Search Engine Optimization for Threaded-Conversations

Ashish Pathak, Steven P. Crain
Department of Computer Science,
State University of New York at Plattsburgh
December 2014

apath001@plattsburgh.edu

Abstract

Online discussion communities are becoming increasingly popular among web users, where an extensive amount of discussion and commenting takes place. However, it is difficult to search these conversations as search engines are not optimized for the conversation-structure of online communities. In this paper, we purpose a method of ranking search results based on the conversation-structure and using it to optimize a domain-specific search engine. To solve this problem, we design several extension to a general-purpose search engine, focusing on maintaining the conversation-structure throughout the extraction, calculating and ranking of search results. Results from our experiments show that users can find optimized search results quicker as it is ranked higher than keyword based results.

1 Introduction

In recent years, we have seen the internet progressing as a community for people that share common interest. From social networks to online newspapers, discussions between users who share knowledge, ideas and suggestions have become very common.

For example, TuDiabetes, an online community of people affected by Diabetes has over 30,000 registered members. In their active web forums, extensive amounts of discussion and commenting take place, which helps members support, learn and share the steps of healthy living with each other. Typically, these conversations carry on over a certain period and the structure that various posts create after several replies to the main discussion or other posts forms a threaded structure.

Threaded conversation is a hierarchal arrangement of discussion and comments. Commonly used in an online discussion forum, threaded conversations maintain a structure that is very useful for determining the importance of an individual post. Here are some basic concepts of a threaded conversation structure:

Groups: On the highest level, groups are the sections of the community that mainly includes discussions on similar topics. Users of TuDiabetes can join a group as a member. Groups in TuDiabetes allows members to post quick messages in an informal area called wall.

Discussions: Within a group, user can start discussions, where they can post an article, question or start a dialogue related to the main topic of the group.

Post: Posts are the lowest level of entry, where users continue the conversation of the discussion by replying directly to the original discussion or to another post in the discussion.

Threaded conversation are hard to recreate when users searches a specific query. Since exact words that matches the query are not always mentioned in each post on a thread, many relevant parts of discussions are not returned to the users. In addition, general-purpose search engines (SE) do not maintain the thread structure in their results, as it falls just out of scope for the task of a general-purpose SE. In most cases, it does know anything special about the page or its structure. These results in losing data that thread structures retain which, if used could benefit users in choosing the correct result.

General-purpose SE poorly maintain the integrity of the conversation structure. In case of TuDiabetes, its default SE provided by the Ning platform is not optimized for threaded conversations, which makes it difficult to keep track of and access specific contents.

For example, the current SE displays all search results (including images and videos) in a single location without filtering ability. Results that are text based do not specify if it is a group, discussion or an individual post. This results in a non-intuitive display of results that affects the overall user experience. Use of platform-specific SE is common because

web-forum/discussion platform such as Ning comes with a built-in search tool. Additionally, building a custom SE can be labor-intensive and its accuracy varies depending on the maintenance and the amount of indexed data [6].

This problem leads many users away from the platform-specific SE such as Ning to other general-purpose SE such as Google or Bing, which are still inefficient, as they do not produce precise results while searching on a specific domain. In contrast, domain-specific SEs can leverage characteristics of the domain to provide a more satisfying user experience [6]. This paper purposes a domain-specific SE for an online community that can be optimized to present results to intuitively reflect the thread structure of a discussion forum.

Developing a domain-specific SEs requires precise crawling of documents on the domain. A web crawler is a script that traverses through the web, adding it to the indexing database of the search engine. While a general-purpose SE indexes each web page as a single document, in discussion communities this method does not work as every page is dynamically created. The domain-specific web crawler indexes each individual post, discussion and thread with its own permalink, allowing us to keep track of individual posts and it's URL.

Thread structures are the relation between the individual posts in a conversation. Each post can have a parent that is another post or a discussion, while each discussion has a parent group. This hierarchal structure of conversation allows us to rank results of a query even when the query terms are missing from the post or a discussion. For example, in the discussion topic "Pumps to Shot" a user replies, "Is it ok to do in your arm?" The reply does not have any of the query terms in its content, due to which a query based SE would have ignored this post. However, if there are several replies with the keyword to the user's comment, it becomes relevant.

