University of Essex at LogCLEF 2011: Studying Query Refinement

M-Dyaa Albakour and Udo Kruschwitz

School of Computer Science and Electronic Engineering
University of Essex
Colchester, CO4 3SQ, UK
{malbak,udo}@essex.ac.uk

Abstract. This paper describes the analysis we performed for the query success task of LogCLEF 2011. In particular, we address the issue of query refinement. The motivating assumption of our work is that query success can be improved by a system that can make good query refinement suggestions. We investigate how log files as provided in LogCLEF can assist in learning good suggestions.
We used the distributed search logs of *Deutscher Bildungsserver (DBS)* and the logs of *The European Library (TEL)*. We first processed the logs to extract the actually submitted search queries together with user session information and the browsing information following the submission of a query. Our initial analysis shows that a large proportion of the sessions on DBS in particular resulted in reformulations of the original query.
The focus of our work is to demonstrate that the given log files can be used to acquire structured knowledge that can assist users in searching the collections (and thus shortening the number of steps needed when compared to a system that does not employ query modification suggestions). We use the paradigm of ant colony optimisation to derive query suggestions and evaluate the results by applying the fully automated AutoEval methodology that relies entirely on the search logs.

Keywords: Search Log Analysis, Query Suggestions, Automatic Evaluation

1 Introduction

LogCLEF provides a platform for researchers to conduct analysis on the search logs of live search engines to understand user search behaviour and study the exploitation of search logs as implicit source of feedback. LogCLEF provides a standard evaluation resource which makes systems comparable and makes research more transparent, in particular because large search engines do exploit their search logs but do not grant public access to them.

This year LogCLEF tries to tackle three different tasks, (a) language identification, (b) query classification and (c) query success. In this paper we study a sub-task of the third task which is query refinement. We perform an analysis

on two different datasets provided for LogCLEF 2011, namely the DBS and the TEL datasets. We then apply *Ant Colony Optimisation* as an adaptive learning approach to build knowledge structures from the query refinements in the logs. These knowledge structures can be exploited to aid users in finding what they are looking for and hence accelerate the success of user searches. We evaluate these structures with the *AutoEval* evaluation methodology which relies entirely on the logs and does not involve users [1]. We show that our learning approach can learn better query suggestions over time across different languages and we discuss the differences between the two datasets used in the light of the result of the analysis and the evaluation.

2 Log Files

In this section we described the datasets used in the experiments and present the initial analysis of the datasets. We also highlight the main initial findings and the differences in those datasets.

2.1 DBS Logs Analysis

The German Web site *Deutscher Bildungsserver* is a clearing house for educational resources on the Web. The logs distributed for the LogCLEF task are general logs in a standard format and cover a period of three months from September to November 2009. These logs record all user interactions with the Web site including queries that have been submitted to their search engine and the results that have been viewed upon issuing the queries. Each record in the log represents a request to the server that contains the user session information together with a timestamp of the request, the destination and source URLs and browser information. We processed the logs to extract the queries submitted to their search engine together with the user session information and the browsing information upon submitting the query. These steps were followed to process the logs:

1. Discard all records that are not search-related. Search-related records have the fields 'metasuche' or 'qsuche' in the URLs.
2. Discard all records that are requests from search engine bots (Google, MSN, Yahoo!).
3. Order records by their session ID and their timestamp.
4. Extract the queries and the actions following each query in each session. The queries are passed in the 'feldinhalt'' parameter of the URL.
5. Case-fold all queries and URL-decode the queries as they are a parameter in the URL.

