Context-Aware Web Search Abandonment Prediction

Yang Song[1], Xiaolin Shi[2], Ryen W. White[1], Ahmed Hassan[1]

[1]Microsoft Research,
[2]Microsoft Bing,
One Microsoft Way,
Redmond, WA 98052, USA
{yangsong, xishi, ryenw, hassanam}@microsoft.com

ABSTRACT

Web search queries without hyperlink clicks are often referred to as *abandoned queries*. Understanding the reasons for abandonment is crucial for search engines in evaluating their performance. Abandonment can be categorized as good or bad depending on whether user information needs are satisfied by result page content. Previous research has sought to understand abandonment rationales via user surveys, or has developed models to predict those rationales using behavioral patterns. However, these models ignore important contextual factors such as the relationship between the abandoned query and prior abandonment instances. We propose more advanced methods for modeling and predicting abandonment rationales using contextual information from user search sessions by analyzing search engine logs, and discover dependencies between abandoned queries and user behaviors. We leverage these dependency signals to build a sequential classifier using a structured learning framework designed to handle such signals. Our experimental results show that our approach is 22% more accurate than the state-of-the-art abandonment-rationale classifier. Going beyond prediction, we leverage the prediction results to significantly improve relevance using instances of predicted good and bad abandonment.

Keywords

Web search abandonment; Structured learning; User behavior analysis

1. INTRODUCTION

It is well-known that search engines leverage user feedback to improve result relevance [17]. Beyond explicitly asking human assessors to annotate the relevance of documents with respect to queries, an alternative, more scalable approach is to use implicit feedback from searchers to learn document preferences. User clicks are among the most effective signals to learn ranking functions [17], with the assumption that clicked documents are often more relevant than non-clicked ones. However, when presented with a search engine result page (SERP), users may elect not to click a result hyperlink. We refer to this scenario as *search abandonment*. User studies have analyzed the rationales for abandonment and the associated search behaviors [19, 22, 23, 11, 28]. These studies revealed two primary abandonment rationales: (1) bad abandonment indicating user frustration and dissatisfaction, and (2) good abandonment suggesting satisfaction without needing to click. Good abandonment is not uncommon in modern search engines, where direct answers such as weather and stock quotes are returned for queries with explicit intent. In addition, text from result snippets can also satisfy users' information needs directly [15, 21].

Accurately categorizing abandoned queries into good and bad abandonment can be critical for search engine relevance improvement, search success estimation [13, 14], as well as helping other decision-making tasks such as online experimentation (e.g., A/B testing). In this paper, we extend previous work in the area of abandonment classification/prediction [11], where the task is to predict the rationale for an observed abandonment instance. In particular, we explore the use of contextual information in search sessions to improve prediction accuracy. From user study data collected through an in-situ survey, we observed several key dependencies between abandoned queries within search sessions. Consequently, we elect to approach this problem using a structured learning framework, which can capture interactions between queries, as well as leveraging both query and session-level features for predicting abandonment rationales.

Specifically, this paper makes the following contributions:

1. We study user behavior associated with abandonment in a large-scale data set of logged search behavior. We examine both query-level and session-level behaviors, and the relationship between the abandoned queries and other activities in the full search session. We discover several key characteristics that distinguish bad abandonment from good abandonment, including query/session length and inter-query time. More importantly, we also discover that adjacent abandoned queries tend to share the same rationale.

2. Inspired by these characteristics, we model the abandonment prediction problem using a structured learning framework. We propose a set of joint-label/observation features to help fully capture dependencies between abandonment instances within the same search session. The new

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGIR'14, July 6–11, 2014, Gold Coast, Queensland, Australia.
Copyright 2014 ACM 978-1-4503-2257-7/14/07 ...$15.00.
http://dx.doi.org/10.1145/2600428.2609604.

framework more accurately predicts abandonment rationales than state-of-the-art classifiers [11].

3. We further leverage the prediction framework to improve search engine relevance. We propose a variant of click-through rate that consider abandonment rationale, and propose a new way to extract pair-wise click preference data for training a new ranker. Experiments show that these using enhancements as features can improve ranking performance.

The remainder of this paper is organized as follows. Section 2 presents related work on Web search abandonment; Section 3 introduces the user survey data used in this study; Section 4 details the user behavior analysis from search engine logs; Section 5 presents our structured learning framework for abandonment prediction; Section 6 shows the empirical study, and Section 7 concludes with future work.

2. RELATED WORK

Traditionally, search-result clicks have been treated as positive signals from which search engines can learn to estimate query-document relevance [17]. In the absence of clicks, it is difficult to explain whether SERP content was relevant and satisfied the searcher. In such cases, researchers often resort to editorial judging to gather feedback regarding abandonment rationales. Li et al. [19] explored the concept of good abandonment to distinguish between good and bad abandonment instances. In their definition, good abandonment describes a situation where a query was successfully addressed by the SERP without requiring clicks or query reformulations. They analyzed such scenarios on both desktop and mobile devices in three geographic locales, by randomly sampling a small number of queries from search engine logs, and applied their own judgment criteria to categorize each query into one of the two cases. They showed that good abandonment is more common and more likely on mobile devices. The authors also discovered that the type of abandonment varies significantly by locale and modality. Thuma et al. [24] showed that those who were accustomed to finding answers quickly tended to abandon their searches faster if they were dissatisfied. Chuklin and Serdyukov examined various aspects of abandonment, including its relationship with query extensions [9], and correlations between abandonment and editorial judgments [8]. Most relevant to our work, Chuklin and Serdyukov also developed models to predict good abandonment using topical, linguistic, and historic features [10]. However, they did not use session interactions or in-situ judgments from abandoning searchers, as we do here. This meant that they could only predict "potentially good" abandonment.

Rather than gathering third-party assessments of abandonment rationales, Stamou et al. [22] employed a small number of participants to complete an external survey for each query they performed on a single day. In their study, 13% of queries had no clicks, categorized as either *unintentional* such as no search results retrieved and search got interrupted, or *intentional* such as spelling check or examining the retrieved snippets to understand the query. In a follow-up study, the same authors recruited six participants, who each installed a browser plug-in to record search activity and elicited abandonment rationales via a survey [23]. They found that 27% of queries were abandoned, and rationales were split evenly between good and bad. In another small-scale study, Koumpouri et al. [18] studied SERP interactions and found that total dwell time/scrolling time, and copying caption text all correlated with satisfaction.

Diriye et al. [11] conducted a much larger user study of 928 searchers and analyzed their search behavior over a one-month period. They deployed a Web browser plug-in to each participant that displayed a survey whenever a user abandoned a query. The survey asked participants whether their information need was met by the abandoned SERP. Their definition of abandonment was similar to Li et al. [19], with some important differences such as using a timeout of 30 minutes as one of the abandonment criteria, instead of a 24-hour period from Li and colleagues. Diriye et al. found that 41% of abandonments were bad, 32% abandonments were good, with the remaining 27% associated with alternate reasons such as choosing a better query before considering the returned SERP. They developed a classifier to predict abandonment rationale by generating features from sessions, queries and documents and classify the data into four categories: *SAT*, *DSAT*, *Unintentional*, and *Other*.