User Interface plays a huge role on how a user selects a specific link on a set of results. Development of a domain-specific SE allows us to reflect the thread structure while displaying query results. A user searching for a discussion might find a reply to the discussion more important, which can be displayed directly in the SE interface allowing the user to discover relevant results quicker and with fewer mouse-clicks [2].

This paper presents a thread-structure extension to the way a domain specific SE produces search results, which will improve the users experience while searching for a topic on a discussion forum. In Section 2, we review various research related to this topic. We then in Section 3, introduce the method and implementation of SE optimization for threaded-conversation. In Section 4, we evaluate our system using real world data and demonstrate its effectiveness. Finally, we analyze the results and conclude in Sections 5 and 6.

2 Related Works

Domain-specific SEs have been widely studied and implemented over the years. General-purpose SEs have options that allows users to search within a specific domain such as Google Site Search or with query option *"site: URL."* Most of them implement page-level indexing, which cannot be used to solve our problem [5] [4] [7] [6].

Object-level web indexing are methods that return individual results for a query terms that a page-level SE would ignore. Common implementation of object-level indexing is found on platform-specific SE of online community platforms such as Ning, used in TuDiabetes.org. Currently, implementation of object-level SE uses the platform database to retrieve the objects, which is not always available to SEs. In addition, object-level implementation of web results do not integrate the threaded-conversation structure or display results using threaded-structure. [7] [10] [2].

Along with object-level indexing, various studies show that use of personalized search results also contribute towards optimization of a domain-specific search engine [9] [1].

Common analysis of structure of threaded-conversation is found in methods used to detect and track topics in discussion forums and news. Implementation of object-level SE with analysis of the conversation structure allows us to solve the problems an online-community user faces [10] [8].

3 Search Engine Optimization for Threaded Conversation

3.1 Overview

Methods used to optimize the Search Engine for threaded-conversation are discussed in this section. Figure 1 demonstrates the structure of our domain-specific Search Engine. The SE uses a web crawler that copies object-level web content and conversation structure to an indexing database. The data is indexed into an Apache Solr Search Engine (`http://lucene.apache.org/solr`), which returns query-based search results to a users query. In post-processing query-based results from SOLR are com-

pared to the conversation-structure and re-ranked based on topic relevance. Re-ranked results are populated with relevant posts from the conversation structure. These results are returned to the user in a threaded-conversation optimized user interface.

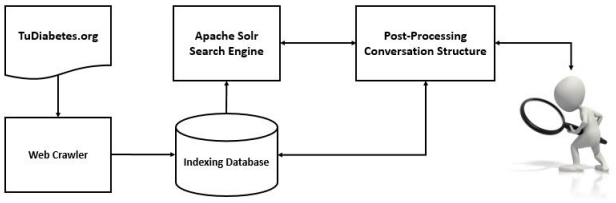

Figure 1: Structure of the domain-specific SE

3.2 Preliminary User Study

The first phase of this study was to conduct open-ended interviews with users of TuDiabetes to help us understand their search and information needs.[1] The study was conducted using video-conferencing, telephone and in-person interviews.

During the interview, we asked the subjects following questions while they used TuDiabetes.org on their personal computer:

1. How long have you been using the internet?
2. Have you used TuDiabetes.org or EsTuDiabetes.org
3. In what ways have you found TuDiabetes helpful?
4. How often do you participate in each of the following activities on TuDiabetes?
   Read a blog, write your own blog, read a forum, reply to a post in a forum, start a new discussion in a forum, follow activity in a group, actively participate in a group, events, view pictures or videos, post pictures or videos, search for information and content.
5. Describe your experience while searching on TuDiabetes. What kinds of things do you look for? How do you find them? Why do you use that method?
6. Have you used the search box in TuDiabetes? How well did it help you?
7. What are some things you would like to be able to do in TuDiabetes search?

