Evaluating Search Engines by Clickthrough Data

Jing He and Xiaoming Li

Computer Network and Distributed System Laboratory, Peking University, China
{hejing,lxm}@pku.edu.cn

Abstract. It is no doubt that search is critical to the web. And it will be of similar importance to the semantic web. Once searching from billions of objects, it will be impossible to always give a single right result, no matter how intelligent the search engine is. Instead, a set of possible results will be provided for the user to choose from. Moreover, if we consider the trade-off between the system costs of generating a single right result and a set of possible results, we may choose the latter. This will naturally lead to the question of how to decide on and present the set to the user and how to evaluate the outcome.

In this paper, we introduce some new methodology in evaluation of web search technologies and systems. Historically, the dominant method for evaluating search engines is the Cranfield paradigm, which employs a test collection to qualify the systems' performance. However, the modern search engines are much different from the IR systems when the Cranfield paradigm was proposed: 1) Most modern search engines have much more features, such as snippets and query suggestions, and the quality of such features can affect the users' utility; 2) The document collections used in search engines are much larger than ever, so the complete test collection that contains all query-document judgments is not available. As response to the above differences and difficulties, the evaluation based on implicit feedback is a promising alternative employed in IR evaluation. With this approach, no extra human effort is required to judge the query-document relevance. Instead, such judgment information can be automatically predicted from real users' implicit feedback data. There are three key issues in this methodology: 1) How to estimate the query-document relevance and other useful features that useful to qualify the search engine performance; 2) If the complete "judgments" are not available, how can we efficiently collect the most critical information from which the system performance can be derived; 3) Because query-document relevance is not only feature that can affect the performance, how can we integrate others to be a good metric to predict the system performance. We will show a set of technologies dealing with these issues.

1 Introduction

Search engine evaluation is critical for improving search techniques. So far, the dominant method for IR evaluation has been the Cranfield evaluation method. However, it also has some disadvantages. First, it is extremely labor intensive to creating relevance judgments. As a result, we often have a limited number of

P.F. Patel-Schneider et al.(Eds.): ISWC 2010, Part II, LNCS 6497, pp. 339–354, 2010.

queries to experiment with. Second, the Cranfield paradigm was proposed for evaluating traditional information retrieval (IR) systems, but it cannot reflect some new features in modern search engines. The modern search engines usually provide more than a ranked document list, such as snippet and related query suggestion. The quality of such new features can affect search users' utility.

As a very promising alternative, automatic evaluation of retrieval systems based on the implicit feedback of users has recently been proposed and studied. One important direction is to leverage the large amount of clickthrough data from users to evaluate retrieval systems. Since clickthroughs are naturally available when users use a search engine system, we can potentially use this strategy to evaluate ranking methods without extra human effort. On the other hand, the clickthrough data can not only reflect the quality of the retrieved documents, but also some other features of search engines. Both factors make it an attractive alternative to the traditional Cranfield evaluation method.

In this paper, we introduce two categories of methods of evaluating search engines based on clickthrough data. The methods of first category are for comparing two search engines. The basic idea of this method is to interleave the retrieved results from two search engines, and the search engine which gets more click on its results wins. The methods of second category infer the document relevance first and utilize the relevance information to evaluate the search engines. The document relevance is usually estimated from a probabilistic click model, and we can reorder the retrieved documents to more efficiently collect the relevance information for evaluation. Finally, we propose a new metric that is able to embed snippet generation quality in the evaluation.

2 Rule Based Methods: Interleaving and Comparing

The basic idea of methods in this category is to interleave the search results returned by different systems for the *same* query in a somewhat random manner and present a merged list of results to the user. The users would then interact with these results in exactly the same way as they would with normal search engine results, i.e., they would browse through the list and click on some promising documents to view. The clickthroughs of users would then be recorded and a system that returned more clicked documents would be judged as a better systems.

We can easily see that any method in this category consists of two functions: (1) an *interleaving function* which determines how to combine the ranked lists of results returned from two systems, and (2) a *comparison function* which decides which ranked list is preferred by the user based on the collected user clickthroughs. In general, the interleaving function and the comparison function are "synchronized" to work together to support relative comparison of retrieval systems. There have been two major methods proposed to instantiate these two functions: the balanced interleaving method [1] and the team-draft interleaving method [2].

2.1 Methods

The balanced interleaving method was proposed in [1]. It merges two lists by taking a document from each list alternatively so as to ensure that the numbers of documents taken from both lists differ by at most one [1]. Specifically, its interleaving function works as follows. It starts with randomly selecting a list (either A or B) as the current list L. It then iteratively pops the top document d from the current list L and appends d to the merged list M if d is not already in M. After each iteration, it would update the current list so that it would point to a different list. This process is repeated until the two lists are empty. To tell which system (list) performs better, the balanced interleaving method looks at which documents are clicked by the user and assumes that list A performs better than B if A has contributed more clicked documents in the stage of constructing the merged list than B.

