A Task Level Metric for Measuring Web Search Satisfaction and its Application on Improving Relevance Estimation

Ahmed Hassan
Microsoft Research
Redmond, WA
hassanam@microsoft.com

Yang Song
Microsoft Research
Redmond, WA
yangsong@microsoft.com

Li-wei He
Microsoft
Redmond, WA
lhe@microsoft.com

ABSTRACT

Understanding the behavior of satisfied and unsatisfied Web search users is very important for improving users search experience. Collecting labeled data that characterizes search behavior is a very challenging problem. Most of the previous work used a limited amount of data collected in lab studies or annotated by judges lacking information about the actual intent. In this work, we performed a large scale user study where we collected explicit judgments of user satisfaction with the entire search task. Results were analyzed using sequence models that incorporate user behavior to predict whether the user ended up being satisfied with a search or not. We test our metric on millions of queries collected from real Web search traffic and show empirically that user behavior models trained using explicit judgments of user satisfaction outperform several other search quality metrics. The proposed model can also be used to optimize different search engine components. We propose a method that uses task level success prediction to provide a better interpretation of clickthrough data. Clickthough data has been widely used to improve relevance estimation. We use our user satisfaction model to distinguish between clicks that lead to satisfaction and clicks that do not. We show that adding new features derived from this metric allowed us to improve the estimation of document relevance.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: search process

General Terms

Algorithms, Experimentation, Measurement

Keywords

search engine evaluation, user behavior models, web search success, clickthrough data

CIKM'11, October 24–28, 2011, Glasgow, Scotland, UK.
Copyright 2011 ACM 978-1-4503-0717-8/11/10 ...$10.00.

1. INTRODUCTION

Web search systems are traditionally evaluated using classical methodologies that use query sets and relevance judgments. Search quality is then measured for individual queries using measures like mean average precision (MAP), and normalized discounted cumulative gain (NDCG). A single score for an information retrieval system is then computed by averaging across many queries. However, user information needs are usually complex and the search process may involve query reformulations. Hence, individual queries may only represent a part of the underlying information need. This gave rise to research efforts that try to evaluate the entire search task rather than individual queries.

To better understand this problem, we developed a Web browser add-in to monitor user search activity across search engines and collect explicit judgments of user satisfaction with the entire search goal. In addition to explicit satisfaction ratings, the add-in also collected several pieces of information describing user behavior throughout the search process. This resulted in a dataset of search goals and the corresponding satisfaction labels. The labels are very reliable because they were directly collected from users performing the search rather than guessed by judges. Results were analyzed using sequence models that incorporate user behavior to predict whether the user ended up being satisfied with her search or not. The trained model can be used as a search quality metric to compare search engines, compare different versions of the same engine, or optimize document relevance estimation.

Web search engines logs have been widely used for optimizing search engine performance. Clickthrough data can be perceived as a strong signal from users telling us which documents they find relevant. However, interpreting clickthrough data for use in optimizing search engines faces several challenges. For example, several methods that exploit clcikthrough data assume that the user examined all the documents in the ranked list and clicked on the relevant ones [14, 20]. Others assume that perceived relevance (i.e. attractiveness) is the same as actual relevance [6, 9], or assume that all clicks result in some information gain for the user [5].

In this work, we propose a new model for estimating document relevance using the search logs of a web search engine. Current relevance models, that use clickthrough data, make the simplistic assumption that the mere fact that a URL has been clicked is a strong indicator of its relevance to the query. Even though, clickthrough data is correlated with relevance, not all clicks result in an information gain for the user. Given a model to predict user satisfaction with search results, we can provide a more accurate interpretation of clicks and hence better estimation of document relevance. We show empirically that user behavior models trained using explicit judgments of user satisfaction outperform several other search quality metrics. We also show that the resulting metric can be used to compute features that better interpret clickthrough data. These features are used to improve a machine learning algorithm that learns to rank documents. Comparing the ranking function with and without the new features, we observe improvement in the ranking quality.

The reminder of this paper is organized as follows. In Section 2, we discuss related work in the areas of measuring search quality and improving query-URL relevance estimation using search log data. A summary of our contributions is presented in Section 3. We then describe a user study where we collected explicit satisfaction ratings from users in Section 4. Our user satisfaction model is described in Section 5, followed by a description of the model that we use to improve relevance estimation in Section 6. Finally we present experiments in Section 7, and conclude in Section 8.

2. RELATED WORK

In this section, we describe previous work on evaluating the quality of search results. We review methods that try to measure search quality on both query level and goal level basis. We also describe work on improving query URL relevance estimation using search logs.

2.1 Evaluating Search Results Quality

Ad hoc information retrieval evaluation had focused on mean average precision (MAP). MAP is optimized to cover both recall and precision aspects. It also takes the entire search result ranking into consideration. Studies of Web search results evaluation have shown that users are only interested in the first few results and hence high early precision is desirable [11]. State of the art measurements of query result relevance use discounted cumulative gain (DCG) [13]. DCG metrics lend themselves to a user model of scanning a ranked list of results to some depth. DCG can be calculated using manual query URL relevance judgments, or estimated query URL relevance [2, 17]. The problem with this approach is that it misses the complete picture by looking only at individual queries. User information needs may result in more than one query, and same queries may have different underlying information needs. Hence, individual query URL relevance may not always mean user satisfaction.

Fox et al. [7] found that a strong correlation between search log features and user satisfaction. They used Bayesian modeling and decision trees to model explicit satisfaction ratings using several kinds of features. They used clickthrough rate features, as well dwell time features, and features describing how users end their search sessions.

