Modeling Dwell Time to Predict Click-level Satisfaction

Youngho Kim[1]*, Ahmed Hassan[2], Ryen W. White[2], and Imed Zitouni[2]
[1] University of Massachusetts, 140 Governors Drive, Amherst, MA 01003, USA
[2] Microsoft, One Microsoft Way, Redmond, WA 98052, USA
yhkim@cs.umass.edu, {hassanam, ryenw, izitouni}@microsoft.com

ABSTRACT
Clicks on search results are the most widely used behavioral signals for predicting search satisfaction. Even though clicks are correlated with satisfaction, they can also be noisy. Previous work has shown that clicks are affected by position bias, caption bias, and other factors. A popular heuristic for reducing this noise is to only consider clicks with long dwell time, usually equaling or exceeding 30 seconds. The rationale is that the more time a searcher spends on a page, the more likely they are to be satisfied with its contents. However, having a single threshold value assumes that users need a fixed amount of time to be satisfied with any result click, irrespective of the page chosen. In reality, clicked pages can differ significantly. Pages have different topics, readability levels, content lengths, etc. All of these factors may affect the amount of time spent by the user on the page. In this paper, we study the effect of different page characteristics on the time needed to achieve search satisfaction. We show that the topic of the page, its length and its readability level are critical in determining the amount of dwell time needed to predict whether any click is associated with satisfaction. We propose a method to model and provide a better understanding of click dwell time. We estimate click dwell time distributions for SAT (satisfied) or DSAT (dissatisfied) clicks for different click segments and use them to derive features to train a click-level satisfaction model. We compare the proposed model to baseline methods that use dwell time and other search performance predictors as features, and demonstrate that the proposed model achieves significant improvements.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Search Process.

Keywords
Dwell time analysis; User behavior; Click satisfaction.

1. INTRODUCTION
Implicit measures of user behavior are important for enhancing web search quality [14][42]. Since obtaining explicit feedback from users is prohibitively expensive and challenging to implement in real-world Information Retrieval (IR) systems, commercial search engines have exploited implicit feedback signals derived from user activity (e.g., search result clickthrough statistics). Previous work has extensively studied implicit feedback measures (e.g., mouse scrolling, gaze tracking, physiological signals, etc.), and verified their effectiveness in predicting search satisfaction (or dissatisfaction) (e.g., [6][13][14][16][20][31]).

Table 1: Click examples. Label means whether a click is satisfied (SAT) or dissatisfied (DSAT)

No.	Query	Label	Dwell Time	Readability Level
1	*abdominal aortic aneurysm*	DSAT	55s	Difficult
2	*what to serve for dinner along with chicken parm*	SAT	19s	Easy

Among many implicit measures, *click dwell time* (i.e., the time that the user spends on a clicked result) is one of the most important features because it is clearly correlated with result-level satisfaction or document relevance [6][10][14]. Longer dwell time on a clicked page has traditionally been used to identify satisfied (SAT) clicks. While clickthrough statistics can be sometimes misleading due to order and caption biases, click dwell time is a more robust measure. Click dwell time has been successfully used in a number of retrieval applications (e.g., implicit relevance feedback [42], re-ranking [2], and query expansion [6]). In those applications, SAT clicks are simply identified by some predefined time threshold (i.e., a click is SAT if its dwell time equals or exceeds that threshold). A dwell time equaling or exceeding 30 seconds, as proposed in [14], has typically been used to identify clicks with which searchers are satisfied. However, the dwell time depends on page content and has been shown to vary based on other factors such as search task and user [32]. A more robust interpretation of click dwell time is therefore needed to more accurately predict SAT (or DSAT (dissatisfaction)) clicks. We address this important challenge with the research presented in this paper. Note that we use the terms "dwell time" and "click dwell time" interchangeably to refer to the time spent on a search result.

We assume that the dwell time required by a given user to locate the required information on a landing page varies depending on factors such as query type, page topic, page content, and page readability level. We refer to these factors as the *query-click attributes*. For example, longer dwell time would be required when the user clicks on pages related to technical topics, even if the pages are not relevant. Table 1 shows two click examples drawn from our experiments where one click is dissatisfied with a long dwell time and the other is satisfied with a short dwell time. In the first example, the query is on a medical topic, and the contents of the clicked page have a high readability level. In contrast, the clicked page in the second example is related to food, and the user can easily comprehend this page in less time. In both examples, the 30-second threshold typically used in previous work (e.g., [6][42]) appears ineffective in determining whether a click is satisfied.

* This work was conducted while interning at Microsoft Research

WSDM'14, February 24–28, 2014, New York, New York, USA.

http://dx.doi.org/10.1145/2556195.2556220

To better understand click dwell time, we propose a method to model it given query-click attributes (e.g., readability level, search topics, etc.). This model helps to identify dwell time distributions of SAT and DSAT clicks for different segments of clicks. From this, we address the following important research questions:

RQ1. How can we model $p(t|\text{SAT}, att)$ (where t and att denote dwell time and query-click attributes, respectively)?

RQ2. How can $p(\text{SAT}|t, att)$ or $p(\text{DSAT}|t, att)$ be calculated?

RQ3. How can SAT dwell time thresholds be identified for particular queries, topics, or reading levels?

To model click dwell time, we collect click instances and label them as either SAT or DSAT. We then generate query and click attributes for each instance, and use them to identify different click segments (i.e., groups of clicks). Given these attributes, we could have used every attribute as a click segment (e.g., query type is "navigational"). However, some of these attributes are very broad and can result in general dwell time models that are similar to global models derived from all data. To address this problem, we apply a data mining approach for identifying attribute associations, and each association serves as a click segment. Once segments are identified, we derive distributions to model dwell time of SAT and DSAT clicks from each segment using Maximum Likelihood Estimation (MLE).

After deriving dwell time distributions, we develop an estimator to predict SAT and DSAT dwell time distributions for new (unseen) clicks. Since the generated segments may not cover all possible clicks, this estimator is required to expand the coverage of our method. The estimator is then used to devise a classification model to predict click satisfaction. This model includes dynamic dwell time features which identify probabilistic distances of a click instance to SAT and DSAT distributions, and we show that these features can significantly improve prediction performance.

The remainder of the paper is structured as follows. Section 2 describes relevant related work in the areas of implicit feedback analysis, search satisfaction, and search performance prediction. In Section 3, we describe how we generate and label the click data used for our method. Section 4 proposes our method to model click dwell time and provides dwell time analysis. In Section 5, we describe a click satisfaction model. We present experimental results in Section 6 and conclude in Section 7.

2. RELATED WORK

There are three main areas of related work that are relevant to the research presented in this paper: (1) implicit feedback, where inferences about intentions and relevance can be made from search behavior; (2) search satisfaction modeling, where search behavior is used to estimate user satisfaction, and; (3) search performance prediction, where estimates are made about result quality in advance of querying the search engine, or in some cases based on the results returned. We cover each of these areas in turn.

2.1 Implicit Feedback Analysis

The utility of implicit feedback as a relevance estimator has been extensively studied. Early research [29] found that implicit feedback is useful for inferring relevance. Kelly and Belkin [32] demonstrated a correlation between the amount of time spent examining a document and its relevance.

Implicit feedback based on search-result clicks has also been used to infer search preferences [3]. Radlinski et al. [39] showed that interleaving the results of two ranking functions and presenting the interleaved results to users can serve as a good predictor of relative search engine performance. In their analysis, they discovered that metrics including abandonment and reformulation did not predict relative performance as accurately as interleaving. Another way to use implicit feedback is to provide training data for learning-to-rank algorithms. Joachims [26] presented one of the earliest studies on using clickthrough data for optimizing learning to rank algorithms. Agichtein et al. [2] showed that incorporating behavior data can significantly improve the relevance of the top results in Web search. Other forms of feedback such as eye gaze and cursor movements have also been used for estimating document relevance [6][16].