Table 1. Example for Interleaving Strategies

interleaving	A	(a, b, c, d)
lists	B	(b, c, a, d)
balanced	A-B	(a, b, c, d)
	B-A	(b, a, c, d)
team draft	A-B-A-B	$(a^{(A)}, b^{(B)}, c^{(A)}, d^{(B)})$
	A-B-B-A	$(a^{(A)}, b^{(B)}, c^{(B)}, d^{(A)})$
	B-A-B-A	$(b^{(B)}, a^{(A)}, c^{(B)}, d^{(A)})$
	B-A-A-B	$(b^{(B)}, a^{(A)}, c^{(A)}, d^{(B)})$

The team-draft interleaving method was proposed in [2] to prevent some bias in the balanced interleaving method. In each round, it start with randomly selecting a list L(either A or B), and appends M with L's most preferred document that has not been in M. And then it turns to the other list and does the same thing. The rounds continue until all the documents in A and B are in M. To predict which system(list) performs better, the team-draft method counts the number of clicked documents selected from list A and B respectively. It assumes that A performs better than B if there are more clicked documents selected from A. Formally, it scores system A by $score(A) = \sum_{d_c} \delta(d_c \in T_A)$, where d_c is a clicked document and δ is a binary indicator function.

In Table 1, we show examples of merging lists A and B using balanced and team-draft function.

The common drawback of both methods is that they are not sensitive to the positions of the clicked documents in the ranked lists. We propose an improvement to the balanced method to overcome this limitation. Specifically, in this new method (called preference-based balanced interleaving), we would interleave the ranked lists in the same way as the balanced method, but make prediction about which system is better based on a new preference-based comparison function (ppref).

A preference relation between two documents indicates that one document is more relevant than the other (with respect to a query), which we denote by $d_i >_p$

d_j. It has been shown that some preference relations extracted from clickthrough data are reliable [3]. We would first try to convert the collected clickthroughs into a set of preference relations based one two rules: (1) a clicked document are preferred to the skipped documents above it; (2) a clicked document is more relevant than the next unclicked one. Both rules have been justified in some previous work [3]. Now, our key idea is to design a preference-based measure to score each ranked list by treating these inferred *incomplete* preference relations between documents as our golden standard. In this study we use precision of preference(ppref)[4].

2.2 Evaluation of Interleaving Strategies

In this section, we propose a simulation-based approach for evaluating and comparing different interleaving methods systematically. Our basic idea is to first systematically generate many test cases, each being a pair of ranked lists of documents, and then apply each interleaving method to these test cases and evaluate its performance. Thus our overall approach consists of two components: 1) metrics to be used for measuring the performance of an interleaving method or comparing two methods, and 2) simulation strategies to be used to generate the test cases.

Metrics. Intuitively, we would expect an ideal method to merge the two lists in such a way that we can differentiate the ranking accuracy of the two systems accurately based on the collected clickthroughs. Thus our first criterion is the *effectiveness of differentiation* (EoD) of a method. Moreover, it is also important that the utility of the merged list from a user's perspective is high. Thus a second criterion that we consider is the *utility to users*(UtU) of a method.

To quantify the utility for users of an interleaving method, in general, we may apply any existing IR relevance evaluation measures (MAP is this paper) to the merged result list generated by the interleaving method. The effectiveness of differentiation of a method can be measured based on the accuracy of the method in predicting which system is better. Again, since we use simulation to generate test cases, we will have available the utilities of the two candidate lists (by MAP is this paper) and decide which is better. By comparing the prediction result of an interleaving method with this ground truth, we may measure the accuracy of the prediction. Since there are three types of results when comparing system A and B, i.e., (1) System A wins; (2) System A loses; or (3) the two systems tie. In general, we can associate different costs with different errors. However, it is not immediately clear how exactly we should set these costs. Leaving this important question for future research, in this paper, we simply assume that all correct predictions have zero cost, all "tie errors" have a cost of 1, and all opposite predictions have a cost of 2. With these costs, given two candidate ranked lists and the prediction of an interleaving method, we will be able to measure the accuracy of the method with the cost of the prediction; a lower cost would indicate a higher accuracy. Since the cost value is presumably not sensitive to specific queries, the absolute value of cost is meaningful, but we mainly use it to compare different interleaving methods in this paper.

Generation of Test Cases. A test case for our purpose consists of the following elements: (1) two ranked lists of documents of length n: A and B, (2) a set of n_r documents assumed to be relevant, R, and (3) the number of documents a user is assumed to view, K ($K \leq n$). We assume that once a merged list of results are presented to the user, the user would view sequentially (starting from the top) K results and click on any relevant document encountered. Thus, given a merged list of results, the clickthroughs can be uniquely determined based on R and K. Thus in general, we can use a straightforward strategy to generate a large random sample to compare different interleaving methods. However, such a blind test provides limited information on how an interleaving method performs in different scenarios.

