TeamA$, $TeamB$

	Input Ranking		Interleaved Rankings *Team-Draft*			
Rank	A	B	AAA	BAA	ABA	...
1	a	b	a^A	b^B	a^A	
2	b	e	b^B	a^A	b^B	
3	c	a	c^A	c^A	e^B	
4	d	f	e^B	e^B	c^A	
5	g	g	d^A	d^A	d^A	
6	h	h	f^B	f^B	f^B	
⋮	⋮	⋮	⋮	⋮	⋮	

Figure 1: An example showing how the Team-Draft method interleaves input rankings A and B for different random coin flip outcomes. Superscripts of the interleavings indicates team membership.

can be thought of as solving the inverse problem of conventional hypothesis testing, and we present an empirical evaluation on real data from an operational search engine for research papers.

2. RELATED WORK

The traditional Cranfield methodology [20] relies solely on editorial judgments for evaluation, where human judges assign explicit relevance ratings to results. Though effective at small scales, this approach quickly becomes infeasible as the evaluation tasks grow in size and number. While methods exist that reduce the labeling requirements to some extent (e.g., [6, 3, 4]), or that reduce the cost of collecting explicit feedback [5, 19], leveraging usage logs has been steadily gaining in popularity due to their inexpensive availability.

Our work is closely related to the topic of learning user behavior models, since implicit feedback is essentially a reflection of human behavior. Fox et al. [11] learned an association between implicit feedback gathered by an instrumented browser and explicit judgments of satisfaction. Other existing approaches typically learn user behavior or document relevance models from passively collected usage logs (e.g., [2, 7, 10, 21, 9], often with the goal of aiding the retrieval function in providing more relvant results (e.g., [1, 8]).

Our work is distinguished from the aforementioned work on user modeling in at least two aspects. First, it is set within the framework of a well-controlled paired experiment design. We will be able to exploit the properties of this experiment design when deriving and theoretically justifying the learning methods, as well as when interpreting the results. Second, prior work on user modeling for this purpose focused largely on evaluation at the result (i.e., document) level (e.g., [1, 14, 22]). In contrast, we are interested in performing more holistic evaluations for ranking functions.

Radlinski & Craswell [17] recently conducted a study comparing conventional measures (e.g., NDCG, MAP) to interleaving metrics based on clickthrough data. They found that manually increasing the weight of clicks lower in the ranking improved the statistical power of the interleaving metric. We will provide automatic learning methods for optimizing the power of the test statistic, making use of many attributes beyond the rank of the clicks.

3. INTERLEAVING EVALUATION

In analogy to experiment designs from sensory analysis (see e.g. [15]), interleaving experiments [13, 18] provide paired preference tests between two retrieval (i.e., ranking) functions. Such paired experiments are particularly suitable in situations where it is difficult or meaningless to assign an absolute rating (e.g., rate this taste on a scale from 1 to 10), but a relative comparison is easy to make (e.g., do you like taste A better than taste B). To elicit such pairwise preferences, both alternatives have to be presented side-by-side and without presentation bias. For example, the order in which a subject tastes two products must be randomized, and the identity of the products must be "blind" to the user.

For the case of comparing pairs of retrieval functions, interleaving experiments are designed to provide such a blind and unbiased side-by-side comparison of two retrieval functions h and h'. When a user issues a query q, the rankings $A = h(q)$ and $B = h'(q)$ are computed but kept hidden from the user. Instead, the user is shown a single interleaved ranking I computed from A and B, so that clicks on I provide feedback on the users preference between A and B under reasonable assumptions.

In this paper, we focus on the Team-Draft Interleaving method [18] that is summarized in Algorithm 1. Team-Draft Interleaving creates a fair (i.e. unbiased) interleaved ranking following the analogy of selecting teams for a friendly team-sports match. One common approach is to first select two team captains, who then take turns selecting players in their team. Team-Draft Interleaving uses an adapted version of this approach for creating interleaved rankings. Suppose each document is a player, and rankings A and B are the preference orders of the two team captains. In each round, captains pick the next player by selecting their most preferred player that is still available, add the player to their team and append the player to the interleaved ranking I. We randomize which captain gets to pick first in each round. An illustrative example from [18] is given in Figure 1.

To infer whether the user prefers ranking A or ranking B, one counts the number of clicks on documents from each team. If team A gets more clicks, A wins the side-by-side comparison and vice versa. Denoting the sets of clicks on the respective teams with C and C' for query q, the mean or median value of the test statistic

$$\delta(q, C, C') = |C| - |C'| \tag{1}$$

over the distribution $P(q)$ of queries reveals whether one of h and h' is consistently preferred over the other. Section 3.1 discusses three possible tests that detect whether the mean or median of $\delta(q, C, C')$ is significantly different from zero.

Note that the presentation is unbiased in the sense that A and B have equal probability of occupying each rank in I. This means that any user that clicks randomly will not generate a significant preference in either direction.