We interviewed four TuDiabetes users; their response for question (4) is described in table (1) using scale 1 = Never to 10 = Daily.

Users/Activity	**1**	**2**	**3**	**4**
Read a blog	10	5	5	6
Write your own blog	2	3	1	1
Read a forum	10	10	5	8
Reply to a post in forum	10	10	2	6
Start to a new discussion	3	6	2	6
Follow activity in a group	8	6	5	5
Actively participate in a group	9	10	2	5
Events	8	10	10	8
View pictures or videos	10	8	8	6
Post pictures or videos	10	8	1	2
Search for information and content	10	10	3	10

Table 1: Evaluation of each activity by subjects

From the preliminary user study, it was found that most users find searching on the TuDiabetes.org difficult. Using the Ning SE, even simple query-based search did not return relevant results and did not have any options for filtering or advanced search. Results were presented in random order with images, videos and textual content all being displayed together in a non-hierarchal method. Some users mentioned that results were not always accurate. On average, only half of the result were something that they were looking for. The majority of the subjects mentioned they used alternative search engine such as Google to search for content on TuDiabetes.

The user study allowed us to evaluate feature that were going to be useful for TuDiabetes users. Most people wanted well-structured results that clearly identified the type of content for each result. Users also wanted the ability to separate different types of content using filters and separation of textual and multimedia contents. These suggestions influenced several user interface improvements.

Through web analytics, we discovered that groups and discussion forums were the most important aspects of TuDiabetes, yielding about 42% user traffic, which corroborates with the response from users in the study.

This study allows us to learn that TuDiabetes users are very active and find discussions with other users valuable. Most of the discussion takes place in TuDiabetes group and discussion, and blogs written by users are very popular. While searching for content, users

[1]This study was conducted under approval of the Committee for the Protection of Human Subjects #1259.

like to use general-purpose SE instead of the default search function provided through TuDiabetes.org.

3.3 Implementation

The web-crawler traverses through individual posts, discussion and groups continuously. To keep the traffic congestion low, the crawler retrieves each page in intervals of 20 seconds. Using the consistent structure of the thread, we are able to extract parent-child relation and the details of groups, discussions and posts.

- Each post is stored individually in an indexing database with its own unique id. It also stores its parents id and id of the discussion it belongs to. TuDiabetes has permalink for each individual post, which is also stored along with its content.
- Each discussion is individually stored with a unique id. It also stores the title and content along with the id of group it belongs to and the number of views.
- Groups are individually stored with a unique id, its title and number of members who subscribe to it.

After crawling, the contents from the indexing database are indexed in Apache Solr (`http://lucene.apache.org/solr`), an enterprise-grade open source search platform built on Apache Lucene. Solr provides familiar SE features, including reliable and precise results, text-normalization, stemming and query suggestions.

When a user enters a query, Solr processes the query and returns a set of documents with scores (Solr score) that reflect how well they match the query. These documents correspond to individual groups, forums, discussions or posts in TuDiabetes.

3.4 Post-Processing

Documents returned through Solr are very useful for finding posts that match the exact words in the query. However, a discussion may very well be relevant even though the opening post of the discussion does not contain the keywords. However, this can be overcome by making use of the thread structure in a post-processing step.

A post that Solr returns (matching exact keyword from query) increases the relevance for its parent post to the query and likewise for its parent discussion and parent group. This is achieved by keeping track of each post and its parent threads.

To determine a score for each parent thread p of a Solr result r based on initial Solr score r_{child}, we first assign height to the parent thread based on the discussion tree, a set of parent-child relation for each post, discussion and forum. Then the score for each parent p_{parent} based on its height is assigned as:

$$p_{parent} = \sum_{child} (m^{height(parent,child)} . r_{child}) \quad (1)$$

To get a reasonable sample, Solr returns 200 documents that are distributed between groups, discussions and posts ordered by individual scores. Each document is processed using equation (1), where $m = 0.9$, including the initial document r. Parent information and their height for r is stored in a tree table. Uploading the initial Solr results and equating scores based on the tree table, we can re-rank results in SQL. This architecture allows multiple different re-ranking algorithms to be tested side-by-side. The set of new documents are populated with the initial Solr results out of which 50 documents are returned to the user as result ordered by the new score.