It would be much more informative and useful to compare different interleaving methods in various scenarios such as known item search vs. high-recall search (modeled by relevant document number n_r), comparing two systems that have similar retrieval results vs. very different retrieval results (modeled by kendall's τ, comparing two similar-performance or different-performance retrieval systems (modeled by relative average precision RAP) or obtaining clickthroughs from a patient user vs. an impatient user (modeled by viewing number K). Thus each such possible scenario should be simulated separately to understand relative strengths and weaknesses of two methods in these different scenarios. It is possible that we may find out that some method tends to perform better in some scenarios, while others perform better for other scenarios. We can stop the sampling process when we have sufficient test cases to obtain a relatively reliable estimate of the UtU and EoD of the interleaving methods being evaluated.

Results. In our experiments, we control the first parameter n by setting it to 10 and vary all the other parameters to simulate different evaluation scenarios. The ground truth about which system is better is decided based on the average precision of the top 10 documents for each system. Variations of other parameters and the scenarios simulated are summarized in Table 2.

We show the results of these three methods in all the different scenarios in Table 3. Because the UtU results for these three methods are very similar, it is

Table 2. Summary of Evaluation Scenarios

Scenario	Variation	Parameter Setting
Topic	Known-Item Search	$n_r = 1$
	Easy Topic	Prec@10Doc = 0.6
	Difficult Topic	Prec@10Doc = 0.3
Result Similarity	High	$0.5 < \tau < 1.0$
	Low	$-1 < \tau < -0.5$
Precision Similarity	High	$0 < RAP < 0.2$
	Low	$0.3 < RAP$
User Patience	High	$k = 8$
	Medium	$k = 5$
	Low	$k = 3$

Table 3. MAP and Cost for Specific Scenarios

Topic	Result Sim.	Prec. Sim.	K	Cost of Prediction		
				Balanced	Team	Preference
Known-item	Low	Low	3	**0.61**(0.24)	0.63(0.25)	**0.61**(0.24), $(0.00, 0.03^-)$
			5	**0.38**(0.23)	0.41(0.27)	**0.38**(0.23), $(0.00, 0.07^-)$
			8	**0.15**(0.13)	0.18(0.19)	**0.15**(0.13), $(0.00, 0.17^-)$
	Low	High	3	**1.00**(0.00)	**1.00**(0.00)	**1.00**(0.00), (0.00, 0.00)
			5	**0.93**(0.06)	0.95(0.08)	**0.93**(0.06), $(0.00, 0.02^-)$
			8	**0.60**(0.24)	0.70(0.32)	**0.60**(0.24), $(0.00, 0.14^-)$
	High	High	3	**1.00**(0.00)	**1.00**(0.00)	**1.00**(0.00), (0.00, 0.00)
			5	**0.80**(0.23)	0.87(0.29)	**0.80**(0.23), $(0.00, 0.08^-)$
			8	**0.26**(0.19)	0.48(0.47)	**0.26**(0.19), $(0.00, 0.46^-)$
Hard	Low	Low	3	**0.31**(0.23)	0.31(0.26)	**0.31**(0.21), (0.00, 0.00)
			5	0.26(0.22)	0.27(0.22)	**0.11**(0.11), $(0.58^-, 0.59^-)$
			8	0.35(0.33)	0.27(0.25)	**0.13**(0.20), $(0, 63^-, 0.52^-)$
	Low	High	3	0.85(0.27)	0.86(0.34)	**0.81**(0.40), $(0.05^-, 0.06^-)$
			5	0.77(0.36)	0.79(0.42)	**0.68**(0.55), $(0.12^-, 0.14^-)$
			8	0.73(0.44)	0.72(0.47)	**0.57**(0.64), $(0.22^-, 0.21^-)$
	High	High	3	0.78(0.24)	0.84(0.31)	**0.75**(0.47), $(0.04^-, 0.11^-)$
			5	0.68(0.37)	0.73(0.56)	**0.56**(0.35), $(0.18^-, 0.23^-)$
			8	0.69(0.48)	0.66(0.57)	**0.39**(0.36), $(0.43^-, 0.41^-)$
Easy	Low	Low	3	0.29(0.22)	0.32(0.24)	**0.06**(0.08), $(0.79^-, 0.81^-)$
			5	0.17(0.15)	0.19(0.15)	**0.01**(0.02), $(0.94^-, 0.95^-)$
			8	0.15(0.16)	0.07(0.07)	**0.00**(0.00), $(1.00^-, 1.00^-)$
	Low	High	3	0.71(0.48)	0.72(0.55)	**0.54**(0.63), $(0.24^-, 0.25^-)$
			5	0.66(0.46)	0.66(0.53)	**0.41**(0.50), $(0.28^-, 0.28^-)$
			8	0.65(0.42)	0.56(0.43)	**0.34**(0.48), $(0.48^-, 0.39^-)$
	High	High	3	0.69(0.31)	0.74(0.56)	**0.59**(0.36), $(0.14^-, 0.20^-)$
			5	0.77(0.35)	0.67(0.60)	**0.45**(0.35), $(0.41^-, 0.33^-)$
			8	0.85(0.38)	0.58(0.57)	**0.37**(0.35), $(0.56^-, 0.36^-)$

omitted here due to the space constraint. In general, the preference method performs better than the other two methods and the balanced method is preferred to the team-draft method. For known-item search, it's always better to use the balanced or preference method. For searches with more relevant documents, if users are expected to view very few top documents, the balanced and preference method is also preferred, but when the user is patient and willing to view more documents, team-draft and preference method may be more appropriate.

