Evaluating Web Search Engines Using Clickthrough Data

Ben Carterette*
Center for Intelligent Information Retrieval
Computer Science Department
University of Massachusetts Amherst
Amherst, MA 01003
carteret@cs.umass.edu

Rosie Jones
Yahoo! Research
3333 Empire Ave
Burbank, CA 91504
jonesr@yahoo-inc.com

ABSTRACT

The web is a highly dynamic environment: documents disappear or become outdated, new documents appear, the query distribution changes. In order to keep up, search algorithms must be continuously tuned and re-tuned. Each iteratively tuned algorithm needs to be evaluated against the current web corpus. But since the web is so frequently changing, relevance judgments acquired at one point in time may not be accurate later. New relevance judgments must be made on a regular basis. Since these require human assessors to read and judge documents, they can be quite expensive to obtain.

We propose a model that leverages the millions of clicks web search engines receive each day to predict document relevance. After an initial training phase using a set of relevence judgments paired with click data, our model can predict the relevance of documents that have not been judged. These predictions can be used to evaluate the performance of a search engine. When no relevance judgments are available, we can identify the better of two ranked lists up to 85% of the time, and with only two relevance judgments for each query, we can identify the better ranking 80%–94% of the time.[1]

1. INTRODUCTION

An important, but often overlooked, part of search engine design is *evaluation*. In order to know whether one ranking function is better than another, we need to evaluate them over a common set of queries (ideally a random sample from the query distribution generated by web searchers) and a common corpus so that they can be directly compared. But evaluation is an expensive process: it requires *relevance judgments* that indicate the degree of relevance of each document retrieved for each query in a testing set. In a corpus like the web, containing billions of documents, it is impossible to judge them all, or even a substantial fraction.

Even beyond the sheer size of the corpus, web evaluation presents special challenges: the corpus is constantly changing as new documents appear, documents disappear or become obsolete, and the distribution of queries entered changes [13]. This requires even greater effort for evaluation, since new documents must be continually judged and new queries must be put into the test set. For example, a year ago today the web page for SIGIR 2006 was probably the most relevant result for the query 'SIGIR'. Today, while the SIGIR 2006 web page is still accessible, surely the SIGIR 2007 web page is more relevant to the same query. The relevance judgments must be updated; over time, the process becomes very expensive.

However, we have a readily-available source of data that could be leveraged to approximate relevance judgments: clicks. Each time a user enters a query and clicks on a result, he or she is making, in some sense, a "relevance judgment". Of course, these are very noisy: users get distracted, fail to specify their query well enough, change their interests, and so on. But looked at *in aggregate*, they may provide valuable information about the relevance of each document.

A naive approach to evaluation might compare two ranked lists based solely on the average click-through rate on both of them. This has some problems, though: a ranking with a single perfectly relevant document might have a lower overall click-through rate than one for the same query that lists many somewhat relevant documents, as the latter requires users to hunt for the information they need.

Our approach instead will be to use *discounted cumulative gain* (DCG), an evaluation metric commonly used in search engine evaluation. Using click data, we can estimate the *confidence* that a difference in DCG exists between two rankings without having any relevance judgments for the documents ranked. Furthermore, a simple algorithm guides the selection of additional documents to judge to improve confidence.

The general problem with using clicks as relevance judgments is that clicks are biased. They are biased to the top of the ranking [11], to trusted sites, to attractive abstracts; they are also biased by the type of query and by other things shown on the results page. To cope with this, we shall introduce a family of models relating clicks to relevance. By conditioning on clicks, we can predict the relevance of a document or a set of documents.

We will show how a comparison of ranking functions can be performed when clicks are available but complete rele-

*Work done while author was at Yahoo!

[1]We are submitting this paper for confidential review to be considered for publication in SIGIR July 23-27 2007.

vance judgments are not. After an initial training phase with a few relevance judgments, the relevance of unjudged documents can be predicted from clickthrough rates. The confidence in the evaluation can be estimated with the knowledge of which documents are most frequently clicked. Confidence can be dramatically increased with only a few more sporadic relevance judgments.

In section 2 we describe previous work on using clickthrough rates and on estimating evaluation metrics. Section 3 describes the evaluation of web retrieval systems using the metric *discounted cumulative gain* (DCG) and shows how to estimate the confidence that a difference exists when relevance judgments are missing. Our model for predicting relevance from clicks is described in Section 4. After a brief discussion of our data (Section 5), we return in Section 6 to the task of estimating relevance for the evaluation of search engines.

2. PREVIOUS WORK

Our work investigates low-cost evaluation in a web setting. There has been a great deal of work on low-cost evaluation in TREC-type settings ([18, 5, 14, 4] are a few), but we are aware of little for the web. Therefore we discuss previous work using clicks and other user interactions, as well as work on defining relevance in web search and evaluating web search engines.

2.1 Clicks and User Interaction

Joachims [9] was one of the first to publish on clickthrough rates. He viewed a click as a "preference judgment", so that a click at rank i indicated that the user preferred the document at i to everything ranked above it. He used these preference judgments to train a "ranking SVM".

