Beyond Clicks: Query Reformulation as a Predictor of Search Satisfaction

Ahmed Hassan
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
hassanam@microsoft.com

Xiaolin Shi, Nick Craswell, Bill Ramsey
Microsoft Bing
One Microsoft Way
Redmond, WA 98052, USA
xishi,nickcr,brams@microsoft.com

ABSTRACT

To understand whether a user is satisfied with the current search results, implicit behavior is a useful data source, with clicks being the best-known implicit signal. However, it is possible for a non-clicking user to be satisfied and a clicking user to be dissatisfied. Here we study additional implicit signals based on the relationship between the user's current query and the next query, such as their textual similarity and the inter-query time. Using a large unlabeled dataset, a labeled dataset of queries and a labeled dataset of user tasks, we analyze the relationship between these signals. We identify an easily-implemented rule that indicates dissatisfaction: that a similar query issued within a time interval that is short enough (such as five minutes) implies dissatisfaction. By incorporating additional query-based features in the model, we show that a query-based model (with no click information) can indicate satisfaction more accurately than click-based models. The best model uses both query and click features. In addition, by comparing query sequences in successful tasks and unsuccessful tasks, we observe that search success is an incremental process for successful tasks with multiple queries.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *selection process, search process.*

Keywords

Re-querying behavior, success prediction, search tasks

1. INTRODUCTION

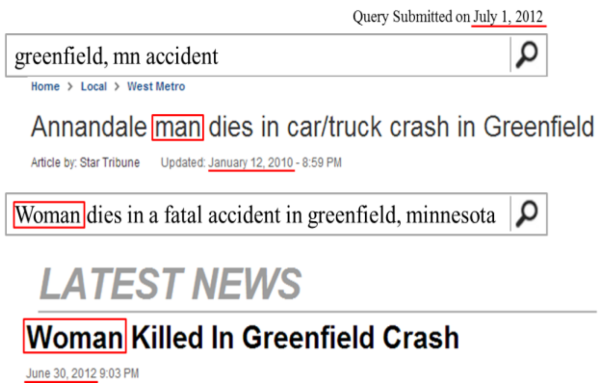
Search is an interactive process that starts with a user submitting a query to a search system. Depending on the results returned by the system and the user's information need, the user may click zero or more results and may submit zero or more follow-up queries. Search log data can provide implicit feedback from which the search system can identify which results are relevant for particular queries [2]. It can also provide insights into retrieval performance. Given search logs, models of searcher satisfaction can be developed at the query level or at the session/task level [16].

A user's search activity has one or more queries and zero or more clicks on results. Such activity is motivated by one or more higher-level goals, which we call tasks, although tasks are not our focus of study in this paper. Instead, we focus on query-level satisfaction. To understand the difference between query-level and task-level success, consider the task of booking a holiday. The user might enter a query "*expedia*" with navigational intent. In that case, reaching the Expedia site constitutes query-level success without necessarily indicating task-level success, since we do not know if the user's task was completed.

The notion of satisfaction at the query level could have many scenarios and aspects relating to the quality and usefulness of search results. These include but are not limited to: successful navigation to a known item, finding the answer to a question, learning about a new topic, finding the required information without clicking (i.e. good abandonment) [30], or gathering evidence that the required item/information doesn't exist. Rather than studying these separately, or modeling degrees of success, we follow state of the art work on search success (e.g. [1],[18]) by choosing a simple view that satisfaction is binary: if a user is satisfied with the current results then the query is a successful one; otherwise, it is not.

Click-based metrics have been widely used as a way to predict whether a given user is satisfied with the search results or not. Clicking on a result page does not necessarily indicate that the query was successful if taken out of context. To better understand this, consider the example in Figure 1. We see an example of a user submitting the query "*greenfield, mn accident*". Apparently, the user is looking for information about an accident that took place in Greenfield, MN. The user clicks a result, dwells for 36 seconds, then types a second query *"woman dies in a fatal accident in greenfield, minnesota"* and clicks another result. The first clicked result is from 2010, the second is from July 2012, and the documents describe different incidents. A likely interpretation of this is that the user was looking for the 2012 accident, and failed to find it on the first query, especially because the queries were submitted just one day after the 2012 accident. The 2012 document was not even in the top-20 results of the first query. The 2010 document does not mention the 2012 accident.

Had we seen only the first query and click, we might have thought the user was satisfied. In this work, we introduce and evaluate models of query-level satisfaction that consider the next query submitted by the user and not only based on whether a user clicks on a result or not. Given a stream of queries submitted by a user, we only consider the immediate follow-up query. This means we


Query Submitted on July 1, 2012

greenfield, mn accident

Home > Local > West Metro

Annandale man dies in car/truck crash in Greenfield

Article by: Star Tribune Updated: January 12, 2010 - 8:59 PM

Woman dies in a fatal accident in greenfield, minnesota

LATEST NEWS

Woman Killed In Greenfield Crash

June 30, 2012 9:03 PM

Figure 1. The next query is evidence of dissatisfaction even though the original query received a long dwell time click (> 30 seconds).

make minimal use of other queries, but also we are using the query that best reflects the user's reaction to the current query.

The next query may be a manual reformulation of the current query because the user is dissatisfied with the current query results (e.g. *"greenfield, mn accident"* → *"woman dies in a fatal accident in greenfield, mn"*). If the user is satisfied, the next query might be a related query on the same topic (e.g., *"best gre practice tests"* → *"gre powerpre"*), or a new query on a different topic. Hence, we build models to predict the current query success using query pair information, click information or both. More specifically, we try to answer the following research questions:

Research Question 1: What is the correlation between click signals and query pair features such as overlap and inter-query time? If both are indicators of satisfaction, there should be some correlation.

Research Question 2: Can we accurately predict user satisfaction using query pair data alone?

Research Question 3: Can we improve query success prediction using both click and query pair signals?

Research Question 4: Using search tasks which may have more than two queries, how does our query-level prediction relate to task-level success?

The remainder of this paper is structured as follows. Section 2 describes related work. A formal problem definition is given in Section 3. Section 4 motivates this research by conducting a large scale exploratory analysis of user behavior logs, considering the correlation between click-based and query-based satisfaction indicators. In Section 5, we present a method for identifying query reformulation behavior, which is used to predict query-level satisfaction In Section 6. In Section 7, we present the experiments performed to evaluate model effectiveness and discusses query reformulation patterns in search tasks. We conclude in Section 8.

2. RELATED WORK

There are four areas of work related to the research presented in this paper: (i) query document relevance, (ii) search satisfaction, success, and frustration, (iii) search tasks boundary identification, and (iv) query refinement. We cover each of these areas in turn.