In this paper, we address one shortcoming of the test statistic

in (1): the test statistic scores all clicks equally, which is likely to be suboptimal in practice. For example, a user clicking back immediately after a click is probably an indicator that the page was not good after all. The goal of this work is to learn a more refined function $score(q,c)$ that scores different types of clicks according to their actual information content. This scoring function can then be used in the following rule

$$\delta(q,C,C') = \left[\sum_{c\in C} score(q,c)\right] - \left[\sum_{c'\in C'} score(q,c')\right].$$

Note that this reduces to (1) if $score(q,c)$ is always 1.

In the following, we will use a linear model $score(q,c) = w^T \varphi(q,c)$ to score clicks, where w is a vector of parameters to be learned and $\varphi(q,c)$ returns a feature vector describing each click c in the context of the entire query session q. We can now rewrite $\delta(q,C,C')$ as

$$\delta_w(q,C,C') = w^T \Phi(q,C,C')$$

where

$$\Phi(q,C,C') = \sum_{c\in C} \varphi(q,c) - \sum_{c\in C'} \varphi(q,c) \quad (2)$$

Feature vectors $\varphi(q,c)$ will contain features that describe the click in relation to position in the interleaved ranking, order, and presentation. In Section 5.2, we will decribe the feature construction used in our empirical evaluation.

3.1 Hypothesis Tests for Interleaving

To decide whether an interleaving experiment between h and h' shows a preference in either direction, one needs to test whether some measure of centrality (e.g. median, mean) of the i.i.d. random variables $\Delta_i \equiv \delta(q,C,C')$ is significantly different from zero. For conciseness, let $(\delta_1,...,\delta_n)$ denote the values of $\delta(q,C,C')$ on a random sample. We consider the following three tests, which will also serve as the baseline methods in our empirical evaluation.

The simplest test, and the one previously used in [12, 13, 18], is the *Binomial Sign Test* (see e.g. [16]). It counts how often the sign of δ_i is positive, i.e. $S = \sum_{i=1}^{n} [\Delta_i > 0]$. This sum S is a binomial random variable, and the null hypothesis is that the underlying i.i.d. Bernoulli random variables $[\Delta_i > 0]$ have $p = 0.5$.

Unlike the Binomial Sign Test, the *z-Test* (see e.g. [16]) uses the magnitudes of the Δ_i and tests whether their sum is zero in expectation. The z-Test assumes that $S = \frac{1}{n}\sum_{i=1}^{n} \Delta_i$ is normal, which is approximately satisfied for large n. The ratio of the observed value $s = \frac{1}{n}\sum_{i=1}^{n} \delta_i$ and standard deviation $std(S)$, called the z-score $z = s/std(S)$, monotonically relates to the p-value of the z-test. While $std(S)$ has to be known, an approximate z-test results from estimating $std(S) = \frac{1}{\sqrt{n}}\sqrt{\frac{1}{n}\sum_j (s-\delta_j)^2}$ from the sample. The t-test accounts for the additional variability from the estimate of the standard deviation, but for large samples z-test and t-test are virtually identical.

Finally, we consider the *Wilcoxon Signed Rank Test* (see e.g. [16]) as a non-parametric test for the median of the Δ_i being 0. To compute the test statistic, the observations are ranked by $|\delta_i|$. Let the resulting rank of δ_i be r_i. The test statistic is then computed as $W = \sum sign(\delta_i) r_i$, and W is tested for mean 0 using a z-test.

4. LEARNING METHODS

The idea behind learning is to find a scoring function that results in the most sensitive hypothesis test. To illustrate this goal, consider the following hypothetical scenario where the scoring function $score(q,c) = w^T \varphi(q,c)$ differentiates the last click of a query session from other clicks within the same session. The corresponding feature vector $\varphi(q,c)$ would then have two binary features

$$\varphi(q,c) = \begin{pmatrix} \text{1, if c is last click; 0 else} \\ \text{1, if c is not last click; 0 else} \end{pmatrix}.$$

Assume for simplicity that every query session has 3 clicks, with "not last clicks" being completely random while "last clicks" favoring the better retrieval function with 60% probability. Using the weight vector $w^T = (1,1)$ (i.e., the conventional scoring function), one will eventually identify that the better retrieval function gets more clicks (typically after ≈280 queries using a t-test with $p = 0.95$). However, the optimal weight vector $w^T = (1,0)$ will identify the better retrieval function much faster (typically after ≈150 queries), since it eliminates noise from the non-informative clicks.

The learning problem can be thought of as an "inverse" hypothesis test: given data for pairs (h,h') of retrieval functions where we know $h \succ h'$, find the weights w that maximizes the power of the test statistic on new pairs. More concretely, we assume that we are given a set of ranking function pairings $\{(h_1,h'_1),...,(h_k,h'_k)\}$ for which we know w.l.o.g. that h_i is better than h'_i, i.e. $h_i \succ h'_i$. This preference may be known by construction (e.g., h'_i is a degraded version of h_i), by running interleaving until the conventional test statistic that scores each click uniformly becomes significant, or through some expensive annotation process (e.g., user interviews, manual assessments). For each pair (h_i,h'_i), we assume access to usage logs from Team-Draft Interleaving [18] for n_i queries. For each query q_j, the clicks C_j and C'_j for each "team" are recorded in a triple (q_j,C_j,C'_j). Eventually, all triples are combined into one training sample

$$S = ((q_1,C_1,C'_1),...,(q_n,C_n,C'_n)).$$

Note that we are essentially treating all interleaving pairs as a single combined example[2]. After training, the learned w and the resulting test statistic $\delta_w(q,C,C')$ will be applied to new pairs of retrieval functions (h_{test},h'_{test}) of yet unkown relative retrieval quality.

