Does That Mean You're Happy?

RNN-based Modeling of User Interaction Sequences to Detect Good Abandonment

Kyle Williams
Microsoft
One Redmond Way
Redmond, Washington 98052
kyle.williams@microsoft.com

Imed Zitouni
Microsoft
One Redmond Way
Redmond, Washington 98052
izitouni@microsoft.com

ABSTRACT

Queries for which there are no clicks are known as abandoned queries. Differentiating between good and bad abandonment queries has become an important task in search engine evaluation since it allows for better measurement of search engine features that do not require users to click. Examples of these features include answers on the SERP and detailed Web result snippets. In this paper, we investigate how sequences of user interactions on the SERP differ between good and bad abandonment. To do this, we study the behavior patterns on a labeled dataset of abandoned queries and find that they differ in several ways, such as in the number of user interactions and the nature of those interactions. Based on this insight, we frame good abandonment detection as a sequence classification problem. We use a Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN) to model the sequence of user interactions and show that it performs significantly better than other baselines when detecting good abandonment, achieving 71% accuracy. Our findings have implications for search engine evaluation.

CCS CONCEPTS

•Information systems → Evaluation of retrieval results; •General and reference → Metrics;

KEYWORDS

Good abandonment, satisfaction, user interaction modeling, LSTM, mouse movements

1 INTRODUCTION

It has become increasingly common for search engines to provide an enhanced user experience beyond 10 blue links. These experiences take many forms, such as showing images and videos interspersed with search results; or having news headlines show up on a Search Engine Results Page (SERP) to save the user the hassle of having to visit a news site in order to stay up to date. Alternatively, answers on a SERP have become increasingly common as an attempt to satisfy a user's information need without them ever needing to visit other Web pages. For instance, Figure 1 shows an example of a SERP from a commercial search engine for the query *weather seattle*. As can be seen from the figure, the SERP displays the current weather as well as the weather forecast for the coming days, thereby making it less likely for the user to click on any search results in order to be satisfied.

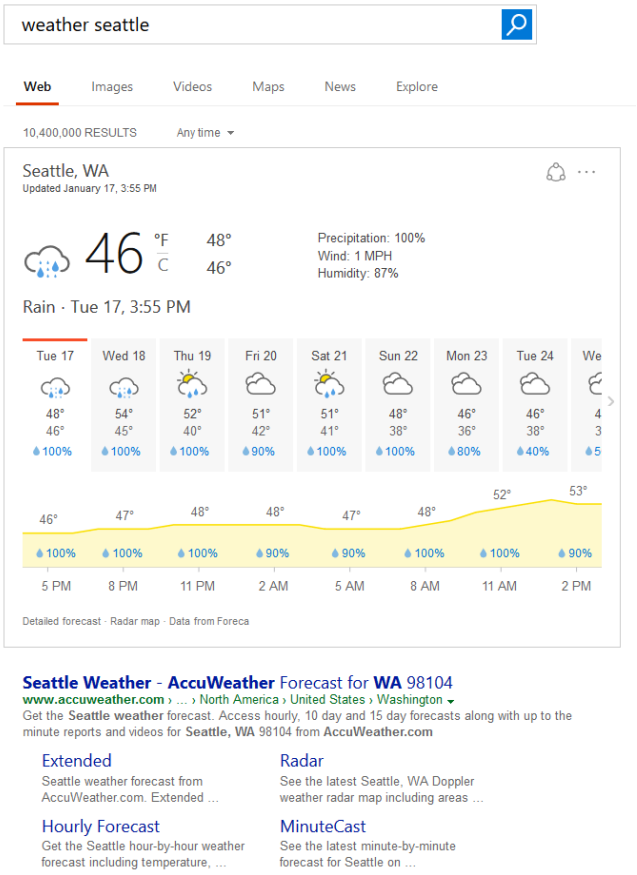

Figure 1: An example of a weather answer on a SERP for the search query *weather seattle*. The presence of this answer may make it less likely for the user to click on Web search result in order to be satisfied.

Traditionally, abandonment in search was seen as a negative indicator of search quality, since it implied that users were not satisfied by any of the search results among the 10 blue links. However, due to search engine features, such as those mentioned above among

CIKM'17, November 6–10, 2017, Singapore.
 ISBN 978-1-4503-4918-5/17/11…$15.00
DOI: 10.1145/3132847.3133035

others, there has been increasing awareness that abandonment can also be good [1, 6, 32, 36, 39]. A challenge for search engines then exists in differentiating between good and bad abandonment since most traditional approaches for evaluating search engine performance are based on clicks and dwell times [8, 16, 17, 24, 25].

Previous approaches have attempted to address the shortcomings in click and dwell time-based evaluations by considering other signals, such as properties of the query [17] and session [7, 36], gaze and viewport tracking [31], and signals associated with the SERP and its elements, such as the distance scrolled or the attributed reading time for web result snippets and answers [39].

We take a different approach and propose to differentiate between good and bad abandonment based on the sequence of interactions that a user makes on an abandoned SERP. Every time a user submits a query the search engine returns a SERP. A user then engages with the SERP in various ways. For instance, they may scroll to inspect the search results, pause to read, scroll again, then move the mouse, and finally abandon the query. This type of user behavior can be viewed as a sequence of interactions over time. We hypothesize that this sequence of user interactions provides meaningful information that can be used to differentiate between good and bad abandonment. This is in contrast to other approaches that have analyzed static user behaviors [31, 39] or properties that are not user behaviors [7, 17].

Thus, the main research question addressed by this research is:

> *Can a sequence of user interactions over time be used to differentiate between good and abandonment?*

We break this main research question into two sub-research questions.

> **RQ1:** *How do sequences of user interactions differ between good and bad abandonment?*
>
> **RQ2:** *How well can sequences of user interactions be used to differentiate between good and bad abandonment?*

We hypothesize that user behavior differs between good abandonment and bad abandonment. Therefore, we analyze a collection of labeled abandoned queries and study how their interactions differ to answer **RQ1**. Based on the insight gathered in **RQ1**, we then set out to answer **RQ2**, which attempts to use the differences in user interactions for good and bad abandonment queries to build a classifier to differentiate between them. To do this we propose to use a Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN) to model the sequence of interactions and treat the problem as a sequence classification problem. We choose to use an LSTM since LSTMs have been shown to be state of the art in various sequence modeling applications, such as language modeling and translation [9]. To our knowledge, this is the first paper to address the good abandonment problem by framing it as a sequence classification problem.

In summary, this paper makes the following contributions:

- We study the difference between the interaction sequences for good abandonment queries and bad abandonment queries.
- Based on insight gathered on these difference, we model the good abandonment detection problem as a sequence classification problem.
- We propose to use an LSTM to predict good abandonment and bad abandonment and compare the performance of the proposed method to two baselines.