Huffman and Hochster [12] study the correlation between user satisfaction and simple relevance metrics. They report a relatively strong correlation between user satisfaction and linear models encompassing the query URL relevance of the first three results for the first query in the search task. Hassan et al. [10] showed that modeling action sequences representing user behavior is better than models based on the query URL relevance of the first three results for the first query. The main reasons are that different information needs sometimes underlie the same query, and that the first query does not tell the complete picture in terms of the entire search task.

Radlinski et al. [22] show that interleaving the results of two ranking functions and presenting the interleaved results to users is a good way for predicting relative search engine performance. They also look at using metrics including abandonment rate, reformulation rate, and time to first click to predict relative performance. They found out that those metrics do not perform as well as interleaving results. Using aggregated features extracted from user behavior is certainly correlated with satisfaction. However, it has been shown that modeling the transitions between actions performed by users during search is a stronger predictor of satisfaction [10].

The most relevant work to this study is that of Hassan et al. [10]. They use two Markov models characterizing the behavior of users in both successful and unsuccessful searches. Our study improves upon this model in several ways. We will describe this model in detail, and discuss how it relates to this study in Section 3.

2.2 Improving Relevance Estimation

One of the earliest studies that use search logs to improve search results ranking was presented in [14]. In this work, clickthrough data based features were used to train a support vector machine (SVM) ranking function.

Radlinski and Joachims [20] extended this work to account for the concept of query chains. They observed that users searching the web often perform a sequence, or a chain, of queries with similar information needs. They extract those chains and generate preference judgments from search logs taking the chains into consideration.

In [21], the authors approach the problem of learning ranking functions from clickthrough data in an active exploration setting. They develop a Bayesian approach for selecting rankings to present to users such that their interactions result in more informative training data. They test their hypothesis on the TREC Web corpus, and on synthetic data. Their experiments demonstrate that their strategy leads to improving the learned ranking function.

Unlike previous work that has focused mainly on clickthough data, Agichtein et al. [2] incorporated user behavior information to improve ranking. They used implicit features aggregated over a large number of users. They included traditional implicit feedback features such as clickthrough rate, and aggregate features such as deviation of the observed clickthrough number for a given query-URL pair from the expected number of clicks. They show that incorporating implicit feedback improves the performance of the ranking algorithm. A similar study that aims at predicting user preferences has been presented in [1].

All the methods we discussed so far assume that the user examined all the documents in the ranking list and clicked on the relevant ones. Hence, the relevance of any document may be estimated simply by counting the number of times it was clicked. However, users do not browse the whole list [6]. For example, documents that appear highly in the ranking have higher probability of being clicked regardless of their relevance [8]. This suggests that taking the probability of examining a result into consideration would improve the ranking. Dupret and Piwowarski [6] addressed this problem by estimating the probability of examination of a document given the rank of the document and the last clicked document. The model assumes that the user goes through the list of results sequentially in the order in which they are presented. The user clicks on a URL only if it is both *examined* and found *attractive*. The probability that a document is examined depends on its rank and the rank of the last examined document. The probability that a document is attractive depends on the title, snippet, etc. A similar model is described in [9], where the authors present two different models to interpret multiple clicks: the independent click model, and the dependent click model which takes into consideration dependencies between multiple clicks.

The work we just described tries to model the relation between multiple clicks and study how this affects relevance estimation. However, it assumes that the attractiveness of a particular URL equals its relevance. Obviously, this may not always be the case. This problem has been addressed in [5]. In this work, the authors present a model for estimating intrinsic relevance as opposed to perceived relevance. They estimate intrinsic relevance by examining the documents that have been clicked after the current document. They assume that if a user clicks on document B after clicking on document A, then the amount of utility the user gained from A was not enough to satisfy her information need. They proposed a model to predict whether, given a certain amount of utility, the user will stop or continue her search and use that to generate a utility feature that is later used to improve ranking. This model has several advantages over previous models that try to interpret clickthrough data and was shown to be useful. However, it makes the simplistic assumption that all goals with clicks are successful. Obviously this assumption is not verified in reality as we will show later. Our model goes one step beyond this model by differentiating between successful and unsuccessful search tasks. We introduce the notion of a user failing to fulfill her information need and abandoning the search out of despair. Given a user satisfaction model, we can assign *utility* values to clicks in satisfied search goals, and *despair* values to clicks in unsatisfied search goals, hence providing a better interpretation of clicks with respect to the entire search task.

3. CONTRIBUTIONS

Our contributions include (1) we perform a large scale user study where thousands of explicit satisfaction ratings were collected from users, (2) we use and improve the user satisfaction model based on user behavior that was presented in [10], (3) we perform a large scale evaluation of the model using real search traffic, and (4) we present an application of the user satisfaction model on Improving relevance estimation.

Our user study allowed us to collect reliable satisfaction ratings from users regarding their search experience. The collected data encompasses several search engines, and several search verticals. The data was manually checked to verify the integrity of all labels.

Our user satisfaction model is based on modeling user behavior [10]. Our model has several advantages compared to previous related work. The quality of the search metrics depends on both the method and the training data. In this work, we train the model using a more reliable training data collected directly from users right after performing search. This is clearly better than asking judges to predict whether a user has been satisfied with her search or not, because judges, at best, are only guessing on the user's information intent; especially because several queries may have different underlying information need. Unlike query based metrics, we predict search goal success by analyzing user behavior. This allows us to capture the whole picture and predict satisfaction with respect to the entire information need rather than individual queries. Hassan et al. [10] used server side logging to collect behavioral signals. We used, and compared, both client side logging and server side logging for modeling user behavior. Client side logging gives us access to more information and hence a better characterization of user behavior during search. Our dataset contained data from several search engines and several verticals. This ensures that our metric generalizes well for all search engines and hence can be used to compare their performance. Finally, we performed a large scale evaluation using millions of search goals from actual user traffic to a major commercial search engine to evaluate how powerful the proposed metric is.