Beyond search result interactions, other researchers have tried to understand browsing behavior by examining time spent on pages [35]. This work was not limited to Web search and did not differentiate between visits to useful pages and other non-relevant pages. In this work, we focus on predicting click-level satisfaction using dwell time as an implicit feedback signal. This work differs from previous research in this area by moving beyond the use of clicks or dwell time as signals to richer models of click dwell time that take contextual, topical, and content information into consideration.

Most recently, Yin et al. [44] used dwell time to capture user's voting actions for recommending social media content. They interpreted longer dwell time on the content as "pseudo votes", and improved recommendation performance by modeling the user's expectation levels that cause viewing and voting behaviors.

2.2 Search Satisfaction Modeling

There is significant literature on predicting satisfaction from user behavior. Proposed methods include correlating search behavior, such as clicks and dwell time, with satisfaction labels collected either first hand from users or using third-party judges.

Recent work has targeted behavioral modeling and using behavior to construct models of user satisfaction [1][19][20][22]. Fox et al. [14] found a strong correlation between search activity and user satisfaction labels gathered in-situ via a browser plugin. They modeled explicit satisfaction ratings provided by searchers using features such as clickthrough rate, click dwell time, and features associated with session termination. Hassan et al. [20] developed models of user behavior to accurately estimate search success on a task level, independent of the relevance of documents retrieved by the search engine. Hassan [19] proposed another satisfaction prediction model that can learn from both labeled and unlabeled data. Ageev et al. [1] proposed a formalization of different types of success for informational search, and presented a scalable game-like infrastructure for crowdsourcing studies of search behavior, specifically targeting the capture and evaluation of successful search strategies on informational tasks with known intent. Their model can predict search success effectively on their data and on a separate set of logs comprising search engine sessions.

Rather than relying on explicit search satisfaction labels, Huffman and Hochster [25] used a different approach where they studied the correlation between user satisfaction and simple relevance metrics. They reported a fairly strong correlation between user satisfaction and linear models encompassing the query-document relevance of the first three results for the first query in the search task.

In this work, we focus on *click level satisfaction* as opposed to task or session level satisfaction. Previous work on satisfaction prediction has focused on user action sequences and on generating a prediction for the full task. However, we argue that being able to

determine satisfaction at the individual click level is important for a number of applications, such as ranking, where satisfied clicks are used. As stated earlier, to date, work on click level satisfaction has employed a fixed threshold on dwell time to distinguish satisfied and dissatisfied clicks, with the most popular threshold being 30 seconds [14]. This is a crude estimate that ignores many factors that can affect dwell time. We show that with our method we can attain more accurate satisfaction estimates than with dwell time alone.

2.3 Search Performance Prediction

Another related line of work is research on search performance prediction. Several search performance predictors have been proposed in the literature (e.g., query clarity [12], inverse collection term frequency [23], and weighted information gain [46]). These predictors do not require relevance judgments and can be applied at query time to generate an estimate of how the search engine will perform, and hence how likely the user is to be satisfied. These predictors mostly utilize features derived from query information, except [17] which examines the effectiveness of interaction information (e.g., clickthrough statistics). Query clarity proposed in [12] measures relative entropy between query and collection models, i.e., how query terms are discriminant from the search collection. He and Ounis [23] examined several pre-retrieval features (e.g., query scope, query length) and found that inverse collection term frequency and (simplified) query clarity are strongly correlated with precision. Leskovec et al. [34] leveraged graphical properties recognized from the link structure of search results to predict the quality of the results. Zhou and Croft [46] proposed a weighted information gain approach which combines term probability and proximity features. Although their approach is more effective in predicting the quality of top retrieved documents, the proposed predictor requires additional query classification because their method is adapted for specific query types (i.e., named page finding or content-based). Guo et al. [17] utilized user-behavior information for predicting query performance, and found that interaction features mined from search log data can significantly improve prediction performance. Most recently, Kim et al. [33] proposed a method to generate association rules by combining topical and lexical features from the query, and showed that the generated rules are effective for predicting query performance. Continuing this line of work, we use multiple query attributes and also page attributes with the new objective of building better models of click dwell time and estimating click-level search satisfaction rather than query performance.

We now describe our methods and the experiments that we conducted to evaluate their performance. We begin with the data acquired to provide labels from which to characterize SAT and DSAT dwell times and train our models to learn more accurate satisfaction estimates for individual clicks.

3. DATA ACQUISITION

3.1 Human-Labeled Data

Our data was sampled from 10 days of search logs from a large commercial Web search engine. Each log entry contains a query, URLs of every clicked page, and the timestamps of the query and the clicks. The click dwell time was calculated by measuring the time between the click time and the next seen click or query. From this data, we randomly sampled 7,500 queries and their clicks, and asked human assessors to review the search session and estimate the satisfaction associated with every observed click.

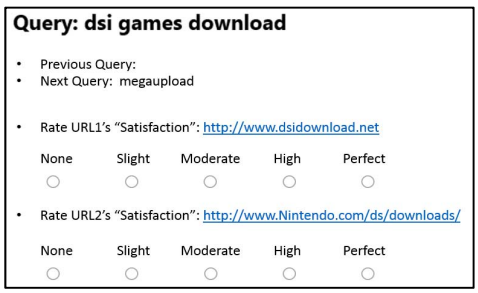

Figure 1: An example of the judgment interface from which we obtained the click-level satisfaction judgments in HLD.

Judges were shown the query, along with the previous and the next query to help them identify the context. For every click, the judges were asked to examine the query and the content of the clicked URL and then rate user satisfaction with the click on a five-point scale with the following response options: *none*, *slight*, *moderate*, *high*, *perfect*. To derive a binary satisfaction score from these multi-point ratings, clicks labeled as *high* or *perfect* were considered as satisfied (SAT); otherwise, the clicks were labeled as dissatisfied (DSAT). Figure 1 shows a screenshot of the judgment interface.

To obtain a ground truth satisfaction estimate for each click, we first collected the judgment results from two annotators. If the two annotators agreed at the binary label, then we use that label. If they disagreed, we ask another judge to label the instance and we use the majority label among the three judges as the satisfaction estimate.

Once every click was judged, we initially identified that 82.8% of all clicks are SAT. This skewness is not surprising because typically most queries end up being satisfied [1][19]. However, such imbalanced data is more challenging to deal with from a statistical learning point of view because the model trained by an imbalanced data would be biased to the majority class (i.e., SAT) and less effective for the minority class (i.e., DSAT). To alleviate this problem, we randomly downsample the majority class to obtain a 50/50 balanced dataset. The final dataset includes 3,204 click instances (1,602 instances are SAT and the others are DSAT). Throughout the paper, we refer to this as the Human-Labeled Data (HLD). This data set is used for evaluating the SAT click prediction model (described in Section 5.2). In addition to this dataset, we will describe a "pseudo labeled" data set in the next subsection that leverages post-click interaction behaviors to estimate click satisfaction. Due to the relatively small size of the human labeled data, we used the pseudo-labeled data to derive probability distributions of click dwell time across click segments as will be described later in the paper.