In making these contributions, the rest of this paper is structured as follows. Section 2 presents related work and is followed by Section 3 which frames the problem and describes the user interactions we use in this study. Section 4 studies the difference between interactions sequences for good and bad abandonment, while Section 5 describes the methods that we use to differentiate between good and bad abandonment, including our LSTM-based neural network. Section 6 then presents a set of experiments comparing our proposed method to two baselines and Section 7 presents the conclusions.

2 RELATED WORK

Related work falls into three categories: satisfaction in search; detecting good abandonment; and user interactions. We do not review general related work on sequence classification due to space constraints; however, a good overview is available in [41].

2.1 User Satisfaction in Search

Satisfaction is a measure of a user's search experience and can be thought of as the extent to which the goals or desires of a user are fulfilled [21]. Satisfaction is a subjective measure and may be influenced by many factors, such as: the latency of the search engine; the effort required to search; the relevance of search results; and even the query itself [23]. In this sense, it is different from more objective relevance-based metrics, such as: Precision@k, NDCG, and MAP, in that it attempts to measure the whole user experience rather than just the relevance of results. Furthermore, it has been shown that search success does not necessarily indicate search satisfaction [14]. In addition to this, there has been work on personalized measurement of satisfaction [18] and fine-grained levels of satisfaction [20]. Query abandonment is related to satisfaction in that good or bad abandonment may contribute to satisfaction.

There have been several methods proposed for modeling and predicting satisfaction. For instance, query reformulation has been used to measure search success [17] and it was shown that predicting satisfaction and success based on query features results in better performance than predicting satisfaction by click-based features. It has also been shown that clicks followed by long dwell times are correlated with satisfaction [8].

Kim et al. [24] study the effect of dwell time on search satisfaction and consider three different measures of dwell time. In [25] it is shown that the dwell times on a landing page are affected by the topic of a page and the authors propose query-click complexities in modeling dwell times on landing pages. Since we only consider abandoned queries in our study, landing page dwell times do not exist; however, we do consider a similar user interaction, which is dwell times or pauses on the SERP.

In work similar to ours, Hassan et al. [16] model the search process as a sequence of actions that include clicks and queries and build models that characterize successful and unsuccessful search sessions. This work is extended in [15], where a generative model is used. Our work is similar in that we also use sequences of user behaviors; however, it differs in that we model fine-grained user

interactions on the SERP, such as scrolls and mouse movements, rather that session interactions, such as clicks, queries, and reformulations. Furthermore, our work differs in that we use these interactions to differentiate between good and bad abandonment rather than detect search success. Other work involving success in search has analyzed the difference between struggling and success in search and predicted signs of struggling [35].

Kiseleva et al. [27] predict satisfaction when using intelligent assistants. Using features derived from voice commands and touch interactions, they achieve better prediction accuracy than click and query features alone in data collected from a user study [28]. Importantly, they evaluate satisfaction at the task level since often interactions with intelligent assistants involve multiple queries.

2.2 Good Abandonment

Abandonment plays a large role in search satisfaction. For instance, in [38] it was found that 27% of searches were performed with the the pre-determined intent of having the search satisfied directly by information on the SERP. Of these searches, about 75% were satisfied this way. Diriye et al. [7] surveyed 186 participants and found that satisfaction accounted for 32% of abandonment. They also sampled 39,606 queries and gathered reasons for abandonment via a popup window on the 22% of those queries that had been abandoned. In the cases where feedback was provided, it was found that satisfaction accounted for about 38% of abandonment.

Chilton and Teevan [3] study the effect that answers on a SERP have on a user's interactions. They observed that answers *cannabilize* clicks by reducing interaction with the SERP. Williams et al. [40] extend this work and consider the effect that answers on mobile SERPs have on good abandonment. The authors found that the type of answer affects user click and abandonment behavior and argue that all answers on a SERP should not be treated equally. In [5] it was found that high quality SERPs lead to increased abandonment and less clicks. One reason for this is that high quality SERPs are more likely to contain rich information in snippets and answers.

Several different features have been used to predict good abandonment. In [6], topical, linguistic features are used to detect potential good abandonment. Song et al. [36] consider context when predicting good abandonment. The authors make use of query level features, such as query length and reformulation, SERP features, and session features. Context is incorporated by also including these features from neighboring queries.

Li et al. [32] provide an upper limit on a good abandonment rate. They found that, of queries that could potentially lead to good abandonment, 70% of those on mobile could clearly or possibly be satisfied by the mobile SERP and 56% on desktop could clearly or possibly be satisfied by the SERP. The authors hypothesize that the reason for the high value on mobile is that mobile users are more likely to formulate their queries in such a way so as to increase the possibility of them being directly answered on the SERP.

Williams et al. [39] study good abandonment in mobile search and find that there are several reasons for good abandonment in mobile search, including search result snippets, images, and answers. The authors propose to detect good abandonment based on user behaviors on the SERP, such as reading times and scrolls. This work is similar to ours in that it considers user behaviors; however, it differs in that a) the authors consider mobile search, whereas we consider desktop search and, b) the authors consider static features, such as the number of scrolls, or average reading times for answers. By contrast, we choose to model the sequence of interactions that a user makes over time.

In [22], the authors use similar features to those of [39] to design an online metric for use in an A/B experiment. They show the metric to perform better at detecting user satisfaction compared to a metric that only considers click and query-based signals.

2.3 User Interactions for Relevance & Satisfaction

User interactions have been used in several studies to detect satisfaction in search. For instance, [33] use mouse movement information to predict search satisfaction. The authors consider both direct feedback from users on their satisfaction as well as labels from external judges and interestingly find that they consider different factors when determining whether a search led to satisfaction. Scroll and mouse movement behaviors have also been considered in other studies for detecting satisfaction [2, 10, 11]. Guo and Agichtein [11] consider scroll and cursor behavior on landing pages to estimate landing page relevance and similar features are used in [13] to predict search success.

There have been several studies on the use of interactions on mobile devices to detect search satisfaction. For instance, in [12], mobile user behaviors, such as zooms, swipes, and dwell times on landing pages are used to predict Web search satisfaction. Similar features along with click through data are in used in [12] to predict search success. Williams et al. [39] used similar features specifically to detect good abandonment in mobile search and similar features are once again used in [27] to predict task satisfaction with a mobile intelligent assistant.

Viewport- and eye-tracking were used in [31] to measure attention and satisfaction and the correlation between gaze time and viewport time were established. Attention was also combined with satisfaction and clicks in [4] and in [30] attention was inferred by jointly modeling interactions and the saliency of Web page content.

In [29] the authors automatically discover frequent mouse movement patterns and use it for relevance prediction and ranking. Our work differs from that work since we focus on detecting good abandonment. Furthermore, instead of general cursor movement patterns, we use a set of predefined mouse movements that capture how users interact with specific elements on the SERP.