We now propose three learning methods, with each corresponding to opimizing a specific inverse hypothesis test.

4.1 Maximize Mean Difference

In the simplest case, we can optimize the parameters w of $score_w(q,c)$ to maximize the mean difference of scores between the better and the worse retrieval functions,

$$\begin{aligned} w^* &= \operatorname*{argmax}_w \sum_{j=1}^{n} \delta_w(q_j,C_j,C'_j) \\ &= \operatorname*{argmax}_w \sum_j w^T \Phi(q_j,C_j,C'_j) \end{aligned}$$

To abstract from different scalings of w and to make the problem well posed, we impose a normalization constraint $||w|| = 1$, leading to the following optimization problem:

$$w^* = \operatorname*{argmax}_w \sum_j w^T \Phi(q_j,C_j,C'_j) \text{ s.t. } ||w|| = 1,$$

which can be written more compactly using $\Psi_j = \Phi(q_j,C_j,C'_j)$,

$$w^* = \operatorname*{argmax}_w \left[\sum_j w^T \Psi_j\right] \text{ s.t. } ||w|| = 1.$$

[2]A better approach may be to explicitly treat each interleaving pair as a separate example.

This has the the following closed-form solution that can be derived via Lagrange multipliers:

$$w^* = \frac{\sum_j \Psi_j}{\sqrt{(\sum_j \Psi_j)^T (\sum_j \Psi_j)}} \sim \sum_j \Psi_j.$$

While maximizing the mean difference is intuitively appealing, one key shortcoming is that variance is ignored. In fact, one can think of this method as an inverse z-Test, where we assume equal variance for all w. Since the assumption of equal variance will clearly not be true in practice, we now consider the following more refined methods.

4.2 Inverse z-Test

The following learning method removes the assumption of equal variance and optimizes the statistical power of a z-Test in the general case (with the null hypothesis that the mean is zero). Finding the w that maximizes the z-score (and therefore the p-value) on the training set corresponds to the following optimization problem:

$$\begin{aligned} w^* &= \operatorname*{argmax}_w \frac{\frac{1}{n}\sum_j \delta_w(q_j,C_j,C'_j)}{\frac{1}{\sqrt{n}}\sqrt{\frac{1}{n}\sum_j \delta_w(q_j,C_j,C'_j)^2 - \left[\frac{1}{n}\sum_j \delta_w(q_j,C_j,C'_j)\right]^2}} \\ &= \operatorname*{argmin}_w \frac{\sum_j \delta_w(q_j,C_j,C'_j)^2}{\left[\sum_j \delta_w(q_j,C_j,C'_j)\right]^2} \end{aligned} \quad (3)$$

While (3) has two symmetric solutions, we are interested only in the one where $\sum_j \delta_{w^*}(q_j,C_j,C'_j) > 0$. Using the abbreviated notation $\Psi_j = \Phi(q_j,C_j,C'_j)$, this optimization problem can be rewritten as

$$w^* = \operatorname*{argmax}_w \frac{(w^T \sum_j \Psi_j)^2}{w^T \left[\sum_j \Psi_j \Psi_j^T\right] w}.$$

For any w solving this optimization problem, cw with $c > 0$ is also a solution. We can thus rewrite the problem as

$$w^* = \operatorname*{argmax}_w \left[w^T \sum_j \Psi_j\right] \text{ s.t. } w^T \left[\sum_j \Psi_j \Psi_j^T\right] w = 1.$$

Using the Lagrangian

$$L(w,\alpha) = w^T \sum_j \Psi_j - \alpha \left(w^T \left[\sum_j \Psi_j \Psi_j^T\right] w - 1 \right),$$

and solving for zero derivative w.r.t. w and α, one arrives at a closed form solution. Denoting $\Psi = \sum_j \Psi_j$ and $\Sigma = \sum_j \Psi_j \Psi_j^T$ the solution can be written as

$$w^* = \frac{\Sigma^{-1}\Psi}{\sqrt{\Psi^T \Sigma^{-1} \Psi}}.$$

While not used in the experiments for this paper, a regularized version Σ_{reg} of the covariance matrix Σ can be used to prevent overfitting. One straightforward approach is to add a ridge term $\Sigma_{reg} = \Sigma + \gamma I$, where I is the identity matrix and γ is the regularization parameter.