3 Evaluation Based on Click Models

The interleaving method can compare the performance of two ranked list for a specific query. However, it has two problems. First, we cannot get the confidence level about the comparison result. Second, it is a little expensive to compare the search engine pairly to get the relative performance for a large number of search engines.

To address the these problems, various unsupervised graphical click models have been recently proposed [5,6,7,8]. Click models connect the document relevance and users' behaviors with probability graphical models. They provide a principled approach to infer relevance of the retrieved documents. In general, an examined document is inferred to be more relevant if it is clicked. The click models can not only give the document relevance estimation, but also give how reliable it is.

3.1 Click Models

Click models usually model two types of search user behaviors: *examination* and *click*. Once a user submitted a query, the search engine returns a ranked list of snippets, each of which corresponds to a retrieved document (the document list is denoted as $D = \{d_1, \ldots, d_M\}$). Then, the user usually *examine* these snippets and *click* on those which seem to be relevant to his information need. Usually, the examination and click events on a document d_i are modeled as two binary variable (denoted as E_i and C_i respectively). Then the documents relevance are connected with these events by some hypotheses.

One category of hypotheses define the click behavior. The most commonly used hypothesis in this category is *examination hypothesis* [9]. For a document d_i, it defines that the click behavior (C_i) depends on both examination (E_i) and document relevance (r_{d_i}). When a document is not examined, it cannot be clicked. And once it is examined, the probability of click is proportional to its relevance.

Another category of hypotheses defines the examination behavior. Most existing work has a common assumption called *cascade hypothesis*, stating that users examine the document in a top-down manner, i.e., if one document at rank i is not examined, all documents below it would not be examined. Besides the *cascade hypothesis*, click models define different functions determining the examination probability. The *cascade model* [10] assumes that the users would continue examine until the first click, and they stop examining after first click. The *dependent click model* [8] assumes that users continue examining until a click, and the probability of keeping examining after a click depends on the position. The *click chain model* [5] assumes that users may stop even when they does not click, and the probability of keeping examination after click depends on clicked document's relevance. While the *user browsing model* [6] does not use the *cascade hypothesis* and assumes that the examination probability is determined by its absolute rank and its distance to the previous clicked document.

All examination and click events can be modeled as nodes in the graphical click model, and the hypotheses on relationship between these events are modeled as edges in the graph. In this graphical click model, only the click variables are observed. The task is to estimate the parameters such as document relevance, and it usually can be done in EM algorithm (or other more efficient method for some specific model). With these models, we can estimate the how these document relevances distribute, so we can not only get the expected relevance value, but also know how reliable the estimation is. In general, the relevance

estimation for a document is more reliable if it is examined more frequently. It has shown that *click chain model* (denoted as CCM) can predict user behavior accurately with the document relevance it estimates. In the later experiments, we use CCM to predict the document relevance.

3.2 Efficiently Collecting Relevance Information

With the relevance information collected by a click model, we can use them to evaluate a search engine with some graded relevance based IR metrics (such as *DCG* or *RBP*). One challenge of using this method is that the evaluation results may be questionable for some tail queries, due to the limited number of query submissions. Intuitively, the retrieved documents contribute differently for evaluating IR systems. The main idea here is to measure the benefit of collecting a document's relevance, thus it can guide us collect the relevance of documents which can bring more benefit.

The first intuition is that the benefit of examining a document is affected by its relevance uncertainty reduction. For example, if we have been very confident about the relevance of a document, it provides little information by further examining this document. Therefore, it is a good choice to move up the document with larger relevance uncertainty reduction. Naturally, the relevance uncertainty of a document can be measured by the variance of its inferred relevance, i.e., the larger variance is, we are more uncertain about the relevance. We can formulate the inferred relevance variance reduction from examining a document d_i as follows:

$$\begin{aligned}\Delta V(r_i) = & P(c_i = 1|\hat{r_i})\Delta_1 V(r_i) \\ & + P(c_i = 0|\hat{r_i})\Delta_0 V(r_i)\end{aligned} \tag{1}$$

Where r_i and $\hat{r_i}$ are the inferred and actual relevance of d_i respectively; $P(c_i = 1|\hat{r_i})$ and $P(c_i = 0|\hat{r_i})$ are clicking and skipping probability given the actual relevance level $\hat{r_i}$ respectively. According to *examination hypothesis*, $P(c_i = 1|\hat{r_i} = r) = r$. Unfortunately, we don't know the exact value of $\hat{r_i}$ in reality, so we approximate by replacing it with inferred relevance r_i. $\Delta_1 V(r_i)$ and $\Delta_0 V(r_i)$ are variance reduction for clicking and skipping cases respectively.