Our work is similar to the work mentioned above in that we also consider user interactions to detect satisfaction. However, it differs in two important ways: 1) we do not measure satisfaction in general but instead only measure satisfaction due to good abandonment, and 2) instead of considering the user interactions as static features, we instead propose to model them as a sequence over time and thus frame the problem as a sequence classification problem.

3 SEQUENCES OF USER INTERACTIONS

A search session consists of various interactions, such as submitting queries, visiting Web pages, returning to the search engine, reformulating the query, etc. Similarly, one can also think of a sequence

of interactions taking place on the SERP; however, unlike the examples of interactions in a search session, the interactions on the SERP are more fine-grained. For instance, they may include interactions such as mouse movements, scrolls, and dwells or pauses.

3.1 Why User Interactions?

An important question to ask is: why consider user interactions when differentiating between good and bad abandonment abandonment? An important concept in search engine evaluation is the difference between *endogenous* and *exogenous* features. Endogenous features are features that the search engine has control over. For instance, if the presence of a weather answer on the SERP (such as in Figure 1) is used in a metric to determine whether or not good abandonment occurred, then a ranker optimizing for this metric may unintentionally learn to put a weather answer on a page. Exogenous features differ from endogenous features in that the search engine does not have direct control over them. For instance, a user scrolling or pausing on a page is a user decision and not a search engine decision. That being said, the difference between endogenous and exogenous features is not absolute but rather falls along a spectrum. The user interactions we use in this study and that are described in Section 3.2 are exogenous. They involve user behaviors in their attempt to satisfy their information needs. Therefore, unlike the presence of, say, a weather answer, they directly capture user behavior and we attempt to model how these behaviors relate to good and bad abandonment.

3.2 Interactions

In this study, we consider the following user interactions that could potentially occur on an abandoned SERP.

- **Short Pause (SP)** The user paused on the SERP for $s \leq 5$ seconds.
- **Medium Pause (MP)** The user paused on the SERP for $5 > s \leq 15$ seconds.
- **Long Pause (LP)** The user paused on the SERP for $15 > s \leq 30$ seconds.
- **Very Long Pause (VLP)** The user paused on the SERP for $s > 30$ seconds.
- **Scroll Down (SD)** The user scrolled down on the SERP.
- **Scroll Up (SU)** The user scrolled up on the SERP.
- **Scroll (S)** The user scrolled on the SERP but no down or up direction was detected.
- **Move-Web (MW)** The user moved their mouse pointer onto a Web result.
- **Move-Answer (MA)** The user moved their mouse pointer onto an Answer.
- **Mouse Read (MR)** The user moved their mouse in a left-to-right horizontal direction for a minimum amount of pixels, thereby emulating reading from left to right.

For each abandoned query, we build a sequence of interactions made up of the interactions described above.

3.3 Interaction Timeline

A user's interaction on the SERP create a sequence of interactions over time. In the case of an abandoned query, a user begins at time step t_0, when they land on the SERP. They then take interactions in a sequence of timesteps t_1-t_{n-1}, afterwhich they abandon the query at timestep t_n. Therefore, we can represent the set of user interactions on an abandoned SERP as the sequence $a_{t_0}, a_{t_1}, a_{t_2}, ..., a_{t_{n-1}}, a_{t_n}$, where a_t is the action at timestep t

4 SEQUENCES OF ACTIONS IN ABANDONED QUERIES

One of the main contributions of this paper is to measure how sequences of user interactions differ for queries leading to good abandonment and those leading to bad abandonment. In the previous section we described user interactions on an abandoned query as forming a sequence of interactions over time and presented a set of 10 interactions that we consider in this study. In this section, we present an analysis of the properties of those sequences of interactions in abandoned queries. This analysis is based on a dataset of labeled abandoned queries, which we describe in the next section.

4.1 Dataset

Labeled data was gathered via crowdsourcing in order to conduct experiments. The crowdsourcing platform included tools for judge qualification and automatic spam detection and judges were trained in the good abandonment detection task. The data came from a commercial search engine and about 1000 abandoned queries were sampled for judgment. We follow a similar approach to other studies where crowdsourced judgments were collected for measuring user satisfaction and showed the judges queries in context [34, 39]. Judges were shown the following information: the query that was abandoned; the previous and next queries to provide query context; the location of the user when the query was submitted; and a screenshot of the SERP for the abandoned query. Judges were the asked to provide feedback as to whether the abandonment was good, bad, or ambiguous. At least 3 judgments were collected for each query and the majority vote was take. The overall judge agreement rate on the labels was 88.8%.

In order to increase the size of our labeled data set we perform automatic data expansion. The data expansion is based on the assumption that, if a judge were to rejudge a previously seen query in the same context, then they would assign it the same label. Based on this assumption, the data was then expanded as follows. For an abandoned query, q_n, a triple in the form of (q_{n-1}, q_n, q_{n+1}) was generated where q_{n-1}, q_n, and q_{n+1} were successive queries in a user session. Then, this triple was matched against all triples in which the abandoned query occurred in the dataset that we sampled the original 1000 queries from and all matching occurrences were given the same label. For instance, if the label for the triple (*information retrieval conferences*, *cikm 2017*, *cikm 2017 submit date*) was labeled as good abandonment, then any query with the same triple was labeled with good abandonment. It is important to note that, while automatically expanded query triples may have the same label, their interaction sequences may differ.

We gathered a total of 21,262 labeled judgments for abandoned queries. Of these, 10,032 were labeled as good abandonment and 11,230 were labeled as bad abandonment. We use this dataset for our sequence analysis as well as for our experiments in Section 6.

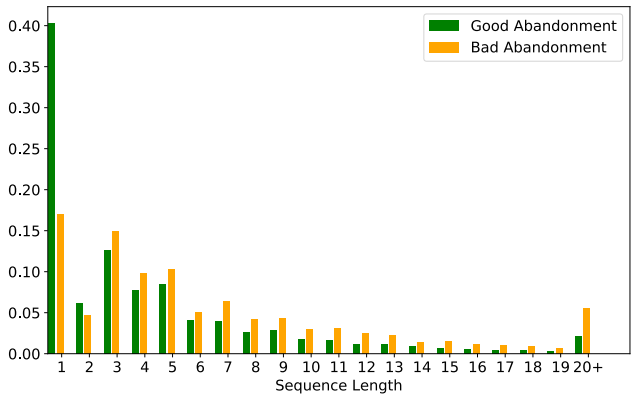

Figure 2: Length distribution of sequences of interactions in abandoned queries.