4 Experiment

In order to evaluate the performance of our proposed method, we conducted experiments reported in this section.

Our experiment data is collected from the community forum of TuDiabetes.org. All posts until Dec. 5, 2014 is indexed by our web crawler. Each document extracted from the web page is evaluated as a post, discussion or forum based of the page URL. During which the title, content, views, members, URL, posted date and time and the thread relation is also extracted. After which, the document is stored in our indexing database then indexed periodically to Apache Solr. As of Dec. 5, 2014 there 450 groups, 11,791 discussions and 125,451 posts stored on the indexing database.

We conducted the experiment between Dec. 1 and Dec. 5 using four different Search Engine:

1. Proposed Search Engine (based on Threaded-Conversation and Solr result)
2. Google site search (using google query attribute site:TuDiabetes.org)
3. TuDiabetes.org default Search Engine (based on Ning community platform)
4. Apache Solr search engine (based on query keyword)

The proposed Search Engine (1) evaluates the effectiveness of the proposed model. Whereas, using Apache Solr (4) we can evaluate the effectiveness of the query keyword only based result. Since Google site search (2) is the "goto" solution for TuDiabetes users, it is valuable to compare results to the SE. Ning SE *(not the official title)* (3) is the in-house SE for TuDiabetes.org and the only SE with access to original content database, while all other SE (1, 2 and 4) make use of web crawler to extract documents.

Initially 10 document were selected at random from the indexing database. From each document, we selected four keywords (a, b, c and d). Each keyword is queried in the Search Engine (1, 2, 3 and 4) using query patterns,

(a) a

(b) a b

(c) a AND b

(d) a b c

(e) a AND b AND c

(f) a b c d

(g) a AND b AND c AND d

"AND" is a Boolean supported by most search engines.

For every document, the order of SE is reversed to normalize learning. For every query pattern, time t in seconds, valid mouse-clicks m, pagination pg and result index c is recorded. Paginations are result distributed in dynamic web pages, for which in SE (1 and 4) there were eight results per page, SE (2 and 3) had 10 results per page.

5 Results

In this section, we report the results of our experiment. We begin by comparing basic statistics pertaining each Search Engine and then present our experiments result.

5.1 Time-based Evaluation

This evaluation describes the time (in seconds) it took to find a specific document on different Search Engines. In figure (2), the average time it took a user to find the result in each SE on each query pattern is displayed. We observe that the numbers of query words sharply decreases the time a user takes to find the specific result.

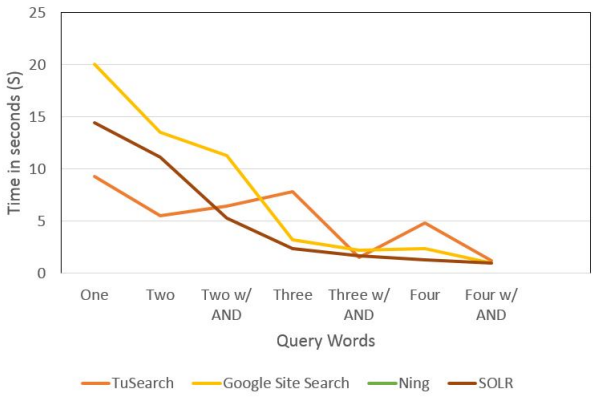

Figure 2: Results of Time-based Evaluation

5.2 Order-based Evaluation

This evaluation describes the order the result is displayed in each SE. We measure the ranking quality using Discounted Cumulative Gain (DCG), which allows us to measure the effectiveness of our SE algorithm. DCG is calculated under the assumption (1) that highly relevant documents are more useful when appearing earlier in search engine result list and (2) highly relevant documents are more useful than marginally relevant documents, which are in turn more useful than irrelevant document [3].