The model we propose for improving relevance estimation is different from all previous work on using clickthough data for optimizing ranking algorithms. Most of previous work makes the simplistic assumption that a click on a URL is always associated with information gain for the user. We show that this is not always the case and introduce the notions of *utility*, which users gain when their information need is satisfied, and *despair*, which users develop when their information need is not met. The closest work to our model is that presented in [5]. However, it does not distinguish between satisfied and unsatisfied search goals.

4. DATA

Collecting the data required for this research was particularly challenging. Asking human judges to predict whether a user ended up being satisfied with her search or not is very tricky. No matter how hard judges try to re-enact the user's experience, they end up to be only guessing on the actual user intent. As a result, we decided to directly collect explicit ratings of user satisfaction from the actual users performing search. For that, we developed an add-in for a widely distributed browser within a client server architecture. The add-in consists of a browser helper object that monitors the user's search experience, and a toolbar that interacts with the user. The browser helper object detects when a user submits a query to any of the three major search engines (Google, Bing, and Yahoo). Users are instructed to submit an explicit satisfaction rating at the end of their search goal, where a search goal is defined as follows:

Definition A search goal is a single information need that may result in one or more queries.

In the rest of this paper, we will use the terms “goal” and “task” interchangeably to refer to an atomic information need that may result in any number of queries.

4.1 User Study

The objective of this user study is to collect explicit satisfaction ratings from users with respect to their search experience. Participants were instructed to vote only after their information need is met. We are collecting goal-level labels rather than query-level labels. A search goal is different from a physical session. The later usually uses the idea of a “timeout” cutoff between queries. A timeout is the time between two successive activities, and it is used as a session boundary when it exceeds a certain threshold. The later definition of a physical session means that it may include several information needs.

A typical search goal would begin with a user issuing a query to a search engine (Bing, Google, or Yahoo) in order to fulfill an information need. The user can read the search results page, read the pages linked by the result page (up to many levels of depth), reformulate the queries, or click on search features such as spelling correction, ads, or other search verticals (such as Images, News, etc.). The user's information need is either satisfied (SAT) or the effort is abandoned (UNSAT).

Participants were encouraged to use the add-in both at home and at work. That helped us collect a more diverse set of information goals. They were also encouraged to use the search engine that best suits their needs. That allowed us to collect data from all major web search engines.

Depending on the query and information need, users had to choose one of the following three outcomes for each information need:

1. Satisfied: user found what she wants
2. Unsatisfied: no matter what user tried, she could not find what she needs, and hence gave up.
3. No Opinion: user was not able to tell whether she is satisfied or not, or did not want this goal logged.

In addition to explicit satisfaction ratings, we collected several other pieces of information that characterize the behavior of users during search. Our log entries included a unique identifier for the user, a timestamp for each page view, a unique browser window and tab identifiers, and the URL of the visited web pages. Secure (https) URLs have not been collected. Any personally identifiable information was removed from the data prior to analysis. We further processed the collected URLs to identify queries, search result clicks, sponsored search clicks, related search clicks, spelling suggestion clicks, and browsing clicks that originated from a search result click. For each query, we identified the corresponding search engine and vertical. In addition, we collected the timestamp of every action. The start of a search goal was automatically detected by the add-in whenever the user submits a query to any major web search engine. The end of every goal is the time when the user submitted her satisfaction rating.

An example of an unsatisfied goal is shown in Table 1. The user started with the query "steven colbert painting". After two clicks with short dwell time, she decided to switch to the "Images" tab. Apparently, she still could not find what she was looking for so she decided to reformulate her query to "steven colbert recursive painting". That still did not help her fulfill her information need. Hence, she gave up and ended her search. An example of a satisfied goal with a related information need is shown in Table 2. Here, the user started with the query "van gogh self portrait", and quickly decided to click on an image results. She spent some time examining the result then she successfully ended her search.

Table 1: An Example of an Unsatisfied Goal

Action		Dwell Time
Query	steven colbert painting	12
CLICK	url1	4
BACK	steven colbert painting	11
CLICK	url2	8
TAB SWITCH	IMAGES	8
Query	steven colbert recursive painting	8
END		-

Table 2: An Example of a Satisfied Goal

Action		Dwell Time
Query	van gogh self portrait	7
IMG CLICK	url1	37
END		-

4.2 Participants

We collected data from 115 regular employees and college interns in a large company who volunteered for the experiments. We initially invited 1200 users to participate in the study. The group of invited users included both technical and administrative employees. 115 user opted in to participate in the study resulting in a 9.5% response rate. 75% of the opt in users were males, while the rest were females. Participants were instructed to use the add-in both on the internal corporate network and outside. We also encouraged participants to use the add-in both at home and at work. That helped us collect a more diverse set of information goals. Our user base included different genders, different age groups (i.e. regular employees vs. college interns), and different educational backgrounds (i.e. technical vs administrative employees). This helped reduce any bias in the collected data.

4.3 Data Characteristics

We collected data from 115 users over a span of 6 weeks. During this period, we collected more than 12,000 search goals and more than 33,000 page visits. The average number of queries per search goals was 2.1 queries. This is somewhat smaller than the values of 2.4 and 2.5 reported in Fox et al. [7], and Spink et al. [23] respectively. The average query length was 2.84 words which is similar the values of 2.99 and 2.35 reported in Fox et al. [7], and Spink et al. [23] respectively.

We collected data from all major web search engines. We also collected data from all search verticals. Expectedly, most of the goals had the first query submitted to the web vertical, however some of those goals switched to a different vertical later in the search process. Our data contains search goals involving almost all available search verticals. For example, we have data from the following verticals: images, video, news, shopping, academic,...etc.

We excluded users who submitted very small (less than 10) or very large (more than 200) number of queries. Most of the users (approximately 88%) submitted 10 to 200 votes.