The second intuition is that we should encourage the users to examine deeper, because it helps to collect more documents' relevance information. The users would generally stop examining when their information need has been satisfied. To encourage the users to examine deeper, we can delay their satisfaction by moving the relevant documents down. Though this strategy obviously sacrifices the user utility, the effect may not be very serious because only a very limited percent of queries are employed for evaluation purpose in real search engine. Assuming the ranked list is $(d_1, \ldots, d_n)$, the *list benefit function* $b(d_1, \ldots, d_n)$ can be defined as:

$$b(d_1, \ldots, d_n) = \sum_i P(e_i = 1) b_1(d_i) \tag{2}$$

Generally, the users examine the top document, but the probability of examining deeper documents depends on the documents ranked above. The above document relevance factor can be plugged in the list benefit function:

$$b(d_1, \ldots, d_n) = b_1(d_1) + P(e_2 = 1|\hat{r_1})b(d_2, \ldots, d_n)$$

Where $P(e_2|\hat{r_1})$ is the probability of continuing examining document d_2 given actual relevance $\hat{r_1}$.

Obviously, an optimal presented document list is to maximize the benefit function. Unfortunately, this problem is intractable. We approximate it by a greedy algorithm: at each step, we select the document that leads to maximal benefit and append it to the end of the presented list. We approximate the benefit of examining below by the maximal document benefit of the unselected documents. Thus, the weight of a document d_i in an unselected document set D can be formulated as:

$$w(d_i) = b_1(d_i) + P(e_{next}|r_i) \max_{\substack{d_j \in D \\ i \neq j}} b_1(d_j) \tag{3}$$

The third intuition is that the highly ranked documents contribute more to the overall performance score. Most IR metrics model this in their formulas. For example, in DCG, the document at rank k can weighted as $\frac{1}{log_2(k+1)}$; Thus it generally requires to infer relevance of highly ranked document more accurately. Click models can infer a distribution of document relevance, so the mean and variance of DCG value distribution can be expressed as:

$$E(DCG(A)) = \sum_{d_i \in A} \frac{E(r_i)}{\log_2(A_i + 1)}$$

$$V(DCG(A)) = \sum_{d_i \in A} \frac{V(r_i)}{\log_2^2(A_i + 1)}$$

In evaluating one search engine, the purpose is to reduce the uncertainty of the metrics score, so the contribution of each document does not only depend on its variance reduction but also its original rank. The weight of document at rank k is $1/\log_2^2(k+1)$, so the benefit of examining document d_i is:

$$b_2(d_i) = \frac{\Delta V(r_i)}{\log_2^2(A_i + 1)} \tag{4}$$

For list benefit function(Equation 2) and document weight function(Equation 3), we can replace the document benefit function $b_1(d_i)$ by $b_2(d_i)$.

Finally, in the context of comparing two systems, retrieved documents have different effect on distinguishing two systems. Therefore, in addressing the comparison problem, it benefits from moving up the documents that are ranked

differently. The mean and variance of DCG score difference from two ranked lists A and B can be expressed as:

$$E(\Delta DCG_{A,B}) = \sum_{d_i \in A \cup B} (W_A(d_i) - W_B(d_i))V(r_i)$$

$$V(\Delta DCG_{A,B}) = \sum_{d_i \in A \cup B} (W_A(d_i) - W_B(d_i))^2 V(r_i)$$

$$W_S(i) = \begin{cases} \frac{1}{\log_2(S_i+1)} & \text{if } d_i \in S \cdot \\ 0 & \text{otherwise} \end{cases}$$

We expect to reduce the uncertainty of ΔDCG value, so one document's contribution to the overall performance difference can be formulated as:

$$b_3(d_i) = (W_A(d_i) - W_B(d_i))^2 \Delta V(r_i) \tag{5}$$

The corresponding list benefit and document weight can be expressed by replacing $b_1(d_i)$ to $b_3(d_i)$ in Equation 2 and 3 respectively.

3.3 Experiments and Results

The first experiment is a user study, which is designed to resemble the common search engine usage. We recruited 50 college students to answer ten questions with the help of our system. These questions contain both open and close questions and vary in difficulty and topic. For each question, we designed a query and collected 10 results from search engines A and B respectively. A user was presented with a question following ten ranked snippet results in a page. The user can answer the question by examining and click the results.

We implemented five reordering functions in our system: three of them presented the results A from one search engine, and two of them interleaved results A and B from two search engines. For one-system results reordering, the baseline function ($base1$) presents the results A unchanged. Two other reordering functions ($fun1$ and $fun2$) use benefit function in Equation 1 and Equation 4 respectively. For two-system results reordering, the baseline function($base2$) presents the results A and B alternatively, i.e., balanced interleaving method used in [1]. The another reordering function($fun3$) determines the presented list using benefit function in Equation 5.

As a golden standard, we asked three assessors to provide the relevance judgments for the results. The relevance level is then normalized into a value between 0 and 1. The engine's performance on a query can be measured by DCG based on the actual relevance judgments(denoted as $D\hat{C}G$). As mentioned, DCG can also be calculated based on the inferred relevance(denoted as DCG).

For one-system evaluation task, it is measured by the relative difference (denoted as $rerr$) between $D\hat{C}G$ and DCG. For two-system comparison task, it is measured by the ratio that predicted comparison result is incorrect.