Later, Joachims et al. [11] investigated whether the preference assumption held. They used eye-tracking devices to track what documents users looked at before clicking. They concluded that the preference relationship, while noisy, generally can be taken to be true: users tend to look at results ranked higher than the one they click on more often than they look at results ranked lower.

The problem with using clicks as preference judgments for learning is that they will tend to learn to reverse the list. The reason is that a click at the lowest rank is preferred to everything else, while a click at the highest rank is preferred to nothing else. Radlinski and Joachims [12] suggest an antidote to this: randomly swapping adjacent pairs of documents. This ensures that users will not prefer document i to document $i+1$ solely because of rank. A drawback to this approach is that we may not wish to show a suboptimal document ordering in order acquire data.

Agichtein et al.[2, 1] used and applied models of user interaction to predict preference relationships and to improve ranking functions. They use many features beyond clickthrough rate, and show that they can learn preference relationships using these features. Our work is superficially similar, but we explicitly model dependencies among clicks for results at different ranks with the purpose of learning probabilistic relevance judgments. These relevance judgments are a stronger result than preference ordering, since preference ordering can be derived from them. In addition, given a strong probabilistic model of relevance from clicks, better combined models can be built.

Dupret et al. [6] give a theoretical model for the rank-position effects of click-through rate, as well as empirically estimating the rank-position effect from data. They also build theoretical models for modeling search engine quality using them. They do not evaluate estimates of document quality, while we empirically compare relevance estimated from clicks to manual relevance judgments.

2.2 Relevance on the Web

When we are judging the relevance of web search results, there are two kinds of relevance we need to consider. The document itself, which is traditionally judged for relevance in information retrieval, has not yet seen by the user who is deciding whether to click. Instead the user sees a summary or *abstract* which may be manually or more often automatically generated to convey the contents of the document and its relevance to the query in an encapsulated fashion. We will refer to the relevance of the document as the *target* relevance, while the relevance of the abstract (judged without reference to the document) is the *perceived relevance*. In our work we shall focus on the perceived relevance.

Broder [3] introduced the idea of different classes of web search queries that reflect different user needs. He distinguishes between "informational" queries, entered by a user with a specific information need; "navigational" queries, when the user's goal is to find a certain site, and "transactional" queries, when the user wants to make a purchase. The different types of queries entail different ideas of relevance. For example, the site `monster.com` is perfectly relevant to the navigational query "monster.com", but less relevant to the (presumably) informational query "monster".

For this reason we should apply ordinal relevance judgments to web results. For a navigational query, a single document, say a home-page, may be the only possible "perfect" result, whereas for an informational query there may be many "good" results.

2.3 Evaluation

Joachims [10] investigated the use of clickthrough rates for evaluation. He showed that with some light assumptions, relative differences in performance could be measured by interleaving results from two ranking functions, then observing which function produced results that are more frequently clicked. As we will show, interleaving results can change user behavior, and not necessarily in a way that will lead to the user clicking more relevant documents.

Soboroff [13] investigated the effect of a deteriorating corpus on evaluation. He proposed methods for maintaining the relevance judgments in a corpus that is constantly changing.

Carterette et al. [4] introduced the idea of treating an evaluation measure as a *random variable* with a distribution over all possible relevance judgments. This can be used to create an optimal sampling strategy to obtain judgments, and to estimate the *confidence* in an evaluation measure. In this work we extend their methods to DCG.

3. EVALUATING SEARCH ENGINES

Discounted Cumulative Gain (DCG) [8] is an evaluation measure frequently used in web search evaluation. It is a precision-based measure: a system that ranks relevant documents highly is rewarded; the reward is discounted as documents get ranked lower. It does not include any recall component, though there are variations that contain a recall-like component as a normalization factor. DCG is more flexible

than traditional information retrieval measures such as average precision in that it supports multi-valued relevance judgments. A system that ranks very relevant documents higher is rewarded more than a system that ranks only somewhat relevant documents.

DCG takes two parameters: the maximum rank and the base of the logarithm to use in discounting. We have elected to use base 2 throughout this work.

$$DCG_\ell = rel_1 + \sum_{i=2}^{\ell} \frac{rel_i}{\log_2 i}$$

The constants rel_i indicate the relevance of the document at rank i. As described in section 2.2, relevance on the web is typically judged ordinally, with labels such as "Perfect", "Excellent", "Good", "Fair", and "Bad". In order to use these labels for evaluation, they must be mapped to constants in a way that allows more relevant documents to contribute more to the overall score. We will denote five levels of relevance a_j, with $a_1 > a_2 > a_3 > a_4 > a_5$.

3.1 Estimating DCG

Like all evaluation measures, DCG requires that the ranked documents have been judged with respect to a query. Assuming any unjudged document is not relevant could have disastrous consequences. In this section we develop the idea that DCG has a distribution over possible assignments of relevance to the unjudged documents; this will allow us to estimate DCG without knowing the relevance of some of the documents.