5. MODELING USER SATISFACTION

5.1 Goals as Search Trails

The user satisfaction model is based on user behavior while interacting with the search engine. We use search trails to represent this behavior by recording all user Web search activities. Every search goal is represented by a trail. A search trail originates with the submission of a query to a search engine and contains all queries and all post-query navigation trails (i.e., pages viewed on the click stream following the query being issued) [27]. It terminates when user stops their information seeking activity. This happens either when the information need is met, or when the user gives up and decides not to pursue his information need anymore. The trails contain any reformulation of queries, and all page visits. A page visit could be a visit to the search engine homepage, a search results page, or by following a link from a result page.

A user search trail is represented by an ordered sequence of user actions along with the time between those actions. We included several types of actions in our search trail representation like a query submission, an algorithmic search click, a sponsored search click, a related search click, a spelling suggestion click, an answer click, a tab switch, a click on a link on a search result page, or navigating back to SERP (i.e. the search engine results page).

Consider the following example: A user enters the query “steven colbert painting”, then 12 seconds later she clicks on one of the search results. She spends only 4 seconds on this page and then goes back the SERP which she examines for 12 seconds then clicks on another search result. She spends 5 seconds examining the new result page, then switches to the “images” tab where she reformulates her query to “steven colbert recursive painting” then ends her information seeking activity. This user goal can be represented by the following trail:

Q 12s SR 4s SERP 11s SR 8s IMG 8s Q 8s END

Here, Q represents a query, SR is a search result click, SERP represents going back to the search engine results page, IMG is a switch to the “images” tab, and finally END denotes the end of the user information seeking activity. We notice that we have a single state “Q” that represents both the first query and any further query rewrites. The reason is that we model transitions rather than states, and hence the transition to the first query will be always “START Q”, and that will be different from transitions to further query rewrites.

5.2 Learning Model Parameters

If we examine the example of the user searching for “steven colbert painting” from Table 1, we will find out that her information need was not met and hence she was not satisfied with her search experience. This goal is represented by this trail: Q 12s SR 4s SERP 11s SR 8s IMG 8s Q 8s END. Now let us contrast this example with the example in Table 2. This goal is represented by the following trail: Q 7s SR 37 END. The later user fulfilled her information need and ended up being satisfied with her search experience. If we examine the trails, we notice that the behavior that leads to satisfaction is different from the behavior that leads to dissatisfaction. Hence, if we are able to build a model to characterizes the behavior of users in the case of satisfaction, and another model that characterizes the behavior of users in the case of dissatisfaction, we can use those models to predict whether any new unseen goal is satisfied or not as in [10].

Given a set of trails representing a set of search goals, we can build a graph $G = V, E, w$ where V is the set of all possible tokens (i.e. actions) that may appear in the trails. $E = V \times V$ is the set of possible transitions between any two tokens. $w : E \rightarrow [0..1]$ is a weighting function that assigns to every pair of states (i, j) a weight $w(i, j)$ representing the probability that we have a transition from state i to state j.

This graph corresponds to a Markovian model. The set of states are the vocabulary, and the transition probabilities between states are estimated using Maximum Likelihood estimation as follows:

$$P_{ij} = \frac{N_{ij}}{N_i}$$

where N_{ij} is the number of times we saw a transition from i to state j, and N_i is the total number of times we saw state i in the training data. We used smoothing to account for any data scarcity.

We build two such models. The first model is built using all trails that appeared in the training dataset and was labeled as satisfied, and the second model is built using all trails in the training dataset that are not satisfied.

To model time, we assume that there is a distinctive distribution that governs the amount of time the user spends at each transition. We assume that transition times follow a gamma distribution. We again split our training data into two splits; the first containing all satisfied goals and the second containing all unsatisfied goals. We then estimate the gamma distribution parameters for every transition once in each model as in [10].

5.3 Classifying New Goals

We split our training data into two splits; the first containing all trails of satisfied goals and the second containing all trails of unsatisfied goals. Given the methodology described in the previous section, we build a pair of Markov models: One using all satisfied goals and the other using all unsatisfied goals. The first model characterized the behavior of satisfied users and the second model characterizes the behavior of unsatisfied users. We also estimate the time distribution parameters for every transition in both models. Given any new goal, we extract the corresponding trail and estimate the log likelihood that this sequence of actions was generated from every model. If a certain goal ended with user satisfaction, we expect that the likelihood of the corresponding behavior being generated from the SAT model will be higher and vice versa.

Given a model M, and sequence of actions (i.e. search trail) $T = (T_1, T_2, \ldots TS_n)$, the probability of this trail being generated from M is:

$$P_M(T) = \prod_{i=2}^{n} P(T_i|T_1, \ldots, T_{i-1}) = \prod_{i=2}^{n} W(T_{i-1}, T_i)$$

where n is the number of actions in the trail, and W is the probability transition function.

The log likelihood is then defined as:

$$LL_M(T) = \sum_{i=2}^{n} \log W(T_{i-1}, T_i)$$

Then we use the log likelihoods ratio to predict whether the search goal was satisfied or not :

$$f = \frac{LL_{M_{sat}}(T)}{LL_{M_{unsat}}(T)}$$

where T is the search trail, $LL_{M_{att}}(T)$ is the log likelihood of the trail given the satisfaction model, and $LL_{M_{unsat}}(T)$ is the log likelihood of the trail given the non-satisfaction model. If f is greater than a particular threshold, we classify the goal as SAT, otherwise we classify it as UNSAT. The threshold is estimated empirically using a development set.