As an alternative to running a user study, abandonment rationales can be estimated from logged behavioral data. Castillo et al. [5] sought to improve the accuracy of click-based metrics by identifying truly bad abandonments. The authors developed a method to automatically assess each searcher in terms of their tenacity, i.e., how likely they were to abandon a query without trying hard enough before they really failed. They then picked a set of tenacious users and measured the relative abandonment ratio between queries with direct answers and queries without. For the task of identifying bad abandonment, their method demonstrated over 80% precision for weather and reference queries. Chilton et al. [7] used searchers' repeat search behavior to understand SERP relevance sans interaction behavior such as clicks. The authors focused on direct answers such as flight status, stock, and weather. They showed that certain types of good answers can compensate for missing clicks and still satisfy users' information needs. Following this work, Bernstein et al. [3] combined log mining with paid crowdsourcing to aggregate direct answers for tail queries.

Huang et al. [15] proposed to leverage mouse cursor movements as an implicit signal to improve search relevance in the absence of user clicks. They reported a gaze-tracking study in which they demonstrated that cursor position is closely related to eye gaze on SERPs. To improve search quality, the authors gathered relevance labels for 1200 query-URL pairs and estimated the correlation between the labels and cursor features. When a query has no clicks, hover features show a fair correlation with human labels. The authors also leveraged cursor features such as movement speed and trail length to distinguish good and bad abandonment, and highlighted some interesting trends (e.g., when answers were present, users moved their mouse cursor more slowly).

Sakai et al. [21] labeled good abandonment as direct information access. They also defined immediate information access as situations where searchers can locate, for example, the result snippet quickly from the SERP. The authors proposed a new way of evaluating *direct and immediate information access* using methods similar to multi-document summarization or question-answering evaluations. They considered the rank position of the snippet matches during evaluation and showed that their measure can correctly reward systems that quickly return relevant snippets.

Our research extends previous work in a number of ways. First, we characterize key aspects of abandonment behavior within search sessions, especially the close relationship between adjacent abandonment instances. Second, we apply a structured learning framework, which is more sophisticated than the standard learning methods applied to this problem thus far. This allows us to capture key dependencies between abandonment instances hitherto ignored in abandonment modeling. Importantly, our analysis is performed using in-situ judgments from abandoning searchers, allowing us to more accurately model the abandonment process. Finally, we apply our predictions in a retrieval setting and show that we can significantly improve search relevance by directly modeling abandonment using our framework.

Previous work has thoroughly studied the problem of modeling search satisfaction at the session or task level [13, 14, 1]. Our work is related to this line of research because we use the entire session to provide context for every abandonment instance. However, our work also differs since we do not predict satisfaction at the session level; rather we do so for every abandoned query.

3. DATA DESCRIPTION

This section introduces the data set used in this study. We first describe the data collection process and then present some basic statistics of the data.

3.1 Data Collection Process

The data used were graciously provided by Diriye and colleagues, the authors of a prior investigation on search abandonment [11]. In that study the authors developed a Web browser plug-in that displayed a survey in a popup window to the searcher asking for an explanation whenever SERP abandonment was detected on the Bing search engine. Diriye et al. deployed the plug-in within Microsoft Corporation, where it was installed by employees who agreed to provide data on the URLs they visited and provide labels explaining the rationale for the abandonment when it occurred. The plug-in captured participant responses to questions in the popup as well as Web interaction data (including clicks and cursor movements using the methodology described in [4]) and SERP contents (including the top 10 results and presence/absence of direct answers). In the remainder of this section, we focus on two aspects of the data important to our current study: (1) how abandonment was defined (i.e., when the popup survey should be shown to searchers), and (2) what information was collected by the popup. We refer the reader to [11] for specific details, but provide an overview here for completeness.

Abandonment in the data that we received was defined as a situation when the SERP was displayed and the following two conditions were met: (1) no hyperlink clicks on results, advertisements, or direct answers, *and* (2) a trigger event occurs. For (2), in addition to defining what counts as abandonment, Diriye et al. also defined the point in time that the determination of no clicks (condition 1) should be made. The authors defined a number of abandonment *triggers* comprising one the following actions: manual re-query, browser tab closure, manual URL entry, click related search or spelling suggestion, change search scope, or timeout. After these two criteria were met, a popup survey was shown asking that respondents indicate their abandonment rationale. The survey captured four reasons: (1) *SAT*: Satisfied with the content of the SERP; (2) *DSAT*: Dissatisfied with the results presented; (3) *Interrupted or Unimportant*: The user was interrupted in their search task (leading to a timeout or other action), or the abandonment was unintentional (e.g., closed browser accidentally), or (4) *Other*: Any reason beyond what was captured in 1-3, with the rationale recorded directly in free text. For SAT, the popup also captured whether participant satisfaction was related to direct answers on the SERP, the captions of the search results, or other SERP content.

To prevent the popup from appearing too often, Diriye et al.: (1) employed a trigger control mechanism that suppressed the popup for 50% of all SERP abandonments on a per user basis, and (2) imposed a popup limit of 10 times per day per user. This has implications for the current study since although we record all instances of abandonment, we do not have the abandonment rationales for all of them.

3.2 Data Statistics

We now briefly describe the data set. The authors in [11] gathered a total of 7,419 labeled abandonment instances[1] from 928 participants. They also recorded participants' click and browse activities, totaling of 739,505 URLs and 39,606 queries. After filtering irrelevant queries (e.g., "test"), the resultant data set contained the following distribution of abandonment instances: 3,104 SATs, 2,524 DSATs, 501 Interrupted or Unimportant, and 1077 Other. On average, a user contributed 7.4 labels, while the most productive user labeled 174 queries. We sorted the queries by their frequencies and noted that (1) top-ranked good abandonment were mostly stock or weather related (e.g., "nok", "boston weather"), and (2) bad abandoned queries were more diverse, but in general longer than good abandoned queries (e.g., "percentage nfl games predicted winner percentage"). In the next section, we analyze user behavior surrounding abandonment at the query and session level.