Let X_i be a random variable representing the relevance of document i. Since relevance is ordinal, the distribution of X_i is multinomial. Let

$$p(X_i = a_j) = p_{ij}, \quad 1 \le j \le 5$$
$$\sum_{j=1}^{5} p_{ij} = 1$$

The expectation and variance of X_i are:

$$E[X_i] = \sum_{j=1}^{5} p_{ij} a_j \tag{1}$$

$$Var[X_i] = \sum_{j=1}^{5} p_{ij}(a_j^2 - a_j)$$

We can then express *DCG* as a random variable:

$$DCG_\ell = X_i + \sum_{i=2}^{\ell} \frac{X_i}{\log_2 i}$$

Its expectation and variance are:

$$E[DCG_\ell] = E[X_1] + \sum_{i=2}^{\ell} \frac{E[X_i]}{\log_2 i} \tag{2}$$

$$Var[DCG_\ell] = Var[X_1] + \sum_{i=2}^{\ell} \frac{Var[X_i]}{(\log_2 i)^2} \tag{3}$$
$$+ 2\sum_{i=1}^{\ell} \frac{Cov(X_1, X_i)}{\log_2 i} + 2 \sum_{1<i<j} \frac{Cov(X_i, X_j)}{\log_2 i \cdot \log_2 j}$$

If the relevance of documents i and j are independent, the covariance $Cov(X_i, X_j)$ is zero.

When some relevance judgments are not available, Eq. (2) and (3) can be used to estimate confidence intervals for DCG. Thus we can compare ranking functions without having judged all the documents.

3.2 Comparative Evaluation

We may be less interested in knowing the true value of *DCG* than knowing whether there is a *difference* in the *DCG* for two or more systems. As we will see, this is a decision that can be made with very few relevance judgments. It takes considerably more to accurately measure the magnitude of the difference.

We first need to redefine *DCG* to allow arbitrary indexings of documents, instead of the indexing by rank we used in the previous section. Let $r_j(i)$ be the rank at which document i was retrieved by system j. Define $\log_2^\ell y$ such that

$$\log_2^\ell y = \begin{cases} 1 & y = 1 \\ \log_2 y & 1 < y \le \ell \\ \infty & y > \ell \end{cases}$$

Then we will define the *discounted gain* $g_{ij} = \frac{X_i}{\log_2^\ell r_j(i)}$ (defining $\frac{x}{\infty} = 0$). This is the amount that document i contributes to the total DCG_ℓ of system j.

Then we can write the difference in *DCG* for systems 1 and 2 as

$$\Delta DCG_\ell = DCG_{\ell 1} - DCG_{\ell 2}$$
$$= \sum_{i=1}^{N} g_{i1} - g_{i2} \tag{4}$$

where N is the number of documents in the entire collection. In practice we need only consider those documents returned in the top ℓ by either of the two systems. ΔDCG_ℓ is also a random variable with an expectation and variance:

$$E[\Delta DCG_\ell] = \sum_{i=1}^{N} \frac{E[X_i]}{\log_2^\ell r_1(i)} - \frac{E[X_i]}{\log_2^\ell r_2(i)}$$

$$Var[\Delta DCG_\ell] = \sum_{i=1}^{N} Var[X_i] \left(\frac{1}{\log_2^\ell r_1(i)} - \frac{1}{\log_2^\ell r_2(i)} \right)^2$$

This expression for variance assumes independence between the relevance of all pairs of documents. We do not in fact make use of the variance in this work, but we present the expression for completeness.

3.3 Confidence in DCG

Following Carterette et al. [4], we define the *confidence* in a difference in *DCG* as the probability that $\Delta DCG = DCG_1 - DCG_2$ is less than zero. If $P(\Delta DCG < 0) \ge 0.95$, we say that we have 95% confidence that system 1 is worse than system 2, given the incomplete set of relevance judgments we used to evaluate the probability.

To compute this probability, we must consider the distribution of ΔDCG. We will be calculating *DCG* with no more than 12 documents, which is not enough for a normal approximation to be accurate: Figure 1 shows a possible distribution of ΔDCG for $k = 10$. Thus we have to estimate the probability by direct calculation or Monte Carlo simulation. Simulation is trivial to implement: simply draw relevance scores for the ranked documents according to the multinomial distribution $p(X_i)$ and calculate ΔDCG using

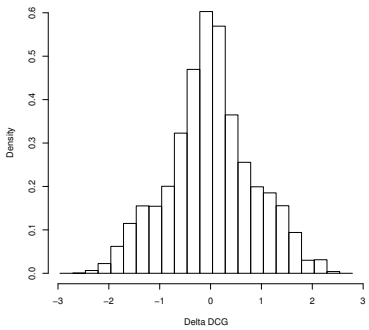


Figure 1: Simulated distribution of ΔDCG_{10} when $p(X_i)$ is assumed to be uniform. The distribution appears to be symmetric, but not normal.

those scores. After T trials, the probability that ΔDCG is less than 0 is simply the number of times ΔDCG was less than 0 divided by T.

How can we estimate the distribution $p(X_i)$? In the absense of any other information, we may assume it to be uniform over all five relevance labels. As we shall see, clicks are a useful source of information that we can leverage to estimate this distribution.