4.3 Inverse Rank Test

Last but not least, we consider a learning method that relates to inverting the Wilcoxon Rank Sign test. A good scoring function $\delta_w(q,C,C')$ for the Wilcoxon test should optimize the Wilcoxon statistic, which can be computed as follows. Assuming $h \succ h'$ w.l.o.g., we denote a prediction as "correct" if $\delta_w(q,C,C') > 0$; otherwise, we denote it as incorrect. Ranking all observations by $|\delta_w(q,C,C')|$ (assuming no ties), the Wilcoxon statistic is isomorphic to the number of observation pairs where an incorrect observation is ranked above a correct observation. One strategy for minimizing the number of such swapped pairs, and therefore optimizing the p-value of the Wilcoxon test, is to choose

$$\delta_w(q,C,C') = \Pr(h \succ h'|q,C,C') - 0.5, \quad (4)$$

where $\Pr(h \succ h'|q,C,C')$ is the estimated probability that h is better than h' given the clicks observed for query q.

We estimate $\Pr(h \succ h'|q,C,C')$ from the training data S using a standard logistic regression model

$$\ln \frac{\Pr(h \succ h'|q,C,C')}{\Pr(h' \succ h|q,C,C')} = w^T \Phi(q_j,C_j,C'_j).$$

Using again the convention that $h \succ h'$ for the training data and abbreviating $\Psi_j = \Phi(q_j,C_j,C'_j)$, the parameters w are chosen via maximum likelihood,

$$w^* = \operatorname*{argmax}_w \prod_{j=1}^{n} \frac{1}{1+e^{-w^T \Psi_j}}.$$

w^* denotes the logistic regression solution on the training data. We used the LR-TRIRLS package[3] to solve this optimization problem. The final ranking function can be simplified to the linear function $\delta_w(q,C,C') = w^T \Phi(q,C,C')$, since it produces the same rankings and signs as (4).

5. EMPIRICAL SETUP

5.1 Data Collection

We evaluated our methods empirically using data collected from the Physics E-Print ArXiv[4]. In particular, we used two datasets of click logs collected while running Team-Draft Interleaving experiments. For both datasets, we recorded information for each query (e.g., the entire session) and click (e.g., rank, timestamp, result information, source ranking function, etc). This information is used to generate features for learning (see Section 5.2 below). One could also collect user-specific information (e.g., user history), but we have not done so in the following experiments.

"Gold standard". Our first dataset is taken from the Team-Draft experiments described in [18]. In these experiments, the incumbent retrieval function was corrupted in multiple ways to provide pairs of retrieval functions with known relative quality. This provides cheap access to a "gold standard" dataset, since one knows by construction which retrieval function is superior within each pair. A total of six pairs was evaluated, with each yielding slightly over 1000 query sessions.

New interleaving experiments. Our second dataset was generated via interleaving pairs of retrieval functions without necessarily having knowledge of which retrieval function is superior within each pair. For example, one retrieval function that we considered modifies the incumbant retrieval function by giving additional weight to query/title similarity. It is a priori unclear whether this would result in improved retrieval quality. Ideally (and intuitively), learning a test statistic on the gold standard dataset should help us more quickly determine the superior retrieval function within these interleaving pairs. We examine this hypothesis further in Section 6.4. A total of six different retrieval functions are considered in this setting. We collected click data from interleaving every possible pairing of the six, resulting in fifteen interleaving pairs with each

[3] http://komarix.org/ac/lr/

[4] http://arxiv.org

yielding between 400 and 650 query sessions. We then removed three of the fifteen interleaving pairs from our analysis, since all methods (including the baselines) showed poor performance (p-value greater than 0.4), making them uninteresting for comparison purposes.

5.2 Feature Generation

The features that are used in the following experiments describe a diverse set of properties related to clicking behavior, including the rank and order of clicks, and whether search result clicks led to a PDF download in ArXiv[5]. Let C_{own} and C_{other} denote the clicks from the own team and the other team, respectively, for a single query session. Recall from (2) that our feature function $\Phi(q, C_{own}, C_{other})$ decomposes as

$$\Phi(q, C_{own}, C_{other}) = \sum_{c \in C_{own}} \varphi(q,c) - \sum_{c \in C_{other}} \varphi(q,c).$$

We will construct $\varphi(q,c)$ for $c \in C_{own}$ in the following way:

1. 1 always
2. 1 if c led to a download
3. $\frac{1}{|C_{own}|}$ if C_{own} gets both more clicks and downloads
4. If $|C_{own}| == |C_{other}|$:
   (a) $\min\left\{\frac{number\_of\_bolded\_words\_in\_title}{number\_of\_query\_words}, 1\right\}$
   (b) $\min\left\{\frac{number\_of\_bolded\_words\_in\_abstract}{number\_of\_query\_words}, 2\right\}$
5. If it is a single-click query:
   (a) 1 if c is not at rank 1
   (b) 1 if c is on first page (top 10)
6. If it is a multi-click query:
   (a) 1 if c is first click
   (b) 1 if c is last click
   (c) 1 if c is first click and not at rank 1
   (d) 1 if c is at rank 1
   (e) 1 if c is at ranks 1 to 3
   (f) 1 if c is on first page (top 10)
   (g) 1 if c is followed by click on a higher position (regression click)

Analogously, we construct $\varphi(q,c)$ for $c \in C_{other}$ by swapping C_{own} and C_{other} in the preceding feature definitions.