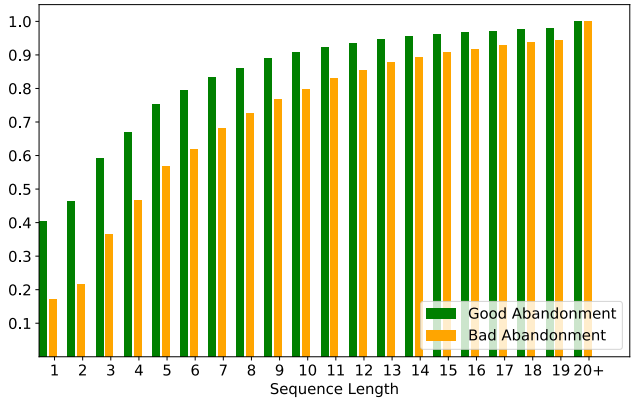

Figure 3: Cumulative distribution of sequence lengths.

4.2 Sequence Length

We first analyze the lengths of the sequences of interactions. Figure 2 shows the distribution of these lengths. As can be seen from the figure, as much as 40% of good abandonment queries contain only one interaction on the SERP compared to about 17% for bad abandonment queries. As will be shown later, the majority of these are *Move* interactions. The same observation can be made for sequences of length 2, where a larger proportion of good abandonment sequences contain two interactions relative to the proportion of two interaction sequences for bad abandonment. However, for sequences with 3+ interactions this relation is reversed. At each sequence length of 3 or higher, bad abandonment has relatively higher occurrence compared to good abandonment. The data suggests that bad abandonment is generally associated with longer sequences of interaction on the SERP. This result is in line with the finding in [39], where it was shown that the total number of mobile swipe actions was negatively correlated with user satisfaction.

Figure 3 shows the cumulative distribution of sequence lengths. Analyzing the difference between good abandonment and bad abandonment reveals some interesting trends. For instance, about 80% of good abandonment queries contain 6 or fewer interactions and 90% of them contain 10 or fewer interactions. Thus, the vast majority of satisfaction due to good abandonment is achieved with 10 or fewer interactions. This is in contrast to bad abandonment where 10 and 15 interactions are required to account for 80% and 90% of queries, respectively. This analysis suggests that more interactions on the SERP could be associated with less satisfaction and more bad abandonment. One way of interpretation this finding is that more interactions are a result of users struggling to satisfy their information need.

4.3 N-Gram Frequency

The previous section showed how sequence lengths tend to be longer for bad abandonment queries. The next question that we seek to answer is: how do the actual interactions differ among good and bad abandonment? Are some interactions or sub-sequences of interactions more common? To answer this question, we analyze the relative frequency of the top 10 occurring unigrams, bigrams, and trigrams in the data. The results are shown Table 1.

As can be seen from Table 1, the most common occurring interaction is a *Move* interaction, occurring in 98% and 92% of good abandonment and bad abandonment queries, respectively. The next most common unigram in both cases is *Move-Answer*, where the user moves their mouse onto an answer on the SERP. However, the relative frequency differs by a large amount as this interaction only occurs in 46% of good abandonment queries compared to 71% for bad abandonment. One interpretation of this is that, when the answer satisfies the user then there is less of a need for them to interact with it. Table 1 also shows that user are also more likely to dwell on a bad abandonment query. For instance, the *Short Pauses (SP), Medium Pauses (MP)* and *Long Pauses (LP)* occur on 53%, 40% and 14% of bad abandonment queries, respectively. By contrast, SP, MP and LP occur on 34%, 22%, and 7% of good abandonment queries. The more frequent pauses in bad abandonment queries suggests that users spend more time reading the SERP. This hypothesis is supported by the frequency of *Scroll Downs (SD)* and *Scroll Ups (SU)*. For good abandonment queries, these two types of scrolls occur in 24% and 15% of queries, respectively; whereas in bad abandonment they occur in 35% and 31% of queries. As was the cases with longer pauses for bad abandonment queries, this increased scrolling behavior is indicative of users spending more time looking for information on the SERP.

One can observe similar behavior when looking at the bigrams and trigrams. For instance, the bigram *Scroll Down-Short Pause (SD-SP)* occurs twice as frequently in bad abandonment queries than it does in good abandonment queries at 22% and 11%, respectively. Once again, this bi-gram is indicative of information seeking behavior. Similarly, the *Scroll Down-Short Pause-Scroll Up* trigram is the most common trigram for bad abandonment and occurs in 15% of queries. This trigram captures the process by which a user scrolls down a page, reads the results and likely finds nothing useful, and then scrolls back up.

4.4 Discussion

this analysis has shown, the length and types of interactions differ among good and bad abandonment and allows us to answer **RQ1:** *How do sequences of user interactions differ between good and bad*

Table 1: Top 10 occurring unigrams, bigrams, and trigrams for abandoned queries in labeled data. The relatively frequency in which each n-gram occurs in the data is shown in parentheses as a percentage.

Unigrams	
Good	**Bad**
M (98%)	M (92%)
MA (46%)	MA (71%)
SP (34%)	SP (53%)
SD (24%)	MP (40%)
MP (22%)	SD (35%)
MW (20%)	MW (33%)
SU (15%)	SU (31%)
MR (10%)	LP (14%)
LP (7%)	VLP (12%)
VLP (6%)	MR (10%)
Bigrams	
Good	**Bad**
M,MA (22%)	MA,M (28%)
MA,M (18%)	M,SP (25%)
MA,SP (15%)	M,MA (24%)
SP,M (13%)	SD,SP (22%)
M,SP (13%)	MA,SP (22%)
SD,SP (11%)	SP,MA (21%)
SP,SD (11%)	SP,M (21%)
SP,MA (10%)	M,MP (19%)
SP,SU (9%)	SP,SU (18%)
MA,MP (8%)	SP,SD (18%)
Trigrams	
Good	**Bad**
M,MA,M (9%)	SD,SP,SU (15%)
MA,SP,M (8%)	M,SP,MA (14%)
SD,SP,SU (7%)	MA,SP,M (14%)
M,SP,MA (7%)	M,MP,MA (11%)
MA,M,MA (7%)	SP,SD,SP (9%)
M,MA,SP (6%)	M,MA,M (9%)
SP,SD,SP (5%)	MP,SD,SP (8%)
MA,MP,M (4%)	MA,MP,M (7%)
MA,SP,SD (4%)	SP,MA,M (7%)
M,SP,SD (4%)	M,SP,SD (7%)

abandonment? Bad abandonment queries usually lead to more user interaction as users spend more time searching for information to satisfy their information needs. This is in contrast to good abandonment queries where interaction sequences are relatively short. Furthermore, the types of interactions differ with dwell and scroll actions being relatively more common among bad abandonment queries than good abandonment queries. These findings suggest that there are sufficient differences between the sequences of user interactions for good abandonment and bad abandonment to warrant the use of them for differentiating between the two. We present an experiment testing this idea in the next section.