To illustrate how this is done, let us consider the entries of the logs in Figure 1. All these entries represent HTTP requests to the server and are within one user session identified by the first string hash. They can also be ordered by their

timestamp. The first two entries (1) and (2) are discarded as they are not related to search. The third entry is a search request where the query can be extracted from the URL parameters 'schulpflicht'. The fourth entry represents a request that follows the search but is discarded as it is a browser request to a script file. The fifth entry is another search request with a query 'schulpflichtiger Kinder'. Entry (6) is a click request on the results displayed for the query 'schulpflichtiger Kinder', whereas the last entry (7) is a further browsing step after viewing the result.

```
..
(1) bc76a65ad4be44e158b2d9ad17674f3dd4fa7296.rwth-aachen.de - - [23/Oct/2009:09:17:41 +0200]
"GET /dbs.js HTTP/1.1" 200 236 http://www.bildungsserver.de/zeigen.html?seite=641 "Mozilla/5.0
(Windows; U; Windows NT 5.1; de; rv:1.9.0.14) Gecko/2009082707 Firefox/3.0.14"
(2) bc76a65ad4be44e158b2d9ad17674f3dd4fa7296.rwth-aachen.de - - [23/Oct/2009:09:17:41 +0200]
"GET /zeigen.html?seite=641 HTTP/1.1" 200 46606 http://www.google.de/search?client=firefox-
a&rls=org.mozilla%3Ade%3Aofficial&channel=s&hl=de&source=hp&q=schulamt&meta=&btnG=Google-
Suche "Mozilla/5.0 (Windows; U; Windows NT 5.1; de; rv:1.9.0.14) Gecko/2009082707 Firefox/3.0.14"
(3) bc76a65ad4be44e158b2d9ad17674f3dd4fa7296.rwth-aachen.de - - [23/Oct/2009:09:17:54 +0200]
"GET /metasuche/qsuche.html?feldinhalt1=schulpflicht&bool1=AND&finden=finden&searchall=ja&daten
banken%5B%5D=dbs_seiten&DBS=1&art=einfach HTTP/1.1" 200 139848
http://www.bildungsserver.de/zeigen.html?seite=641 "Mozilla/5.0 (Windows; U; Windows NT
5.1; de; rv:1.9.0.14) Gecko/2009082707 Firefox/3.0.14"
(4) bc76a65ad4be44e158b2d9ad17674f3dd4fa7296.rwth-aachen.de - - [23/Oct/2009:09:17:55 +0200]
"GET /metasuche/dbs.js HTTP/1.1" 200 236 http://www.bildungsserver.de/metasuche/qsuche.html?
feldinhalt1=schulpflicht&bool1=AND&finden=finden&searchall=ja&datenbanken%5B%5D=dbs_seiten
&DBS=1&art=einfach "Mozilla/5.0 (Windows; U; Windows NT 5.1; de; rv:1.9.0.14) Gecko/2009082707
Firefox/3.0.14"
(5) bc76a65ad4be44e158b2d9ad17674f3dd4fa7296.rwth-aachen.de - - [23/Oct/2009:09:18:10 +0200]
"GET /metasuche/qsuche.html?feldinhalt1=schulpflichtiger+Kinder &bool1=AND&finden=finden
&searchall=ja&datenbanken%5B%5D=dbs_seiten&DBS=1&art=einfach HTTP/1.1" 200
109350 http://www.bildungsserver.de/metasuche/qsuche.html?feldinhalt1=schulpflicht
&bool1=AND&finden=finden&searchall=ja&datenbanken%5B%5D=dbs_seiten&DBS=1&art=einfach
"Mozilla/5.0 (Windows; U; Windows NT 5.1; de; rv:1.9.0.14) Gecko/2009082707 Firefox/3.0.14"
(6) bc76a65ad4be44e158b2d9ad17674f3dd4fa7296.rwth-aachen.de - - [23/Oct/2009:09:18:45 +0200]
"GET /zeigen.html?seite=21 HTTP/1.1" 200 25648 http://www.bildungsserver.de/metasuche/qsuche.html?
feldinhalt1=schulpflichtiger+Kinder&bool1=AND&finden=finden&searchall=ja &daten-
banken%5B%5D=dbs_seiten&DBS=1&art=einfach "Mozilla/5.0 (Windows; U; Windows NT 5.1;
de; rv:1.9.0.14) Gecko/2009082707 Firefox/3.0.14"
(7) bc76a65ad4be44e158b2d9 ad17674f3dd4fa7296.rwth-aachen.de - - [23/Oct/2009:09:19:24 +0200]
"GET /zeigen.html?seite=136 HTTP/1.1" 200 45273 http://www.bildungsserver.de/zeigen.html?seite=21
"Mozilla/5.0 (Windows; U; Windows NT 5.1; de; rv:1.9.0.14) Gecko/2009082707 Firefox/3.0.14"

..
```

Fig. 1. Sample entries in the DBS log.

