Detecting Search Sessions Using Document Metadata and Implicit Feedback

Tomáš Kramár
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology
Bratislava, Slovakia
kramar@fiit.stuba.sk

Mária Bieliková
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology
Bratislava, Slovakia
bielik@fiit.stuba.sk

ABSTRACT

It has been shown that search personalization can greatly benefit from exploiting user's short-term context – user's immediate need and intent. However, this requires that the search engine must be able to divide user's activity into segments, where each segment captures user's single goal and focus. Several different approaches to search session segmentation exist, each considering different features of the queries, but it may be helpful to also consider user's implicit feedback on the search results clicked in response to the query. We propose a method for segmenting queries into search sessions which is based on document metadata and incorporates implicit feedback. Our approach also considers multitasking, where user shifts her current interest, but afterwards proceeds with the original task. We evaluated our approach on manually segmented query log and compared the results of our approach with results from other methods and showed that using implicit feedback can improve the performance of the segmentation task.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Query formulation, Selection process, Search process

General Terms

Information retrieval

Keywords

personalization, search, short-term contexts, search sessions, search session segmentation

1. INTRODUCTION

Web contains an ever-growing amount of documents. Accessing these document poses great difficulties, especially after the rise of Web 2.0 where users have been given the ability to create the content, which allows for more social-based approaches to personalization [1], but also contributes to information overload. According to Technorati1 , a service which tracks user-generated media, the content on the Web is growing with a pace at 2 blog posts per second, and this number does not include the growth of other content.

Search engines play a crucial role in accessing this amount of content. Users interact with search engines by entering few keywords, which describe their intent and expect the machine to provide a list of relevant documents. This model has several known disadvantages:

- the number of keywords is usually low, typically 1-3 keywords [17] and this often leads to ambiguity and unclear intent;
- many of the words are ambiguous; a word "jaguar" can refer to an animal, a car and even has less-known meanings such as a game console or German battle tank;
- the queries are almost never accurate [9], they are either too generic or too specific, but almost never exactly aligned with the specific intent the user has in mind.

The combined impact of these problems leads to a conclusion that finding the relevant document when we do not have enough information about it is indeed a difficult task, both for the user and the search engine.

To mitigate this problem, several approaches to search personalization have been researched, each with the ultimate aim to help users find the relevant content, without trying to change how humans think, or work.

There is relevance feedback, query expansion, search intent detection, alternative ranking schemes and many others. These techniques leverage and act upon some form of search context. Generally, the term context refers to attributes of the environment [7], such as location, time, or weather, but in the domain of personalized search the term is commonly used to describe user's needs, goals and intents (e.g. [21, 27]). Based on the time span that is used to build the search context, the context may be long, or short-term.

Long-term search context is composed of the goals and intents that can be recognized by observing the complete user activity, beginning with the first known information about the user and her activity.

Short-term search context is composed of the goals and intents that the user has in the moment of search. These represent the current focus and are obtained by observing the user activity beginning in a recent point of time.

To be able to use short-term search context a personalization system must know the exact moment the user changes her intent, so that it can start and use a new context. The task of detecting this change is referred to as search session detection (segmentation). The term search session was never formally defined in the literature and its meaning differs in different works. In this work, we assume that search session is a sequence of search related actions with the single underlying informational intent, similarly to [22].

The goal of search session segmentation is to partition the stream of user queries into segments of queries, where each segment is the search session, i.e., holds the condition that all queries that it contains are related to a single underlying goal.

Several existing approaches to search session segmentation exist, but they have various disadvantages. When a segmentation approach acts solely on the features provided by the query itself, the amount of understanding of the underlying intent is quite limited. Therefore, many approaches also consider the documents that were clicked in response to the query, but these approaches do not evaluate user's feedback that is implicitly left in each document. Many existing approaches also do not consider interruptions in Web browsing and multitasking (i.e. having multiple intents).

In this work we aim to contribute to the area of search session segmentation by matching the queries with the metadata of the documents clicked from the search results to get better insight into the purpose of the query by aggregating more data than only the query itself provides. We also evaluate the level of page usefulness for the particular query by collecting and analyzing the implicit feedback indicators that the user provides for each page view. Our approach also considers user interruptions and is able to separate intermingled sessions and reconnect interrupted sessions.

The paper is structured as follows. Section 2 describes the related work done in the area of search session segmentation. In Section 3 we describe our data collection methodology. Section 4 describes the proposed method for search sessions segmentation. Experimental results are described in Section 6.