Note that some features are more naturally expressed at the query level. For example, feature 3 can be equivalently expressed directly as feature of $\Phi(q, C_{own}, C_{other})$ as

$$\begin{cases} 1 & \text{if } C_{own} \text{ gets both more clicks and downloads} \\ -1 & \text{if } C_{other} \text{ gets both more clicks and downloads} \\ 0 & \text{otherwise} \end{cases}.$$

For clarity, we focus our formulation on click-level features, since most features we used are more naturally understood at the click level.

[5]All search results correspond to research papers that are available for download.

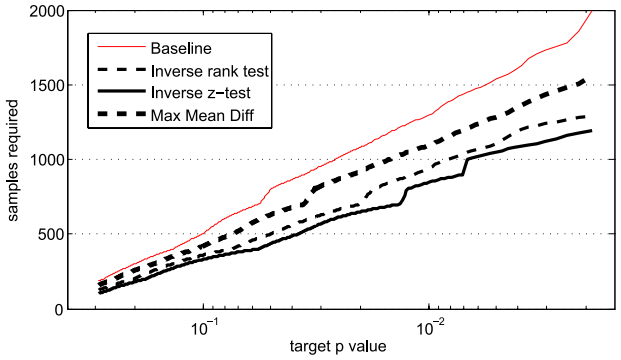

Figure 2: Comparing the sample size required versus target t-test p-value in the synthetic experimental setting. Measurements taken from 1000 bootstrapped subsamples for each subsampling size.

6. EMPIRICAL EVALUATION

For ease of presentation, we will only show comparisons against the t-test baseline; our empirical results also hold when comparing against the other baselines. In general, we find the inverse z-test to be the best performing method, with the inverse rank test often being competitive as well.

6.1 Synthetic Experiment

We first conducted a synthetic experiment where all six gold standard interleaving pairs in the training set are mixed together to form a single (virtual) interleaving pair. From this, 70% of the data was used for training and the remaining 30% for testing. Intuitively, this setup satisfies the assumption that the click distribution we train on is the same as the click distribution we test on – a core assumption often made when analyzing machine learning approaches.

Figure 2 shows how the required sample size grows with decreasing target t-test p-value. This plot (and all similar plots) was generated by subsampling the test set (with replacement) at varying subset sizes and computing the p-value. Subset sizes increase in increments of 25 and each subset size was sampled 1000 times. Our goal is to reduce the required sample size, so lower curves indicate superior performance.

We observe in Figure 2 that our methods consistently outperform the baseline. For example, for a target p-value of 0.01, the inverse z-test requires only about 800 samples whereas the baseline t-test requires about 1200 – a 50% improvement. In all of our subsequent experiments, we find that the max mean difference method consistently performs worse than the inverse z-test. As such, we will focus on the inverse rank test and the inverse z-test in the remaining empirical evaluations.

6.2 Analyzing the Learned Scoring Function

To give some insight into the scoring function $\delta_w(q, C, C') = w^T \Phi(q, C, C')$ learned by our methods, Table 1 shows the weights w generated by the inverse rank test on the full gold standard training set. Since the features are highly correlated, it is difficult to gain insight merely through inspection of the weights. As such, we now provide a selection of prototypical example queries for which we will compute the feature vector $\Psi = \Phi(q, C, C')$ and the value of $\delta_w(q, C, C')$.