3.2 Pseudo-Labeled Data

Our method relies on partitioning the set of clicked URLs into smaller segments and learning the dwell time distribution for SAT and DSAT clicks for every segment. If we only partition the human labeled data into segments, we end up with a small number of instances in each segment and hence we may have insufficient data to reliably estimate the dwell time distributions in each segment. To resolve this problem, we build a pseudo-labeled dataset from the search logs and use it to estimate the dwell time distribution. Note that the pseudo labeled data set is only used for training purposes, not for testing; all evaluations are conducted on the human labeled dataset (HLD) only.

To generate the pseudo-labeled dataset, we randomly extracted hundreds of thousands of query impressions from the query logs of the same search engine that provided the data for the HLD assessment. To assign pseudo satisfaction labels to clicks, we assume that a click followed by a query reformulation is a dissatisfied click, while a click not followed by a query reformulation is a satisfied click. A query reformulation is the act of submitting a follow up query to modify a previous search query in hope of retrieving better results. Post-click query reformulation has been used as a predictor of search satisfaction in previous work [21]. The intuition here is that dissatisfied users will reformulate their queries, while satisfied users will not. This is a crude estimate of click satisfaction but it allows us to easily generate large numbers of pseudo labeled instances without leveraging any information about the click itself (which could lead to confounding). We will show later that these pseudo labels can be used to significantly improve click success prediction as measured on the HLD.

To identify query reformulations we use a method similar to that described in [5], where features of query similarity (e.g. edit distance, word overlap, etc.) and time between queries are used to identify query reformulations. Using these assumptions, we randomly collected 104,000 click instances from the search logs of the engine described earlier from March 2013. The data contains the same number of SAT and DSAT instances and does not overlap with any of the human labeled data. We refer to this data set as the Pseudo-Labeled Data (PLD), and randomly divide it into two sets: mining (training) and validation. The mining set contains 91,000 instances, and is used to model click dwell time (Section 4) and training an estimator for dwell time distributions (Section 5.1). Besides, the validation set includes the rest of the data and is used for optimizing the parameters of learning algorithms for SAT click prediction (described in Section 5.2). Note that both data sets are balanced such that there an equal number of SAT and DSAT instances.

4. CLICK DWELL TIME MODEL

In this section, we present our method to model click dwell time. We first identify segments (groups) of clicks, each of which effectively characterizes some of the click instances (Section 4.1). After that, we derive probabilistic models of SAT and DSAT dwell times for each click segment (Section 4.2).

4.1 Click Segment Generation

We generate click segments (i.e., groups of labeled click instances) using query-click attributes (e.g., query types, search topics, readability levels, etc.). In this paper, we assume that these attributes would affect the user's dwell time on a clicked result, and simply each attribute can form a click segment. However, some of these attributes are very broad (e.g., query type is "navigational"). This could result in dwell time models that are too general (similar to the global model which can be derived without the attributes). To avoid this problem, we mine click segments by associating important query-click attributes (e.g., reading difficulty of the text is high and the search topic is "drugs & medicines"). Specifically we first generate many query-click attributes for each click instance. After that, we identify associations of the attributes using decision tree learning. These groups of attributes are used to characterize the click segments.

4.1.1 Attribute Generation

For each click instance, we know the text of the search query, the URL of clicked result, and the downloaded content of the URL. To characterize each instance, we generate the following attributes:

Query Topic Attributes: Open Directory Project (ODP, dmoz.org) provides a Web hierarchy that lists topically similar pages in the same category (predefined) including sub-categories. ODP has been widely used for topical classification of query text (e.g., [4]) or web pages (e.g., [43]). We use the query-based ODP classifier, proposed in [4], to assign topic labels to queries. The classifier used query text and clicked pages associated with every query from aggregated historic search data from the search engine to assign a topic label to each query (see [4] for more details about performance and coverage of the classifier). For each instance, the classifier assigns the five most likely categories with their associated probabilities, and we use the categories whose probabilities are higher than 0.5 (empirically set) as attributes.

Query Type Attributes: Query type classification has been the subject of various research efforts with the objective of understanding search intent (e.g., homepage finding task [28]). We leverage such query-type categories to characterize query-click instances. To do this, we train text classifiers that can label a query as query-type categories, and use them as binary attributes. It is worth noting that the classifiers are trained using human-labeled query examples since such human involvement can help to improve the accuracy of the classification.

Page Topic Attributes: We use the ODP categories to assign topic labels to every clicked page. Given the large number of pages in our set we need to label them automatically. We perform automatic classification of pages into ODP categories using an approach similar to [41], which involves looking up the URL for each page in ODP and backing off on the URL path as necessary. URLs in the directory are directly classified according to their corresponding categories. Missing URLs are incrementally pruned one level at a time until a match is found in the ODP or a miss declared. The top two levels of the ODP hierarchy are used as topic labels.

Reading Level Attributes: Given a document, the automatic identification of its reading difficulty has been studied in [11]. The reading difficulty of each document is represented on a 12-point scale mapped to American school grade levels. The reading level predictor we use adopts a language modeling approach using a multinomial naïve Bayes classifier, proposed in [11], and this approach has been shown to be effective, as reported in [30]. For each click instance, we first download the content of the page, and the classifier then predicts the reading level of the content.

Overall, we use 600 attributes (219 query-based ODP categories + 363 URL-based ODP categories + 6 query types + 12 reading levels) as binary attributes for mining click segments.

4.1.2 Click Segment Mining

To generate click segments, we can use various algorithms leveraged for data mining (e.g., decision tree learning [38], k-means clustering [36], etc.). We selected decision tree learning because of its simplicity (e.g., clustering approaches need to define a similarity between examples) and efficiency in applying decision rules to instances (i.e., additional computation to identify a membership of a new instance is not necessary). Click segments (i.e., decision rules) are mined as follows. We are given a set of SAT and DSAT click instances, each of which contains 600 binary attributes, and a binary decision tree can be trained using this data to classify an instance into either SAT or DSAT (i.e., the non-leaf nodes in the tree contain any of the 600 attributes). Once a decision tree is learned, we can extract decision rules (where each rule is formed by the path from the root to a leaf node). Each decision rule contains an association of the attributes (e.g., $\text{Shopping} \wedge \text{Computer}$)

ALGORITHM: Iterative Decision Tree Learning

INPUT:

- Set of labeled click instances, $C = \{c_1, c_2, \dots, c_n\}$ where the label of c is $DSAT$ or SAT, i.e., $L(c) \in \{\text{DSAT}, \text{SAT}\}$
- Set of attributes $A = \{a_1, a_2, \dots, a_d\}$
- Min. number of examples in a leaf node, *m*.

OUTPUT: A set of segment rules

PROCESS:

1) Initialize $R = \{\}, T = \{\}, R' = \{\}$
2) Do
3) $N = |R|$
4) $A \leftarrow A - T$
5) Train a binary decision tree, DT which uses A as features and C as examples where $L(q)$ indicates a label of $c \in C$ and the minimum number of examples in a leaf node is set as *m*
6) For each leaf node
7) Generate P the set of nodes (features) in the path from the root to the leaf node in DT
8) $R \leftarrow R \cup \{P\}$
9) End For
10) $T \leftarrow T \cup \{a_t\}$ where a_t is the root node of DT
11) $N' = |R|$
12) While $N \neq N'$
13) For each rule r in R
14) Generate r' by excluding the attribute associated with negation in r
15) $R' \leftarrow R' \cup \{r\}$
16) End For
17) Return R'

Figure 2: Iterative decision tree learning algorithm.

and can form a click segment. To extend the rule coverage, we ignore negated attributes (e.g., ¬Recreation).