2.1 Query Document Relevance

State of the art measurement in information retrieval uses a test collection comprising a document corpus, query topics and relevance judgment. These are then used with relevance metrics such as MAP and discounted cumulative gain (DCG) [21]. This process requires costly manual judgments of the relevance of documents in the search result list to individual queries. Previous work has also estimated query document relevance using models derived from user click behavior [2][8][28]. Other research work has used the click patterns to compare different ranking functions, i.e. to derive a metric [8][11][24].

Even though Click data is very useful for predicting query document relevance, it is also very noisy. Some of the reasons behind that are document snippets that do not accurately represent the content, and the bias resulting from the position of the document in the result set [24]. Our work shows that using information about the next query submitted by the user can allow us to filter out noisy click signals that are not indicative of query success.

2.2 Search Satisfaction

Extensive literature exists on deriving indicators of task success or failure from online user behavior. Fox et al. [13] used an instrumented browser to determine whether there was an association between explicit ratings of user satisfaction and implicit measures of user interest and identified the measures that were most strongly associated with user satisfaction. They found that there was a link between user activity and satisfaction ratings, and that clickthrough, dwell time, and session termination activity combined to make good predictors of satisfaction for Web pages. For example, they found out that a short dwell time is an indicator of dissatisfaction, while long dwell time is correlated more with satisfaction. Behavioral patterns were also used to predict user satisfaction for search sessions. Huffman and Hochster [21] found a relatively strong correlation with session satisfaction using a linear model encompassing the relevance of the first three results returned for the first query in a search task, whether the information need was navigational, and the number of events in the session. Hassan et al. [20] developed models of user behavior to accurately estimate search success on a session level, independent of the relevance of documents retrieved by the search engine. Ageev et al. [1] propose a formalization of different types of success for informational search, and presented a scalable game-like infrastructure for crowdsourcing search behavior studies, specifically targeted towards capturing and evaluating successful search strategies on informational tasks with known intent. They show that their model can predict search success effectively on their data and on a separate set of log data comprising search engine sessions. Feild et al. [12] developed methods to predict user frustration. They assigned users difficult information seeking tasks and monitored their degree of frustration via query logs and physical sensors.

Our work is different from this work in that we focus on query-level satisfaction. However, we also try to understand the difference between query-level and task-level satisfaction and we study the patterns of query sequences that form a task.

2.3 Search Task Boundary Identification

The problem of classifying the boundaries of the user search tasks within sessions in web search logs has been widely addressed before. Boldi et al. [7] presented the concept of the query-flow graph. A query-flow graph represents chains of related queries in query logs. They use this model for finding logical session boundaries and query recommendation. Ozmutlu [34] proposes a method for identifying new topics in search logs. He demonstrates that time interval, search pattern and position of a query in a user session, are effective on shifting to a new topic. Radlinski and Joachims [35] study sequences of related queries (query chains).

Table 1. Relative CTR for different subsets of pairs, using word overlap and a 5 minute time threshold

	overall	*non-overlap*	*overlap*
overall	0%	11%	-21%
non-quick	25%	24%	29%
quick	-29%	-17%	-39%

Table 2. Relative CTR-30 for different subsets of pairs, using word overlap and a 5 minute time threshold

	overall	*non-overlap*	*overlap*
overall	0%	6%	-12%
non-quick	6%	-1%	40%
quick	-7%	20%	-30%

They use that to generate new types of preference judgments from search engine logs to learn better ranked retrieval functions.

Arlitt [3] found session boundaries using a calculated timeout threshold. Murray et al. [31] extended this work by using hierarchical clustering to find better timeout values to detect session boundaries. Jones and Klinkner [26] also addressed the problem of classifying the boundaries of the goals and missions in search logs. They showed that using features like edit distance and common words achieves considerably better results compared to timeouts. Lucchese et al. [31] uses a similar set of features as [26], but uses clustering to group queries in the same task together as opposed to identifying task boundary as in [26]. In this work, we present better models for predicting the relation between pairs of queries and we use it toward a higher level goal, which is predicting query level success.

2.4 Query Refinement

Existing research has studied how web search engines can propose reformulations, but has given less attention to how people perform query reformulations. Most of the research on manual query reformulation has focused on building taxonomies of query reformulation. These taxonomies are generally constructed by examining a small set of query logs.

Anick [3] classified a random sample of 100 reformulations by hand into eleven categories. Jensen et al. [23] identified 6 different kinds of reformulation states (New, Assistance, Content Change, Generalization, Reformulation, and Specialization) and provides heuristics for identifying them. They use them to predict when a user is most receptive to automatic query suggestions. The same categories were used in several other studies **Error! Reference source not found.**[29].

Huang and Efthimis [19] proposed another reformulation taxonomy. Their taxonomy was lexical in nature (e.g., word reorder, adding words, removing words, etc.). They also proposed the use of regular expressions to identify them. While studying re-finding behavior, Teevan et al. [36] constructed a taxonomy of query re-finding by manually examining query logs, and implemented algorithms identify repeat queries, equal click queries and overlapping click queries.

None has built an automatic classifier distinguishing reformulation queries from other. Heuristics and regular expressions have been used in [19] and [23] to identify different types of reformulations.

3. PROBLEM DEFINITION

We start by defining some terms that will be used through-out the paper:

Definition: A *Search Session* is group of queries and clicks demarcated with a 30-minute inactivity timeout, such as that used in previous work [35].

Definition: A *SAT (Satisfied) Query* is a query where the information need of the searching user has been successfully addressed.

Definition: A D*SAT (disatisfied) Query* is a query where the information need of the searching user has not been successfully addressed.

Definition: *Query Reformulation* is the act of submitting a *Next Query* Q2 to modify a previous search query Q1 in hope of retrieving better results.

Assume we have a stream of queries submitted to a search engine. In response to each query, the engine returns a search results page. The user may decide to click on one or more elements on the page, reformulate the query, or end the search. So given a query *Q1*, clicks on *Q1*'s results, and the next query *Q2*, our objective is to predict whether the user was satisfied with *Q1* or not (i.e. *Q1* was successful, we use the terms satisfied and successful interchangeably throughout the paper). To build toward this goal, we start with a large-scale motivating exploratory analysis of search logs (Section 4), build methods for predicting query reformulation (Section 5), and build methods for query success prediction (Section 6).

4. CLICKS AND NEXT QUERY: A LARGE SCALE EXPLORATORY ANALYSIS

We begin with some motivating exploratory analysis of user behavior logs, considering the correlation between click-based and query-based satisfaction indicators. With one week of activity from a large number of users, we identify all query pairs such that a single user entered *Q1* then *Q2* with no intervening queries. 67% of the queries in the dataset had a next query.