5 CLASSIFYING SEQUENCES OF USER INTERACTIONS

Section 4 presented an analysis of the differences between sequences of interactions for good abandonment queries and bad abandonment queries. It was found that differences exist in terms of the lengths of sequences of interactions as well as the interactions themselves. A natural question then arises is given by **RQ2:** *How well can sequences of user interactions be used to differentiate between good and bad abandonment?*

In answering this question we seek to determine if the differences observed in the interactions for good abandonment and bad abandonment are useful to differentiate between the two.

Since this study has focused on sequences of user interactions, sequence classification provides a natural solution to the problem. In sequence classification, the input is a sequence s and the goal is to find the most likely class c given s. There are several ways one might go about building a classifier to determine $P(c|s)$ and, in this paper, we propose to use a Long Short Term Memory (LSTM) Recurrent Neural Network (RNN). LSTMs have the benefit over standard RNNs in that they are able to better handle long term dependencies in the data [19]. As a result of this, LSTMs have been able to achieve state of the art performance in several sequence classification problems [9]

We compare the performance of the LSTM-based architecture that we use to two baseline methods. The first is a classifier based on the most frequent n-grams in a large dataset of abandoned queries. The second is a generative model that previously achieved state of the art in measuring search success [15]. In this section, we describe the baseline approaches as well as the LSTM-based neural network architecture used in this study.

5.1 Baseline 1: Top N-Grams

The first approach we use involves the top N-grams. A set of 1 million abandoned queries was analyzed and the top 10 most frequent unigrams, bigrams, and trigrams were identified. Then, for any given training or testing query, the presence of each of those unigrams, bigrams or trigrams is identified by binary indicator in the feature vector. Unlike the other approaches used in this paper, the N-gram-based approach does not model the actual sequence of user interactions over time. Instead, it simply considers whether a short sub-sequence of actions occurs or not.

When N-grams are used as features, we make use of the Gradient Tree Boosting ensemble method as a classifier.

5.2 Baseline 2: Generative Model

The second approach we use is the generative model proposed in [15]. This model was originally used to measure search success from a sequence of actions in a session and thus included actions such as clicks, reformulations, and queries. By contrast, we only consider the interaction within a single abandoned query.

The generative model is a mixture model, which is composed of two components: good abandonment and bad abandonment. A

sequence of interactions is generated by the mixture model and, given a sequence of user interactions, the goal during classification is to determine whether that sequence was generated by the good abandonment component of the mixture model or the bad abandonment component. If $C = [c_g, c_b]$ is the set of classes corresponding to good and bad abandonment, respectively, then a sequence s_i is generated according to a distribution $P(s_i|C, \theta)$, where θ are the parameters of the probability distribution of the mixture model. Therefore, from [15], the likelihood of seeing any sequence s_i is given by:

$$P(s_i|\theta) = \sum_{c \in [c_g, c_b]} P(c|\theta) P(s_i|c; \theta) \tag{1}$$

If we assume that every interaction that a user takes is only dependent on the previous action, then:

$$P(s|c, \theta) \propto \prod_{j=1}^{n} P(a_j|a_{j-1}; c; \theta), \tag{2}$$

where n is the length of the sequence.

We therefore have the parameters $\theta_{a,a',c} \equiv P(a'|a; c; \theta)$, for each class c and each pair of sequential interactions (a, a'). The final parameters of the model are the prior class parameters, $P(c_g|\theta)$ and $P(c_b|\theta)$. By expanding Equation 1, the probability of generating any sequence of user interactions is given by:

$$P(s_i|\theta) \propto \sum_{c \in [c_g, c_b]} P(c|\theta) \prod_{j=1}^{n} P(a_j|a_{j-1}; c; \theta) \tag{3}$$

5.2.1 Learning the Parameters. The parameters $\hat{\theta}$ of the generative model need to be learned from the training data. To do that we follow [15] and use maximum a posterior (MAP) estimates with Dirichlet priors.

The action transition parameters are estimated as:

$$\hat{\theta}_{a,a',c} \equiv P(a'|a; c; \hat{\theta}) = \frac{1 + N_{a_i, a_j, c}}{|A| + N_{a_i, c}}, \tag{4}$$

where $|A|$ is the total number of interactions, $N_{a_i, a_j, c}$ is the number of transitions from interaction a_i to a_j in class c, and $N_{a_i, c}$ is the count of interaction a_i. The prior class probability parameters are given by:

$$\hat{\theta}_c \equiv P(c|\hat{\theta}) = \frac{1 + N_c}{2 + N}, \tag{5}$$

where N_c is the number of interaction sequences in the training data belonging to class c and N is the total number of interaction sequences in the training data.

After these parameters have been estimated, the conditional probability of a new sequence s_i is given by:

$$P(C|s_i, \hat{\theta}) = \frac{P(c|\hat{\theta}) P(s_i|c; \hat{\theta})}{P(s_i|\hat{\theta})}. \tag{6}$$

Finally, a sequence is classified as the most likely class given the sequence of actions as:

$$Prediction(s_i) = \arg\max_{c \in c_g, c_n} P(c|s_i, \hat{\theta}) \tag{7}$$

5.3 Long Short-Term Memory Recurrent Neural Network

As previously mentioned, we propose to use an LSTM-based RNN since it naturally fits the problem of sequence classification and has achieved state of the art performance in other domains.

A Recurrent Neural Network (RNN) is a neural network where the connections between units form a cycle, thereby enabling them to process sequences of data. This is in contrast to a feed forward network where the input data is of a fixed length. Long Short-Term Memory (LSTM) units are a type of unit that can be used in RNNs and that have become increasingly popular in the literature and in industry due to their state of the art performance, which often comes from their ability to better handle long sequences of data and long term dependencies [19].

A unit in a LSTM, is a recurrently connected unit that has, input, output and forget gates, as well as a memory cell. The effect of the gates is to modulate the incoming, outgoing, and historical signals. The memory cell captures the long term memory of the unit. Formally, we use an LSTM unit that receives x_t as input at time t and is defined by:

$$i_t = \sigma(\boldsymbol{W}_i x_t + \boldsymbol{U}_i h_{t-1} + b_i) \tag{8}$$

$$f_t = \sigma(\boldsymbol{W}_f x_t + \boldsymbol{U}_f h_{t-1} + b_f) \tag{9}$$

$$o_t = \sigma(\boldsymbol{W}_o x_t + \boldsymbol{U}_o h_{t-1} + b_o) \tag{10}$$

$$\tilde{C_t} = \tanh(\boldsymbol{W}_c x_t + \boldsymbol{U}_c h_{t-1} + b_c) \tag{11}$$

$$C_t = i_t * \tilde{C_t} + f_t * C_{t-1} \tag{12}$$

$$h_t = o_t * \tanh(C_t), \tag{13}$$

where i_t, f_t, and o_t, are the input, forget, and output gates. $\tilde{C_t}$ is the candidate value for the state of the memory cell and C_t is the final state of memory cell at timestep t. Lastly, h_t is the output of the unit. The $\boldsymbol{W}$ and $\boldsymbol{U}$ are the weight matrices for the current input and previous output, respectively, and the b are the bias vectors.