To mine more segments, we devise an iterative learning algorithm that iteratively learns multiple decision trees with different attribute sets. Specifically, we first learn a decision tree using all attributes (defined in Section 4.1.1), and extract all rules from the learned tree. In the next iteration, we identify the root node (i.e., top attribute) of the previous tree and exclude it from the current learning iteration. In other words, the decision tree learned from the current iteration does not generate the rules containing the root node. By doing this, we can discover new rules that are not extracted from the previous iteration because every previously-mined rule contains the top attribute (root node) of the previous decision tree. This process is repeated until no new rules can be extracted. Figure 2 describes this algorithm. In that, we control the number of examples in a click segment by *m* (i.e., the minimum number of examples in a leaf node). A value of *m* that is too small is less effective because it causes the learning process to take a long time and results in rules with a small number of clicks. A small number of examples for each rule (segment) would be insufficient to derive a probabilistic model (Section 4.2). Besides, too large an *m* is inappropriate since only a few attributes are selected and the number of generated rules would be insufficient.

In experiments, we used the C4.5 algorithm implemented in [18], and ran our iterative learning algorithm on the mining set of PLD (Section 3.2) by setting *m* as 100,200, …,1000. We obtained 269 rules (segments). Table 2 shows some examples of the rules. From that, we can identify how each segment is characterized by the query-click attributes, e.g., QueryTopic(Business) ∧ RLV(12)

Table 2: Segment examples. #SAT and #DSAT indicate the number of SAT and DSAT click examples, respectively.

Segment (Rule)	#SAT	#DSAT
PageTopic (Computers/Companies)	1,423 (46%)	1,659 (54%)
QueryTopic(Business) ∧ RLV(12)	916 (58%)	652 (42%)

means that the queries of the corresponding clicks are related to "business" topics and their clicked pages are difficult to understand (i.e., their readability level is 12).

4.2 Model Fitting

Now we describe how to fit a probability distribution to the SAT and DSAT examples of each segment. The Gamma distribution has been frequently used to model waiting time [24]. In this paper, we assume that the Gamma distribution can govern the amount of time that the user dwells at each clicked page, and the dwell times of the same-segment clicks would follow the same distribution. The distribution includes two parameters: a shape, *k* and scale, θ, and the probability density function (PDF) is given as:

$$\mathrm{p}(t|k,\theta) = \frac{t^{k-1}}{\Gamma(k)\theta^k}\exp\left(-\frac{t}{\theta}\right) \quad (1)$$

where $k > 0, \theta > 0$ and *t* is a click dwell time where $t\epsilon(0,\infty)$.

Given a click segment, we first collect SAT and DSAT examples that belong to the segment, and generate two lists of dwell times from the SAT and DSAT examples. For each dwell time list $\{t\}_1^n$, we estimate the parameter values to fit the distribution by Maximum Likelihood Estimation (MLE) as follows.

The log-likelihood for *n* i.i.d. dwell time samples are given as:

$$\ell(k,\theta) = (k-1)\sum_{i=1}^{n}\ln t_i - \frac{1}{\theta}\sum_{i=1}^{n} t_i - n\ln\Gamma(k) - nk\ln\theta \quad (2)$$

The maximum of θ is found by taking the derivative of Eq. (2).

$$\hat{\theta} = \frac{1}{kn}\cdot\sum_{i=1}^{n} t_i \quad (3)$$

Substituting Eq. (3) into Eq. (2) gives:

$$\ell = (k-1)\sum_{i=1}^{n}\ln t_i - kn - n\ln\Gamma(k) - nk\ln\left(\frac{\sum_{i=1}^{n} t_i}{kn}\right) \quad (4)$$

To find the maximum of *k*, we take the derivate of Eq. (4) as:

$$\ln k - \psi(k) = \ln\left(\frac{1}{n}\sum_{i=1}^{n} t_i\right) - \frac{1}{n}\sum_{i=1}^{n}\ln t_i \quad (5)$$

where ψ is the digamma function.

A numerical solution for Eq. (5) is introduced in [9] which gives the Newton-Raphson iteration as:

$$\frac{1}{k^{(t+1)}} \leftarrow \frac{1}{k^{(t)}} + \frac{\ln k^{(t)} - \psi\left(k^{(t)}\right) - s}{(k^{(t)})^2\left(1/k^{(t)} - \psi'(k^{(t)})\right)} \quad (6)$$

where $s = \ln\left(\frac{1}{n}\sum_{i=1}^{n} t_i\right) - \frac{1}{n}\sum_{i=1}^{n}(\ln t_i)$, ψ' denotes the trigamma function, and $k^{(t)}$ is the *t*-th update of *k*.

More details of the derivation are described in [37], and by the equations above we can estimate two parameter sets, i.e., $\langle \hat{k}_{\mathrm{SAT}}, \hat{\theta}_{\mathrm{SAT}} \rangle$ and $\langle \hat{k}_{\mathrm{DSAT}}, \hat{\theta}_{\mathrm{DSAT}} \rangle$. For each parameter set, its goodness is determined using the Kolmogorov-Smirnov test (K-S test) [27] which is typically used for evaluating the goodness of an estimation. The Kolmogorov-Smirnov statistic measures a probabilistic distance between the empirical Cumulative Distribution Function (eCDF) of a sample and the CDF of a reference distribution (i.e., one-sample K-S test). In our case, the reference distribution is the Gamma distribution, and the null hypothesis is that the sample (dwell times of click instances) is drawn from the Gamma distribution. We reject the null hypothesis at the 5% significance level ($p < 0.05$), and an estimated parameter set is rejected if its testing result is rejection. In which case, the rule (segment) is discarded. In experiments, we initially extracted 269 segments (rules), and accordingly the same number of SAT or DSAT parameter sets were obtained. After testing every parameter set, we removed a segment if its SAT or DSAT estimation is rejected. As a result, 56 segments are discarded, and 213 segments are retained.

4.3 Dwell Time Analysis

In this section, we provide an empirical analysis of SAT and DSAT dwell times using our dwell time distributions. For each attribute or segment, we extract corresponding instances from mining the PLD, estimate SAT and DSAT parameters, and plot their PDFs. From this, we can understand the difference between SAT and DSAT dwell times across attributes or segments. Note that for each estimation the K-S test is performed and the estimation is rejected if it could not pass the test.

We first analyze SAT and DSAT dwell times by different reading levels. Figure 3, 4, and 5 show the PDFs estimated by SAT and DSAT examples whose reading levels are 2 (easy), 6 (moderate), and 12 (difficult), respectively. In each figure, a click which belongs to the corresponding reading level is likely to be SAT if its dwell time is placed within the region where SAT probability is higher than DSAT probability; otherwise, the click is likely to be DSAT. Based on this interpretation, we empirically mark the dwell-time threshold where a click is likely to be SAT if its dwell time is longer than that (i.e., (1)). For the easy reading level, such threshold is found at around 160 seconds, the threshold of the moderate reading level is slightly increased (i.e., recognized at around 175 seconds), and the difficult reading level requires more than 200 seconds before a satisfied determination can be made. This suggests that the user requires different dwell times to examine clicked pages according to their reading difficulty. Moreover, in each figure, we can identify the dwell time when its DSAT probability is maximized (i.e., the highest DSAT probability). In the easy reading level, the shorter dwell time a click has, the more likely to be DSAT. However, in the medium reading level, about 30 seconds can maximize the DSAT probability whereas in the difficult reading level, such maximum point moves to over 50 seconds. That is, a longer dwell time is required for more complex pages even if their clicks are DSAT.

Next, we provide a similar analysis focused on different topics and rules other than reading level. Figure 6 plots the SAT and DSAT PDFs estimated by an ODP category of "Reference/Knowledge Management". Since this category generally contains "technical" topics (e.g., knowledge discovery, knowledge retrieval, etc.), the dwell time threshold (i.e., (1)) is quite increased, and the highest DSAT probability is obtained when the dwell time is about 70 seconds (similar to that of the difficult reading level).