For each pair we are interested in the user's query-level satisfaction with *Q1*. Since this dataset has no relevance judgments of any kind we use clicks as a satisfaction indicator. For a large set of pairs, we can calculate a clickthrough rate (CTR) that is the proportion of pairs where *Q1* has at least one click. Since for some clicks the user backs out immediately, we also calculate CTR-30, which is the proportion of pairs where *Q1* has at least one click with a dwell time of 30 seconds or greater (we see no further search activity for at least 30 seconds). Previous work has shown that dwell time exceeding 30 seconds is highly correlated with satisfaction [13]. Clicks are a noisy indicator of relevance, but for a very large set of pairs a higher CTR and higher CTR-30 is some indication of greater satisfaction with *Q1*.

Our query-based satisfaction indicators are based on query similarity and time between queries. Here we say that *Q1* and *Q2* overlap if, after lowercasing, tokenization, and removing stop-words, the queries have at least one token in common. Consider the query *Q1* "*la map*", where the user's intention is to find maps of Louisiana (abbreviated as LA). If the results of *Q1* consist of maps of Los Angeles, then *Q2* is more likely to reformulate the query, for example "*Louisiana map*". Issuing another "*map*" query would be less likely if *Q1* returned relevant results. In this case, reformulation is an indicator of dissatisfaction. In this section, we use word overlap as a proxy for reformulation (i.e. *Q2* is considered a reformulation of Q1 if they have at least one non-stop-word term overlap). Note, we later build on this intuition by considering richer notions of *Q1*-*Q2* similarity.

Our other indicator of satisfaction is the time between *Q1* and *Q2*. Using our previous example, if *Q1* has the wrong maps, *Q2* may show up sooner as the user searches for the right ones. More precisely, we characterize the time between queries as either *quick* (less than or equal to 5 minutes) or *non-quick* (greater than 5 minutes). This threshold was tuned using the dataset described in Section 5. Note, we later build richer models that do not use any hard thresholds on time between queries. Now let us reconsider the maps search, if *Q1* has the right maps, we have more chance of ceasing search activity for 5 minutes or more. In this case, a quick *Q2* is an indication of dissatisfaction. We note that a low CTR-30 and a quick *Q2* are both associated with quick user interactions, so should be correlated in our analysis, though they use quite different thresholds (30 seconds and 5 minutes).

4.1 CTR Analysis of Query Pairs

We analyze the CTRs of various sets of pairs. We show CTR relative to the CTR of all pairs. Reading the first row of Table 1, the pairs with Q1-Q2 overlap had CTR that was 21% below average (relative), while non-overlapping pairs had 11% above average CTR. Quick pairs had 29% below average CTR, with non-quick pairs being 25% above average. The remaining cells show interactions. Lowest CTR is found in quick overlapping pairs (-39%). Interestingly, all three values in the non-quick row are similar, indicating that for pairs with 5 or more intervening minutes overlap is not such a useful indicator of dissatisfaction.

Table 2 presents the same analysis but for CTR-30. As before, although overlap and quick seem like good dissatisfaction indicators on their own (-12% and -7%), there are interactions between the two, and it is really pairs that are quick and overlapping that are the interesting case, with CTR-30 that is 30% below average.

4.2 Query Pair Examples

Via manual sorting and grouping of the query pairs, we can find some illustrative examples of agreement and disagreement between our satisfaction indicators. For example, in pairs where *Q1* is "*chicago tribune*" we see a high CTR-30, and relatively few cases where *Q2* is quick and overlapping. These all indicate query-level success and we agree, it seems like successful navigational behavior.

By contrast, it is possible for a single user session to confound all our indicators. A user searching for "*how tall is X*" for many celebrities named X will be typing many overlapping queries in quick succession. If the search engine has the factoid answer on the results page, the user also does not need to click (good abandonment). To identify that the user is actually satisfied at each query, and indeed we think they saw the factoid answer, we will need a more nuanced definition of query similarity, as will be presented later in the paper. The surprisingly high CTR of overlapping non-quick pairs could also be related to our simple definition of similarity. We observed high CTR, high overlap rate, low quick rate for pairs where Q1 is "*christmas crafts for kids*". In this case, the user may have query-level SAT but naturally carries on and searches for related queries such as "*easy snowman christmas crafts*".

Pairs where Q1 is "*chiropractor*" have a relatively low CTR-30 and relatively high chance of being followed by a quick and overlapping Q2. The most frequent overlapping Q2 cases are "*chiropractor directory*", "*what is a chiropractor*", "*chiropractor salary*" and "*chiropractor school*". Many other Q2 cases add a location, for example "*chiropractor pittsburgh*".

If the relevance of Q1 improved, by adding informative results for the user to click, we might see higher CTR-30 and fewer quick overlapping follow up queries. If Q1 relevance was improved by adding an inline answer to the "*what is*" question, requiring no clicks, then CTR-30 would give an incorrect indication that users were dissatisfied. However, we note that our click-based analysis in Tables 1 and 2 are affected by cases such as good abandonment and the general noisiness of click data. This highlights the importance of now moving to datasets with explicit success judgments.

In the next sections, we build on the observations, from the large scale log analysis described in this section, to build richer models of query success prediction using both click data and query reformulation data. The analysis is this section highlighted the relation between query reformulation and click through rate, where click through rate is used as a proxy for success. The analysis also emphasized the importance of building more nuance query reformulation prediction models (Section 5), richer query success prediction models (Section 6), and datasets with explicit success judgments (Section 7).

5. QUERY REFORMULATION PREDICTION

Query Reformulation is the act of submitting a query Q2 to modify a previous search query Q1 in hope of retrieving better results. Hence, query reformulation is considered an indication of dissatisfaction with the previous query. For Q2 to be considered a reformulation of Q1, both queries must be intended to satisfy the same information need. Note that a related query on the same topic addressing a different information need is *not* considered as query reformulation for our purpose (e.g., *"best gre practice tests"* → *"gre powerpre"*). In this section, we propose methods for automatically predicting whether the current query is a reformulation of the previous query.

5.1 Query Normalization

We perform standard normalization where we replace all letters with their corresponding lower case representation. We also replace all runs of whitespace characters with a single space and remove any leading or trailing spaces.

In addition to the standard normalization, we also break queries that do not respect word boundaries into words. Word breaking is a well-studied topic that has proved to be useful for many natural language processing applications. This becomes a frequent problems with queries when users do not observe the correct word boundaries (for example: "*southjeseycraigslist*" for "*south jersey craiglist*") or when users are searching for a part of a URL (for example "*quincycollege*" for "*quincy college*"). We used a freely available word breaker Web service available[1] that has been described at [37].