In this study we use an LSTM RNN as defined above to model the sequence of user actions. The input x_t to our model are word embeddings of the action labels. These embeddings were learned from over 10 million sequences of actions using the *word2vec* tool using the cbow method, a window size of 1, and an output dimension of 100. We also allow these vectors to be fine-tuned as part of the training process. The effect of this is that each action presented in Section 3.2 is represented as a 100 dimensional vector. When we feed data into this network, we feed this 100-dimensional vector for each timestep in the interaction sequence. The embedding layer is connected to a block of 32 LSTM units with the possibility of dropout. Dropout is an approach to preventing overfitting in neural networks by randomly dropping units and their connections in order to prevent unit co-adapting [37]. The output from the last timestep in the LSTM is presented to a standard feed-forward neural network that contains a single output neuron that uses the *sigmoid* activation function. The output of this layer is then used as the final prediction. To train the network, we make use of the Adam optimization algorithm [26].

Table 2: Performance of classifiers on abandoned data. 10,032 good abandonment queries; 11,230 bad abandonment queries. Bold results indicate the best performance for the metric. † indicates statistical significance in a Wilcoxon signed-rank test at the 5% level ($p < 0.05$) for the best performing classifier for each metric compared to the others.

Classifier	Accuracy	Good P	Bad P	Good R	Bad R	Good F1	Bad F1
Top N-Grams	0.6922	0.6730	0.7079	0.6761	**0.7065**	0.6744	0.7080
Generative Model	0.6095	0.5743	0.6520	0.6662	0.5589	0.6168	0.6018
LSTM	**0.7096**†	**0.6769**	**0.7445**†	**0.7356**†	0.6864	**0.7049**†	**0.7141**†

Table 3: Performance of classifiers on labeled abandoned data from Flights A-D. Bold results indicate the best performance for the metric. † indicates statistical significance in a Wilcoxon signed-rank test at the 5% level ($p < 0.05$) for the best performing classifier for each metric compared to the others.

Flight A							
Classifier	**Accuracy**	**Good P**	**Bad P**	**Good R**	**Bad R**	**Good F1**	**Bad F1**
Top N-Grams	0.6471	0.9857	0.0056	0.6526	0.1723	0.7853	0.0109
Generative Model	0.5955	0.9864	**0.0078**†	0.5992	**0.2747**†	0.7455	0.0151
LSTM	**0.7286**†	**0.9867**†	0.0057	**0.7354**†	0.1323	**0.8427**†	0.0109
Flight B							
Classifier	**Accuracy**	**Good P**	**Bad P**	**Good R**	**Bad R**	**Good F1**	**Bad F1**
Top N-Grams	0.6078	0.9743	0.0154	0.6153	0.2701	0.7542	0.0291
Generative Model	**0.6777**†	**0.9815**†	**0.0288**†	**0.6834**†	**0.4214**†	**0.8058**†	**0.0539**†
LSTM	0.6551	0.9753	0.0159	0.6642	0.2443	0.7902	0.0299
Flight C							
Classifier	**Accuracy**	**Good P**	**Bad P**	**Good R**	**Bad R**	**Good F1**	**Bad F1**
Top N-Grams	0.7352	**0.9101**†	**0.2175**†	0.7749	**0.4498**†	0.8371	**0.2931**†
Generative Model	0.7177	0.9060	0.1994	0.7570	0.4352	0.8248	0.2735
LSTM	**0.7542**†	0.8985	0.2012	**0.8117**†	0.3410	**0.8529**†	0.2530
Flight D							
Classifier	**Accuracy**	**Good P**	**Bad P**	**Good R**	**Bad R**	**Good F1**	**Bad F1**
Top N-Grams	0.6374	0.9487	0.0489	0.6534	**0.3351**†	0.7738	0.0853
Generative Model	0.6480	0.9424	0.0357	0.6703	0.2296	0.7834	0.0618
LSTM	**0.7076**†	**0.9511**†	**0.0547**†	**0.7296**†	0.2946	**0.8257**†	**0.0923**†

6 EXPERIMENTS

6.1 Classification Performance

We evaluate the performance of the classifiers presented in Section 5 on the dataset presented in Section 4.1. This dataset contain 10,032 samples of good abandonment, and 11,230 samples of bad abandonment and we used 10-fold cross validation for the experiment.

For the top N-gram features, we used a Gradient Tree Boosting Ensemble. During training, we perform a grid search on the cross validation training set in order to find the best values for the number of leaves, min samples to split, tree depth, and number of learners. For the LSTM RNN, we perform a grid search over the learning rate and dropout rate. When the dropout rate is set to 0, there is zero probability of units and their connections being dropped. We set the batch size during training to 128. Training LSTMs can be costly, therefore we employ early stopping as follows. For each training fold we retain 10% of the fold as a validation set. We then stop training when the validation loss does not change by a delta of 10^{-8} over 3 epochs with *logloss* as the loss function.

Table 2 shows the performance of the different classifiers. As can be seen from the table, the best performance is achieved by the LSTM, which reaches an accuracy of 70.96%, which is significantly better than the other methods. The next best accuracy of 69.22% is achieved by the Top N-Grams approach, while the lowest is achieved by the Generative Model. The LSTM approach also achieves the highest precision for both good abandonment and bad abandonment, doing significantly better than the other methods in each case. Of particular interest is the bad abandonment precision, which is 74.45% compared to the next best 70.79% by the Top N-Grams approach. The Generative Model has the worst performance of all the classifiers, while the Top N-Grams approach achieves the best recall for bad abandonment queries. Overall, the LSTM

Table 4: Flight Statistics

Flight	Size	Good	Bad
Flight A	150,353	148,655	1,698
Flight B	44,724	43,748	976
Flight C	2,407	2,113	294
Flight D	1,768	1,862	94

outperforms all other methods significantly on 6 of the 7 metrics. While these findings show that the LSTM-based RNN provides a good approach to modeling sequences of user interactions, a natural question that arises relates to how well it generalizes for use across different datasets. We address this question in the next section.

6.2 Generalizing Across Datasets

The previous experiment showed the LSTM to be the best performing classifier on the dataset described in Section 4.1. An interesting question that naturally arises is: how well do the different methods perform when evaluated on datasets that may be drawn from other distributions? This question naturally mimics the setting that occurs in production for a commercial search engine where the addition of new features, such as new answers, Web page summarizations, etc., lead to different user interaction behavior. As an example of this, consider the case where an answer triggers for the query *cikm 2017 dates* where it never triggered before. This hypothetical triggered answer may show the dates of the conference and its presence may lead to different scroll and mouse movement behavior compared to the case where the answer did not trigger.