6. IMPROVING RELEVANCE ESTIMATION

We propose a new model for estimating document relevance using the search logs of a Web search engine. The model is based on analyzing the overall activity of users during search goals. In particular, we propose a better interpretation of clickthrough data by putting the click in its context. Current relevance models, that use clickthrough data, make the simplistic assumption that the mere fact that a URL has been clicked is a strong indicator of its relevance to the query. Even though clickthrough data is correlated with relevance, there are several problems that arise when using it as an explicit relevance measure. When a user decides to click on a particular URL, he is usually affected by the results presentation. That includes the snippet, the position of different URLs, other displayed results, etc. Hence when a user decides to click on a URL, she is only guessing that this URL is relevant to her information need, which may or may not be correct. On the other hand, if we know whether the information need of the user has been met or not, we would be able to provide a much better estimation of the relevance of a particular document given the clickthrough data.

The proposed model makes use of the user satisfaction model we described earlier. It assumes that users, who end up being satisfied with their search, keep clicking on search results till they collect enough "utility" to satisfy their information need. On the other hand, users who end up being dissatisfied with their search develop "despair" as they click on more and more irrelevant results. They give up when they develop a certain amount of despair. We collect a large number of search goals from the search logs and use our overall goal success metric to predict whether users ended up being satisfied or not. We then estimate the average utility, and despair for every query document pair. These will be used as features to train a machine learning ranking function. We compare this ranking function to a baseline using both content features and features extracted from clickthrough data.

Let us reconsider the example from Section 5.1 where a user was searching for "steven colbert painting". In this example, the user clicked on two different URLs, yet this did not fulfill her information need. Systems that use clickthrough rate (CTR) as a predictor of actual relevance will incorrectly assume that the two URLs are relevant to the query "steven colbert painting". The model we propose will not only avoid this, but will also reduce the predicted relevance of those URLs causing other, possibly more relevant, results to move up in the ranking.

6.1 Utility Despair Model

Assume our unit of analysis is a search goal as described earlier in Section 5. Assume a user U_1 starts a search to satisfy a certain information need. During her search she clicked on three URLs U_1, U_2, and U_3. Also assume that our satisfaction model predicted that this user was satisfied with her search. The basic intuition behind the model we propose is that the user kept clicking on results until she collected enough utility. after which her information need was met and she concluded her search activity. Every result she clicked contributed a certain amount of utility toward satisfying her need.

We tried two different methods to assign a particular amount of utility to every result. The first assumes that utility is distributed uniformly between all clicked results and hence:

$$Util(u_i) = \frac{1}{n} \tag{1}$$

where n is the number of clicked results.

The second method assumes that the more time a user spends on a search result, the more utility she gains from it. Hence the utility gain from a particular click is proportional to its dwell time. Hence utility is defined as:

$$Util(u_i) = \frac{T_i}{\sum_{j=1}^{n} T_j} \tag{2}$$

where T_i is dwell time of the click with order i and n is the number of clicked results.

Now assume the same user starts another search where she clicks on some results . But in this case, our satisfaction model predicted that this user was not satisfied with her search. This user kept trying to fulfill her information need till she finally gave up. we assume that unsatisfied users keep trying to fulfill their information need till they develop a certain amount of "despair", after which they give up. Every click the user makes contributes to that despair. In cases, where a user does not click on any results, we assume that she at least spent an equal amounts of time examining the titles and snippets of the first two results, and did not find them relevant. Hence, we treat them as if they were clicked. The choice of the first two results in case of no clicks is justified by the eye tracking studies [8] that show that the mean time users fixate on a presented result is almost equal for links ranked 1 and 2, and that it drops off sharply after the second link. Despair is distributed among URLs in the same way we described for utility (i.e. either uniformly or using dwell time). When using dwell time, it is assumed that the less time the user spends on the result, the more despair she develops.

7. EXPERIMENTS

7.1 User Satisfaction Model

7.1.1 Evaluation using Labeled Data

Our experiments use the search goals collected during the user study described in Section 4. Our search goals come from several search engines and several verticals. Labels were directly collected from users in the form of explicit satisfaction ratings. We evaluate our results in terms of *Precision*, *Recall*, *F-measure*, and *Accuracy*. We used ten fold cross validation for all experiments and evaluated statistical significance using a 1-tailed paired t-test. All results are statistically significant unless otherwise stated.

We compare our results to two different baselines. The first is based on the time to first click which is the difference between the timestamp from serving up the page, and the timestamp of the first user click on the page. It has been previously shown that this number is highly correlated with satisfaction. The second baseline is based on the number of successful clicks in the search goal. A successful click is defined to be either a click on a result link (answer, ad, or web result) that has a dwell time of 30 second or the last click in a user's goal was on an answer, ad, or a web result.

We compare our method to the baselines using 10-fold cross validation on the labeled data. We also perform a large scale evaluation using real user traffic. In the later scenario, we use two ranking functions; one of which is hypothesized to be better by the ranking function developers. Results from the two functions were shown to a small fraction of a commercial search engine system and millions of queries and clicks are observed. We use the proposed method and the baselines to compare the two ranking functions and examine which methods succeed and which methods fail in finding the difference between the two ranking functions.

More than 83% of our data was for goals where users ended up being satisfied with their search experiment. Hence the class distribution was not balanced. Imbalance in the class distribution often causes machine learning algorithms to perform poorly on the minority class [25]. In our case, we are more interested in the minority (DSAT) class. Hence, the cost of misclassifying the minority class is usually higher than the cost of misclassifying the majority class. Several researchers have experimentally evaluated the benefit of using sampling to artificially balance imbalanced data [4, 18, 26]. A common way of addressing class imbalance is to artificially rebalance the training data. To do this we down-sample the majority class (SAT goals) by holding the M DSAT goals constant, randomly selecting without replacement M SAT goals, and training our model on the 50/50 split. This was repeated several times until all SAT goals have been used which resulted in 5 different models. Finally decisions are made by taking the majority vote among all models.