3.4 Selecting Documents to Judge

The simulation procedure above tells us how much confidence we should have in an evaluation when only incomplete sets of relevance judgments are available. We may need to obtain more relevance judgments to improve our confidence. In order to do as little work as necessary, we should not select documents arbitrarily; we should select documents that are likely to tell us a lot about ΔDCG and therefore tell us a lot about our confidence.

The method is motivated by a few simple rules of thumb:

1. If a document is retrieved at the same rank by both systems, it makes no difference to the estimate of ΔDCG. This suggests that we can ignore those documents.

2. A document ranked at the top by one system but unranked by the other is very informative.

3. A document that has a higher expected relevance is more interesting than a document with low expected relevance.

The most informative document is the one that would have the greatest effect on ΔDCG. Since ΔDCG is linear, it is quite easy to determine which document should be judged next. Eq. (4) tells us to simply choose $\max_i |E[g_{i1}] - E[g_{i2}]|$.

Algorithm 1 shows how relevance judgments would be acquired iteratively until confidence is sufficiently high.

Algorithm 1 Iteratively select documents to judge until we have high confidence in ΔDCG.

while $1 - \alpha \leq P(\Delta DCG < 0) \leq \alpha$ **do**
  $i^* \leftarrow \max_i |E[g_{i1}] - E[g_{i2}]|$
  judge document i^*
  (human annotator provides rel_{i^*})
  $P(X_{i^*} = rel_{i^*}) \leftarrow 1$
  $P(X_{i^*} \neq rel_{i^*}) \leftarrow 0$
  estimate $P(\Delta DCG)$ using Monte Carlo simulation
end while

4. MODELING CLICKS AND RELEVANCE

Our goal is to model the relationship between clicks and relevance in a way that will allow us to estimate a distribution of relevance $p(X_i)$ from the clicks on document i and on surrounding documents.

We first introduce a joint probability distribution including the query q, the relevance X_i of each document retrieved (where i indicates the rank), and their respective clickthrough rates c_i:

$$p(q, X_1, X_2, ..., X_\ell, c_1, c_2, ..., c_\ell) = P(q, \mathbf{X}, \mathbf{c}) \quad (5)$$

Boldface $\mathbf{X}$ and $\mathbf{c}$ indicate vectors of length ℓ.

If we are missing information about clicks but have information about relevance, or we are missing information about relevance but have information about clicks, we'd like to infer the missing data. We can do this by training models conditioned on different subsets of the data:

1. $p(\mathbf{c}|q, \mathbf{X})$ to predict clicks from relevance and the query;

2. $p(\mathbf{X}|q, \mathbf{c})$ to predict relevance from clicks and the query;

3. $p(q|\mathbf{X}, \mathbf{c})$ to predict the query from clicks and relevance;

4. $p(c_1, X_1|q, \mathbf{c}/c_1, \mathbf{X}/X_1)$ to predict the clickthrough rate and relevance of the first document from the clicks and relevances of the rest.

In this work we consider only the first model, but the other examples show the variety of ways this joint distribution can be used.

4.1 Predicting Relevance from Clicks

Suppose we have a query for which we have no relevance judgments (perhaps because it has only recently begun to appear in the logs, or because it reflects a trend and numerous new documents concerning the query have appeared in the corpus). We can obtain click-through data, and we would like to use that to predict relevance. In this case we are interested in the conditional probability $p(\mathbf{X}|q, \mathbf{c})$. How can we infer the relevance from the query and clicks?

Note that $\mathbf{X} = \{X_1, X_2, \cdots\}$ is a vector of ordinal variables: each X_i may take on five values, and those values are ranked from best to worst. Doing inference in this model is not easy. To simplify, we make the assumption that the relevance of document i and document j are conditionally independent given the query and the clickthrough rates:

$$p(\mathbf{X}|q, \mathbf{c}) = \prod_{i=1}^{\ell} p(X_i|q, \mathbf{c}) \quad (6)$$

This gives us a separate model for each rank, while still conditioning the relevance at rank i on the clickthrough rates at all of the ranks. We do not lose the dependence between relevance at each rank and clickthrough rates on other ranks.

The independence assumption allows us to model $p(X_i)$ using ordinal regression. Ordinal regression is a generalization of logistic regression to a variable with more than two

outcomes that can be ranked by preference. Implementations of proportional odds logistic regression can usually be found in statistical software packages such as R.

The proportional odds model for our ordinal response variable is

$$\log \frac{p(X > a_j|q, \mathbf{c})}{p(X \leq a_j|q, \mathbf{c})} = \alpha_j + \beta q + \sum_{i=1}^{\ell} \beta_i c_i + \sum_{i<k}^{\ell} \beta_{ik} c_i c_k$$

where a_j is one of the five relevance levels. The sums are over all ranks in the list; this models the dependence of the relevance of the document to the clickthrough rates of everything else that was retrieved, as well as the dependence between the clickthrough rates at any two ranks.