4. USER BEHAVIOR ANALYSIS

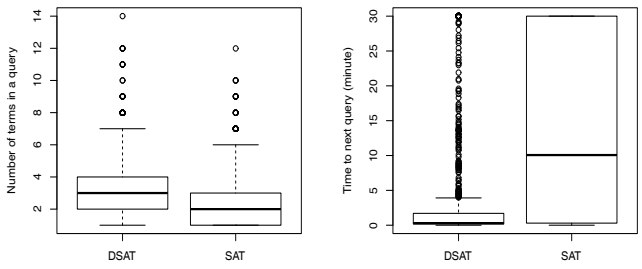
In this section, we examine the search abandonment behavior and investigate the behavioral patterns associated with abandonment in satisfaction or dissatisfaction. The main focus of this section is to understand the differences between good and bad abandonment in a qualitative way. Since abandonment is directly related to user satisfaction, we will be using good abandonment and *SAT*, bad abandonment and *DSAT*, interchangeably.

4.1 Query-level abandonment behavior

As we discussed in Section 3, the labels of our data are gathered at the query level, i.e. if a query does not have any hyperlink result click on its SERP (i.e., the query is abandoned), a popup may be shown and the searcher is asked to provide the abandonment rationale. We first investigate the differences between good and bad abandonment on the query level, beginning with some query characteristics.

The box plots in Figure 1(a) show the differences in query lengths (in terms) of good (SAT) abandoned queries and bad

[1]The authors in [11] reported using 1,799 labels collected over a 30-day period. However, they continued to collect data beyond the 30 days used in their original study. Our data set is substantially larger since it included abandonment instances collected during this additional time.



(a) (b)

Figure 1: Box plots of (a) query lengths (i.e. number of terms) and (b) time to next query of bad abandoned queries and good abandoned queries.

(DSAT) abandoned queries. Although both types of abandoned queries do not have hyperlink clicks, from Figure 1(a), we see that good abandoned queries are on average shorter than bad abandoned queries with statistical significance (mean query length: SAT=2.42, DSAT=3.01; $t(2996.0) = -9.65$; p-value $< 2.2e-16$), which is consistent with prior work [1]. Since it is well-known that query length is an effective indicator of query difficulty, Figure 1(a) suggests that DSAT abandonments are associated with more difficult queries than those abandoned with SAT.

We find that query length is not the only feature that differs significantly between good and bad abandonment. Figure 1(b) shows the difference of time to next query of these two types of abandonment. The difference there is even more pronounced. For bad abandoned queries, the median time to next query is 0.3 minutes; and for good abandoned queries, the median is 10.07 minutes ($t(2866.4) = 19.03$ with p-value $< 2.2e-16$). A short time interval (e.g., under 5 minutes) between two consecutive queries may imply dissatisfaction the first query and the subsequent reformulation of a similar search intent. Figure 1(b) illustrates that bad abandoned queries are more likely to be followed by reformulation than good abandoned queries, and examining if an abandoned query is followed by a reformulation is a very effective way in identifying user satisfaction (and this was also shown in the analysis of features in the classifier developed by [11]).

4.2 Session-level Abandonment Behavior

In addition to considering abandonment at the individual query level, we also seek to understand how abandoned queries correlate with other closely related queries and the abandonment behavior within search sessions. We use search sessions identified using a 30-minute inactivity timeout [11].

From our recorded search log, we collect all sessions that contain at least one labeled abandonment query, 2,483 such sessions in total. Among these sessions, 446 sessions are single-query sessions. 74.9% are abandonment queries labeled as SAT, and only 25.1% of them are labeled as DSAT.

Furthermore, for sessions with multiple queries, we find DSAT abandonment queries are more likely before the last query in a session (i.e., 84.8% of DSAT abandonments happen before the last queries of sessions versus 15.2% DSAT abandoned queries which are the last queries of sessions). Comparing with DSAT abandoned queries, SAT abandoned queries are less likely to occur before the last queries of sessions (60.3%) and more likely to be the last queries of sessions (39.7%). Both facts regarding abandoned queries in single-query sessions and multiple-query sessions are consistent with Figure 1(b), i.e. DSAT abandoned queries are more likely to be followed by reformulations and thus are more likely to co-exist with other queries and be followed by other queries in the same sessions.

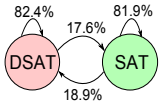


Figure 2: The transition probability between abandonment labels within the same session.

The findings of the analysis presented thus far consistently show that bad abandoned queries are more likely to be followed by reformulations, while good abandoned queries are more likely to be either the single query of a session or the last query of a session. Now we further examine the cases when there multiple labeled abandoned queries appear in the same search sessions. Specifically, we want to understand the likelihood of observing a session with both DSAT and SAT abandonment and understand the transitions between the two rationales in the same session.

The transition between labeled abandonment query pair is defined as follows: if two labeled abandoned queries q_1 and q_2 are in the same session and there is no other labeled abandoned query between them (non-labeled query is permitted), then the transition between this query pair is $T_{L_1L_2}$, where L_1 and L_2 are the SAT or DSAT labels of q_1 and q_2. From our data, we observe 268 cases of transition from DSAT to DSAT, 177 transitions from SAT to SAT, 57 transitions from DSAT to SAT, and 39 transitions from SAT to DSAT. The transition probabilities are illustrated in Figure 2. This means that good abandoned queries tend to co-exist in a session and same is true for bad abandoned queries. Conversely, it is much less likely for both SAT and DSAT abandoned queries to be observed in the same session.

We notice that usually when good abandoned queries co-occur in sessions, they are mostly related to simple facts. For example, a query "9 litres in quarts" (labeled as SAT) is followed by another query "9 litres in gallons", which is followed by a third query "how many quarts in a gallon?" (labeled as SAT), all in the same session. In contrast, DSAT abandoned queries of the same sessions are more likely to be reformulations of a single search intent. For example, in one session we observe "texas mom death brain" (no label), "texas mom brain injury" (no label), "texas mom head injury" (labeled as DSAT) and "texas mom football death" (labeled as DSAT), and the queries labeled as bad abandonment clearly have the same search intent.

Finally, we examine the *difficulty* of sessions that are abandoned due to satisfaction or dissatisfaction. For this purpose we use the value of the last abandonment label as a session label, and the number of queries in a session to estimate session difficulty, i.e., the amount of effort that a user expends.

Figure 3 shows the difference of sessions abandoned with SAT or DSAT in terms of the numbers of queries in these sessions. We observe that users tend to spend less effort in sessions that are abandoned with SAT than those abandoned with DSAT (means are SAT=2.26 and DSAT=2.93; $t(314.3) = -2.37$; p-value $= 0.018$). Even if we exclude single-query sessions (because we have already shown that

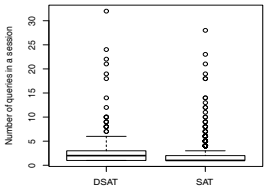


Figure 3: Box plots of query numbers in sessions abandoned with DSAT and SAT. A session is abandoned with DSAT if its last query is a DSAT abandoned query, and a session is abandoned with SAT if its last query is a SAT abandoned query.

single-query sessions are more likely to have SAT abandoned queries), SAT abandoned sessions still trend shorter than DSAT abandoned ones (mean number of queries SAT=3.84; DSAT=4.76; $t(172.0) = -1.84$; p-value $= 0.067$).