1. *Single click on result from h at rank 1*: Feature vector Ψ has

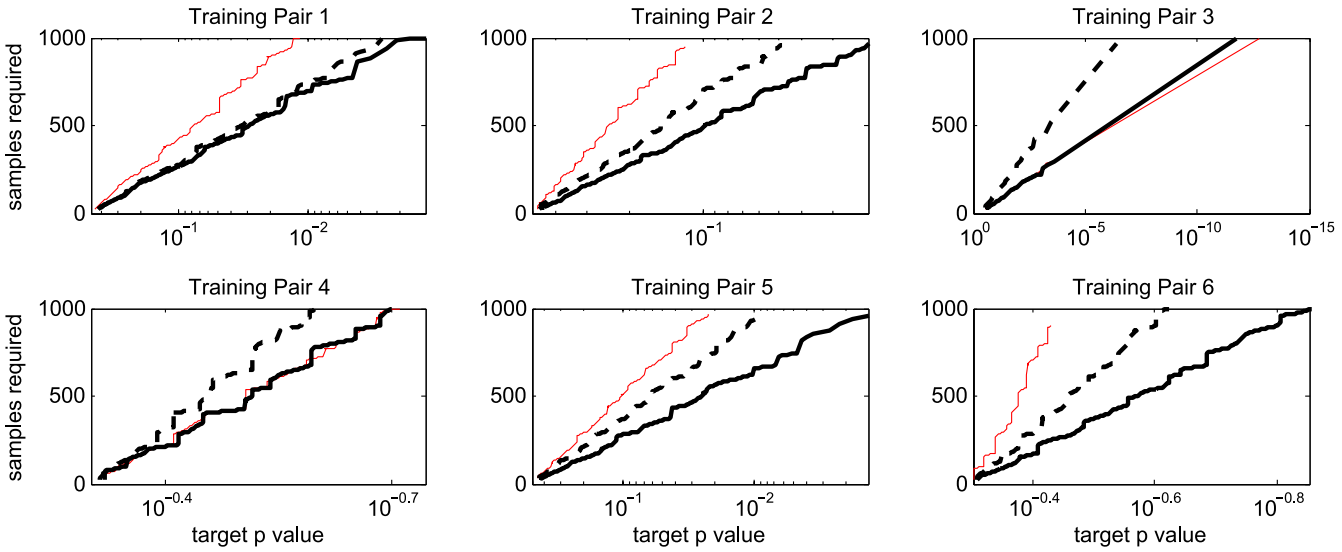

Figure 3: Comparing sample size required versus target t-test p-value in leave-one-out testing on the training set. Methods compared are baseline (red), inverse rank test (black dotted) and inverse z-test (black solid). The inverse z-test consistently performs as well as the baseline, and can be much better. Note that the different graphs vary dramatically in scale.

Table 1: Weights learned by the inverse rank test on the full gold standard training set. See Section 5.2 for a full description of the features.

ID	Feature Description (w.r.t. $\varphi(q,c)$)	Weight
1	Click	0.056693
2	Download	0.020917
3	More clicks & downloads than other team	0.052410
4a	**1**[# Clicks equal] $\times$ Title bold frac	0.083463
4b	**1**[# Clicks equal] $\times$ Abstract bold frac	0.118568
5a	Single click query AND Rank > 1	0.149682
5b	Single click query AND Rank ≤ 10	0.004950
6a	Multi-clicks AND First click	0.063423
6b	Multi-clicks AND Last click	0.000303
6c	Multi-clicks AND First click AND Rank > 1	0.015217
6d	Multi-clicks AND Click at rank $= 1$	0.018800
6e	Multi-clicks AND Click at ranks ≤ 3	-0.00419
6f	Multi-clicks AND Click at ranks ≤ 10	0.067362
6g	Multi-clicks AND Regression click	0.033067

value 1 for features 1 and 5b, leading to $\delta_w = 0.062$ (we are assuming no downloads in this scenario).

2. *Single click on result from h at rank 3*: Feature vector Ψ has value 1 for features 1, 5a and 5b, leading to $\delta_w = 0.211$. As expected, this query is judged to be more informative, since a click at rank 3 indicates a more careful selection.

3. *Single click on result from h at rank 3 followed by download*: Feature vector Ψ has value 1 for features 1, 2, 3, 5a, 5b, leading to $\delta_w = 0.285$. The download adds further evidence, which follows our intuition.

4. *One click on result from h at rank 1, followed by another click on result from h' at rank 2. Rank 2 has bolded title terms, while rank 1 has not*: Feature vector Ψ has value 1 for features 6a, 6d, and value -1 for 4a and 6b. This leads to $\delta_w = -0.002$, indicating a slight preference for h'.

6.3 Cross Validation Experiments

In this setting, we trained our models on five of the gold standard interleaving pairs and tested on the remaining one, repeating this process for all six pairs. This provides a controlled way of evaluating generalization performance. Figure 3 shows how required sample size changes as the target p-value decreases. Again, lower curves indicate superior performance. We observe the inverse z-test performing at least as well as the baseline on all except training pair 3. Note, however, that training pair 3 is an exceptionally easy case where one can achieve confident p-values with very little data. We observe the inverse rank test to also be competitve, but with somewhat worse performance.

6.4 New Interleaving Experiments

To further evaluate the methods in a typical application scenario, we trained our models on all six of the gold standard interleaving pairs, and then tested their predictions on new interleaving pairs. It should be noted that we did not examine the new interleaving dataset when developing the features described in Section 5.2. As such, this evaluation very closely matches how such methods would be used in practice.

Figure 4 shows, for all twelve test cases, how required sample size changes as the target t-test p-value decreases. We observe both learning methods consistently performing at least as well as, and often much better than, the baseline t-test (with the exception of Exp 1). We also verified that all methods and baselines agree on the direction of the preference in all cases (since we are using a two-tailed test).

Table 2 provides numerical comparisons for several standard significance thresholds. For half of the twelve test cases, the inverse z-test reduces the required sample size by at least 10% for a target significance of $p = 0.1$. For a quarter of the cases, the inverse z-test achieves a significance of $p = 0.05$ using the available data whereas