Learning the coefficients β_i and β_{ik} is done by likelihood maximization using iteratively reweighted least squares (IRLS). There are five intercepts α_j; these are learned by a variant of Newton's method.

After the model is trained, we can obtain $p(X \leq a_j|q, \mathbf{c})$ using the inverse logit function. Then $p(X = a_j|q, \mathbf{c}) = p(X \leq a_j|q, \mathbf{c}) - p(X \leq a_{j-1}|q, \mathbf{c})$.

Ordinal regression has one weakness: it requires a linear relationship between relevance and click-through rate. In real data, there is no such relationship; instead, perfectly relevant documents tend to be clicked on more than twice as often as less-relevant documents. The relationship between query and relevance is even less predictable and nonlinear: queries with low average clickthrough rates tend to have very high clickthrough rates on perfectly relevant documents, while queries with high average clickthrough rates tend to have a more erratice relationship to relevance.

A generalization to logistic regression is the *generalized additive model* (GAM). By fitting a function to a variable or sets of variables it can model nonlinear relationships as well as complicated dependencies. The general form of a GAM that predicts a binary response y from vector $Z = (z_1, z_2, ...)$ is:

$$\log \frac{p(y|Z)}{1 - p(y|Z)} = \alpha_0 + f(Z)$$

f is a "smoothing function" that may be fit by a method such as piecewise regression. It is in this function that the GAM can model nonlinearity and dependencies. If $f(Z) = \beta_1 z_1 + \beta_2 z_2 + \cdots$, the GAM is equivalent to logistic regression. This is therefore a very flexible model.

As GAM is a generalization of logistic regression, the *vector generalized additive model* (VGAM) is a generalization of ordinal regression. The general form of the VGAM for our model would be:

$$\log \frac{p(X > a_j|q, \mathbf{c})}{p(X \leq a_j|q, \mathbf{c})} = \alpha_j + f(q, \mathbf{c})$$

where f is a smoothing function as described above. Since the smoothing is done by a type of piecewise regression, we would need huge amounts of data to learn a function of 10 or more variables. Therefore we break the smoothing function into additive components:

$$\log \frac{p(X > a_j|q, \mathbf{c})}{p(X \leq a_j|q, \mathbf{c})} = \alpha_j + s(q) + \sum_{i=1}^{\ell} f_i(c_i) + \sum_{i<k}^{\ell} g_{ik}(c_i c_k)$$

We lose a small amount of flexibility, but gain in that we need much less data to fit the model and it is much less prone to overfitting.

Methods for finding the smoothing functions are described in Yee and Wild [17]. The VGAM is implemented in the R library VGAM. Once the model is trained, we have $p(X = a_j)$ using the same arithmetic as for the proportional odds model.

5. DATA

We obtained actual data from a large search engine company. Although we limited the data to sponsored search (advertisements), there is no reason in principle our method should not be applicable to general web search.

In this section we describe the subset of data we used, how we aggregated and cleaned it, and the constraints that it imposes.

5.1 Relevance Judgments

We have a total of 28,961 relevance judgments for 2,021 queries. The queries are a random sample of all queries entered in late 2005 and early 2006. Relevance judgments are based on details of the advertisement, such as title, summary, and URL. There are two separate scales for relevance: one consisting of five labels described earlier, the other of six labels. We will focus on the former in this work.

The relevance judgments were made by the staff assessors of the search engine company. The assessors work from instructions on how to interpret queries, guidelines for each level of relevance, etc. They are provided with some click results to give them an idea of the user's intent. Unfortunately, measurements of inter-assessor agreement are not available.

5.2 Click Logs

We obtained the sponsored search logs from the search engine company for the month of April 2006. Each record consists of a query, a search identification string, a canonicalized query, the "query class" (described below), an advertisement id, the rank the advertisement appeared at, and whether the advertisement was clicked. Thus if a user enters the query "monster.com" and receives 12 results, we will have downloaded 12 database records: one record for the result at each rank. Each of these records would have the same search ID, query, canonical query, and query class, but different advertisement ids and ranks.

We filtered out queries that we had no relevance judgments for. We used canonical query forms to filter, so that e.g. queries "Home Depot" and "home depot" would be considered the same. We then aggregated records into distinct lists of advertisements for a query as follows:

First we aggregate records by query and search ID, so for each query/search ID, we have a list of the ads displayed to the user upon entering the query and which one (if any) was clicked. Call this list L. Next we aggregate Ls over search IDs. This gives us the number of times L was displayed to all users who entered the same canonical query and the number of times each ad in L was displayed. Call the aggregated lists $\mathcal{L}$. $\mathcal{L}$ can be seen as an ordered set, which each element being a count of clicks on the ad at the corresponding rank. The clickthrough rate on each ad is simply the count in $\mathcal{L}$ divided by the impressions, the number of times L was shown to any user. After aggregation, we had about three million unique lists $\mathcal{L}$. The majority of these are missing at least one relevance judgment.