The results presented in this section provide some valuable insights about the abandonment process, in particular the observed dependencies between an abandonment instance and the session within which it occurs. Structured learning has been shown to be particularly useful at handling sequences where there are such dependencies [25, 2]. For this reason, we developed a structured learning framework for modeling abandonment. The findings of the analysis presented in this section helped inform the need for that framework and the design of some of the features it uses. Before proceeding, it is worth noting that in this analysis, we reported results using sessions, while we also performed similar analysis based on tasks [20]. The results from tasks were quite similar to sessions. Due to space, we will focus our analysis on sessions in this paper.

5. STRUCTURED LEARNING FOR ABANDONMENT PREDICTION

We can model our abandonment prediction task as structured learning, where each label of the abandoned query can be chosen from a finite set, while there exists strong dependencies among the labels within each session. We leverage the linear structural Support Vector Machines (SVM) algorithm, shown to outperform other structured output prediction algorithms such as conditional random fields (CRF), for sequential labeling [25]. In particular, we extend the structural SVM which is isomorphic to a linear chain Hidden Markov Model (HMM), by incorporating not only the label transition probability but also the features that can depend on any arbitrary pairs of labels. We first review the structural SVM algorithm and then discuss our extension.

5.1 Preliminary

In label sequence tagging, each input $\mathbf{x} = (\mathbf{x_1}, ..., \mathbf{x_m})$ is a sequence of feature vectors $\mathbf{x_i} \in \mathbb{R}^n$, associated with a sequence of labels $\mathbf{y} = (y_1, ..., y_m)$, where $y_i \in \{1, ..., k\}$ is chosen from a finite set of predefined categories, e.g., $\mathcal{Y} = \{N, NP, V...\}$ for natural language parsing. The goal of linear structured SVM is to learn a discriminant function F, parameterized by $\mathbf{w}$, where the features are the combined representation of both inputs and outputs $\Psi(\mathbf{x}, \mathbf{y})$,

$$F(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y}) \rangle . \quad (1)$$

To derive a maximum margin formulation, we first define the margin of a training example $\mathbf{x_i}$ as

$$\gamma_i = F(\mathbf{x_i}, y_i; \mathbf{w}) - \max_{y \in \mathcal{Y} \neq y_i} F(\mathbf{x_i}, y; \mathbf{w}), \quad (2)$$

so that the objective becomes to select $\mathbf{w}$ with $||\mathbf{w}|| \leq 1$ that maximizes $\min_i \gamma_i$. By allowing errors in the training set, a set of slack variables ξ is used along with the dual to optimize a soft-margin problem:

$$\min_{\mathbf{w}, \xi} \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{n}\sum_i^n \xi_i,$$
$$s.t. \forall i, \xi_i \geq 0,$$
$$\forall i, \forall y \in \mathcal{Y} \neq y_i : \gamma_i \geq 1 - \xi_i, \quad (3)$$

where γ_i is the margin defined in eq. (2), and $C > 0$ is the parameter that balances the trade-off between two terms.

The combined feature representation $\Psi(\mathbf{x}, \mathbf{y})$ for sequence tagging contains both the emission features and the histogram of label transition. The emission feature is defined as, for each $\mathbf{x_i}$ and y_i

$$\Psi_{emission}(\mathbf{x_i}, y_i) = (0, ...0, \mathbf{x_i}, 0, ...0)' \quad (4)$$

where the feature vector $\mathbf{x_i}$ is placed in the j's position for $y_i = j$. The combined feature then becomes

$$\Psi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} \begin{pmatrix} \Psi_{emission}(\mathbf{x_i}, y_i) \\ [y_i = 1][y_{i-1} = 1] \\ [y_i = 1][y_{i-1} = 2] \\ . \\ . \\ [y_i = k][y_{i-1} = k] \end{pmatrix}, \quad (5)$$

where $[P]$ is the indicator function for the predicate P. The biggest challenge in optimizing this maximum margin framework is the large number of linear constraints, which is $n|\mathcal{Y}| - n$ for n training examples with the label space $|\mathcal{Y}|$. Consequently, standard quadratic programming solvers are unable to handle such a large search space. Efficient learning algorithms have been developed to address this issue, such as cutting-plane methods [25] and dual coordinate descent (DCD) methods [26]. So far, DCD is the fastest implementation so we decided to use this for our problem, which has more advanced features and thus even larger feature space.

5.2 Extending Structured SVM

A limitation of the above framework is the simplified feature representation for label dependencies. As can be seen from eq. (5), the dependency feature is only available in the format of label transition histograms, e.g., $[y_i = j][y_{i-1} = k]$. This may not be an issue when the label space $|\mathcal{Y}| = k$ is large and the length of the training sequence $|\mathbf{x}| = m$ is sufficient to capture the strong dependencies between different labels. Nevertheless, in our scenario of abandonment rationale prediction, the label space is quite small: either 0 or 1, and the length of training sequence is often not as adequate as in other applications such as NLP tagging.

Therefore, we propose to extend the previous structured SVM framework by considering more label dependency features, which include the feature vector $\mathbf{x}$ along with two adjacent labels y_i and y_{i-1}. These so-called joint-label/observation features have a general form of

$$f_s(\mathbf{x}, y_i, y_{i-1}) = [y_i = j][y_{i-1} = k]\psi_s(\mathbf{x}, i-1, i)$$
$$j, k \in \mathcal{Y}, \psi_s(\cdot) \to \mathbb{R}, s \in S \quad (6)$$

where $\psi_s(\cdot)$ maps a certain sequence between $i-1$ and i of $\mathbf{x}$ into a value, and S denotes the set of all such functions.

We use a specific example to illustrate eq. (6) in the abandonment scenario: let us denote $\psi_s(\mathbf{x}, i-1, i)$ the total query dwell time between $i-1$ and i, while $[y_i = 0]$ and $[y_{i-1} = 0]$ means whether both queries are bad abandonments. In this setting, we implicitly assume that if a user abandoned a query with dissatisfaction, there is a high chance that the user would abandon another one in a short period of time, which is also a bad abandonment. Another example of such a function could be the query similarities of all queries between $i-1$ and i.

We want to clarify the notation in particular the subscripts $i-1$ and i here. Unlike the traditional sequence labeling tasks where y_i and y_{i-1} are always assumed to be adjacent, in our scenario, there could exist several queries between y_i and y_{i-1}. Recall that labels are assigned to abandoned queries only. For those queries with clicks in a user search session, no labels are assigned to them. An example is illustrated in Figure 4, which shows five queries in a user session with two abandoned queries $\mathbf{x_j}$ and $\mathbf{x_l}$, where the two adjacent training labels y_j and y_l are separated by three queries. The joint-label/observation function of these two labels are therefore defined between the three feature vectors $\mathbf{x_j}$, $\mathbf{x_k}$ and $\mathbf{x_l}$.