5.2 Queries to Keywords

Lexical similarity between queries has been often used to identify related queries [24]. The problem with lexical similarity is that it introduces many false negatives (e.g. synonyms), but this can be handled by other features as we will describe later. More seriously, it introduces many false positives. Take the following query pair as an example Q1: "*weather in new york city*" and Q2: "*hotels in new*

[1] http://web-ngram.research.microsoft.com/info/

Table 3. Examples of queries, and the corresponding segmentation into keywords. Different tokens in a keyword are separated by "_"

Query	*Phrases and Keywords*
hotels in new york city	hotels in new_york_city
hyundai roadside assistance phone number	hyundai roadside_assistance phone_number
kodak easyshare recharger chord	kodak_easyshare echarger_cord
user reviews for apple iphone	user_reviews for apple_iphone
user reviews for apple ipad	user_reviews for apple_ipad
tommy bhama perfume	tommy_bhama perfume

york city". 80% of the words are shared between Q1 and Q2. Hence, any lexical similarity feature would predict the

user submitted Q2 as a rewrite of Q1. What we would like to do is to have a query representation that recognizes that the first query has two keywords: "*weather*" and "*new york city*" and the second has also two keywords "*hotels*" and "*new York city*" and that only 50% of the keywords are shared between the queries.

To build such a representation, we segment each query into keywords. Query segmentation is the process of taking a user's search query and dividing the tokens into individual phrases or semantic units [6]. Consider a query $x = \{x_1, x_2, \dots x_n\}$ consisting of n tokens. Query segmentation is the process of finding a mapping: $x \rightarrow y \in Y_n$, where y is a segmentation from the set Y_n. Many approaches to query segmentation have been presented in recent research. Some of them pose the problem as a supervised learning problem [6] [43]. Many of the supervised methods though use expensive features that are difficult to re-implement.

On the other hand many unsupervised methods for query segmentation have also been proposed [14][27]. Most of these methods use only raw web n-gram frequencies and are very easy to re-implement. Additionally, Hagen et al. [15] have shown that these methods can achieve segmentation accuracy comparable to current state-of-the-art techniques using supervised learning. We opt for the unsupervised techniques to perform query segmentation. More specifically, we adopt the mutual information method (MI) used throughout the literature. A segmentation S for a query q is obtained by computing the pointwise mutual information score for each pair of consecutive words. More formally, for a query $x = \{x_1, x_2, \dots x_n\}$:

$$PMI(x_i, x_{i+1}) = log\frac{p(x_i, x_{i+1})}{p(x_i).p(x_{i+1})}$$

where $p(x_i, x_{i+1})$ is the joint probability of occurrence of the bigram (x_i, x_{i+1}) and $p(x_i)$ and $p(x_{i+1})$ are the individual occurrence probabilities of the two tokens x_i and x_{i+1}.

A segment break is introduced whenever the point wise mutual information between two consecutive words drops below a certain threshold τ. The threshold we used, τ = 0.895 , was selected to maximize the break accuracy [24] on the Bergsma-Wang-Corpus [6]. In our implementation, the probabilities for all words and n-grams have been computed using the freely available Microsoft Web N-Gram Service [20]. Table 3 shows different examples of queries, and the corresponding phrases.

5.3 Matching Keywords

Two keywords may have full term overlap, partial term overlap, or no direct overlap yet are semantically similar. To capture phrase similarity, we define four different ways of matching phrases ranked from the most to the least strict:

1- Exact Match: The two phrases match exactly.

2- Approximate Match: To capture spelling variants and misspelling, we allow two keywords to match if the Levenshtein edit distance between them is less than 2.

3- Semantic Match: We compute the keyword similarity by measuring the semantic similarity between the two phrases representing the keywords. Let $Q = q_1 \dots q_J$ be one phrase and $S = s_1 \dots s_I$ be another, the semantic similarity between these two phrases can be measured based on WordNet [32]. WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept [32]. Synsets are interlinked by means of conceptual-semantic and lexical relations.

To capture semantic variants, we match two terms if their similarity according to the WordNet Wu and Palmer measure (wup) is greater than 0.5. The Wu and Palmer measure [42] calculates relatedness by considering the depths of the two synsets in the WordNet taxonomies, along with the depth of the Least Common Subsumer (LCS). The measure is computed as follows:

$$wup(t_i, t_j) = \frac{2 * depth(LCS)}{depth(t_i) + depth(t_j)}$$

where the depth of any synset in WordNet is the length of the path connecting it to the root node plus one.

To measure the similarity between the two phrases, we calculate the number of matched terms between Q and S and divide it by the sum of the to the of matched terms between Q and S, the terms in Q that did not match any terms in S and the terms in S that did not match any terms in Q. This is similar to computing the Jaccard distance between the terms in Q that did not match any terms in Q. This is similar to calculated the Jaccard distance between Q and S except that terms are considered identically if they can be matched using the Wu and Palmer measure described earlier.

5.4 Features

5.4.1 Textual Features

Jones and Klinkner [24] showed that word and character edit features are very useful for identifying same task queries. The intuition behind this is that sequence queries which have many words and/or characters in common tend to be related. We repurpose those features for detecting satisfaction. The features they used are:

- normalized Levenshtein edit distance
- 1 if lev > 2, 0 otherwise
- num. characters in common starting from the left
- num. characters in common starting from the right
- num. words in common starting from the left
- num. words in common starting from the right
- num. words in common
- Jaccard distance between sets of words

5.4.2 Keyword Features

As we explained earlier the word and character edit features capture similarity between many pairs of queries. However, they also tend to misclassify many other pairs especially when the two queries share many words yet have different intents. We used the keyword representation of queries described in Section 5.2 to compute the following set of features:

- num. of "exact match" keywords in common
- num. of "approximate match" keywords in common
- num. of "semantic match" keywords in common
- num. of keywords in Q1
- num. of keywords in Q2
- num. of keywords in Q1 but not in Q2
- num. of keywords in Q1 but not in Q2
- 1 if Q1 keywords all Q2's keywords
- 1 if Q2 keywords all Q1's keywords

5.4.3 Other Features

Other features, that have been also used in [24], include temporal features:

- time between queries in seconds
- time between queries as a binary feature (5 mins,
- 30 mins, 60 mins, 120 mins)

and search results feature:

- cosine distance between vectors derived from the first 10 search results for the query terms.

6. QUERY SUCCESS PREDICTION

Now that we can predict whether a query *Q2* is a reformulation of the previous query *Q1* using the methods from the previous section, we move on to addressing the main problem of this study. The problem we are trying to solve in this section is given a query *Q1*, can we predict whether the user was satisfied with *Q1*'s results or not using information about the next query *Q2* and the clicks on *Q1*'s results if any.