Figure 1 compares the precision-recall curves for the two baselines described earlier and the proposed model for the SAT case. Figure 2 compares the precision-recall curves for the same systems for the DSAT case. If we examine the curves for the SAT class, we notice that the baseline based on the number of successful clicks does pretty well for very low recall regions (i.e. recall less than or equal to 0.2). The proposed model significantly outperforms the baseline based on the number of successful clicks for all operating points where recall is greater than 0.2. The proposed model significantly outperforms the baseline based on time to first click for all operating points. The superior performance of the baseline based on the number of successful clicks at low recall is not surprising. The reason is that the few number of goals with so many successful clicks are almost always satisfied. If we examine the precision-recall curves for the DSAT class, we notice that the proposed model significantly outperforms the baselines at all operating points. The model also subsumes the information in the baselines. This became clear, when we trained a classifier using features from the baselines along with the log likelihood ratio calculated by the proposed model. The difference in performance before and after adding features from the baselines was not statistically significant. The Markov model based on search trails representing user behavior has several advantages compared to the baseline. The most important advantage is that it provides a more accurate picture of the user behavior, while the baselines only use aggregated feature to create a rough approximation of behavior.

We now measure the performance of the proposed method when the system is allowed to abstain from classifying goals for which it have low confidence. We regard the log likelihood ratio as a confidence measure and evaluate the top goals with the highest confidence level at different values of threshold. Figure 3 shows the accuracy and the DSAT precision at different thresholds. We notice that both metrics improve by abstaining from classifying the difficult goals. This is particularly important for the DSAT precision, because one of the possible applications for our metric is to mine dissatisfaction cases. These are cases where a search engine fails and hence need improvement. We notice from the figure that if we abstain from classifying the 30% of the goals where we have the least confidence, we can achieve precision as high as 93%.

Finally we compared the performance of the model when restricted to using information available on the server side logs only to that using the richer information available at the client side. We found out that when using server side logs , accuracy drops by approximately 1%. This shows that the model is useful for use with both kinds of logs, yet it is capable of using the richer information available in the client side logs.

7.1.2 Large Scale Evaluation using User Traffic

The second part of our evaluation uses real user traffic. We used two ranking functions developed by a large commercial search engine. One of the ranking functions is intentionally degraded by the ranker developers. The two ranking functions were shown to a small fraction of a commercial search engine system users and millions of queries and clicks were observed. We extracted $1,000,000$ search goals for each ranking function. We ran the proposed system on all search goals corresponding to the two ranking functions. The p-values for the time to first click baseline, the number of successful clicks baseline, and the proposed system were 0.5694, 0.0048, and 2.2×10^{-8} respectively. This shows that the proposed metric was able to tell the two ranking functions apart with a significantly smaller p-value than the baselines. We also used another pair of ranking functions for testing. All previously mentioned metrics failed to find any statistically significant difference between their performance. The only two metrics that were able to report any statistically significant difference are our metric, and a metric based on an interleaved evaluation [19]. According to recent studies, interleaving experiments can identify large differences in retrieval effectiveness with high reliability. The advantage of our metric is that it eliminates the need for creating and collecting traffic for a third ranking function that interleaves the results of the two functions we need to compare [19].

We also studied how many search goals are needed for our metric to reliably compare two systems. We studied the effect of the amount of user traffic collected on the outcome of the comparison between the first pair of ranking functions. We sample n goals from the user traffic data without replacement for the two ranking functions. We measure the performance of every ranking function and determine which one wins. We repeat the sampling $1,000$ times for every n and report the percentage of trials where the better ranking function wins. Figure 4 shows the fraction of samples where the better ranker gets a better score using the proposed user satisfaction model versus the size of the dataset used for evaluation in terms of the number of goals. For small numbers of goals, each ranker gets a better score for almost 50% of the time. As the number of goals increase, a constant preference for the better ranker to win is observed. More sensitive metrics are more desirable because they can work with less amounts of data. The figure shows that with

100, 000 goals, the better function wins more than 90% of the time. This figures tells us how much data do we need to collect to get meaningful results from the metric.

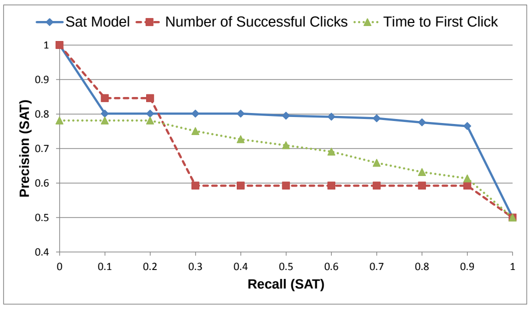

Figure 1: Precision Recall Graph for the SAT Class.

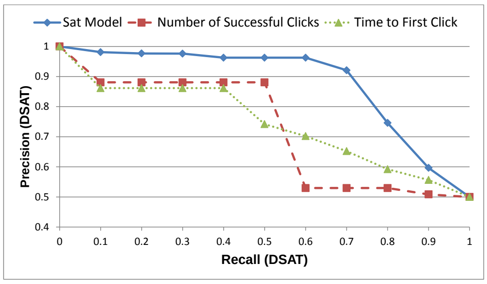

Figure 2: Precision Recall Graph for the DSAT Class.

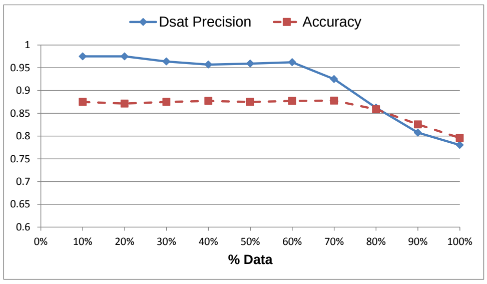

Figure 3: Confidence in Prediction vs. Performance.