To understand how well the different methods generalize, we design an experiment whereby we train the different classifiers on the data presented in Section 4.1. We then test the performance of these classifiers on independent datasets that were drawn independently of the training data. We describe the data in the next section and then present the experiment and its results.

6.2.1 Datasets. To test this scenario, we sampled queries from different online flights, where an online flight refers to an online experiment that isolates the firing of an experience to a specific set of users. For example, an online flight may be created for our hypothetical conference dates answer and only users on that flight will experience that feature. We consider four different online flights in a commercial search engine, hereby labeled A, B, C & D. The flights consist of changes to the search engine, such as: changes to the relevance ranker, and changes to the generation of snippets. We collect data from these flights the same way as described in Section 4.1 and perform the same dataset expansion. Table 4 shows the number of abandoned queries labeled for these flights as well as the number of good and bad abandonment queries. As can be seen from 4, for all of the flights, the number of good abandonment impressions vastly exceeds the number of bad abandonment impressions.

6.2.2 Approach & Results. In this experiment, all data from the dataset described in Section 4.1 was used as training data. Once again, a grid search was performed for the number of leaves, minimum samples to split, tree depth, and number of learners for the Gradient Tree Boosting Ensemble used with the Top N-Gram features. Similarly, for the LSTM RNN, a grid search is used to find the learning rate and dropout rate, with 10% of the training data being used for validation and early stopping. The trained model is then used to classify the labeled data from Flights A,B,C & D, which we repeatedly sample in order to allow us to test for significance. The results are presented in Table 3.

As can be seen from Table 3, the LSTM model achieves significantly better accuracy than the other models on three of the four flights. For flights A, C, and D, the difference in accuracy between the best performing LSTM and the second best classifier is 8.15, 1.9, and 5.96%, respectively. For flight B, where the generative model performed best by achieving an accuracy of 67.77%, the LSTM achieved the second best accuracy of 65.51%. These findings provide evidence that the LSTM model generalizes relatively well in terms of accuracy compared to the other approaches.

When measuring good abandonment, the LSTM achieves the best precision on two of the four flights and the best recall on three of the four flights. Generally, the precision for good abandonment is relatively similar for the different models, differing at most by just over 1% on flight C. However, for good abandonment recall, the magnitude of the improvement is larger with the difference in the recall between the LSTM and second best performing method being 8.28, 3.68, and 7.62% for flights A, C, and D, respectively. As with recall, the good abandonment F1 score is highest for the LSTM method for three of the four flights.

For bad abandonment, the difference between the precision for best and second best performing methods is relatively small at <2%, with the generative model achieving the best performance on two flights, and the LSTM and Top N-Grams model achieving the best performance on the remaining two. For bad abandonment recall, the LSTM never achieves the best performance, while the other two models achieve the best bad abandonment recall twice each.

Across flights the LSTM-based model achieved the overall best accuracy, good abandonment precision, good abandonment recall, and good abandonment F-score. This is consistent with the results presented in Table 2, where the training and testing data were drawn from the same distribution. Similarly consistent with Table 2, the LSTM-based model also achieved the worst overall bad abandonment recall. These findings in general reveal that the LSTM-based model generalizes quite well by having relatively consistent performance across datasets.

6.3 Discussion

We have presented two experiments whereby we used sequences of user interactions to differentiate between good and bad abandonment. This allows us to answer **RQ2:** *How well can sequences of user interactions be used to differentiate between good and bad abandonment?* We achieve a recognition of 71% with the LSTM-based model. Furthermore, we showed how this model is able to behave relatively consistently across datasets.

7 CONCLUSIONS

Measuring good abandonment is becoming an increasingly important problem as search engines become more focused on satisfying users without them needing to leave the SERP. In this study we

proposed to use sequences of user actions to differentiate between good and bad abandonment. We studied the difference between sequences of interactions for good and bad abandonment queries and showed that bad abandonment queries typically involve more user interactions and are more likely to contain scroll and dwell interactions. Based on this insight, we formulated good abandonment detection as a sequence classification problem. We proposed to use an RNN to classify the sequence of user interactions and showed a classification accuracy of 71%. Lastly, we also showed how this model was able to generalize well across datasets.

REFERENCES

[1] Michael S. Bernstein, Jaime Teevan, Susan Dumais, Daniel Liebling, and Eric Horvitz. 2012. Direct answers for search queries in the long tail. In *Proceedings of the ACM Annual Conference on Human Factors in Computing Systems*. 237–246.

[2] Ye Chen, Yiqun Liu, Ke Zhou, Meng Wang, Min Zhang, and Shaoping Ma. 2015. Does vertical bring more satisfaction?: Predicting search satisfaction in a heterogeneous environment. In *Proceedings of the ACM International on Conference on Information and Knowledge Management*. 1581–1590.

[3] Lydia B Chilton and Jaime Teevan. 2011. Addressing People's Information Needs Directly in a Web Search Result Page. In *Proceedings of the International Conference on World Wide Web*. 27–36.

[4] Aleksandr Chuklin and Maarten de Rijke. 2016. Incorporating Clicks, Attention and Satisfaction into a Search Engine Result Page Evaluation Model. In *Proceedings of the ACM International on Conference on Information and Knowledge Management*. 175–184.

[5] Aleksandr Chuklin and Pavel Serdyukov. 2012. Good abandonments in factoid queries. In *Proceedings of the International Conference Companion on World Wide Web*. 483–484.

[6] Aleksandr Chuklin and Pavel Serdyukov. 2012. Potential good abandonment prediction. In *Proceedings of the International Conference Companion on World Wide Web*. 485–486.

[7] Abdigani Diriye, Ryen White, Georg Buscher, and Susan Dumais. 2012. Leaving so soon? Understanding and Predicting Web Search Abandonment Rationales. In *Proceedings of the ACM International Conference on Information and Knowledge Management*. 1025–1034.

[8] Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, and Thomas White. 2005. Evaluating implicit measures to improve web search. *ACM Transactions on Information Systems* 23, 2 (2005), 147–168.

[9] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. 2016. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems* (2016).

[10] Qi Guo and Eugene Agichtein. 2010. Ready to buy or just browsing? Detecting Web Searcher Goals from Interaction Data. In *Proceeding of the ACM International SIGIR Conference on Research and Development in Information Retrieval*. 130–137.

[11] Qi Guo and Eugene Agichtein. 2012. Beyond Dwell Time: Estimating Document Relevance from Cursor Movements and other Post-click Searcher Behavior. In *Proceedings of the International Conference on World Wide Web*. 569–578.

[12] Qi Guo, Haojian Jin, Dmitry Lagun, Shuai Yuan, and Eugene Agichtein. 2013. Mining touch interaction data on mobile devices to predict web search result relevance. In *Proceedings of the ACM International SIGIR Conference on Research and Development in Information Retrieval*. 153–162.