We discussed earlier how search can be viewed as an interactive process that involves the user and the search engine. When a user submits a query Q, sometimes, the user gets satisfying results and ends her search, moves on to another unrelated search, or moves on to another related search but with a different information need. On the other hand, when the user does not get a satisfying result, the user may abandon her search or try to reformulate the query in hope of getting better results.

Previous work often assumes that a click is a strong evidence of satisfaction. This has been evident in the use of click through rate (CTR), long dwell time clicks and other similar variants as features to predict query success and query-URL relevance. However, there are many cases where users click on a result and later find out that it is not relevant. Previous work has shown that the probability of click is influenced by a document's position in the results page [10] which results in more clicks for highly ranked results even if they are not relevant. It has also been shown that the "attractiveness" of the title and snippet of the result may lead to a user to clicking on this result [26], only to later find out that it is not relevant. An example of such behavior has been shown earlier in Figure 1.

Hence, we build models to improve query success prediction using both click information and query reformulation. We build and test the following prediction systems:

Table 4. Heuristics vs. Textual vs. Keyword Features for Reformulation Prediction

	Accuracy	*Reform. F1*	*No-Reform. F1*
Heuristic	77.10%	75.60%	76.08%
Textual	82.90%	71.75%	87.75%
Keywords	85.80%	77.30%	88.15%
All	**87.15%**	**81.06%**	**89.86%**

System 1. Clicks Only: A query Q is successful if it receives at least one click

System 2. SAT Clicks Only: The dwell time is the time the user spends on the results page. It has been shown in previous work that longer dwell time is highly correlated with success [11][19]. We consider a query Q successful if it received a click with dwell time greater than τ. Previous work has often used the threshold of 30 seconds to identify successful clicks [14]. We experiment with this threshold and several other thresholds.

System 3. Reformulation Only: We completely ignore clicks and predict the success of the current query based on the next query only. We use all the features from the previous section to predict whether a query is successful or not. This method assumes that users will always reformulate their queries when not successful.

System 4. Reformulation + Clicks (2 stages): Even though, information about the next query is useful for predicting the user satisfaction with the current query, using it alone is problematic because some users simply give up and abandon their searches without reformulation when not satisfied. Using click information, in addition to information about the next query can help us identify these cases. In this approach, we classify the query as unsuccessful if the user reformulates their queries as predicted by the system in the previous section. If the next query was predicted as not being a reformulation of the current query, then the click information is used. We try both system 1 (Clicks Only) and system 2 (SAT Clicks Only) for this purpose.

System 5. Reformulation + Clicks (classifier): Instead of adopting a staged approach as in the previous system, we learn a classifier using both the next query information and the click information simultaneously. We use all the features from the previous section, as well as whether the query received a click or not and the max dwell time if it did receive a click as features to predict whether the current query has been successful or not.

7. EXPERIMENTS AND RESULTS

7.1 Data

Our data consists of several thousands of query pairs randomly sampled from the queries submitted to a commercial search engine during a week in mid-2011. Every record in our data consisted of a consecutive query pair (Q_i, Q_{i+1}) submitted to the search engine by the same user and in the same session (i.e. within less than 30 minutes of idle time, the 30 minutes threshold has been frequently used in previous work, e.g. [39]). Identical queries were excluded from the data. All data in the session to which the sampled query pair belongs were recorded. In addition to queries, the data contained a timestamp for each page view, all elements shown in response to that query (e.g. Web results, answers, etc.), and visited Web page or clicked answers. Intranet and secure URL visits were excluded. Any personally identifiable information was removed from the data prior to analysis.

Table 5. Query Success Prediction Performance

		Accuracy	*SAT Precision*	*SAT Recall*	*DSAT Precision*	*DSAT Recall*	*SAT F1*	*DSAT F1*
1	**Clicks Only**	38.86%	35.88%	64.21%	46.77%	21.51%	46.04%	29.47%
2	**Sat Clicks Only (τ =10s)**	51.20%	42.19%	54.34%	61.10%	49.05%	47.50%	54.42%
3	**Sat Click Only (τ =30s)**	56.07%	46.22%	49.87%	63.75%	60.31%	47.97%	61.98%
4	**Sat Click Only (τ =50s)**	60.61%	51.80%	43.55%	65.18%	72.28%	47.32%	68.54%
5	**Reformulation Only**	79.17%	64.79%	**97.16%**	**97.58%**	68.41%	77.74%	80.43%
6	**Reformulation + Clicks (2 stages)**	73.17%	68.70%	62.37%	75.78%	80.56%	65.38%	78.10%
7	**Reformulation + SAT Click (τ =10s) (2 stages)**	73.22%	73.85%	52.76%	72.97%	87.22%	61.55%	79.46%
8	**Reformulation + SAT Click (τ =30s) (2 stages)**	73.01%	76.51%	48.42%	71.80%	89.83%	59.31%	79.81%
9	**Reformulation + SAT Click (τ =50s) (2 stages)**	71.99%	78.92%	42.37%	70.06%	**92.26%**	55.14%	79.64%
10	**Reformulation + Clicks (Classifier)**	**84.23%**	77.74%	81.19%	88.53%	86.04%	**79.43%**	**87.27%**

A group of annotators were instructed to exhaustively examine each session and "re-enact" the user's experience. The annotators inspected the entire search results page for each of Q_i and Q_{i+1}, including URLs, page titles, relevant snippets, and other features .. They were also shown clicks to aid them in their judgments. Additionally, they were also shown queries and clicks before and after the query pair of interest. They were asked to then use their assessment of the user's objectives to determine whether the user was satisfied with Q_i's results. Different judges were also asked to determine whether Q_{i+1} is a reformulation of Q_i. Each query pair was labeled by at least three judges and the majority vote among judges was used. Because the number of positive instances is much smaller than the number of negative instances, we use all positive instances and an equal number of randomly selected negative instances leaving us with approximately 6000 query pairs.

7.2 Predicting Query Reformulation

In this section we describe the experiments we conducted to evaluate the reformulation prediction system. We perform experiments using the data described in the previous section. We compare the performance of four different systems:

System 1. Heuristics: The first system is a heuristic that does not need any training data. It simply computes the *similarity* between two queries as the percentage of common words to the length of the longer query in terms of the number of words. When finding common words, it allows two words to be matched if their Levenshtein edit distance is less than or equals 2. The second query is predicted to be a reformulation of the first if $similarity \geq \tau_{sim}$ and the $time\_difference \leq \tau_{time}$ mins. The two thresholds were set to 0.35 and 5 minutes respectively using grid search to maximize accuracy over the training data. We present the results of this baseline to provide a simple method that is easy to re-implement and does not need any training data and can be easily used in future research.