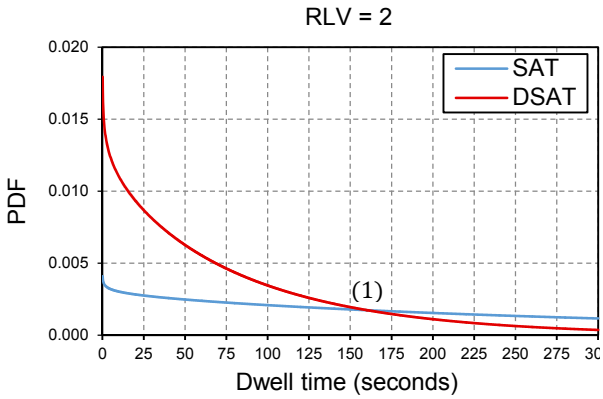

Figure 3: SAT and DSAT PDFs of RLV = 2 (easy).

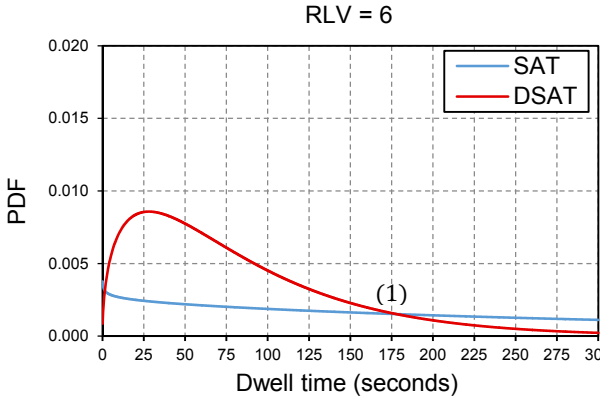

Figure 4: SAT and DSAT PDFs of RLV = 6 (moderate).

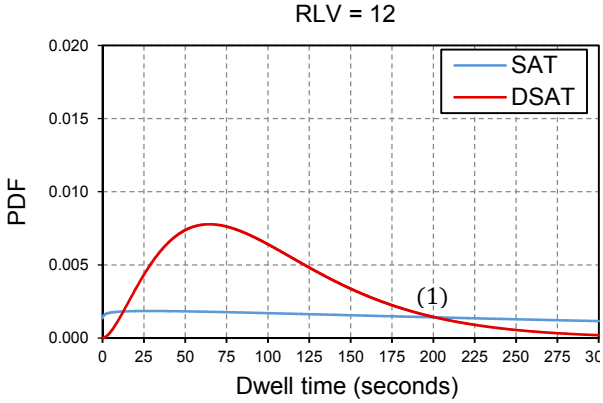

Figure 5: SAT and DSAT PDFs of RLV = 12 (difficult).

Regarding click segments, Figure 7 and 8 show the PDFs generated from two sample segments. The first segment contains the instances related to "Computer/Companies" (from Page Topic) and "Technical Help" (among query types), which form a "technical" segment. Again, the dwell time threshold is high (about 210 seconds) because the searches in this segment relate to technical topics and searchers may require longer to determine content value. Different from this, the clicks from the second segment (i.e.,) are related to less-technical topics (i.e., "shopping") and its dwell time threshold is much shorter than that of the first segment (i.e., technical topics).

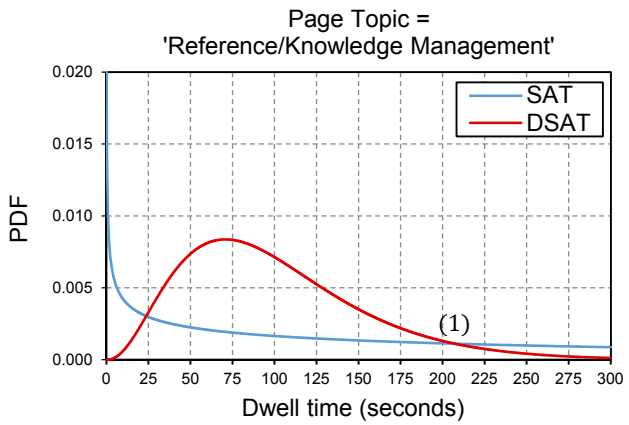

Figure 6: SAT and DSAT PDFs of a sample attribute.

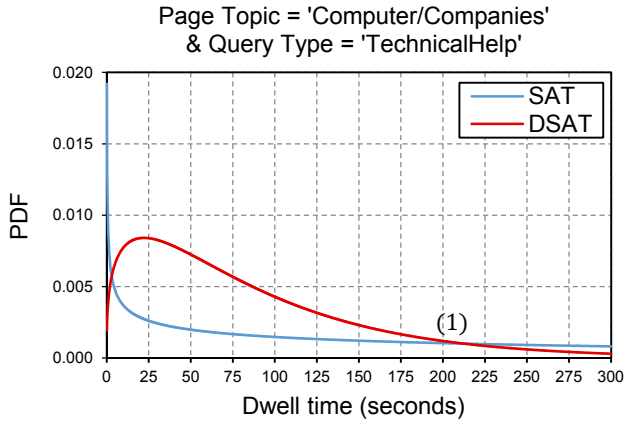

Figure 7: SAT and DSAT PDFs of a "technical" segment.

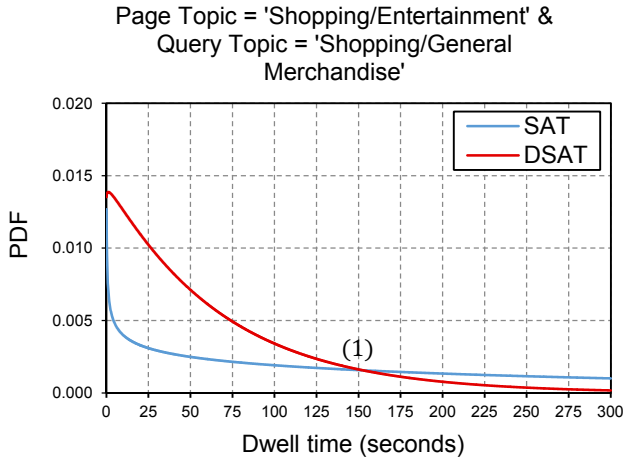

Figure 8: SAT and DSAT PDFs of a "shopping" segment.

It is clear from the findings presented in this section that there are differences in the SAT and DSAT dwell times associated with attributes of the clicked page, such as complexity and topic. This demonstrates that a single satisfaction threshold across all pages may be insufficient. In the next section we describe a click satisfaction model that leverages these dwell time distributions.

5. CLICK SATISFACTION MODEL

In this section, we describe a click-level satisfaction model exploiting our dwell time distributions. In Section 5.1, we describe how to estimate SAT and DSAT dwell time distributions for unseen click instances, and Section 5.2 provides a classification model to predict SAT clicks based on the estimated distributions.

Table 3: Features for estimating dwell time distribution [35].

Feature	Description
PageSize	Size (in bytes) of the clicked page
PageLength	# of words in the clicked page
RenderTime	Time to render the clicked page
HTML Tag	The frequency distribution of the 93 HTML tags (see http://www.quackit.com)
QueryTopic	Query topic (as described in Section 4.1)
PageTopic	Page topic (as described in Section 4.1)
Readability	Page readability level (as described in Section 4.1)

5.1 Estimating Dwell Time Distributions

Given a click instance, we would like to use the SAT and DSAT distributions described in Section 4 to determine whether the click is SAT. Although we generate many segments using a large-scale data set, we still have some unseen clicks that do not belong to any generated segments. Hence, some statistical method is required to estimate the SAT and DSAT dwell time distributions for these unseen clicks.