For this test, where we seek exactly 1 relevant document, the DCG is,

$$DCG_p = \frac{1}{log_2(i+1)} \tag{2}$$

where, i is the index in list of results where document is located [3].

In figure (3), we present the average DCG for each SE based on the experiment conducted on four SEs. We can evaluate that initially with one query word, TuSearch presents the most relevant results first, compared to other SEs. As we increase the number of words in the query, the score for each SE increases with SOLR getting the peak score while using four query words.

We can further observe that our experiment setup favors SOLR based SE, however, in practice, the relevance of document is not just based on the keyword of query but on its context and the information the user is searching for.

Our results show that use of object-level indexing and conversation-structure based re-ranking produces result that a user will find helpful. However, since our experiment favors keyword based search, we further need to evaluate relevance of a result, based on

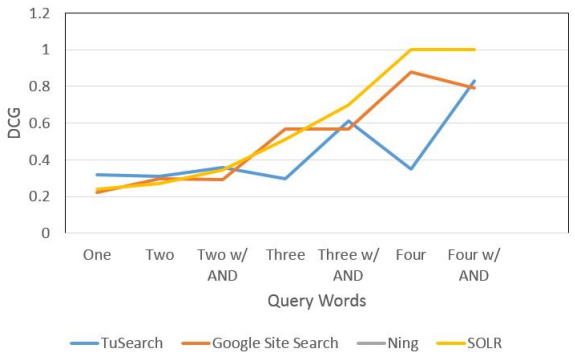

Figure 3: DCG of Order-based Evaluation

whether a user is looking for specific document or general information on the topic.

6 Conclusion and future work

In this paper, we have purposed a method of ranking search results and using it to optimize a domain-specific SE. Our evaluation shows us that discussion and comments that takes place in web forums do not always use words directly related to the topic. However, such posts can be useful when a user enters a query in a SE looking for information regarding that topic. Our method uses a web crawler to extract post information using object-level index and information about a post's relation with other posts, discussion or a group. We also re-rank a keyword based search result using the information obtained from the conversation that takes place in the discussion forum.

Our experiment shows that using this method we can get the information user is looking for higher in the list of results provided to the user. However, users are not always looking for a specific content instead browsing the SEs to find more information regarding a specific topic. This behavior based on individual preference is not evaluate in our experiment. In future, we will be conducting a separate user study to evaluate results based on user looking for general information on the topic rather than a specific document from the site.

References

[1] Carsten Eickhoff and Kevyn Collins-Thompson. Personalizing atypical web search sessions. In *WSDM 2013*, 2013.

[2] Marti A. Hearst and Jan O. Pedersen. Reexamining the cluster hypothesis: scatter/gather on retrieval results. In *SIGIR*, pages 76–84, New York, NY, USA, 1996. ACM Press.

[3] Kalervo Jarvelin and Jaana Kekalainen. Cumulated gain-based evaluation of ir techniques. In *ACM Transaction on Interfmation Systems*, 2002.

[4] Rong Jin, Hamed Valizadegan, and Hang Li. Ranking refinement and its application to information retrieval. In *WWW2008*, 2008.

[5] Xian Lin, Howard D. White, and Jan W. Buzydlowski. Associative seasearch and visualization. In *SSGRR 2001*, 2001.

[6] Andrew Mccallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. A machine learning approach to building domain-specific search engines. In *16th International Joint Conference on Artificial Intelligence*, pages 662–667, 1999.

[7] Zaiqing Nie, Ji-Rong Wen, and Wei-Ying Ma. Object-level vertical search. In *CIDR 2007*, 2007.

[8] Anish Das Sarma, Alpa Jain, and Cong Yu. Dynamic relationship and event discovery. In *WSDM 2011*, 2011.

[9] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR 2005*, pages 449–456, New York, NY, USA, 2005. ACM.

[10] Mingliang Zhu, Weiming Hu, and Ou Wu. Topic detection and tracking for threaded discsssion communities. In *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 2008.