Table 1 lists all these features. Since many feature values are not nominal, e.g., dwell time of queries in seconds, it is often suggested to discretize them into several nominal bins, each of which takes value from either 0 or 1. Thus, for each of these features, we choose to separate them into 4 to 5 bins according to the distribution of their histograms.

With the enhanced features, we can re-write the combined feature representation as

$$\Psi(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} \begin{pmatrix} \Psi_{emission}(\mathbf{x_i}, y_i) \\ [y_i = 1][y_{i-1} = 1] \\ [y_i = 1][y_{i-1} = 2] \\ . \\ X. \\ [y_i = k][y_{i-1} = k] \\ f_1(\mathbf{x}, y_i, y_{i-1}) \\ . \\ f_S(\mathbf{x}, y_i, y_{i-1}) \end{pmatrix}. \quad (7)$$

Therefore, the inner product between two sequences can be calculated as

$$\begin{aligned} \left\langle \Psi(\mathbf{x}, \mathbf{y}), \Psi(\mathbf{x}', \mathbf{y}') \right\rangle &= \sum_{i,ii} [y_i = y'_i i][y_{i-1} = y'_{ii-1}] \\ + \sum_{i,ii} [y_i = y'_{ii}] \left\langle \Psi_{emission}(\mathbf{x_i}, y_i), \Psi_{emission}(\mathbf{x'_{ii}}, y'_{ii}) \right\rangle \\ + \sum_{i,ii,s} [y_i = y'_{ii}] \left\langle f_s(\mathbf{x_i}, y_i, y_{i-1})), f_s(\mathbf{x'_{ii}}, y_{ii}, y_{ii-1}) \right\rangle \quad (8) \end{aligned}$$

At training stage, we need to solve the inference problem to predict the optimal structure $\hat{\mathbf{y}}$ for an input $\mathbf{x}$ with ground-truth label $\mathbf{y}$:

$$\hat{\mathbf{y}} \leftarrow \arg \max_{\hat{\mathbf{y}} \in \mathcal{Y}(\mathbf{x})} \left(\Delta(\mathbf{y}, \hat{\mathbf{y}}) + \mathbf{w}^T (\Psi(\mathbf{x}, \mathbf{y}) - \Psi(\mathbf{x}, \hat{\mathbf{y}})) \right) \quad (9)$$

where $\Delta(\mathbf{y}, \hat{\mathbf{y}})$ defines the loss of the two sequences, e.g., zero-one loss for classification or inverse loss for natural language parsing [25], and $\Psi(\mathbf{x}, \mathbf{y})$ is the enhanced feature representation in eq. (7). Given the large number of linear constraints, we employee a Viterbi decoding method to efficiently find the optimal sequence. To solve the actual training problem as shown in eq. (3), we use a dual coordinate descent (DCD) algorithm which considers the square hinge loss function. For details, readers are referred to [26, 6].

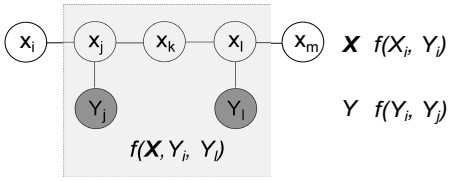


Figure 4: Illustration of our extended structured model. In traditional models, each instance x corresponds to a label y, while in our model, only abandoned instances have labels. Traditional models often have two types of features, $f(\mathbf{x_i}, y_i)$ the emission features and $f(y_i, y_j)$ the label transition features. Our model extends it by also considering joint-label/observation features $f(\mathbf{x}, y_i, y_j)$.

5.3 Feature Set

We now discuss the feature set used for prediction. In the previous section, we presented the structural features used for SVM in Table 1, which correspond to those $f(\mathbf{x}, y_i, y_j)$ features in Figure 4. We have also shown the transition features $f(y_i, y_j)$ as the histogram count in eq. (7). This leaves only the set of emission features, or $f(\mathbf{x}, y_i)$, to be defined. In our study, we divide those features into three main categories, (i) query, (ii) SERP, and (iii) session features.

Query Features: We extract features for each query and its interaction with the immediate preceding and succeeding queries in the session. These features include the query length in terms of characters and tokens, whether the query is the first/last/only query in the session, whether the query is a URL-type query, and whether the current query is a reformulation of the previous query, or the next query is a reformulation of the current one. To determine this, we calculate the Levenshtein distance between two queries after removing stop words. The similarity of two queries are calculated by dividing the Levenshtein distance by the maximum query length. We use a threshold of 0.5 to determine whether a reformulation exists.

SERP Features: We also include the SERP features for the current query and for the immediately preceding and succeeding queries. Note that the current target query has no clicks, however its neighborhood queries may contain clicks, as shown in Figure 4. The SERP features reflect these situations by considering the previous/next query's click count and click position. We also consider the presence of answers on the SERP by adding two answer features: the total number of answers shown and whether the SERP contains one or more *good* answers in the top region. We manually define a white list of good answers which have high probability of satisfying a user's intent without the need for clicks. These good answers include weather, dictionary definitions, and currency conversion.

Session Features: Following the definition used in [11], a search session starts with a query, and terminates with a 30-minute inactivity timeout. Session features contain the overall summary of a search session, including its entry point

$f_s(\mathbf{x}, y_i, y_j)$	Explanation	Discretization Bins
$NQ = \|i - j\|$	Total number of queries between i and j	$0, (0, 2], (2, 4], > 4$
$DT = \sum_{k=i}^{j} DT(\mathbf{x_k})$	Total dwell time of all queries between i and j	$\leq 10, (10, 30], (30, 60], > 60$
$Ans = \sum_{k=i}^{j} Ans(\mathbf{x_k})$	Total answers shown on SERP between i and j	$0, (0, 3], (3, 5], (5, 10], > 10$
$Click = \sum_{k=i}^{j} Click(\mathbf{x_k})$	Total clicked URLs between i and j	$0, (0, 3], (3, 5], (5, 10], > 10$
$SameQ = [Q(\mathbf{x_i}) = Q(\mathbf{x_j})]$	Whether the two abandoned queries are identical	/
$ReformQ = [reform(Q(\mathbf{x_i}), Q(\mathbf{x_j}))]$	Whether one query is a reformulation of the other	/
$Max\text{-}Sim = \max_{k,t \in [i,j]} Sim(Q(\mathbf{x_k}), Q(\mathbf{x_t}))$	Max query similarity of any query pair in $[i, j]$	0, (0, 0.25], (0.25, 0.5], (0.5, 1]
$Min\text{-}Sim = \min_{k,t \in [i,j]} Sim(Q(\mathbf{x_k}), Q(\mathbf{x_t}))$	Min query similarity of any query pair in $[i, j]$	0, (0, 0.25], (0.25, 0.5], (0.5, 1]

Table 1: List of joint-label/observation features used in our framework. Numeric features are discretized into several nominal bins for ease of learning and inference.