The results from the user study is presented in Table 4. In the one-system evaluation results we can find that the both $fun1$ and $fun2$ perform much

Table 4. User Study Results

ID	Evaluation			Comparison	
	base1	fun1	fun2	base2	fun3
1	0.77	0.27	**0.26**	C	C
2	1.33	1.47	**1.32**	E	C
3	**1.64**	1.97	2.07	C	C
4	1.32	**0.45**	0.69	C	C
5	0.43	**0.09**	0.32	E	C
6	1.61	**0.48**	0.58	C	C
7	**0.43**	0.47	1.24	C	C
8	0.70	0.24	**0.02**	C	C
9	**0.01**	0.02	0.17	C	C
10	0.47	0.44	**0.26**	E	C
All	0.87	**0.59**	0.69	0.30	**0.00**

better than $base1$ on most questions. In the two-system comparison results, C denotes correct and E denotes error. We find that $fun3$ can distinguish two search engines accurately on all questions but $base3$ can do only 7 out of the 10 questions.

We further conduct a simulation study. The simulation experiment is deployed in the similar manner as that of Section 2.2. But the user behavior is synthesized from a click model instead of clicking all relevant documents. For one-system evaluation problem, we conduct the experiment for $base1$, $fun1$ and $fun2$. For two-system evaluation problem, besides $base2$ and $fun3$, we also test $fun1$ and $base3$.

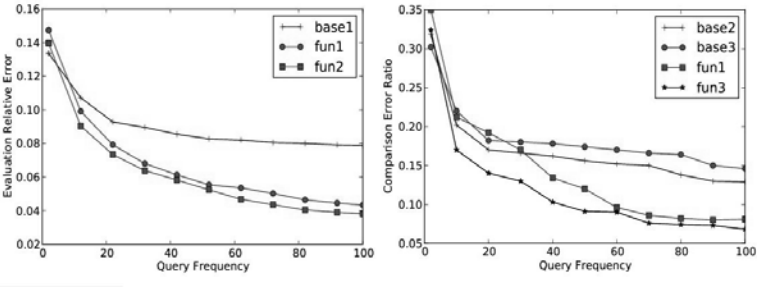

Fig. 1. One-system Evaluation (Left) and Two-system Comparison (Right)

In $base3$, it uses the balanced strategy[1] to merge the documents from the two ranked lists into the presented list. We present the one-system evaluation and two-system comparison results in Figure 1.

The experimental results suggest that at the first few query submissions, $base1$ and $base3$ are good choices for one-system evaluation and two-system comparison. When the query is submitted a little more, $fun2$ and $fun3$ are generally good choices for one-system evaluation and two-system comparison problems respectively.

4 Beyond Document Relevance

All methods use document relevance to evaluate the quality of information retrieval component. There are some research to investigate the correlation between the Cranfield paradigm experiment and user study. Some recent evaluation work [11,12] reported the positive correlation between the Cranfield evaluation based on document relevance and real user study. However, in the use of a real search engine, the effectiveness and the satisfaction of a user is also affected by some factors other than retrieval system performance. Our problem is: how to embed the quality of other components in search engine for the evaluation.

The document snippet is a very important feature for modern search engine. Good snippets can guide the users to find out the relative documents from the retrieved results, or even contain relative information itself. On the contrary, users may miss relevant documents or waste time clicking and examining irrelevant documents due to bad snippets. Turpin et al. [13] investigated this problem and showed that it makes difference by including snippet quality in search engine evaluation. In this paper, we interpret traditional IR metric *precision* as *effective time ratio* of the real user, i.e., the ratio between the time used in reading relevant information and the total search time, and extend it in the scenario when the search engines provide document snippets.

4.1 Effective Time Ratio

Intuitively, for a search engine with snippets, the users' satisfaction is affected by both retrieval system and snippet generation qualities. *Precision* is one of most important metrics in IR. In order to derive the metric that can be used to evaluate search engines with document snippet, we first interpret *precision* as *effective time ratio* (denoted as ETR) when using retrieval system without snippet presentation.

Definition 1. *(Effective Time Ratio): Effective Time Ratio is the ratio between effective time used in get relevant information and the total search time.*

We assume an IR system presents to the user a ranked list of documents for a query. We further assume the time spent on examining each document is identical (denoted as T). Thus a user needs $T \cdot N$ time to examine top N documents, but the *effective time*, which is used to examine relevant documents, is $T \cdot \sum_{i=1}^{N} R(d_i)$, where d_i is the i-th ranked document in the result list, and R is a binary function, indicating relevance of a document. With very simple derivation, we can find that $ETR@N$ is identical to $P@N$, so we can interpret *precision* as *effective time ratio* when using the retrieval system without providing document snippets.

However, the modern search engines usually present the users a list of snippets instead of documents. In this scenario, a user examines i-th snippet, which is generated from i-th ranked document (i is initially assigned 1). If he finds the snippet is relevant, he would click and examine the corresponding document; otherwise, he would examine the next snippet. After examining a document, the

users may quit search or continue to examine $i+1$-th snippet. As in most click models, this process uses examination hypothesis and cascade hypothesis.