[13] Qi Guo, Dmitry Lagun, and Eugene Agichtein. 2012. Predicting web search success with fine-grained interaction data. In *Proceedings of the ACM International Conference on Information and Knowledge Management*. 2050–2054.

[14] Qi Guo, Shuai Yuan, and Eugene Agichtein. 2011. Detecting success in mobile search from interaction. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1229–1230.

[15] Ahmed Hassan. 2012. A Semi-Supervised Approach to Modeling Web Search Satisfaction. In *Proceedings of the ACM International SIGIR Conference on Research and Development in Information Retrieval*. 275–284.

[16] Ahmed Hassan, Rosie Jones, and Kristina Lisa Klinkner. 2010. Beyond DCG: User Behavior as a Predictor of Successful Search. *Proceedings of the ACM International Conference on Web Search and Data Mining* (2010), 221–230.

[17] Ahmed Hassan, Xiaolin Shi, Nick Craswell, and Bill Ramsey. 2013. Beyond Clicks: Query Reformulation as a Predictor of Search Satisfaction. In *Proceedings of the ACM International Conference on Information and Knowledge Management*. 2019–2028.

[18] Ahmed Hassan and Ryen W. White. 2013. Personalized models of search satisfaction. In *Proceedings of the ACM International Conference on Information and Knowledge Management*. 2009–2018.

[19] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[20] Jiepu Jiang, Ahmed Hassan Awadallah, Xiaolin Shi, and Ryen W White. 2015. Understanding and Predicting Graded Search Satisfaction. In *Proceedings of the ACM International Conference on Web Search and Data Mining*. 57–66.

[21] Diane Kelly. 2009. Methods for Evaluating Interactive Information Retrieval Systems with Users. *Foundation and Trents in Information Retrieval* 3, 1-2 (2009), 1–224.

[22] Madian Khabsa, Aidan Crook, Ahmed Hassan Awadallah, Imed Zitouni, Tasos Anastasakos, and Kyle Williams. 2016. Learning to Account for Good Abandonment in Search Success Metrics. In *Proceedings of the ACM International on Conference on Information and Knowledge Management*. 1893–1896.

[23] Youngho Kim, Ahmed Hassan, Ryen W. White, and Yi-Min Wang. 2013. Playing by the rules: mining query associations to predict search performance. In *Proceedings of the ACM International Conference on Web Search and Data Mining*. 133–142.

[24] Youngho Kim, Ahmed Hassan, Ryen W. White, and Imed Zitouni. 2014. Comparing client and server dwell time estimates for click-level satisfaction prediction. In *Proceedings of the ACM International SIGIR Conference on Research & Development in Information Retrieval*. 895–898.

[25] Youngho Kim, Ahmed Hassan, Ryen W. White, and Imed Zitouni. 2014. Modeling dwell time to predict click-level satisfaction. In *Proceedings of the ACM International Conference on Web Search and Data Mining*. 193–202.

[26] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[27] Julia Kiseleva, Kyle Williams, Ahmed Hassan Awadallah, Aidan C Crook, Imed Zitouni, and Tasos Anastasakos. 2016. Predicting user satisfaction with intelligent assistants. In *Proceedings of ACM International SIGIR conference on Research and Development in Information Retrieval*. ACM, 45–54.

[28] Julia Kiseleva, Kyle Williams, Jiepu Jiang, Ahmed Hassan Awadallah, Aidan C Crook, Imed Zitouni, and Tasos Anastasakos. 2016. Understanding user satisfaction with intelligent assistants. In *Proceedings of the ACM International Conference on Human Information Interaction and Retrieval*. ACM, 121–130.

[29] Dmitry Lagun, Mikhail Ageev, Qi Guo, and Eugene Agichtein. Discovering Common Motifs in Cursor Movement Data for Improving Web Search. In *Proceedings of the ACM International Conference on Web Search and Data Mining*.

[30] Dmitry Lagun and Eugene Agichtein. 2015. Inferring Searcher Attention by Jointly Modeling User Interactions and Content Salience. In *Proceedings of the ACM International SIGIR Conference on Research and Development in Information Retrieval*. 483–492.

[31] Dmitry Lagun, Chih-Hung Hsieh, Dale Webster, and Vidhya Navalpakkam. 2014. Towards better measurement of attention and satisfaction in mobile search. In *Proceedings of the ACM International SIGIR Conference on Research & Development in Information Retrieval*. 113–122.

[32] Jane Li, Scott Huffman, and Akihito Tokuda. 2009. Good abandonment in mobile and PC internet search. In *Proceedings of the ACM International SIGIR Conference on Research and Development in Information Retrieval*. 43–50.

[33] Yiqun Liu, Ye Chen, Jinhui Tang, Jiashen Sun, Min Zhang, Shaoping Ma, and Xuan Zhu. 2015. Different Users, Different Opinions: Predicting Search Satisfaction with Mouse Movement Information. In *Proceedings of the ACM International SIGIR Conference on Research and Development in Information Retrieval*. 493–502.

[34] Rishabh Mehrotra, Imed Zitouni, Ahmed Hassan Awadallah, Ahmed El Kholy, and Madian Khabsa. 2017. User Interaction Sequences for Search Satisfaction Prediction. In *Proceedings of the ACM International SIGIR Conference on Research and Development in Information Retrieval*. 165–174.

[35] Daan Odijk, Ryen W. White, Ahmed Hassan Awadallah, and Susan T. Dumais. 2015. Struggling and Success in Web Search. In *Proceedings of the ACM International on Conference on Information and Knowledge Management*. 1551–1560.

[36] Yang Song, Xiaolin Shi, Ryen White, and Ahmed Hassan Awadallah. 2014. Context-aware web search abandonment prediction. In *Proceedings of the ACM International SIGIR Conference on Research & Development in Information Retrieval*. 93–102.

[37] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[38] Sofia Stamou and Efthimis N. Efthimiadis. 2010. Interpreting User Inactivity on Search Results. In *European Conference on Information Retrieval*, Vol. 5993. 100–113.

[39] Kyle Williams, Julia Kiseleva, Aidan C Crook, Imed Zitouni, Ahmed Hassan Awadallah, and Madian Khabsa. 2016. Detecting good abandonment in mobile search. In *Proceedings of the International Conference on World Wide Web*. 495–505.

[40] Kyle Williams, Julia Kiseleva, Aidan C Crook, Imed Zitouni, Ahmed Hassan Awadallah, and Madian Khabsa. 2016. Is This Your Final Answer?: Evaluating the Effect of Answers on Good Abandonment in Mobile Search. In *Proceedings of the ACM International SIGIR Conference on Research and Development in Information Retrieval*. ACM, 889–892.

[41] Zhengzheng Xing, Jian Pei, and Eamonn Keogh. 2010. A Brief Survey on Sequence Classification. *SIGKDD Explor. Newsl.* 12, 1 (Nov. 2010), 40–48.