System 2. Textual: The second system uses the textual features from previous work that have been described in Section 5.4.1 and the temporal and results features described in Section 5.4.3.

System 3. Keywords: The third method uses the keyword features that we presented in Section 5.4.2 and the temporal and results features described in Section 5.4.3.

System 4: All: Finally the last system uses both the textual features, the keyword features and the temporal and results features.

All the last three methods use the temporal and search results similarity features that have been described in Section 5.4.3. We do not report the contribution of temporal and web search results features here due to space limitation and because they have already been studied in previous work [26].

The accuracy, positive (reformulation) F1, and negative (non-reformulation) F1 for the four methods are shown in Table 4. The results show that the keyword features outperform the textual features. Combining them together results in a small gain over using the keyword features only. The keyword features were able to achieve higher precision rates while not sacrificing recall because they were more effective in eliminating false reformulation cases.

7.3 Predicting Query Success

Next, we move on to the experiments we conducted to predict query success using both click information and information about the next query. The results are shown in Table 5. The first row corresponds to the "*Click Only*" method which predicts a query as successful if it received one or more clicks. The following three rows correspond to systems that use click information only but also takes dwell time into consideration to identify satisfied clicks ("*SAT Click Only*"). The three systems require a certain amount of dwell time for a click to count (10, 30, and 50 seconds) respectively. Next we move on to the "*Reformulation Only*" method which completely ignores the clicks and predicts a query as successful if the next query is not a reformulation of the current query. The following 4 rows correspond to the "*Reformulation + Clicks (2 stages)*" methods that predict a query as successful if the next query is not a reformulation of the current query (as the previous one) and the query received at least one click (with different thresholds on dwell time as before). Finally, the last system ("*Reformulation + Clicks (classifier)*") learns a classifier where the reformulation features, from Section 5.4, and click features to predict query success.

We evaluate the methods using the following metrics accuracy, precision, recall and F1 measure of the "Satisfied Queries" class (SAT Precision, SAT Recall and SAT F1), and precision, recall and F1 measure of the "Dissatisfied Queries" class (DSAT Precision, DSAT Recall and DSAT F1).

We notice from the results in Table 5 that the "*Clicks Only*" method performs poorly with low accuracy and very low SAT precision and DSAT recall. This confirms our hypothesis that many queries that receive a click still end up being unsuccessful. As we introduce a threshold on the dwell time of the clicks for them to be considered, the performance improves (rows 2 – 4). As we increase the dwell time threshold, we see an improved accuracy and precision but at the expense of recall. This is expected since the higher the threshold the more likely that a click becomes a true successful click. However as we increase this threshold, we also miss many successful clicks with shorter dwell time. Using dwell time thresholds of less than 10 seconds or more than 50 seconds did not result in any performance improvement.

Interestingly, when we only use the reformulation signal to predict success ("*Reformulation Only*"), we achieve better performance than using clicks only. Notice however that this is limited to the cases where a next query exists. As shown in Section 4, these correspond to two thirds of the queries as estimated using millions of queries submitted to a commercial search engine. Because we predict any query that has no reformulation as successful, we end up getting very high SAT recall and DSAT precision. The SAT precision and DSAT recall are much lower though.

In rows 6-9, we combine the reformulation signal and the click signal by a simple rule that assumes that successful queries should receive a click (with certain dwell time) and should not be reformulated. This improves the SAT precision and DSAT recall compared to using reformulation only because some DSAT queries are not followed by reformulation. This comes at the expense of the SAT recall and DSAT precision as expected. Like the clicks only cases, increasing the threshold on the dwell time improves most metrics except for the SAT recall. Finally, when we allow the learner to learn a classifier using both the reformulation and the click features, we get the best performance in terms of accuracy SAT F1 and DSAT F1.

7.4 Query Sequences and Search Tasks

In this section, we investigate how the user reformulation behavior is related to the success of the entire search task. We obtained labeled search tasks from the authors of the study described in [17]. Each task is labeled as either satisfied or not by the user performing the search. To gather this data, they deployed a plugin that detected when a user submits a query to any of the three major search engines (Google, Bing, and Yahoo). Users were instructed to submit a satisfaction rating at the end of their search task, where a search task is defined as a single information need that may result in one or more queries [26].The data gathered during that study provided in-situ judgments of satisfaction direct from searchers at the point of task termination. In this dataset, we have 7,628 tasks that were labeled by 218 users. Among these tasks, 98% are tasks with fewer or equal to 5 queries. Table 6 shows the counts of tasks that are successful or unsuccessful with lengths (i.e. the numbers of queries) from 1 to 5.

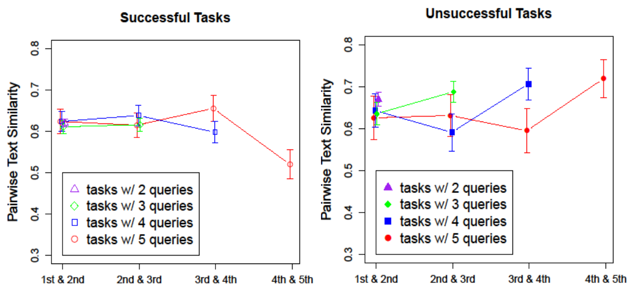
From Table 6, we see that not only the number of tasks is skewly distributed with regard to the task length, the probability of being successful is not uniformly distributed for tasks with different lengths either: the probability for a task to be successful decreases as its length increases. This fact tells us that, in a search task, having more reformulations does not increase the probability of being successful. But rather, having more reformulations is a strong indictation of a possibly difficult task and thus the task has a higher probability of failure.

After examining the relationship between the number of re-queries and the probability for a task to be successful, we control the length of tasks and further examine the different patterns presented by query sequences in successful tasks and unsuccessful tasks of the same length. We compute the similarity between pairs of consecutive queries and compare these similarities for both succesful and unsucesful tasks. The result of this experiment is shown in Figure 2. The figure shows the textual similarity and keyword similarity between consecutive queries in tasks of different lengths with 2 queries, 3 queries, 4 queries and 5 queries.