(e.g., search homepage or browser search bar), browser type (e.g., Firefox, Chrome), the total session length in terms of queries or total query dwell time , the total number of abandoned queries, and so on. Table 2 lists all features.

Note that in the work of Diriye et al. [11], the authors included many historical features such as overall query frequency, which relies on the access of a long-period of search logs and thus not immediately available from the current user session. In practice, calculating and maintaining these features are quite difficult and time-consuming, and thus not desirable as features that are easy to extract. Besides, considering potential applications of this research, e.g., online abandonment prediction for unseen (new) queries, or as a metric for online A/B experimentation, we have excluded these features from our framework. Later we show that even without these features, our framework still significantly outperforms the previous model [11].

6. EMPIRICAL ANALYSIS

This section discusses experimental results on both abandonment prediction and its application to improve the search quality of ranking functions.

6.1 Data Preparation and Evaluation Metrics

From the total 7,419 instances, we selected 5,628 instances that belong to either SAT (good abandonment) or DSAT (bad abandonment) for our prediction task, excluding the other classes where the rationale was unrelated. This left us with 3,104 SAT and 2,524 DSAT cases, which we consider a fairly even distribution (meaning that did not to artificially rebalance the dataset). For learning, we treat SAT as positive class (+1) and DSAT as negative class (-1). For each instance, we extract the 28 emission and session features shown in Table 2, as well as 12 structured features (some of which are shown in Table 1). This leads to 28 features for the baseline binary classifier which we shall discuss shortly, and 216 features for our structured learning framework.

To evaluate the performance of predictions, we measure both the precision and recall for both classes. We report the F_1 score that considers both metrics which is defined as $F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$. We also measure the accuracy of the prediction, defined as $Accuracy = 1 - \frac{misclassified}{all}$.

6.2 Methods Compared

We compare with the boosted decision tree classifier (BDT) [12] which has shown superior prediction performance for binary classification in many tasks. In [11], the authors reported that BDT performed best of all classifiers they tried. So we consider BDT a strong baseline for this study. BDT has three parameters to tune: the number of leaf node L, the shrinkage parameter η and the number of iterations M. In our experiments, we tried both $L = 2$ and $L = 10$. The other two parameters are selected via cross-validation. In particular, we set the maximum value of $M = 1000$ and perform early termination if we observe that the accuracy of the validation set no longer improves.

To evaluate whether the proposed joint-label/observation features in Table 1 work, we also compare with a structured SVM framework that has only the emission features and the histogram of state transitions, i.e., $\Psi(\mathbf{x}, \mathbf{y})$ in eq. (5). We name this baseline *SSVM-Basic* and our method with the entire feature space as *SSVM-Advance*.

6.3 Results and Explanations

In the learning procedure, we randomly split the 5,628 instances into training and test data with a 1:1 ratio, grouped by user sessions. We repeat the experiments 10 times for each algorithm and report the average performance. For SSVM, we further split the training data into validation and training and used two-fold cross validation to determine the balancing parameter C. It emerged that the optimal value of C for both models are fairly small — $C = 0.15$ for SSVM-Basic and $C = 0.2$ for SSVM-Advance.

Table 3 lists summarizes the prediction performance. Both SSVM framework significantly outperformed the baseline BDT method. In particular, our best SSVM-Advance model improved accuracy by over 22% relative to the strong baseline BDT method. Even without the advanced joint-label/observation features, we see that SSVM-Basic still outperformed BDT significantly with the addition of the label transition features. It confirmed that the structural dependencies between abandoned labels within sessions are truly helpful for prediction. Most noticeably, we see that the performance of the DSAT class benefits more from the structured learning framework. In the baseline BDT method, the classifier did much worse on DSAT than SAT. With the introduction of structured features, both SSVM models had better predictive power for DSAT than SAT class. Paired t-test results indicated that both SSVM's performance improvements were statistically significant with p-value < 0.0001. Note that improving the accuracy on DSAT queries has a *profound impact* on search engine relevance. A recent study[2] shows that the majority of user search tasks

[2]http://www.experian.com/hitwise/press-release-experianhitwise-reports-google-share-of-searche.html

Query Features	SERP Features	Session Features
Query Length (Char/Token) [**All**]	Answer Count [**All**]	SessionEntryPoint
Query Dwell Time [**All**]	HasGoodAnswers [**All**]	UserBrowserType
Is First/Last/Only/Url Query [**All**]	QueryClickCnt [**Prev/Next**]	Session Length (Query/Time)
IsNonAlphaNumericQuery [**All**]	QueryClickPosition [**Prev/Next**]	Total Abandoned Query
IsReformulation [**Prev/Next**]		

Table 2: List of all emission features and session-level features. The emission features can be further divided into query features and SERP features. Note here [All] indicates that the feature is available for the current, previous and next query, while [Prev/Next] means the feature can only be calculated from the immediate preceding and succeeding queries.

are satisfied (over 80%). This class imbalance also affects previous studies [1, 14], which report high accuracy in SAT prediction but poor performance in DSAT prediction. We believe that our research is a big step forward in addressing this important issue.

Note that our baseline BDT performed slightly worse than the reported number of Section 6.7 in [11] in terms of $F_{0.5}$ score (0.7520 vs. 0.7847). One reason could be that their data set was much smaller (1,799 instances) than ours (5,628 instances) and may have had a different distribution. However, our SSVM-Basic still outperformed their number noticeably in $F_{0.5}$ score (0.8237 vs. 0.7847), so did the best SSVM-Advance model ($F_{0.5} = 0.9045$).

Recall that at the end of Section 4 we mentioned that a task-based study was also conducted. In our experiments, we also trained a BDT using the task definition adopted by Liao et al. [20]. The accuracy of the task-segmented baseline was 0.7185, slightly worse than session baseline. We therefore believe that a more sophisticated segmentation method does not necessarily improve the prediction performance, and continue our experiments with only session-based segmentation.

In comparing between two SSVM models, we still observe a substantial performance improvement. Since it is not quite straightforward to aggregate the feature weights for individual features in the structured learning framework, we choose two methods to compare the structured features. In the first method, we add individual features in Table 1 to the SSVM-Basic model and compare the model performance. In the second method, we train the SSVM-Advance model with all structured features except for the one to evaluate.

Table 4 lists the model accuracy of individual features. We observe that in both models, the *SameQ* feature is most important, followed by the *ReformQ* feature. At the same time, *NQ* and *DT* perform quite similarly, where both features measure the gap between two abandonments. Overall, the *Min-Sim* feature seems to be least effective, only adding marginal improvement to both models.