To estimate SAT and DSAT dwell time distributions for unseen instances, we develop a regression model to predict the parameters of each distribution. The formal definition of this model is given as follows. Suppose that $S = \{s_1, s_2, \dots, s_n\}$ is a set of n click segments and $\Pi = \{\pi_1, \pi_2, \dots, \pi_n\}$ is a list of estimated parameter sets from each segment. Each parameter set contains the parameters for SAT and DSAT distributions, i.e., $\pi_i = \{y_i^{\text{SAT}}, y_i^{\text{DSAT}}\}$ where π_i includes the corresponding parameters for s_i. In our case, each y involves a shape and scale values because we use the Gamma distribution for model fitting. In addition, for each segment, we have labeled instances used for estimating the parameters, which are expressed as $\{x_i^{\text{SAT}}, x_i^{\text{DSAT}}\}$ where x_i^{SAT} is the set of instances used for estimating y_i^{SAT}. Then, we train regression models using SAT and DSAT instances, i.e., $T_{\text{SAT}} = \{\langle \phi(x_i^{\text{SAT}}), y_i^{\text{SAT}} \rangle\}_{i=1}^{n}$ and $T_{\text{DSAT}} = \{\langle \phi(x_i^{\text{DSAT}}), y_i^{\text{DSAT}} \rangle\}_{i=1}^{n}$ where ϕ is a feature function and y_i is a target regression value for x_i. Note that in experiments, we generate four different regressors since each y in our case includes two parameter values (see Section 6.1). The regression function $f: \phi(x) \mapsto \Re$ maps a feature vector to a real value, and the model is trained to minimize a loss function defined by the difference between the mapped value and the target regression value (y) of every training example. To generate a feature vector for each instance, we leverage the features described in Table 3. These features have been shown to correlate with browsing time such as page length, the distribution of HTML tags, etc. [35]. In addition to these features, we use the query-click attributes (described in Section 4.1) and the segment rules (described in Section 4.2) as features because the clicks would follow the same distribution if they have some common attributes or belong to the same segment.

5.2 Predicting SAT Clicks

We now describe our method to predict click-level satisfaction. In that, we classify a click instance into either SAT or DSAT, i.e., a binary classification. As shown in Section 4.3, our dwell time model (i.e., SAT and DSAT dwell time distributions) can help to

Table 4: Dynamic dwell time features. M_{SAT} is the SAT Gamma distribution estimated by the regressor in Section 5.1, t is the dwell time of a click instance, k_{SAT} and θ_{SAT} are the shape and scale parameter values for M_{SAT}, respectively.

No.	Definition	Description
F1	$k_{\mathrm{SAT}} - k_{\mathrm{DSAT}}$	Difference between SAT and DSAT shape values
F2	$\theta_{\mathrm{SAT}} - \theta_{\mathrm{DSAT}}$	Difference between SAT and DSAT scale values
F3	$\mu_{\mathrm{SAT}} = k_{\mathrm{SAT}}\theta_{\mathrm{SAT}}$	Mean of SAT distribution
F4	$\mu_{\mathrm{DSAT}} = k_{\mathrm{DSAT}}\theta_{\mathrm{DSAT}}$	Mean of DSAT distribution
F5	$\mu_{\mathrm{SAT}} - \mu_{\mathrm{DSAT}}$	Diff. between μ_{SAT} and μ_{DSAT}
F6	$\lvert t - \mu_{\mathrm{SAT}} \rvert$	Time distance to μ_{SAT}
F7	$\lvert t - \mu_{\mathrm{DSAT}} \rvert$	Time distance to μ_{DSAT}
F8	$\mathrm{p}(t\lvert \mathrm{M}_{\mathrm{SAT}})/\mathrm{p}(t\lvert \mathrm{M}_{\mathrm{DSAT}})$	Ratio of the SAT likelihood to the DSAT likelihood
F9	$\ln \mathrm{p}(t\lvert \mathrm{M}_{\mathrm{SAT}})$	Log-likelihood to SAT
F10	$\ln \mathrm{p}(t\lvert \mathrm{M}_{\mathrm{DSAT}})$	Log-likelihood to DSAT
F11	$\ln \mathrm{p}(t\lvert \mathrm{M}_{\mathrm{SAT}}) - \ln \mathrm{p}(t\lvert \mathrm{M}_{\mathrm{DSAT}})$	Difference between log-likelihoods of SAT and DSAT

dynamically determine dwell time thresholds according to attributes and segments, and provide better interpretations of the dwell time of a click. As an example, given a dwell time, the likelihoods to SAT and DSAT distributions can be calculated. Therefore, our click satisfaction model can include more elaborate dwell time features based on the SAT and DSAT dwell time distributions estimated by the regression models described in Section 5.1.

Table 4 shows our dynamic dwell time features to facilitate the click classification. F1, F2, F3, F4, and F5 are calculated only using the distribution parameters estimated by the regression models, and the dwell time of each instance (i.e., *t*) is not necessary. On the other hand, F6, F7, F8, F9, F10, and F11 require *t* for calculating each feature value, but those features may provide better understanding of *t* for classification algorithms. In other words, F6, F7, F8, F9, F10, and F11 would be useful in that they indicate how much each instance is likely to be a SAT or DSAT click. Additionally, we use the segment rules mined by decision trees (Section 4.1) as binary features because those rules contain discriminative attributes to identify SAT or DSAT clicks.

Using the features described above, we train a classification model using labeled click instances, and the formal definition of this is given as follows. Given a set of *n* labeled examples, $\{x_1, x_2, \dots, x_n\}$, we generate a feature vector of each x_i, i.e., $\phi(x_i)$, and the label of each x_i is indicated by $L(x_i)$. A set of training examples is given as $\Phi = \{\langle \phi(x_i), L(x_i)\rangle\}_{i=1}^{n}$, and a classification function map a feature vector to a SAT or DSAT label, i.e., $f: \phi(x) \mapsto \{1,0\}$ where 1 and 0 denote SAT and DSAT, respectively. The model is learned by minimizing the disagreement between a mapped label and original label, $L(q_i)$ for every training example.

6. EXPERIMENTS

6.1 Estimating Dwell Time Distributions

In this section, we discuss the performance in estimating SAT and DSAT dwell time distributions, which is significant for the SAT click prediction. To identify an appropriate regression method, we test various algorithms for our SAT and DSAT distribution estimation (described in Section 5.1). We employ Poisson regression [7], Stochastic Gradient Descent (SGD) [5], and Multiple Additive Regression Trees (MART) [15]. Since a Gamma distribution includes two parameters (i.e., k (shape) and θ (scale)), we generate four regressors for each parameter of $\{k_{\mathrm{SAT}}, \theta_{\mathrm{SAT}}, k_{\mathrm{DSAT}}, \theta_{\mathrm{DSAT}}\}$. We measure the Root Mean Square (RMS) error for each method. For evaluation, we can use the mining set from PLD (Section 3.2), which contains 91,000 click examples (45,500 instances are SAT and the others are DSAT) because this data set is used for deriving dwell time distributions (Section 4.2) and accordingly corresponding regression values for each example are already calculated. Note that in this experiment the SAT or DSAT label of each instance is not used since we only test our regression model. We run each algorithm 10 times and for each run, 10-fold cross-validation is performed with random partitioning.

Table 5: RMS error by regression method. A significant improvement is marked using the first letter of each method. The Wilcoxon rank sum test is performed with $p < 0.01$.