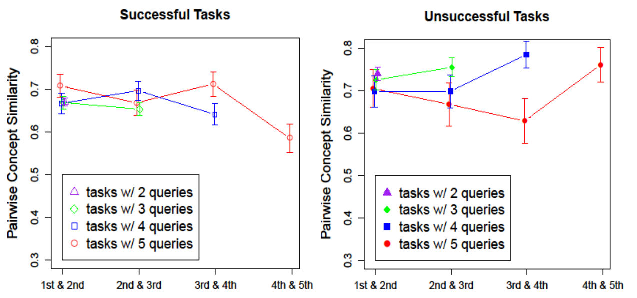
From Figure 2, we see that the relationships between consecutive queries in successful tasks and unsuccessful tasks are showing different patterns. Especially, we notice that the last pair of queries in a successful task is much less similar to each other than an unsuccessful task, using both text and keyword similarity. This is also the case for tasks as short as having a length if 2 (i.e. having two queries only). We also see that the textual and keyword similarity between the only two queries are significantly higher for unsuccessful tasks than successful tasks. This observation tells us that there is significant difference between re-query behavior in a successful task and in an unsuccessful task.

In order to get a better understanding on why successful tasks and unsuccessful tasks are showing such distinct patterns as shown in Figure 2, we examine a few individual examples. We find that it is often the case that, in a successful task, the user adopts some completely new term(s) or novel information from the landing page(s) of URL(s) she clicks after issuing the previous queries in the same task. Figure 3(a) shows an example of a sequence of queries in a successful task. The term "*powerprep*" in the last query is from the landing page the user clicks after issuing the second last query, and it doesn't appear in any of the previous queries in the same task. On the other hand, we observe that for most unsuccessful tasks, the users are not able to get much information from their previous queries, and thus the terms used for the entire tasks are very similar to each other. Figure 3(b) gives an example of an unsuccessful task, in which although the user has a few clicks, she does not get much useful information from those clicks to reformulate the query and achieve success. We also observe that the average time gap between consecutive queries of successful tasks is significantly longer than unsuccessful tasks (i.e. 91.10 seconds vs. 73.07 seconds with p-value < 0.05). This fact further supports our hypothesis that in a successful task, a user tends to spend more time to gather useful information in each intermediate step before reaching full success eventually.

In summary, in this part we investigate the relationship between query reformulation patterns in search tasks. We have the following two major observations regarding users' different re-query behavior in successful tasks and unsuccessful tasks. The first observation is that unsuccessful tasks tend to have more re-queries than successful tasks. Which means, having more re-queries in a task doesn't infer a higher probability of success instead, it is a strong indictation of a possibly difficult task and the probability of failure is higher.



(a) Textual similarity between pairs of consecutive queries of tasks with different lengths.



(b) Keyword similarity between pairs of consecutive queries of tasks with different lengths.

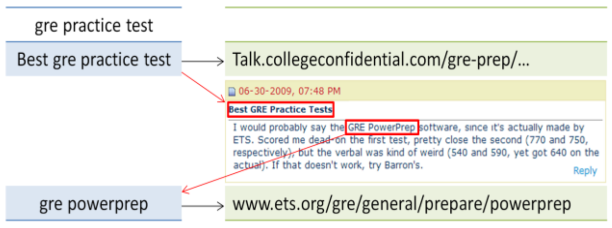
Figure 2: Different patterns of text similarity and keyword similarity of successful tasks versus unsuccessful tasks.

The second observation is more interesting, as we see the pairwise query similarity of successful tasks and unsuccessful tasks are showing very different patterns. In particular, queries of an unsuccessful task tend to be more similar to each other than queries of a successful task. This suggest that queries in a successful task are more likely to be intended to cover different aspects of the information need while queries in an unsuccessful tasks are more likely to be trying to express the same information in different ways.

By investigating individual examples, we find that in successful tasks users are more likely to reformulate their queries with the novel information gained from the results of previous queries. This suggests that users are learning useful information from the results of previous queries and using it to formulate the following queries. However, in unsuccessful tasks, users usually don't gain much novel or useful information from the previous queries and thus the re-queries tend to be similar to each other. This fact suggests that for a task with multiple queries, search success is an *incremental* process. This also agrees with a similar assumption used in previous work [11], where session utility model was presented that is based on the assumption that users reach partial satisfaction at the intermediate steps of a successful task.

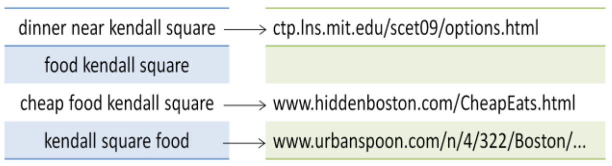
8. CONCLUSIONS

Previous models of query-level satisfaction have focused on clickthrough data as the main signal for predicting query success or query document relevance. Clickthrough information has also been shown to be noisy due to snippet bias and result presentation bias. This bias leads users to clicking on results because they think they are relevant (perceived relevance) only to find out later that some



I would probably say the GRE PowerPrep software, since it's actually made by ETS. Scored me dead-on the first test, pretty close the second (770 and 750, respectively), but the verbal was kind of weird (540 and 590, yet got 640 on the actual). If that doesn't work, try Barron's.

(a) Example: a successful task with 4 queries (queries are shown on the left side and clicks on the right side). The last query has a new term that comes from a landing page of a URL the user clicks after issuing the third query.


dinner near kendall square
ctp.lns.mit.edu/scet09/options.html
food kendall square
cheap food kendall square
www.hiddenboston.com/CheapEats.html
kendall square food
www.urbanspoon.com/n/4/322/Boston/...

(b) Example: an unsuccessful task with 4 queries (queries are shown on the left side and clicks on the right side). Although the user has a few clicks after some of the queries, the terms used in all queries are very similar.

Figure 3: Examples of query sequences of a successful task and an unsuccessful task.

of these clicks are not relevant (Actual relevance). In this work, we addressed this shortcoming by introducing models of query-level success that draw conclusions on query success based on information about the next query submitted by the user in addition to click information. The following query submitted by a user in a search session reveals information about the user's intent as well as the user's satisfaction with the current search results.

Through experimentation via labeled query pairs drawn from logs of a commercial search engine, we show that our proposed models can accurately identify query reformulations by users dissatisfied with the results of their current query. We also showed that our proposed models can predict query-level satisfaction more accurately than baselines that use clickthrough features only. Additionally, we studied the relation between sequences of queries in satisfied vs. dissatisfied search tasks. We observe that search success is an incremental process for most multi-query tasks. That is, in a successful task, a user is more likely to gain some novel information from the search results of intermediate queries before they reach final success.

We used three datasets through this paper: a dataset of reformulation pairs, one-week worth of query logs to conduct the exploratory motivating study, and search tasks data for the task analysis. To reproduce these experiments, the reader may use any search engine log data, such as the AOL data, to conduct the exploratory analysis. For the query success experiments, query pairs sampled from the same log data should be judged by annotators to label satisfaction and reformulation. Finally, for task-level data, the reader may use the data in [1], or collect similar data using human annotators on in-situ judgments.