Since the structured features need at least two labels in each training sequence to take effect, we break down the predictive performance by the length of training sequences. Table 5 lists the accuracy results, where for example 2(867) means a total of 867 training sequences with two abandonment labels. Note that the actual user search sessions can be arbitrarily long including many clicked queries. We observe that when the length of the training sequences increases, both SSVM-Basic and SSVM-Advance improve their performance gradually. Meanwhile, BDT exhibited similar performance regardless of session length. This result again confirms the importance of structured features in abandonment prediction given the high label dependency between queries.

	Accuracy	SAT F_1-score	DSAT F_1-score
BDT [12]	0.7195	0.7545	0.6729
SSVM-Basic	0.8238	0.8174	0.8396
SSVM-Advance	0.8774	0.8639	0.8885

Table 3: Binary Prediction performance of accuracy and F_1 score for SAT and SAT classes. Both structured SVM models substantially outperformed a strong baseline. Our models improve a lot on the DSAT class, see text for details.

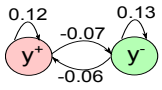
	SSVM-Basic *Plus*	**SSVM-Advance** *Without*
NQ	0.8378	0.8672
DT	0.8321	0.8671
Ans	0.8316	0.8696
Click	0.832	0.8689
SameQ	0.8476	0.8548
ReformQ	0.8392	0.8613
Max-Sim	0.8311	0.8698
Min-Sim	0.8275	0.8705

Table 4: The performance of individual structural features in terms of Accuracy. The second column means the model plus that feature, higher means better. The third column means the model without that feature, so lower means the feature is more important. Top three features are highlighted.

To better understand feature performance among different label transition scenarios, in Figure 5, we plot the feature weights of several structural features. Recall that each of these features is divided into four features during feature construction according to the label transitions $[y_i = (0,1)][y_{i-1} = (0,1)]$. In (a), the label transition weights of y indicate that it is indeed that transition to the same class is much more likely than to the opposite class, which confirms our discovery of label dependency in the user behavior analysis presented earlier. Similarly, (b), (c) and (d) lists feature weights of the three most important structural features. It can be seen, for example, from the *SameQ* feature, similar queries are more likely to share the same abandonment label. Interestingly, from the DT feature, we see that all weights are positive except for the self-transition between DT^-.

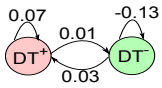
	BDT	SSVM-Basic	SSVM-Advance
2(867)	0.7085	0.7903	0.8579
3(238)	0.7096	0.8195	0.8692
4(88)	0.7155	0.8309	0.8942
5(30)	0.7146	0.8303	0.8867
>5(34)	0.7157	0.8312	0.8893

Table 5: Accuracy result broken down by the length of training sequences. Numbers in parentheses are the total number of sessions with that length.

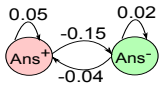


(a) Label transition

(b) Same-Query transition



(c) Dwell-Time transition



(d) Answer transition

Figure 5: Feature weights of four transition features. Transition to the same class often has higher weight.

This indicates that when the query dwell time increases, it is more likely that users are satisfied by the current SERP and thus causes the current query to be good abandonment. It is also less likely (-0.13) that a bad abandonment will occur again after the user has expended more effort in examining the new SERP of the next query.

6.4 Application on Ranking Improvement

We further seek to use the prediction results to improve search quality. Specifically, we contribute a new Clickthrough-Rate (CTR) feature from good abandoned queries. We also propose to use bad abandonment as a signal to extract preference data to enhance the training data used by search engine ranking algorithms.

New CTR Feature: by definition, CTR is calculated by, for each query-document pair, the total number of clicks divided by the total number of impressions, i.e., how many times the document was shown for that particular query. In the case of abandonment, users perform no click on any document, therefore the impression number increases but click number remains the same. However, for good abandoned queries, this calculation is inaccurate since the user's information need is satisfied. We therefore propose to subtract the estimated number of good abandoned query impressions from all impression to get a more accurate CTR, i.e.,

$$\text{New-CTR}(Q,U) = \frac{\text{Clicks}(Q,U)}{\text{Impressions}(Q,U) - \text{GoodAbandonment}(Q)}$$

New Click Preference Data: search engines often learn the ranking functions from user preference data. It has been shown that pair-wise learning frameworks (e.g., RankSVM [16], LambdaMART [27]) consistently outperform point-wise learning methods (e.g., ordinal regression). Pair-wise training data is often constructed by comparing the relevance of two documents returned for the same query. The learning algorithms then take these preferences and try to learn an optimal ranking function by minimizing a cost function, e.g., pair-wise loss [16, 27].

In the case of bad abandonment, a user was dissatisfied with all results. If the user kept reformulating the query and finally clicked one of the results, we can infer an implicit preference from this scenario. To be concrete, given a no-click query Q_a which we labeled as bad abandonment, we examine the subsequent queries in the same session. If a query Q_b has a click $U_b^{(c)}$ and we think Q_b is a reformulation of Q_a (using the same criteria as described in Section 5.3, we construct a set of pair-wise preference data from the session as $\mathbf{S} = \{(U_b^{(c)}, U_a^{(i)})\}$, where $U_a^{(i)}$ is a no-click document for query Q_a. Each pair in $\mathbf{S}$ indicates a preference relationship of $U_b^{(c)} \succ U_a^{(i)}$.

6.4.1 Generating Data for Ranking

We study the impact of the new features from a log analysis perspective using the logs of the Microsoft Bing search engine gathered from December 2012 and April 2013 from the U.S. search market. To form a baseline training file, we first randomly sampled 12,000 queries from the logs during that time. We then extracted 10 to 20 documents for each query and asked human assessors to annotate their query relevance on the following five-point scale: Perfect (5), Excellent (4), Good (3), Fair (2), and Bad (1). Each query-document pair was judged by multiple assessors so its final label was determined by a majority vote. We further extracted 400 ranking features for each pair, including some frequently used features such as BM25 and TF.IDF. Given this feature set and the relevance judgments, we use the LambdaMART [27] algorithm to train a ranker.

To generate the new CTR feature, we ran our predictor on the logs of the same time period. Among the 12,000 queries, 3,765 queries had one or more good abandonment labels. On the other hand, we also discovered that 265,872 sessions had one or more bad abandonment followed by a reformulation. To prevent generating too much training data, we limit the number of training pairs in each session to be three, i.e., top-3 returned documents for the bad abandonment and the first clicked document by the reformulation. We further require a pair to appear at least five times in the logs to be considered in training. We generated 35,742 new training instances.

6.4.2 Ranking Performance

For training and testing, we randomly split the original 12,000 queries into five parts for five-fold cross validation. We use 4/5 for training and 1/5 for testing. The number of leaf nodes is set to be 10 for LambdaMART. The learning rate is set to be 0.1. The maximum training iterations is limited to be 100. We repeat the experiment five times and report on the average performance.