Method	k_{SAT}	θ_{SAT}	k_{DSAT}	θ_{DSAT}
Poisson [7]	0.224	813.968	0.166	111.205
SGD [5]	0.159 [P]	56.876 [P]	0.083 [P]	10.740 [P]
MART [15]	0.014 [PS]	8.905 [PS]	0.012 [PS]	0.762 [PS]

Table 5 shows the regression results using the three different algorithms. We report an average value of the RMS error over every testing example. Note that the evaluation metric denotes a numerical error where a smaller value indicates a better performance. It is quite clear that MART can significantly outperform the others in every regression. In the next series of experiments, we use MART as a learning algorithm of the regression model for dwell time distributions.

6.2 SAT Click Classification

We now describe the experiments that we performed to predict whether observed clicks were satisfied. We begin by describing the experimental setup (including baseline features), and then present the classification results comparing the different methods.

6.2.1 Experimental Setup

We conduct experiments to evaluate our SAT click classification model. For this evaluation, we use HLD (described Section 3.1) that contains 3,204 human-labeled click instances (where the half of them are SAT and the others are DSAT). To each instance, we apply the MART regressors (trained using the mining set of PLD; see Section 6.1) for estimating its dwell time distributions. To learn the classification model, we use MART boosted decision trees [15]. We also conducted the experiment with another classification algorithm, i.e., Linear Support Vector Machine (SVM) [8] using baseline features (described below), and MART performed significantly better than SVM (we omit the results due to space limitation). The classifier was run 10 times, and for each run, 10-fold cross-validation is performed using random partitioning.

Baselines: For a baseline, the dwell time of each click can be used. As analyzed in [14], dwell time (the time spent on a clicked result) is significantly correlated with click satisfaction, and this feature is used as Baseline 1. In addition, we use the attributes described in Section 4.1.1 and the features from previous work on predicting search performance [12][23]. Since dwell time can be influenced by reading difficulty, topics, and query types, these attributes may help to identify the satisfaction of each click. Besides, search performance predictors (e.g., [12][23]) are effective in estimating search quality which would be related to click satisfaction. We choose to use the query clarity score [12], inverse collection term frequency [23], and query term length, which can be easily implemented and effectively work as much as other predictors (e.g., [45][46]).

Table 6: Classification Results on Human-Labeled Data. Baseline 1 contains only the dwell time of each click instance, Baseline 2 includes the search performance predictors [12][23] and the click attributes described in Section 4.1, and Baseline 3 is the combination of Baseline 1 and 2. D and R denote dynamic dwell time features (see Section 5.2) and click segment rules (Section 4.1), respectively. In each column, a significant improvement over each baseline is marked by its number, e.g., B12 indicates an improvement over Baseline 1&2, and a * indicates a significant improvement of each Baseline + D + R over the Baseline + D (i.e., the effectiveness of R for each Baseline + D). The Wilcoxon rank sum test is performed with $p < 0.01$, and the best result in each column is marked in bold.

Method	SAT Precision	SAT Recall	SAT F1	DSAT Precision	DSAT Recall	DSAT F1	Accuracy
Baseline 1	0.5758	0.5180	0.5453	0.5620	0.6184	0.5888	0.5682
Baseline 1 + D	0.7103 ^{B1}	0.6736 ^{B1}	0.6915 ^{B1}	0.6897 ^{B1}	0.7253 ^{B1}	0.7071 ^{B1}	0.6995 ^{B1}
vs. Baseline 1	*+23.36%*	*+30.04%*	*+26.81%*	*+22.72%*	*+17.29%*	*+20.08%*	*+23.10%*
Baseline 1 + D + R	0.7278 ^{B1*}	0.6967 ^{B1*}	0.7119 ^{B1*}	0.7092 ^{B1*}	0.7395 ^{B1*}	0.7240 ^{B1*}	0.7181 ^{B1*}
vs. Baseline 1	*+26.40%*	*+34.50%*	*+30.55%*	*+26.18%*	*+19.57%*	*+22.95%*	*+26.38%*
Baseline 2	0.7564 ^{B1}	0.6979 ^{B1}	0.7260 ^{B1}	0.7196 ^{B1}	0.7753 ^{B1}	0.7464 ^{B1}	0.7366 ^{B1}
Baseline 2 + D	**0.8178** ^{B123}	**0.7951** ^{B123}	**0.8062** ^{B123}	**0.8006** ^{B123}	**0.8228** ^{B123}	**0.8115** ^{B123}	**0.8089** ^{B123}
vs. Baseline 2	*+8.11%*	*+13.93%*	*+11.06%*	*+11.26%*	*+6.13%*	*+8.73%*	*+9.82%*
Baseline 2 + D + R	0.8168 ^{B123}	0.7901 ^{B123}	0.8032 ^{B123}	0.7968 ^{B123}	0.8227 ^{B123}	0.8095 ^{B123}	0.8064 ^{B123}
vs. Baseline 2	*+7.98%*	*+13.22%*	*+10.64%*	*+10.73%*	*+6.12%*	*+8.46%*	*+9.48%*
Baseline 3	0.7570 ^{B1}	0.7116 ^{B1}	0.7336 ^{B1}	0.7279 ^{B1}	0.7715 ^{B1}	0.7491 ^{B1}	0.7416 ^{B1}
Baseline 3 + D	0.8149 ^{B123}	0.7933 ^{B123}	0.8039 ^{B123}	0.7986 ^{B123}	0.8199 ^{B123}	0.8091 ^{B123}	0.8066 ^{B123}
vs. Baseline 3	*+7.66%*	*+11.47%*	*+9.59%*	*+9.71%*	*+6.26%*	*+8.01%*	*+8.76%*
Baseline 3 + D + R	0.8174 ^{B123}	0.7933 ^{B123}	0.8051 ^{B123}	0.7992 ^{B123}	0.8227 ^{B123}	0.8108 ^{B123}	0.8080 ^{B123}
vs. Baseline 3	*+7.98%*	*+11.47%*	*+9.75%*	*+9.79%*	*+6.63%*	*+8.23%*	*+8.96%*

To estimate the inverse collection term frequency, we use term probabilities obtained from the Web N-Gram services [40]. We calculate the sum, standard deviation, ratio of the maximum to the minimum, maximum, arithmetic mean, and geometric mean among the term probabilities of all query terms. Therefore, Baseline 2 contains 600 binary attributes (see Section 4.1.1) and eight search performance predictors. To develop more robust baseline, we combine the features from Baselines 1 and 2 to form Baseline 3. Note that the features for estimating dwell time distributions (i.e., Table 3&4) are not used in any baselines.

6.2.2 Classification Results

We now evaluate our SAT click prediction model that contains dynamic dwell time features (proposed in Section 5.2). With each baseline, we incorporate the dynamic dwell time features, and the combined model would perform better if the proposed features are effective to identify SAT and DSAT clicks. Besides, we additionally test with the segment rules (Section 4.1) as binary features for expecting further improvements. We measure precision, recall, and F1 for each class, and accuracy (i.e., weighted precision) is measured to verify overall classification performance.

Table 6 shows the classification results on HLD. First, the features in Baseline 2 seem to be more effective than Baseline 1. Baseline 2 and 3 including the search performance predictors and query-click attributes can significantly outperform the Baseline 1 containing only dwell time. However, no significant difference is observed between the results of Baseline 2 and 3. Second, the proposed features can significantly improve every baseline. This means that our dynamic dwell time features are quite effective for predicting SAT and DSAT click examples. In particular, only dwell time (Baseline 1) appears somewhat less important (i.e., its overall accuracy is only 0.5682), but combining it with the dynamic dwell time features, the performance is dramatically improved, i.e., +23.10% in accuracy. Third, we can only observe the further improvement achieved by segment rules when combining with Baseline 1 (dwell time only), marked by an asterisk in Table 6. This is because Baselines 2 and 3 already contain the attributes used for generating the segment rules (i.e., a segment rule is an association of discriminant attributes).