Table 1 illustrates the distribution of queries in sessions and clicks after queries. A total number of 98,512 queries were submitted to the search engine in 65,513 unique sessions. In most sessions one query was submitted, however the proportion of the sessions which contain more than one query is around a quarter with an overall average session length of 1.50 queries.

More than half of the queries were not followed by any user clicks on the results. The average number of clicks is quite low 0.69.

The number of distinct queries within those are 31,347. Table 2 lists the top queries submitted to the search engine with their frequencies. Not surprisingly they are all in the German language.

	total	Mean.	Min.	Q1	Median	Q3	Max.
Queries/Session	98,512/65,513	1.50	1	1	1	2	101
Clicks/Query	68,604/98,512	0.69	0	0	0	1	51

Q1: 1st quartile, Q3: 3rd quartile.

Table 1. Distribution of Queries in Sessions and Clicks after queries.

Query	**Frequency**	**Query**	**Frequency**
quereinstieg	2505	englisch	208
suchbegriff	1387	praktika	205
quereinsteiger	689	vorbereitungsdienst	198
stellenangebote	435	deutsch	182
lehrplan	407	bildungsplan	180
lehrpläne	303	praktikum	179
abitur	293	grundschule	159
seiteneinstieg	283	adhs	148
mathematik	224	geschichte	138
referendariat	223	seiteneinsteiger	138

Table 2. Most frequent queries.

2.2 TEL Log Analysis

We have used log data that have been collected on the search engine of the European Library (TEL)[1]. The TEL logs contain an entry for every user interaction with the TEL portal. Log entries contain the type of action performed (e.g. simple or advanced search, changing system options) and attributes such as user ID, session ID, the interface language, query, and timestamp. Figure 2 lists some sample entries, the first one describing a search for *"pomegranate fertilization"* submitted through the simple user interface.

The logs record not just all queries submitted to the search engine but also other activities such as viewing a result. TEL logs have already been used in LogCLEF 2009 and 2010[2]. This year a new log file was distributed and covers the period from 1 January 2010 till 30 December 2010. That is the file we used.

In the logs there is a great inclination towards using simple search compared to using advanced search [3]. In our experiments we do not consider queries submitted via the advanced search interface.

We have used TEL logs in the past [4] and applied a very similar processing pipeline as previously, more specifically:

1. Discard all actions that are not simple search queries
2. Remove all queries that do not have English specified as the query language
3. Remove all queries that contain non-ASCII characters
4. Case-fold all queries, replace all non-alphanumeric characters by space

[1] `http://www.theeuropeanlibrary.org`
[2] `http://www.uni-hildesheim.de/logclef/`

```
...
903779;guest;83.33.xxx.xxx;83et8b7j010eh4vlht3ucj8dl1;en;("pomegranate fertilization");search\_sim;;0;-;;;2007-10-05 13:52:30
...
1889115;guest;71.249.xxx.xxx;8eb3bdv3odg9jncd71u0s2aff6;en;("mozart");search_url;;0;-;;;2008-06-24 22:02:52
...
1889118;guest;71.249.xxx.xxx;8eb3bdv3odg9jncd71u0s2aff6;en;("mozart");view_full;;1;;;;2008-06-24 22:03:03
...
1889120;guest;71.249.xxx.xxx;8eb3bdv3odg9jncd71u0s2aff6;en;Klavierkonzerte;search_res_rec_all;;0;-;;;2008-06-24 22:03:55
1889121;guest;71.249.xxx.xxx;8eb3bdv3odg9jncd71u0s2aff6;en;("klavierkonzerte");view_full;;1;;;;2008-06-24 22:04:10
...
```

Fig. 2. Sample entry in the first TEL log file.