REFERENCES

[1] Ageev., M., Guo, Q., Lagun, D., and Agichtein, E. (2011). Find it if you can: a game for modeling different types of web search success using interaction data. In Proc. SIGIR, 345–354.

[2] E. Agichtein, E., Brill, E., and S. T. Dumais, S.T.. (2006). Improving web search ranking by incorporating user behavior information. In Proc. SIGIR, 19–26.

[3] Anick, P. (2003). Using terminological feedback for web search refinement: a log-based study. In Proc. SIGIR, 88–95.

[4] Arlitt, M. (2000). Characterizing Web user sessions. ACM SIGMETRICS Performance Eval Review, 28(2), 50–63.

[5] Berger, A.L. and Lafferty, J. (1999). Information retrieval as statistical translation. In Proc. SIGIR, 222–229.

[6] Bergsma, S., and Wang Q. I. (2007). Learning Noun Phrase Query Segmentation. In Proc. EMNLP, 819–826

[7] P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna. (2008). The query-flow graph: model and applications. In Proc. CIKM, 609-618.

[8] Carterette, B. and Jones, R. (2007). Evaluating search engines by modeling the relationship between relevance and clicks. In Proc. NIPS, 217–224.

[9] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. (2008). An experimental comparison of click position-bias models. In Proc. WSDM.

[10] Dupret, G., and Liao, C. (2010). A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine. In Proc. WSDM, 181-190.

[11] G. Dupret, V. Murdock, and B. Piwowarski. (2007). Web search engine evaluation using clickthrough data and a user model. In WWW workshop on Query Log Analysis: Social and Technological Challenges.

[12] Feild, H., Allan, J., and Jones, R. (2010). Predicting searcher frustration. In Proc. SIGIR, 34−41.

[13] Fox, S., Karnawat, K., Mydland, M., Dumais, S.T., and White, T. (2005). Evaluating implicit measures to improve the search experience. ACM TOIS, 23(2), 147–168.

[14] Hagen, M., Potthast, M. Stein, B, and Bräutigam, C. (2011). Query segmentation revisited. In Proc. WWW, 97−106.

[15] Hagen, M., Potthast, M. Stein, B, and Bräutigam, C. (2010). The Power of Naïve Query Segmentation. In Proc. SIGIR, 797–798.

[16] Hassan, A., Jones, R., and Klinkner, K.L. (2010). Beyond DCG: user behavior as a predictor of a successful search. In Proc. WSDM, 221–230

[17] Hassan, A., Song, Y., and He, L. (2011). A task level user satisfaction model and its application on improving relevance estimation. In Proc. CIKM, 125–134.

[18] Hassan, A. (2012). A semi-supervised approach to modeling web search satisfaction. . In Proc. SIGIR, 275–284.

[19] Huang, J. and Efthimis N. (2009). Analyzing and evaluating query reformulation strategies in web search logs. In Proc. of CIKM, 77−86.

[20] Huang, J., Gao, J., Miao, J., Li, X., Wang, K. and Behr, F.. (2010). Exploring Web Scale Language Models for Search Query Processing. In Proc. WWW, 451–460.

[21] Huffman, S. and M. Hochster, M. (2007). How well does result relevance predict session satisfaction? In Proc. SIGIR, 567−-574.

[22] K. Jarvelin and J. Kekalainen. (2002). Cumulated gain-based evaluation of IR techniques. ACM TOIS, 20(4), 422–446.

[23] Jansen, B.J., Zhang, M., and Spink, A. (2007). Patterns and transitions of query reformulation during web searching.

[24] Joachims, T. (2002). Evaluating search engines using clickthrough data. Department of Computer Science, Cornell University.

[25] T. Joachims, T., Granka, L,. Pan, B., Hembrooke, H. and Gay, G. (2005). Accurately interpreting clickthrough data as implicit feedback. In Proc. SIGIR , 154–161.

[26] Jones, R. and Klinkner, K.L. (2008) . Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In Proc. CIKM.

[27] R. Jones, B. Rey, O. Madani, and W. Greiner. (2006). Generating Query Substitutions. In Proc. WWW, 387–396

[28] 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.

[29] Lau, T. and Horvitz, E. (1999). Patterns of search: analyzing and modeling Web query refinement. In User Modeling '99, 119-128.

[30] J. Li, S.B. Huffman, and A. Tokuda (2009). Good abandonment in mobile and PC internet search. In Proc. SIGIR, 43–50

[31] C. Lucchese, S. Orlando, R. Perego, F. Silvestri and G. Tolomei. (2011). Identifying Task-based Sessions in Search Engine Query Logs. In Proc. WSDM 2011.

[32] G. A. Miller. 1995. Wordnet: a lexical database for English. Commun. ACM, 38(11):39–41.

[33] Murray, G.C., J. Lin, and A. Chowdhury,. (2006). Identification of User Sessions with Hierarchical Agglomerative Clustering. In ASIS&T '06, 43(1), 1-5.

[34] S. Ozmutlu. Automatic new topic identification using multiple linear regression. Information Processing and Management, 42(4):934-950, 2006

[35] F. Radlinski and T. Joachims. Query chains: learning to rank from implicit feedback. In R. Grossman, R. Bayardo, and K. P. Bennett, editors, KDD, pages 239-248. ACM, 2005.

[36] Teevan, J., Adar, E., Jones, R., and Potts, M. (2007) Information Re-Retrieval: Repeat Queries in Yahoo's Logs. In Proc. SIGIR.

[37] Toutanova, K., Klein, D., Manning, C., and Singer, Y. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In Proc. HLT-NAACL, 252-259.

[38] Wang, K., Thrasher, C., Hsu, B. (2011). Web Scale NLP: A Case Study on URL Word Breaking. In Proc.WWW, 357–366.

[39] Weber, I. and Castillo, C. (2010). The demographics of Web search. In Proc. SIGIR, 523–530.

[40] White, R.W. and Drucker, S.M. (2007). Investigating behavioral variability in Web search. In Proc. WWW, 21−30.

[41] White, R.W. and Dumais, S. (2009). Characterizing and predicting search engine switching behaviour. In Proc. CIKM, 87−96.

[42] Wu, Z. and Palmer, M. (1994). Verb semantics and lexical selection. Proc. ACL, 133–138.

[43] Yu., X. and Shi., H. (2009). Query Segmentation Using Conditional Random Fields. Proceedings of the Workshop on Keyword Search on Structured Data (KEYS), 21–26.

[44] Zhang, X., Anghelescu, H.G.B., and Yuan, X. (2005). Domain knowledge, search behavior, and search effectiveness of engineering and science students. Inf. Res., 10(2), 217.