2. RELATED WORK

The most widely used approach to search session segmentation is to compare temporal distance of the queries. If two queries were issued with a time difference larger than a predefined threshold, it is assumed that a new session started, and the existing session is split at that place. This technique was first described in [4], establishing the threshold of 25.5 minutes. They measured an average time between page requests (9.5 minutes) and added a 1.5 standard deviation. This approach only makes sense for normal distribution, while surfing on the Web shows properties of a long-tail distribution. The established 25.5 minutes are part of the long-tail distribution and it wouldn't make much difference if 20, or 40 minutes were used instead [14]. He et al. [13] experimented with various cutoff settings and established that the optimal cutoff time is between 10 and 15 minutes. Due to its simplicity both in concept and implementation, this technique is widely used and there are modification ranging from use of 5 minutes cutoff [8], through 30 minutes cutoff [24] to a per-user cutoff [23].

- it is unable to detect sessions which are split within a short time – if the search intent changes quickly, without sufficient time passing between consecutive queries, the queries are falsely considered as part of a single session, yet they might bear a different underlying intent;
- the long time between searches may not mean that the user's intent changed – this is a well known downside of any time-window based method. The longer pause between queries may be caused by several factors, such as reading long article, breaks, or any other interruptions that are commonly encountered nowadays [5].

Another approach uses lexical distance of the queries. This idea compares content of two queries to detect if the intent has changed. Example of this approach is stated in [16]. The main drawback of this method is that it leads to a high amount of false positives. There are many instances where two queries are completely dissimilar (they share no common words), yet their underlying intent is the same. Consider a user searching for "IR" and "information retrieval" afterwards. Using the lexical distance approach would incorrectly yield two separate sessions.

Method described in [11] combines both temporal and lexical distances. They use vector-space representation, where each pair of following queries is placed in the space. If the query pair fits into the space bounded by the subplane delimited by two edge cases

- two parallel, but dissimilar queries,
- the same queries, but executed long time apart

it is considered an extension of the current session. Although this combination can achieve better results than each of the methods alone, it is still prone to the aforementioned problems.

As stated, related queries do not provide sufficient data to match similar intents, but there are approaches that use signals from the retrieved documents. In [25], authors used vectors of titles and snippets of 50 top-ranked documents for the query. The session is split as determined by comparing cosine similarities of two following queries. Another approach uses document keywords. In [6], authors extract document's keywords using the TF.IDF scheme and map

them to ODP categories. The session is then split when the ODP category changes. This approach however fails to account for user's real interests. When the user enters a query and clicks and views some documents only to realize that the results are wrong and the query needs to be reformulated, this approach has already used these faulty results to decide upon session segmentation.

Jones and Klinker [18] trained a binary classifier to detect whether two queries are part of the same search task or not, using a set of syntactic, temporal, query log and web search features. They have obtained best results by incorporating a vector derived from the first 50 results for the given query, but this approach again suffers from the query ambiguity.

Currently, the arguably best approach is grouping related queries into sessions by using a clustering methods. The key feature of the approach is wikification of the queries, i.e., building a relevance vector in a high dimensional concept space of Wikipedia articles [22]. The relevance vector describes relevance of the query terms for each Wikipedia article. Although this approach gives good result on the manually annotated set of circa 1300 queries, it is strongly dependent on the Wikipedia articles. Given a query which does not contain a term present on Wikipedia, this approach does not yield any benefit over other methods. Our method is not dependent on other external and limited knowledge base, instead, it depends only on clicked search results which follow naturally after every search.

Most of the existing approaches also do not deal with multitasking. Multitasking might be present in several forms, mainly:

- parallel browsing, when a user is having multiple goals at the same time and working on them at the same time by means of having multiple browsers, or multiple browser tabs, or
- browsing with interruptions, when a user is having a single goal but is interrupted, switches the goal for a while and returns back to the original goal afterwards. The interruption may not be forced, it may be the user's conscious decision to abandon the current task and concentrate on different goal.

The practical effect of multitasking on search session segmentation is that it is causing the sessions to be disconnected. Single search session is interrupted by queries that are part of another session and then the session continues, as the user returns to the original goal.

There are mixed reports on the presence of multitasking on the Web. According to [15] only 25When searching, the multitasking is practically nonexistent for navigational and transactional queries [28], it is only present in 6-15that only 1et al. analyzed AOL query log and report that multitasking in search exists [22].

In [3] a method to detect disconnected sessions is proposed. Their approach is executed in two steps: in the first phase, sessions are segmented using a cutoff time, in the second step, all queries in each session are compared to build a similarity graph where the transitive closure is computed. All queries connected by the closure are considered part of the same session. This approach again combines the disadvantages of the approaches it merges – temporal and lexical distance. In [22] the queries are connected using clustering methods, so the multitasked queries are connected naturally.