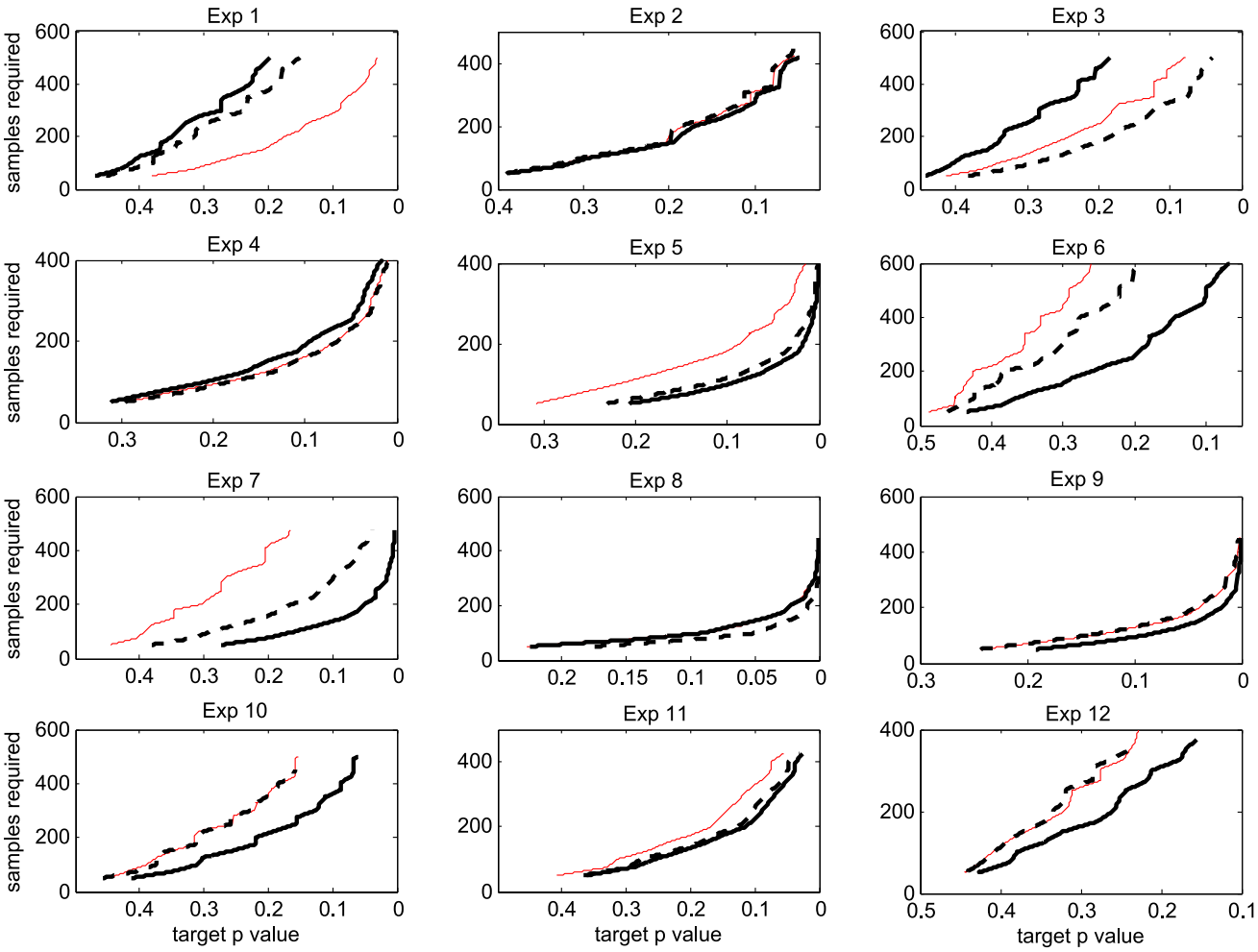

Figure 4: Comparing sample size required versus target t-test p-value in the twelve new interleaving experiments. Methods compared are baseline (red), inverse rank test (black dotted) and inverse z-test (black solid). Both the inverse rank test and inverse z-test methods outperform baseline in most cases.

the baseline t-test fails to do so. These results imply that substantial savings can be gained from employing optimized test statistics.

7. DISCUSSION AND LIMITATIONS

In this section, we discuss and summarize the core assumptions and limitations of our approach.

While the learned test statistics generally improved the power of the experiments on new retrieval function pairs (h,h'), there is likely a limit to how different the new pair may be from the training pairs. If the retrieval functions to be evaluated move far from the training data (e.g. after several iterations of improving the ranking function), it might be necessary to add appropriate training data and re-optimize the test statistic. Furthermore, we do not believe that test statistics learned on one search engine would necessarily generalize to a different collection or user population.

A key issue in generalizing to new retrieval function pairs (h,h') lies in the appropriate choice of features $\Phi(q,C,C')$. In particular, if the chosen features allow the learning algorithm to models specific idiosyncracies of the training pairs, this will likely result in poor generalization on new pairs. Furthermore, different systems may record different types of usage behavior (such as maintaining user IDs for personalization purposes). This dictates the types of features that are available to the learning methods.

Pooling the training examples from multiple training pairs (h_i,h'_j) into one joint training set might lead to unwanted results, since the learning methods optimize an "average" statistic over multiple pairs. In particular, the methods might ignore difficult to discriminate pairs in return for increased discriminative power on easy pairs. It would be more robust to minimize the maximum p-value uniformly over all training pairs.

Finally, the empirical results need to be verified in other retrieval domains. Particularly interesting are domains that include spam. It would be interesting to see whether one can learn scoring functions that recognize (and discount) clicks that were attracted by spam.