5. If a query contains one or more Boolean operators, trim the query so that the left-most operator and everything that follows gets removed.
6. Finally, delete all queries which have no session number specified

With this processing we extracted 162,642 queries that have 'English' as the query language in 75,100 unique sessions from the total 806,155 interactions in the file. Figure 3 presents two sample entries in the processed query logs (using the 2008 logs and reproduced from [4]).

```
...
8eb3bdv3odg9jncd71u0s2aff6 xxxx 1889115 xxxx mozart xxxx 2008-06-24 22:02:52
8eb3bdv3odg9jncd71u0s2aff6 xxxx 1889120 xxxx klavierkonzerte xxxx 2008-06-24 22:03:55
...
```

Fig. 3. Sample session records after processing the TEL logs.

In Table 3 we compare both logs DBS and TEL 2010. The DBS search engine has more traffic and the average session length in the DBS logs is shorter.

Table 4 lists the most frequent queries in TEL 2010. By comparing this to the most frequent queries in DBS we can observe that the TEL query logs are more sparse. The frequency of the top queries in TEL 2010 are much less than the top ones in DBS.

	DBS (3 months)	TEL 2010 (12 months)
#Total Queries	98,512	162,642
#Distinct Queries	31,347	37,377
#Sessions	65,513	75,100
#Single Query Sessions	47665	42,200

Table 3. Comparison between the data sets.

3 Query Suggestions with Ant Colony Optimisation

We have recently explored the application of Ant Colony Optimisation (ACO) to build query association graphs from the query logs for the purpose of query

Query	Frequency	Query	Frequency
ditt legeme er ditt	308	maps	96
mozart	287	dante	95
art	189	bach	91
text	155	see	90
napoleon	148	water analysis	88
a	147	shakespeare	86
music	120	harry potter	84
meer dan een baan	115	michalopoulos	82
digital	107	map	80
hippo et hippa	97	france	80

Table 4. Most frequent queries in TEL 2010 after removing the official test queries.

recommendation [2]. ACO is applied to query logs as an adaptive learning process.

A user interaction with the search engine is treated as an individual ant's journey producing some pheromone and over time the collective journeys of all ants result in strengthening more popular paths, with higher pheromone levels, which leads to a corresponding term association graph. Less popular and seasonally incorrect paths will have lower pheromone levels by introducing some evaporation factor which reduces the weights of non-traversed edges over time.

Using this process, the association graph is being updated in a continuous learning cycle. The directed association graph can then be used for query recommendation as follows. Starting from the query node in question, we traverse the graph edges to identify and rank associated query nodes using the weights on the edges.

Figure 4 illustrates a partial association graph extracted by running ACO on the DBS logs.

4 Experimental Setup

The goal of the experiment is to answer these questions:

1. Does ACO learn useful query recommendations over time?
2. How does the performance of ACO in learning these suggestions differ in both logs given the difference we observed in the analysis between the two datasets?
3. Can the query recommendations provided by ACO shorten the number of steps needed by users to find what they are looking for?

We conducted two sets of experiments to answer these questions.