5.3 Query Class

A constraint imposed by our data is the notion of a 'query class'. Queries are manually divided into five classes: we shall refer to them as a, b, c, d, e. The classes are distinguished by differing average clickthrough rates. By modeling each class separately, we can get a sense of how our model might perform in different contexts.

Class b did not have enough data to be modeled accurately. We removed all queries from this class.

5.4 Dependence of Clicks on Entire Result List

We argue that there is a dependence between clicks at rank i and clicks at rank j. Consider a user that clicks on rank two. In that action, she has taken the additional action of *not* clicking on the other ranks. This is a subtle type of dependence. It is similar to an opportunity cost: by taking action x, we have also not taken alternative action y. This is *not* the same type of dependence as a user clicking on one result, return to the search page, and clicking on a different result. The latter is a dependence in actions over time and we do not attempt to model it. The former is a dependence in actions at a single point in time. The examples below show that this dependence is important enough that it must be modeled.

When there is an "Excellent" document ranked first, the clickthrough rate varies depending on the relevance of the document at rank 2 (Figure 2(a)). For example, a "Perfect" document at rank 2 may decrease the likelihood of a click on the "Excellent" document at rank 1, while a "Fair" document at rank 2 may *increase* the clickthrough rate for rank 1. Clickthrough rate at rank 1 more than doubles as the relevance of the document at rank 2 drops from "Perfect" to "Fair".

Likewise, when there is an "Excellent" document ranked second, the clickthrough rate varies depending on the relevance of the document ranked first (Figure 2(b)). Clickthrough rate doubles as the relevance of the first document drops from "Perfect" to "Good", then falls precipitously—perhaps because a poor result at rank 1 discourages users from looking further down the ranking.

This implies that a straight average of clickthrough rates will not accurately reflect the quality of a ranked list. Replacing a document at rank 2 with one that is more relevant will result in the clickthrough rate at rank 1 decreasing even as the clickthrough rate at rank 2 increases. Thus averaging clickthrough rates over rank will only work if the increase at rank 2 is greater than the decrease at rank 1. But this is unlikely given that clicks are so highly correlated with rank.

6. EXPERIMENTS

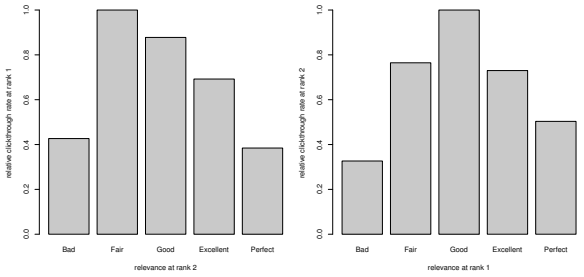
6.1 Fit of Document Relevance Model

We first want to test our proposed model (Eq. (6)) for predicting relevance from clicks. If the model fits well, the distributions of relevance it produces should compare favorably to the actual relevance of the documents.

We will compare it to a simpler model that does not take into account the click dependence. The two models are contrasted below:

$$\text{dependence model:} \quad p(\mathbf{X}|q,\mathbf{c}) = \prod p(X_i|q,\mathbf{c})$$

$$\text{independence model:} \quad p(\mathbf{X}|q,\mathbf{c}) = \prod p(X_i|q,c_i)$$



(a) Varying relevance of the document at rank 2 affects clickthrough rate at rank 1

(b) Varying relevance of the document at rank 1 affects clickthrough rate at rank 2

Figure 2: The relevance of a document at one rank can affect the clickthrough rate of a document at another rank. The relevance of the document for which CTR is measured is fixed at "Excellent" while the relevance of the other document is varied.

The latter models the relevance being conditional only on the query and its own clickthrough rate, ignoring the clickthrough rates of the other items on the page.

We removed all instances for which we had fewer than 500 impressions, then performed 10-fold cross-validation for each of four query classes. For simplicity, the query q is modeled as the aggregate clickthrough rate over all results ever returned for that query.

Both models produce a multinomial distribution for the probability of relevance of a document $p(X_i)$. Predicted relevance is the expected value of this distribution:

$$E[X_i] = \sum_{j=1}^{5} p(X_i = a_j) a_j$$

Table 1 shows the correlation between predicted relevance and actual relevance. The correlation trends downward as we move down the list. This is because lower ranks are clicked less often; there are fewer clicks to provide evidence for relevance. Correlations are highest for query classes d and e; those are the two that have the highest clickthrough rates overall. Although the correlations for the independence model are not shown, they are significantly lower at each point.

Figure 3 depicts boxplots for each value of relevance for both the models. Each box represents the distribution of predictions for the true value on the x axis. The center line is the median prediction; the edges are the 25% and 75% quantiles. The whiskers are roughly a 95% confidence interval, with the points outside being outliers.

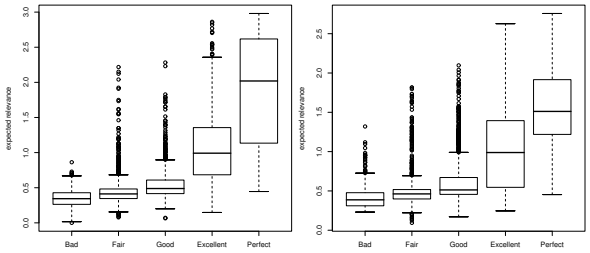
The figures show that when dependence is modeled (Figure 3(a)), the distributions are much more clearly separated from each other, as shown by the fact that there is little overlap in the boxes. The correlation between predicted and acutal relevance is 18% higher, a statistically significant difference.