7.2 Relevance Estimation Evaluation

We explained in previous sections how to use the user satisfaction model and a large amount of user traffic to assign utility and despair values to query URL pairs. We evaluate those values by using them as features to train a learning to rank algorithm.

We compare the results of two ranking functions. The first uses content based features (i.e. BM25F [24], word overlap,..etc.), and link based features. We then added the utility and despair values as features and create a new ranking function.

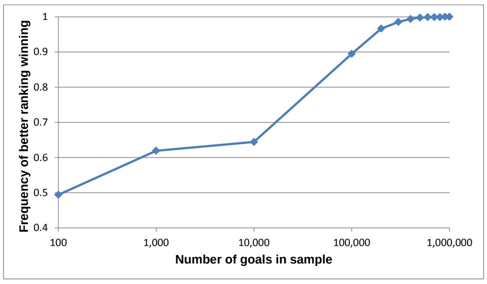

Figure 4: Number of Goals in Test Data vs. Frequency of Better Ranking Winning.

We use Normalized Discounted Cumulative Gain (NDCG) [13] for evaluation. It is defined as:

$$NDCG@K = \frac{1}{Z@K} \sum_{i=1}^{k} \frac{2^{g(i)} - 1}{\log(i+1)} \qquad (3)$$

where K is the cut-off threshold, and $g(i)$ is the relevance of the ith document. $g(i)$ can take any integer value between 1, and 5 which corresponds to the ”Perfect”, ”Excellent”, ”Good”, ”Fair”, and ”Bad” relevance grades. $Z@K$ is a normalization factor that represents the DCG of the ideal ranking.

Our test data consists of 750 queries for which we have human relevance judgments. For every query result pair, a human annotator was asked to examine the pair and decide whether the result is relevant to the query or not. Annotators used a five point scale, where every pair is judged as either “Perfect”, “Excellent”, “Good”, “Fair”, or “Bad”. We also collected a large amount of user traffic over the span of a two weeks period from users using a commercial search engine. We segmented the search log data into search goals using a simple linear classifier that uses the time between queries, word level edit distance and character level edit distance as features. The parameters of the classifier were set using grid search. We can also use more sophisticated techniques for boundary identification following the method proposed in Jones and Klinkner [16] which reports accuracy of around 92%. After identifying search goal boundaries, we keep only goals that has one or more of the queries for which we have human relevance judgments. Then, we apply our user satisfaction model to every goal and calculate a utility and/or despair values for every query URL pairs as discussed in Section 6. We use those feature to train a learning to rank algorithm. We use SVMRank [15] for all ranking experiments. All experiments use 10-fold cross validation. All experiments distribute utility/despair based on dwell time unless otherwise stated.

Figure 5 shows the NDCG@K for different values of K for the baseline ranking function with and without the utility and despair features. We notice from the figure that adding the features improves the performance of the baseline for all values of K. Next, we tried to add click-through rate features to the baseline to test whether adding the utility and despair features will still improve performance. The result of this experiment is shown in Figure 6. The figure shows an improvement in NDCG@K for all values of K even after

Table 3: Example of an Unsatisfied Goal

NDCG @k	Uniform	Dwell Time
1	0.116	0.129
2	0.239	0.251
3	0.326	0.338
4	0.372	0.385
5	0.402	0.414

we add the CTR features to the baseline. All improvements are statistically significant at the 0.05 level.

We compare the NDCG gain for informational versus navigational queries [3] in Figure 7. We notice that the improvement in case of informational queries is larger than the improvement in case of navigational queries. This is a desirable property because informational queries are usually more challenging.

Finally we compare the two utility distribution methods in Table 3. The table shows the NDCG@K values for the uniform distribution method, and the one based on dwell time. We notice that the dwell time yields the better results. However, the results are only numerically better but not statistically significant. This supports the hypothesis that the more time a user spends on a result page, the more utility she gains from it.

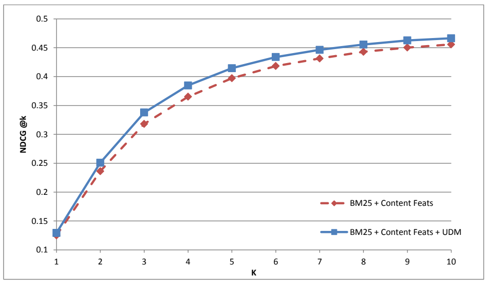

Figure 5: NDCG@n Improvement when Adding Utility and Despair Features.

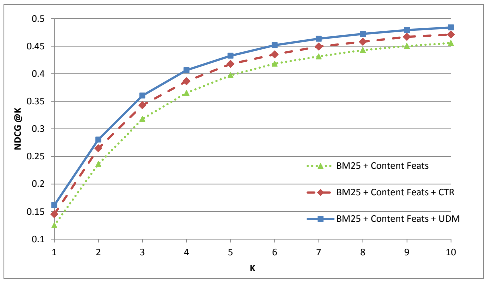

Figure 6: NDCG@n Improvement when Adding CTR vs. Improvement when Adding Utility and Despair Features.

8. DISCUSSION AND CONCLUSIONS

We performed a user study to collect explicit satisfaction ratings from users across search engines. We developed a browser add-in for a well distributed web browser to collect satisfaction ratings from participants, as well as monitor their behavior during search. We managed to collect approximately 12,000 search goals and 33,000 page visits from 115 users over a span of 6 weeks. The dataset we collected contains a reliable satisfaction ratings for a diverse set of search goals from several search engines.