7. CONCLUSIONS

Current methods to estimate search satisfaction use a single dwell time threshold applied to all result clicks. In this paper, we proposed a method to model dwell time distributions of SAT and DSAT clicks related to specific query-click attributes, i.e., reading difficulty of clicked pages, search topics, and query types. Our dwell time model can provide better interpretations of a click dwell time, e.g., the likelihood of the click to be associated with SAT or DSAT. In addition, we identified changes in SAT or DSAT dwell times according to the attributes, e.g., more dwell time is required for pages which have sophisticated content (high reading level) although its click is DSAT. To the best of our knowledge, this is the first study to identify and characterize the effect of query-click attributes on click dwell time. Based on the dwell time model, we also proposed regression models that estimate SAT and DSAT dwell time distributions for new clicks. Using these regressors, we could generate dynamic dwell time features for predicting SAT clicks, which resulted in significant improvements in prediction performance. For future work, we will explore other techniques to determine dwell time (e.g., client side monitoring rather than server side as reported here), study other factors that may affect click dwell time (e.g., pages revisited by the same user may lead to lower dwell times given increased familiarity with the page), and analyze the impact of these variations on the performance of our click-level satisfaction predictor. We will also explore the use of the satisfaction predictions as more reliable implicit feedback in applications such as search result ranking.

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. *Proc. SIGIR*, 345-354.

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

[3] Agichtein, E., Brill, E., Dumais, S.T., and Ragno, R. (2006). Learning user interaction models for predicting web search result preferences. *Proc. SIGIR*, 3-10.

[4] Bennett, P.N., Svore, K., and Dumais, S.T. (2010). Classification-enhanced ranking. *Proc. WWW*, 111-120.

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

[6] Buscher, G., van Elst, L., and Dengel, A. (2009). Segment-level display time as implicit feedback: a comparison to eye tracking. *Proc. SIGIR*, 67-74.

[7] Cameron, A.C. and Trivedi, P.K. (1998). *Regression Analysis of Count Data*. Cambridge University Press.

[8] Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(27): 1-27.

[9] Choi, S.C. and Whette, R. (1969). Maximum likelihood estimation of the parameters of the gamma distribution and their bias. *Technometrics*, 11(4): 683-690.

[10] Claypool, M., Le, P., Wased, M., and Brown, D. (2001). Implicit interest indicators. *Proc. IUI,* 33-40.

[11] Collins-Thompson, K. and Callan, J. (2004). A language modeling approach to predicting reading difficulty. *Proc. HLT*, 193-200.

[12] Cronen-Townsend, S., Zhou, Y., and Croft, W.B. (2002). Predicting query performance. *Proc. SIGIR*, 299-306.

[13] Feild, H., Allan, J., and Jones, R. (2010). Predicting searcher frustration. *Proc. SIGIR*, 34-41.

[14] Fox, S., Karnawat, K., Mydland, M., Dumais, S., and White, T. (2005). Evaluating implicit measures to improve web search. *ACM TOIS*, 23(2): 147-168.

[15] Friedman, J.H. (2001). Greedy function approximation: a gradient boosting machine. *The Annals of Statistics*, 29(5): 1189-1232.

[16] Guo, Q. and Agichtein, E. (2012). Beyond dwell time: estimating document relevance from cursor movements and other post-click searcher behavior. *Proc. WWW,* 569-578.

[17] Guo, Q., White, R.W., Dumais, S.T., Wang, J., and Anderson, B. (2010). Predicting query performance using query, result, and user interaction features. *Proc. RIAO*, 198-201.

[18] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I.H. (2009). The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1): 10-18.

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

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

[21] Hassan, A., Shi, X., Craswell, N., and Ramsey, B. (2013). Beyond clicks: query reformulation as a predictor of search satisfaction. *Proc. CIKM*, 2019-2028.

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

[23] He, B. and Ounis, I. (2006). Query performance prediction. *Information System*, 31(7): 585-594.

[24] Hogg, R. V., McKean, J., and Craig, A.T. (2012). *Introduction to Mathematical Statistics*. Pearson, 7th Ed.

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

[26] T. Joachims. (2002). Optimizing search engines using clickthrough data. *Proc. SIGKDD*, 132-142.

[27] Justel, A., Peña, D., and Zamar, R. (1997). A multivariate Kolmogorov-Smirnov test of goodness of fit. *Statistics & Probability Letters*, 35(3): 251-259.

[28] Kang, I.-H. and Kim, G. (2003). Query type classification for web document retrieval. *Proc. SIGIR*, 64-71.

[29] Kelly, D. and Teevan, J. (2003). Implicit feedback for inferring user preference: a bibliography. *SIGIR Forum*: 37(2).

[30] Kidwell, P., Lebanon, G., and Collins-Thompson, K. (2009). Statistical estimation of word acquisition with application to readability prediction. *Proc. EMNLP*, 900-909.

[31] Kelly, D. and Belkin, N.J. (2001). Reading time, scrolling, and interaction: exploring implicit sources of user preferences for relevance feedback. *Proc. SIGIR*, 408-409

[32] Kelly, D. and Belkin, N.J. (2004). Display time as implicit feedback: understanding task effects. *Proc. SIGIR*, 377-384.

[33] Kim, Y., Hassan, A., White, R.W., and Wang, Y.-M. (2013). Playing by the rules: mining query associations to predict search performance. *Proc. WSDM*, 133-142.

[34] Leskovec, J., Dumais, S., and Horvitz, E. (2007). Web projections: learning from contextual sub graphs of the web. *Proc. WWW*, 471-480.

[35] Liu C., White, R.W., and Dumais, S. (2010). Understanding web browsing behaviors through Weibull analysis of dwell time. *Proc. SIGIR*. 379-386.

[36] Lloyd, S.P. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2): 129-137.

[37] Minka, T.P. (2002). *Estimating a Gamma Distribution*. Microsoft Research Technical Report.

[38] Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers.

[39] Radlinski, F., Kurup, M. and Joachims, T. (2008). How does clickthrough data reflect retrieval quality? *Proc. CIKM*, 43-52.

[40] Wang, K., Thrasher, C., Viegas, E., Li, X. and Hsu, P. (2010). An overview of Microsoft web n-gram corpus and applications. *NAACL HLT Demo Session*, 45-48.

[41] White, R.W. and Huang, J. (2010). Assessing the scenic route: measuring the value of search trails in web logs. Proc. *Proc. SIGIR*, 587-594.

[42] White, R. W. and Kelly, D. (2006). A study on the effects of personalization and task information on implicit feedback performance. *Proc. CIKM*. 297-306.

[43] Xue, G.-R., Xing, D., Yang, Q., and Yu, Y. (2008). Deep classification in large-scale text hierarchies. *Proc. SIGIR*, 619-626.

[44] Yin, P., Luo, P., Lee, W.-C., and Wang, M. (2013). Silence is also evidence: interpreting dwell time for recommendation from psychological perspective. *Proc. KDD*, 989-997.

[45] Zhou, Y., and Croft, W.B. (2006). Ranking robustness: a novel framework to predict query performance. *Proc. CIKM*, 567-574.

[46] Zhou, Y. and Croft, W.B. (2007). Query performance prediction in web search environments. *Proc. SIGIR*, 543-550.