6.2 Estimating DCG

Since our model appears to work fairly well, we now turn

rank	query class a	c	d	e
1	0.431	0.484	0.754	0.747
2	0.274	0.497	0.799	0.718
3	0.282	0.489	0.626	0.753
4	0.319	0.505	0.525	0.726
5	0.330	0.508	0.527	0.630

Table 1: Correlations between actual relevance and expected relevance given clickthrough rates for four query classes and the top five ranked documents.



(a) Dependence model; $\rho = 0.754$

(b) No dependence modeled; $\rho = 0.638$

Figure 3: Predicted vs. actual relevance for query class *d*, rank 1. Correlation increases 18% when dependence of relevance of the document at rank 1 on clickthrough at all ranks is modeled.

our attention to using relevance predictions to estimate DCG for the evaluation of search engines. Recall that we are interested in comparative evaluation—determining the sign of the difference in DCG rather than its magnitude. Our confidence in the sign is $P(\Delta DCG < 0)$, which is estimated using the simulation procedure described in Section 3.3. The simulation samples from the multinomial distributions $p(X_i)$ that are produced by our model.

6.2.1 Methodology

To be able to calculate the exact DCG to evaluate our models, we need all ads in a list to have a relevance judgment. Therefore our test set will consist of all of the lists for which we have complete relevance judgments and at least 500 impressions. The remainder will be used for training. The size of the test sets for each class is $543, 1601, 1720, 160$ lists, respectively for classes a, c, d, e. The training sets will include all lists for which we have at least 200 impressions, over 5000 lists for each class.

After training each model, we predict relevance for the ads in the test set. We then use these expected relevances to calculate the expectation $E[DCG]$. We will compare these expectations to the true DCG calculated using the actual relevance judgments. As a baseline for automatic evaluation, we will compare to the average clickthrough rate on the list $E[CTR] = \frac{1}{k}\sum c_i$, the naive approach described in our introduction.

We then estimate the confidence $P(\Delta DCG < 0)$ for pairs of ranked lists for the same query and compare it to the actual percentage of pairs that had $\Delta DCG < 0$. Confidence be less than or equal to this percentage; if it is, we can "trust" it in some sense.

class	$\rho(DCG, E[DCG])$	$\rho(DCG, E[CTR])$
a	0.601	0.168
c	0.336	0.203
d	0.876	0.622
e	0.792	0.422

Table 2: Spearman correlations between DCG and $E[DCG]$, and between DCG and $E[CTR]$ for each query class. Our model results in a significantly higher correlation in each query class.

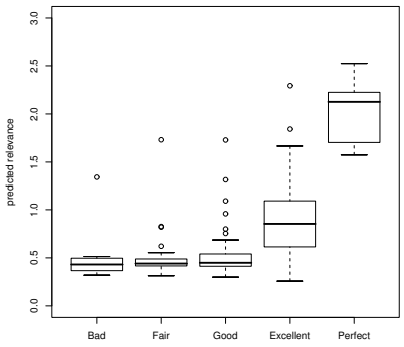


Figure 4: Actual relevance vs. predicted relevance at rank 1 for the testing set. "Perfect" and "Excellent" predictions are well-separated from the others, suggesting our model is doing a good job at distinguishing highly relevant documents.

6.2.2 Results

Table 2 shows the correlations between DCG and $E[DCG]$ calculated using the expected relevance predicted using our model, as well as the correlations between DCG and $E[CTR]$. We can see that DCG and $E[DCG]$ are well-correlated for all classes, while the correlation between DCG and $E[CTR]$ is much lower. This means we can approximate DCG better using our model than just using the mean clickthrough rate as a predictor.

Figure 4 shows actual vs. predicted relevance for ads in the test set. (This is slightly different from Figure 3: the earlier figure shows predicted results for *all* data from cross-validation while this one only shows predicted results on our test data.) The separation of the boxes shows that our model is doing quite well on the testing data, at least for rank 1. Performance degrades quite a bit as rank increases (not shown), but it is important to note that the upper ranks have the greatest effect on DCG—so getting those right is most important.