Our work extends and differs from existing research in several important ways

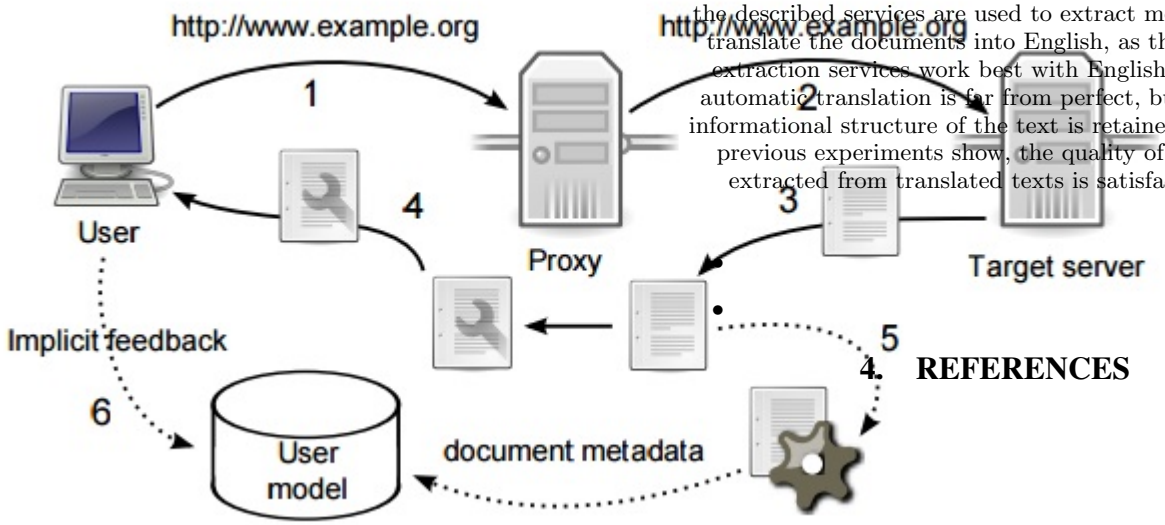
- First, we only consider the documents that the user deemed useful. To measure the usefulness, we partially rely only on the clickthrough data and only consider the search results that the user actually viewed, but we also use implicit feedback indicators to weight the contribution of each of the documents.
- The search sessions are segmented by comparing their metadata (both machine and human extracted), and to increase the chance of a successful match, we extend the metadata with ConceptNet relations.
- Our approach also considers multitasking and user interruptions – we maintain a stack of short contexts (search sessions) and when the query fits within a previously abandoned session, this session can be popped from the stack and continued.

3. DATA COLLECTION

Our work is based on the idea of considering only the documents that the user found useful, therefore we need a way to collect user's actions on search engine and implicit feedback on viewed documents [10]. In order to gain detailed insight into users actions and to collect precise and rich implicit feedback data, we leverage a personalized proxy server, called PeWe Proxy. The collection process is described in more detail in Fig. 1.

The personalized proxy server2 acts as a regular proxy, sitting between the user and the Web [20]. All user's requests are handled by the proxy first and then forwarded to the target server. Similarly, the response from Web server has to pass through the proxy. This means that the proxy has access to complete message processing between client and the server and can even modify the actual content of the request/response.

We are aware, that using proxy server in a production search engine is not a realistic option. We are using proxy server to gain detailed insight into implicit feedback on every accessed page. Some level of implicit feedback can be inferred by analyzing timestamps and patterns in query logs [26].



3.1 Implicit feedback collection

We use the proxy server to inject a tracking JavaScript into each page, which collects the following implicit feedback signals:

- *real time spent on page* – the actual time spent on page is measured in series of short time windows. If there is an observed activity within the page (i.e., mouse movement, scrolling, clicking, writing) during the span of the window, the length of the window is added to the total time on page. In this case, we used the window of 4 seconds;
- number of clicks, scrolls and copying into clipboard;

3.2 Metadata collection

Besides the implicit feedback collection, we use PeWeProxy to create logs of user activity, i.e., we track the documents that the user visited and to associate each log with its corresponding implicit feedback signals.

Each document is processed and following types of metadata are extracted:

- *keywords* – we extract keywords automatically by using the tagthe.net and Alchemy Orchestr8 Web services and JATR library;
- *tags* – we use human created directory of tags, the Web service delicious.com;
- *named entities* extracted by using *OpenCalais* Web service;
- *categories* from the human-maintained project ODP

All metadata is extracted by using a wrapper Web service Metall3 , which strips the document from auxiliary content (ads, menus, header, footer, etc.), leaving only the main text of the document, which is translated into English and the described services are used to extract metadata. We translate the documents into English, as the available extraction services work best with English text. The automatic translation is far from perfect, but the basic informational structure of the text is retained and as our previous experiments show, the quality of metadata extracted from translated texts is satisfactory [2].

4. REFERENCES