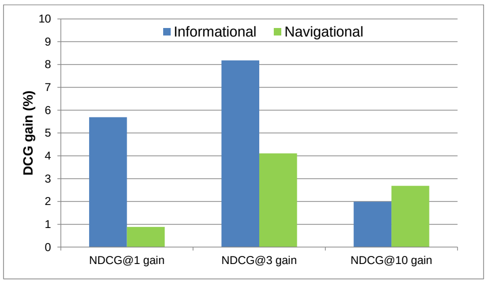

Figure 7: NDCG Gain for Informational vs. Navigational Queries.

We have shown that training a supervised model that learns user behavior in satisfied and unsatisfied scenarios results in an accurate satisfaction/dissatisfaction prediction. We extensively evaluated our model using cross validation on the labeled data. We also evaluated it by comparing two ranking functions using millions of goals from real user traffic on a commercial search engine. The results show that our model is accurate and robust.

We also showed preliminary results for an application of how our user satisfaction model can be used to improve relevance estimation. We presented a new model for interpreting clickthrough data that distinguished between clicks in satisfied, and unsatisfied goals. We introduced the notions of utility and despair, where users gain utility when clicking on results in satisfied goals, and develop despair when clicking or examining results in unsatisfied goals. We described how utility and despair values can be calculated for query-URL pairs and used as features for a learning to rank algorithm. We showed that adding those features improves the performance of a strong ranking function used as a baseline.

One direction of future work is improve the user satisfaction model by adding the notion of effort as a component of predicting over all satisfaction. Another direction is to extend the model to the semi-supervised setting where we learn from both labeled and unlabeled. We also intend to explore how the satisfaction model can be used in real time settings where the search engine predicts that a user will end up being unsatisfied and interfere to prevent that. Another direction of future work is to improve the relevance estimation model by using different ways, that are more sensitive to user behavior, to distribute utility or despair among clicks.

9. REFERENCES

[1] E. Agichtein, E. Brill, S. Dumais, and R. Ragno. User interaction models for predicting web search result preferences. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–10, 2006.

[2] E. Agichtein, E. Brill, and S. T. Dumais. Improving web search ranking by incorporating user behavior information. In *SIGIR 2006: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–26, New York, NY, USA, 2006. ACM.

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

[4] C. Drummond and R. C. Holte. C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In *ICML'2003 Workshop on Learning from Imbalanced Datasets II*, pages 1–8, 2003.

[5] G. Dupret and C. Liao. A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 181–190, New York, NY, USA, 2010. ACM.

[6] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 331–338, New York, NY, USA, 2008. ACM.

[7] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White. Evaluating implicit measures to improve web search. *ACM Transactions on Information Systems*, 23, 2005.

[8] L. A. Granka, T. Joachims, and G. Gay. Eye-tracking analysis of user behavior in www-search. In *Proceedings of the 27th annual international conferenceon Research and development in information retrieval*, pages 478–479, 2004.

[9] F. Guo, C. Liu, and Y. M. Wang. Efficient multiple-click models in web search. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 124–131, 2009.

[10] A. Hassan, R. Jones, and K. L. Klinkner. Beyond dcg: user behavior as a predictor of a successful search. In *WSDM '10: Proceedings of the third ACM international conference on Web search and data mining*, pages 221–230, New York, NY, USA, 2010. ACM.

[11] D. Hawking, N. Craswell, P. Thistlewaite, and D. Harman. Results and challenges in web search evaluation. In *WWW '99: Proceedings of the eighth international conference on World Wide Web*, pages 1321–1330, New York, NY, USA, 1999. Elsevier North-Holland, Inc.

[12] S. B. Huffman and M. Hochster. How well does result relevance predict session satisfaction? In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 567–574, 2007.

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

[14] T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, New York, NY, USA, 2002. ACM.

[15] T. Joachims, T. Finley, and C.-N. Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59–59, October 2009.

[16] R. Jones and K. Klinkner. Beyond the session timeout: Automatic hierarchical segmentation of search topics in query logs. In *Proceedings of ACM 17th Conference on Information and Knowledge Management (CIKM 2008)*, 2008.

[17] S. Jung, J. L. Herlocker, and J. Webster. Click data as implicit relevance feedback in web search. *Information Processing and Management (IPM)*, 43(3):791–807, 2007.

[18] X.-Y. Liu, J. Wu, and Z.-H. Zhou. Exploratory undersampling for class-imbalance learning. *Trans. Sys. Man Cyber. Part B*, 39(2):539–550, 2009.

[19] F. Radlinski and N. Craswell. Comparing the sensitivity of information retrieval metrics. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 667–674, 2010.

[20] F. Radlinski and T. Joachims. Query chains: learning to rank from implicit feedback. In *KDD '05: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 239–248, New York, NY, USA, 2005. ACM.

[21] F. Radlinski and T. Joachims. Active exploration for learning rankings from clickthrough data. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 570–579, New York, NY, USA, 2007. ACM.

[22] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In J. G. Shanahan, S. Amer-Yahia, I. Manolescu, Y. Zhang, D. A. Evans, A. Kolcz, K.-S. Choi, and A. Chowdhury, editors, *CIKM*, pages 43–52. ACM, 2008.

[23] A. Spink, D. Wolfram, B. Jansen, B. J. Jansen, and T. Saracevic. Searching the web: The public and their queries. 2001.

[24] S. J. M. H.-B. Stephen E. Robertson, Steve Walker and M. Gatford. Okapi at trec-3. In *Proceedings of the Third Text REtrieval Conference (TREC 1994)*, 1994.

[25] J. Van Hulse, T. M. Khoshgoftaar, and A. Napolitano. Experimental perspectives on learning from imbalanced data. In *Proceedings of the 24th international conference on Machine learning*, pages 935–942, 2007.

[26] G. M. Weiss and F. Provost. Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, 19:315–354, 2003.

[27] R. W. White and S. M. Drucker. Investigating behavioral variability in web search. In *Proceedings of the 16th international conference on World Wide Web*, 2007.