According to the definition of *effective time ratio*, it is easy to include snippet quality. The effective time used in getting really relevant information include the time spent on reading the snippet and the original text of the relevant documents. The total time is composed by two parts: 1) the time spent on reading all snippets of the (top) retrieved documents, and 2) the time spent on reading the text of the clicked documents, whose snippets seem to be relevant. We further assume that the time spent on reading a snippet and a document are two static values(T_1 and T_2). In the top N documents, only the relevant documents with relevant snippets can provide relevant information, so the *effective time* is $\sum_{i=1}^{N} R(s_i) \cdot R(d_i) \cdot (T_1 + T_2)$(where d_i the i-th ranked document, s_i is its snippet, and R is a function indicating whether a document or a snippet is relevant or not). Total time spent on reading snippets is $N \cdot T_1$. The user would read all documents with relevant snippet, so the total time spent on reading documents is $\sum_{i=1}^{N} T_2 \cdot R(s_i)$. Thus the *effective time ratio* can be formulated as Equation 6. We can rewrite it by the time rate between reading document and reading snippet ($c = T_2/T_1$) in Equation 7.

$$ETR@N = \frac{(T_1 + T_2) \cdot \sum_{i=1}^{N} R(d_i) R(s_i)}{T_1 \cdot N + T_2 \cdot \sum_{i=1}^{N} R(s_i)} \tag{6}$$

$$= \frac{(1+c) \sum_{i=1}^{N} R(d_i) R(s_i)}{N + c \cdot \sum_{i=1}^{N} R(s_i)} \tag{7}$$

To further understand the effect of including snippet quality, we would compare versions of *effective time ratio* implementation with and without snippet quality factor. A snippet is good if it can indicates the document relevance accurately. Otherwise it may mislead users to miss a relevant document or waste time reading an irrelevant document. Here we define that a snippet is relevant iff it indicates the corresponding document is relevant. Therefore the quality of snippet can be qualified by possibility two types of errors.

Definition 2. *(First Type Error) Error of generating a relevant snippet for an irrelevant document*

Definition 3. *(Second Type Error) Error of generating an irrelevant snippet for a relevant document*

The first type of errors would lead the users to waste time clicking and examining irrelevant documents, and the users would miss relevant documents because of the second type of errors. Given a snippet generation algorithm, we define $p_1 = Pr(R(s) = 1 | R(d) = 0)$ as the conditional possibility of making *first type error* and $p_2 = Pr(R(s) = 0 | R(d) = 1)$ as the conditional possibility of making *second type error*. The ratio between expected expected effective time and expected total time (denoted as $EETR$) can be expressed as in Equation 8. Because $P@N$ is also an implementation of *effective time ratio* without considering snippet factor,

this equation describes the relations of *effective time ratio* with and without including snippet quality.

$$EETR@N = \frac{(1+c)(1-p_2)}{c(1-p_1-p_2)+\frac{1+cp_1}{P@N}} \tag{8}$$

We can derive three properties of *expected effective time ratio* from this equation. Proposition 1 shows that a search engine with an error-free snippet generation algorithm has higher *expected effective time ratio* than a retrieval system without snippet.

Proposition 1. $p_1 = 0, p_2 = 0 \Rightarrow EETR@N > P@N$

Proposition 2 validates that the $EETR$ values can reflect the underlying retrieval performance. Similarly, proposition 3 validates that the $EETR$ values can reflect snippet generation quality.

Proposition 2. $p_1(A) = p_1(B), p_2(A) = p_2(B), P@N(A) > P@N(B) \Rightarrow EETR@N(A) > EETR@N(B)$

Proposition 3. $p_1(A) > p_1(B), p_2(A) > p_2(B), P@N(A) = P@N(B) \Rightarrow EETR@N(A) < EETR@N(B)$

4.2 Experiments and Results

To validate the *effective time ratio* metric, a user study is designed to resemble the common search engine usage. 10 college students are employed to collect information for 50 questions with the help of a search engine. The questions are uniformly distributed in open and close categories, topic categories and difficulty degrees. We collect 100 results for each question and present 10 on one page. Once ending a question, the user was asked to answer the question and to report his satisfaction, whose values ranging from 1 to 4 (the higher the better). The satisfaction values are then compared with scores of various IR metrics including the proposed *effective time ratio*.

The first group of IR metrics use the document relevance only, including P@N, DCG, RR and cumulated precision. The reason for using unnormalized version of the metrics is that the total number of relevant documents is unknown. The second group of metrics have the same forms as those in the first group, but one document is considered to be relevant iff the document and its snippet are both relevant [13]. The third groups of metrics are *effective time ratio* and its extensions. As extending precision to cumulated precision, we can also define the *cumulated effective time ratio* as the sum of *effective time ratio* at the cutoffs where both the document and the snippet are relevant.

There are two parameters in the metric *effective time ratio*: cutoff N and document/snippet reading time rate $c = T_2/T_1$. For N, we can tune it in the experiment. For c, we can estimate it from the user study log and a one-month commercial search engine log. The estimated c value is 8.25 for the former log

and is 10.36 for the later one. We also find that most c values are near 10, so we use 10 as c value in ETR.