In Table 3, we have binned pairs of ranked lists by their estimated confidence. We computed the accuracy of our predictions (the percent of pairs for which the difference in DCG was correctly identified) for each bin. If we use only the relevances predicted by clickthrough rates, high confidence is generally not achieved (except for class e), but the

	accuracy							
	no judgments				two judgments			
confidence	a	c	d	e	a	c	d	e
$0.5 - 0.6$	0.569	0.448	0.522	0.504	0.563	0.452	0.572	0.554
$0.6 - 0.7$	0.701	0.650	0.617	0.674	0.576	0.486	0.678	0.711
$0.7 - 0.8$	–	–	0.734	0.947	0.758	0.495	0.697	0.915
$0.8 - 0.9$	–	–	0.818	0.988	0.927	0.706	0.890	0.856
$0.9 - 0.95$	–	–	–	0.997	0.971	0.815	0.918	0.906
$0.95 - 1$	–	–	–	0.996	0.932	0.901	0.940	0.960
overall accuracy	0.619	0.572	0.598	0.848	0.839	0.799	0.849	0.942

Table 3: Confidence vs. accuracy of predicting the better ranking for pairs of ranked lists using the relevance predictions of our model based on clicks alone (left), and with two additional judgments for each pair of lists (right). Except for class c, confidence estimates are good predictions of accuracy.

confidence estimates are generally trustworthy in the sense that they are good predictors of accuracy. If we make only two additional judgments per pair using our algorithm in Section 3, confidence is dramatically improved, and accuracy continues to be trustworthy for classes a, d, and e. Class c is an oddity: the confidence estimates are not trustworthy; in fact even with two judgments we are doing no better than random.

In general, performance is very good: using only the predictions of our model based on clicks, we have a very good sense of the confidence we should have in our evaluation. Judging only two more documents dramatically improves our confidence: there are many more pairs in high-confidence bins after two judgments.

7. CONCLUSION

We have shown how to compare ranking functions using expected DCG. After a single initial training phase, ranking functions can be compared by predicting relevance from clickthrough rates. Estimates of confidence can be computed; the confidence gives a lower bound on how accurately we have predicted that a difference exists. With just a few additional relevance judgments chosen cleverly, we significantly increase our success at predicting whether a difference exists. Using our method, the cost of acquiring relevance judgments for web search evaluation is dramatically reduced, when we have access to click data.

One limitation of our approach is that it is applicable only to ranked result lists for which we have many impressions. Thus we may not be able to estimate engine performance on rare queries, for which we have very few impressions.

Some additional questions to explore include: How many queries and relevance judgments are needed to train the initial models? We observed that the more data available, the better our models fit and the better predictions they provided. Considering that it is a one-off cost, it may be worthwhile to spend a lot on many high-quality judgments.

Would additional user interaction features improve our models? Agichtein et al. [2] showed that features such as last document clicked, time spent on page, and other features can provide high gains over just clicks. If our models were improved, either by training them on more data or by using more features in training, the DCG predictions they provide would be even better.

Finally, our probabilistic relevance judgments could be useful for other tasks beyond evaluation. They may be useful for learning if they are considered to be noisy labels. Thus, this could provide a huge increase in the amount of training data available to search scientists and engineers.

8. ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval and in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-06-C-0023. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect those of the sponsor.

9. REFERENCES

[1] E. Agichtein, E. Brill, and S. T. Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings SIGIR*, pages 19–26, 2006.

[2] E. Agichtein, E. Brill, S. T. Dumais, and R. Ragno. Learning user interaction models for predicting web search result preferences. In *Proceedings SIGIR*, pages 3–10, 2006.

[3] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, 2002.

[4] B. Carterette, J. Allan, and R. K. Sitaraman. Minimal test collections for retrieval evaluation. In *Proceedings of SIGIR*, pages 268–275, 2006.

[5] G. V. Cormack, C. R. Palmer, and C. L. Clarke. Efficient Construction of Large Test Collections. In *Proceedings of SIGIR*, pages 282–289, 1998.

[6] G. Dupret, B. Piwowarski, C. Hurtado, and M. Mendoza. A statistical model of query log generation. In *SPIRE*, LNCS 4209, pages 217–228. Springer, 2006.

[7] T. Hastie and R. Tibshirani. Generalized additive models. *Statistical Science*, 1:297–318, 1986.

[8] K. Jarvelin and J. Kekalainen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[9] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of KDD*, pages 133–142, 2002.

[10] T. Joachims. Evaluating retrieval performance using clickthrough data. In *Text Mining*, pages 79–96. 2003.

[11] T. Joachims, L. A. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of SIGIR*, pages 154–161, 2005.

[12] F. Radlinski and T. Joachims. Minimally invasive randomization fro collecting unbiased preferences from clickthrough logs. In *Proceedings of AAAI*, 2006.

[13] I. Soboroff. Dynamic test collections: measuring search effectiveness on the live web. In *Proceedings of SIGIR*, pages 276–283, 2006.

[14] I. Soboroff, C. Nicholas, and P. Cahan. Ranking Retrieval Systems without Relevance Judgments. In *Proceedings of SIGIR*, pages 66–73, 2001.

[15] L. Wasserman. *All of Nonparametric Statistics*. Springer, 2006.

[16] S. N. Wood. Thin plate regression splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(1):95–114, 2003.

[17] T. W. Yee and C. J. Wild. Vector generalized additive models. *Journal of the Royal Statistical Society, Series B (Methodological)*, 58(3):481–493, 1996.

[18] J. Zobel. How Reliable are the Results of Large-Scale Information Retrieval Experiments? In *Proceedings of SIGIR*, pages 307–314, 1998.