8. CONCLUSION

We have presented learning methods for optimizing the statistical power of interleaving experiments for retrieval evaluation. Given clickthrough data from interleaving pairs of retrieval functions of known relative retrieval quality, our proposed methods learn an optimized test statistic. We showed that these learned test statistics generalize to new retrieval functions, often substantially reducing the number of queries needed for evaluation.

Table 2: Sample size requirements of three target t-test p-values for the twelve new interleaving experiments.

		Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	Exp 6	Exp 7	Exp 8	Exp 9	Exp 10	Exp 11	Exp 12
Baseline	p=0.2	160	169	247	93	111	> 625	415	59	70	352	174	> 400
	p=0.1	288	310	460	161	182	> 625	> 475	95	128	> 500	328	> 400
	p=0.05	406	> 450	> 500	228	259	> 625	> 475	142	174	> 500	> 425	> 400
Inv. rank test	p=0.2	373	149	180	90	64	575	157	< 50	71	353	141	> 400
	p=0.1	> 500	313	330	160	114	> 625	296	74	129	> 500	260	> 400
	p=0.05	> 500	> 450	471	230	162	> 625	423	99	184	> 500	365	> 400
Inv. z-test	p=0.2	491	146	461	104	53	254	76	58	< 50	216	134	308
	p=0.1	> 500	275	> 500	189	97	505	137	95	94	361	222	> 400
	p=0.05	> 500	416	> 500	251	142	> 625	199	144	138	> 500	339	> 400

The idea of evaluating and learning via pairwise comparisons is attractive due to its simplicity in interpretation. In such cases, it is generally quite intuitive to design meaningful hypothesis tests. As such, the general techniques described in this paper for optimizing these statistical tests (e.g., inverse z-test) can also be applied to other domains beyond traditional information retrieval using domain-appropriate feature representations.

Acknowledgements

The work is funded by NSF Awards IIS-0812091 and IIS-0905467. The first author is also supported in part by a Microsoft Research Graduate Fellowship and a Yahoo! Key Scientific Challenges Award.

9. REFERENCES

[1] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incroporating user behavior. In *ACM Conference on Information Retrieval (SIGIR)*, 2006.

[2] E. Agichtein, E. Brill, S. Dumais, and R. Ragno. Learning user interaction models for predicting web search result preferences. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 3–10, New York, NY, USA, 2006. ACM.

[3] J. A. Aslam, V. Pavlu, and E. Yilmaz. A sampling technique for efficiently estimating measures of query retrieval performance using incomplete judgments. In *ICML Workshop on Learning with Partially Classified Training Data*, 2005.

[4] C. Buckley and E. M. Vorhees. Retrieval evaluation with incomplete information. In *ACM Conference on Information Retrieval (SIGIR)*, 2004.

[5] C. Callison-Burch. Fast, cheap, and creative: Evaluating translation quality using amazon's mechanical turk. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2009.

[6] B. Carterette, J. Allan, and R. Sitaraman. Minimal test collections for retrieval evaluation. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2006.

[7] B. Carterette and R. Jones. Evaluating search engines by modeling the relationship between relevance and clicks. In *Conference on Neural Information Processing Systems (NIPS)*, 2007.

[8] O. Chapelle and Y. Zhang. A dynamic bayesian network click model for web search ranking. In *World Wide Web Conference (WWW)*, 2009.

[9] G. Dupret and C. Liao. Cumulating relevance: A model to estimate document relevance from the clickthrough logs. In *ACM Conference on Web Search and Data Mining (WSDM)*, 2010.

[10] G. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *ACM Conference on Information Retrieval (SIGIR)*, 2008.

[11] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White. Evaluating implicit measures to improve web search. *ACM Trans. Inf. Syst.*, 23(2):147–168, 2005.

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

[13] T. Joachims. Evaluating retrieval performance using clickthrough data. In J. Franke, G. Nakhaeizadeh, and I. Renz, editors, *Text Mining*, pages 79–96. Physica/Springer Verlag, 2003.

[14] 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.

[15] D. Laming. *Sensory Analysis*. Academic Press, 1986.

[16] A. Mood, F. Graybill, and D. Boes. *Introduction to the Theory of Statistics*. McGraw-Hill, 3rd edition, 1974.

[17] F. Radlinski and N. Craswell. Comparing the sensitivity of information retrieval metrics. In *ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2010.

[18] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *Conference on Information and Knowledge Management (CIKM)*, 2008.

[19] R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 254–263, Morristown, NJ, USA, 2008. Association for Computational Linguistics.

[20] E. M. Vorhees and D. K. Harman, editors. *TREC: Experiment and Evaluation in Information Retrieval*. MIT Press, 2005.

[21] K. Wang, T. Walker, and Z. Zheng. Pskip: Estimating relevance ranking quality from web search clickthrough data. In *ACM Conference on Knowledge Discovery and Data Mining (KDD)*, 2009.

[22] Y. Yue, R. Patel, and H. Roehrig. Beyond position bias: Evaluating result attractiveness as a source of presentation bias in clickthrough data. In *World Wide Web Conference (WWW)*, 2010.