A good evaluation metric is supposed to reflect the users' satisfaction in using a search engine to find out relevant information for a need. In this paper, we follow Huffman and Hochster's work [12] to use correlation between metric score and user reported satisfaction. If the correlation is larger, the metric can reflect the users' satisfaction better.

Table 5. Correlations for Metrics

D	RR	P3	P5	P10	DCG3	DCG5	DCG10	CP3	CP5	CP10
Rel	**0.497**	0.396	0.407	0.286	0.412	0.431	0.366	0.330	0.345	0.286
S-D	RR	P3	P5	P10	DCG3	DCG5	DCG10	CP3	CP5	CP10
Rel	**0.467**	0.344	0.343	0.253	0.366	0.375	0.320	0.283	0.287	0.222
ETR	ETR3	ETR5	ETR10	ETR20	CETR3	CETR5	CETR10	CETR20		
	0.469	**0.537**	0.513	0.383	0.312	0.314	0.239	0.122		

Table 5 presents the correlation results for three groups of metrics. The highest score in each group is in bold. It shows that the *effective time ratio* has overall highest correlation with the users' satisfaction, and RR also has relative high correlation. Surprisingly, though *average precision* is the most commonly used metric in IR, *cumulated precision* and *cumulated effective time ratio* work not so well when compared with the real users' satisfaction. Another finding is that metrics at cutoff 5 can reflect users' satisfaction better. It may be because the user can see about 5 snippets without scrolling the mouse at the search engine result page.

5 Conclusion

In this paper, we introduce two categories of methods for evaluating search engines based on clickthrough data. Both methods model the noise in click data. The interleaving and comparing method simply combines results from two search engine and counts the clicks for system comparison. We show that the comparison can be more accurate by considering rank information. On the other hand, the click model based methods formulate the user behavior as a graphical click model and can get both value and confidence of document relevance estimation. Thus we can get reliable evaluation results of search engines. Besides, it reminds us to develop an algorithm to reorder the retrieved result to collect relevance information more efficiently. Moreover, we observe that the document relevance alone is handicapped for search engine evaluation, and we propose a new metric called *effective time ratio*. We show that this metric can reflect the users' utility better than the existing metrics employing document relevance only.

References

1. Joachims, T.: Unbiased evaluation of retrieval quality using clickthrough data. In: SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval (2002)

2. Radlinski, F., Kurup, M., Joachims, T.: How does clickthrough data reflect retrieval quality? In: CIKM 2008: Proceeding of the 17th ACM Conference on Information and Knowledge Mining, pp. 43–52. ACM, New York (2008)
3. Joachims, T., Granka, L., Pan, B., Hembrooke, H., Radlinski, F., Gay, G.: Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. ACM Transactions on Information Systems (TOIS) 25 (2007)
4. Carterette, B., Bennett, P.N., Chickering, D.M., Dumais, S.T.: Here or there. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) ECIR 2008. LNCS, vol. 4956, pp. 16–27. Springer, Heidelberg (2008)
5. Guo, F., Liu, C., Kannan, A., Minka, T., Taylor, M., Wang, Y.M., Faloutsos, C.: Click chain model in web search. In: WWW 2009: Proceedings of the 18th International Conference on World Wide Web, pp. 11–20. ACM, New York (2009)
6. Chapelle, O., Zhang, Y.: A dynamic bayesian network click model for web search ranking. In: WWW 2009: Proceedings of the 18th International Conference on World Wide Web, pp. 1–10. ACM, New York (2009)
7. Dupret, G.E., Piwowarski, B.: A user browsing model to predict search engine click data from past observations. In: SIGIR 2008: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 331–338. ACM, New York (2008)
8. Guo, F., Liu, C., Wang, Y.M.: Efficient multiple-click models in web search. In: Proceedings of the Second International Conference on Web Search and Web Data Mining, WSDM 2009, Barcelona, Spain, February 9-11, pp. 124–131 (2009)
9. Richardson, M., Dominowska, E., Ragno, R.: Predicting clicks: estimating the click-through rate for new ads. In: WWW 2007: Proceedings of the 16th International Conference on World Wide Web, pp. 521–530. ACM, New York (2007)
10. Craswell, N., Zoeter, O., Taylor, M., Ramsey, B.: An experimental comparison of click position-bias models. In: WSDM 2008: Proceedings of the International Conference on Web Search and Web Data Mining, pp. 87–94. ACM, New York (2008)
11. Allan, J., Carterette, B., Lewis, J.: When will information retrieval be "good enough"? In: SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 433–440. ACM, New York (2005)
12. Huffman, S.B., Hochster, M.: How well does result relevance predict session satisfaction? In: SIGIR 2007: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 567–574. ACM, New York (2007)
13. Turpin, A., Scholer, F., Jarvelin, K., Wu, M., Culpepper, J.S.: Including summaries in system evaluation. In: SIGIR 2009: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 508–515. ACM, New York (2009)