Table 6 lists the performance in terms of average NDCG at the third rank position (NDCG@3) scores and standard deviation. We see that the new CTR feature indeed captures the document relevance more accurately by improving the NDCG score for 0.4% over the old CTR feature and 1.3% over the baseline. On the other hand, we also tested the performance of adding different portion of the new training data to the original training data. The results clearly indicate that new training data is helpful. In particular, the performance improvement becomes statistically significant after adding 80% or more new training data. The best performance is achieved by combining the new CTR feature and

	Test Set NDCG@3
Baseline	0.6073± 0.045
Baseline + Old CTR	0.6092 ± 0.052
Baseline + New CTR	0.6135± 0.059
Baseline + New Data (20%)	0.6143± 0.028
Baseline + New Data (40%)	0.6167± 0.003
Baseline + New Data (60%)	0.6179± 0.064
Baseline + New Data (80%)	**0.6198± 0.076**
Baseline + New Data (100%)	**0.6201± 0.088**
Baseline + New (CTR + Data) (100%)	**0.6274± 0.065**

Table 6: Ranking performance of the new CTR feature and new training instances from the outcome of abandonment prediction. Bolded results are statistical significant.

the new training data, which achieves 0.6274 NDCG score – an improvement of over 2% in absolute percentage.

7. CONCLUSIONS AND FUTURE WORK

Web search abandonment occurs frequently but is not fully understood. Through analysis of logs from a large-scale user study, we discovered several characteristics of abandonment, namely: (1) bad abandoned queries are longer than good abandoned queries, (2) query dwell time of bad abandonment is less than good abandonment, (3) session length given good abandonment is shorter than bad abandonment, and most importantly (4) bad abandonment is more likely to lead to another bad abandonment in the same session (same applies to good abandonment). Based on these characteristics, we proposed a structured learning framework to model abandonment using emission features and structured features. Experimental results demonstrated a significant improvement of our framework over the state-of-the-art boosted decision tree method presented in earlier work [11].

We leveraged our model to improve search relevance by devising more reliable CTRs and improving the quality of training data. We found strong relevance gains from applying our model to enhance result ranking.

Future work will extend our research to predict abandonment rationales for individual users by considering factors such as user expertise and tenacity. We will also improve our prediction performance, and pursue the application of our model for tasks such as metric development and real-time searcher assistance.

8. REFERENCES

[1] M. Ageev, Q. Guo, D. Lagun, and E. Agichtein. Find it if you can: a game for modeling different types of web search success using interaction data. In *SIGIR '11*, pages 345–354, 2011.

[2] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden markov support vector machines. In *ICML '03*, pages 104–111, 2003.

[3] M. S. Bernstein, J. Teevan, S. Dumais, D. Liebling, and E. Horvitz. Direct answers for search queries in the long tail. In *CHI '12*, pages 237–246, 2012.

[4] G. Buscher, R. W. White, S. Dumais, and J. Huang. Large-scale analysis of individual and task differences in search result page examination strategies. In *WSDM '12*, pages 373–382, 2012.

[5] C. Castillo, A. Gionis, R. Lempel, and Y. Maarek. When no clicks are good news. In *SIGIR 2010 Industry Track*, 2010.

[6] M.-W. Chang and W. tau Yih. Dual coordinate descent algorithms for efficient large margin structured prediction. *TACL*, 1:207–218, 2013.

[7] L. B. Chilton and J. Teevan. Addressing people's information needs directly in a web search result page. In *WWW '11*, pages 27–36, 2011.

[8] A. Chuklin and P. Serdyukov. Good abandonments in factoid queries. In *WWW '12 Companion*, pages 483–484, 2012.

[9] A. Chuklin and P. Serdyukov. How query extensions reflect search result abandonments. In *SIGIR '12*, pages 1087–1088, New York, NY, USA, 2012. ACM.

[10] A. Chuklin and P. Serdyukov. Potential good abandonment prediction. In *WWW '12 Companion*, pages 485–486, New York, NY, USA, 2012. ACM.

[11] A. Diriye, R. White, G. Buscher, and S. Dumais. Leaving so soon?: understanding and predicting web search abandonment rationales. In *CIKM '12*, pages 1025–1034, 2012.

[12] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.

[13] A. Hassan, R. Jones, and K. L. Klinkner. Beyond dcg: user behavior as a predictor of a successful search. In *WSDM '10*, pages 221–230, 2010.

[14] A. Hassan, Y. Song, and L.-w. He. A task level metric for measuring web search satisfaction and its application on improving relevance estimation. In *CIKM '11*, pages 125–134, 2011.

[15] J. Huang, R. W. White, and S. Dumais. No clicks, no problem: using cursor movements to understand and improve search. In *CHI '11*, pages 1225–1234, 2011.

[16] T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02*, pages 133–142, 2002.

[17] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR '05*, pages 154–161, 2005.

[18] A. Koumpouri and V. Simaki. Queries without clicks: Evaluating retrieval effectiveness based on user feedback. In *SIGIR '12*, pages 1133–1134, New York, NY, USA, 2012.

[19] J. Li, S. Huffman, and A. Tokuda. Good abandonment in mobile and pc internet search. In *SIGIR '09*, pages 43–50, 2009.

[20] Z. Liao, Y. Song, L.-w. He, and Y. Huang. Evaluating the effectiveness of search task trails. In *WWW '12*, pages 489–498, 2012.

[21] T. Sakai, M. P. Kato, and Y.-I. Song. Click the search button and be happy: evaluating direct and immediate information access. In *CIKM '11*, pages 621–630, 2011.

[22] S. Stamou and E. N. Efthimiadis. Queries without clicks: Successful or failed searches. In *SIGIR 2009 Workshop on the Future of IR Evaluation*, pages 13–14, 2009.

[23] S. Stamou and E. N. Efthimiadis. Interpreting user inactivity on search results. In *ECIR'2010*, pages 100–113, Berlin, Heidelberg, 2010. Springer-Verlag.

[24] E. Thuma, S. Rogers, and I. Ounis. Evaluating bad query abandonment in an iterative sms-based faq retrieval system. In *OAIR '13*, pages 117–120, Paris, France, France, 2013.

[25] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML '04*, 2004.

[26] M. wei Chang, V. Srikumar, D. Goldwasser, and D. Roth. Structured output learning with indirect supervision. In *ICML '10*, 2010.

[27] Q. Wu, C. J. Burges, K. M. Svore, and J. Gao. Adapting boosting for information retrieval measures. *Inf. Retr.*, 13(3):254–270, June 2010.

[28] T. Yao, M. Zhang, Y. Liu, S. Ma, Y. Zhang, and L. Ru. Investigating characteristics of non-click behavior using query logs. In *AAIRS*, volume 6458 of *Lecture Notes in Computer Science*, pages 85–96. Springer, 2010.