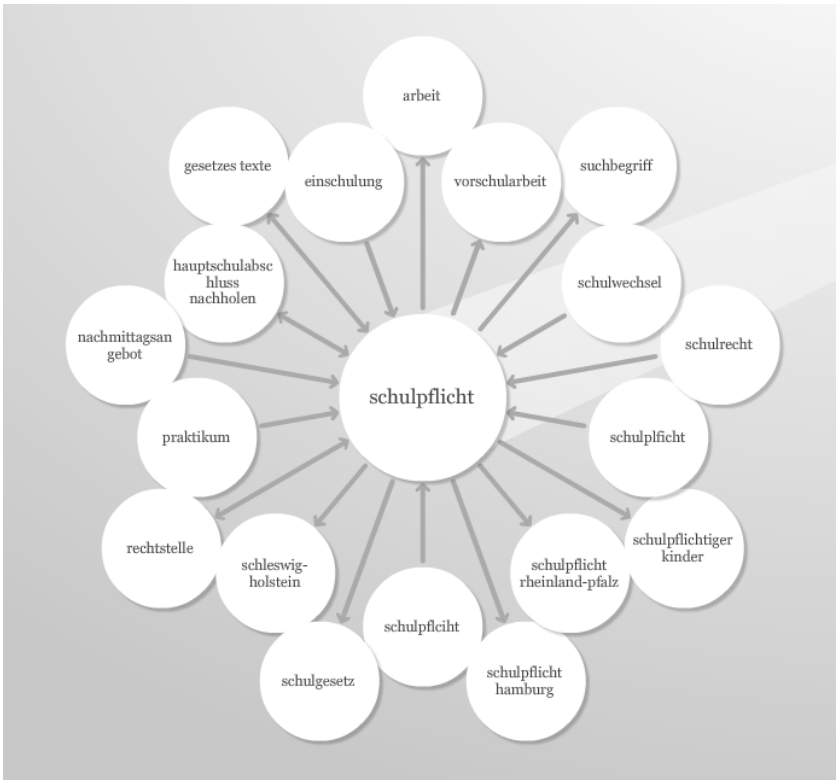

Fig. 4. Partial domain model derived from DBS log data. Weights are not shown on the edges.

4.1 AutoEval Framework

We employed AutoEval which is an automatic evaluation framework for assessing the performance of query suggestion systems over time based on actual query logs.The validity of the framework has been confirmed with a user study [1].

The evaluation is performed on arbitrary intervals, e.g. on a daily basis. For example, let us assume that during the current day, three query modifications have been submitted. For each query modification pair, the domain model is provided with the initial query and returns a ranked list of recommended query modifications. We take the rank of the actual modified query (i.e., the one in the log data) in this list, as an indication of the domain model's accuracy. So for the total of three query modifications in the current day, we can calculate the model's Mean Reciprocal Rank (MRR) score as $(1/r_1+1/r_2+1/r_3)/3$, where r_1 to r_3 are the ranks of the actual query modifications in the list of modifications recommended by the model in each of the three cases. More generally, given a day d with Q query modification pairs, the model's Mean Reciprocal Rank score for that day MRR_d is given by Equation 1 below.

$$MRR_d = (\sum_{i=1}^{Q} \frac{1}{r_i})/Q \quad (1)$$

Note that in the special case where the actual query modification is not included in the list of recommended modifications then $1/r$ is set to zero. The above evaluation process results in a score for each logged day. So overall, the process produces a series of scores for each domain model being evaluated. These scores allow the comparison between different domain models. A model M_1 can therefore be considered superior over a model M_2 if a statistically significant improvement can be measured over the given period.

It is important to mention here that we do not try to identify query modifications within a user session that are actually related. Therefore even subsequent queries that are not related are treated as a query modification pair. However these noisy query modification pairs do not affect the evaluation methodology as this noise is common for all evaluated models.

We ran AutoEval on the DBS logs in the period of September to November 2009 with weekly batches and our ACO algorithm to derive query suggestions. However for the TEL logs, we used monthly batches and ran AutoEval for the entire period of the year 2010. Monthly batches were used to have somehow comparable traffic.

4.2 Shortening the Path

The second set of experiments looked at examining the effect of query suggestions in reducing the number of steps required by user to achieve their search goals. Given a session of more then two queries we want to find out whether our ACO query recommendation system can suggest the last query in the session. Here we are assuming that the last query in the session was a successful one as no more queries were issued afterwards. This assumption can be relaxed by looking at only session which ended up with a query followed by a single click on the results as this may a better indication of search success. For this experiment, using the DBS logs, we ran the ACO algorithm to learn an association graph on the first 45 days of the log. For the next 45 days we extracted sessions with more than two queries. This resulted in 3600 sessions, out of which 1259 ended with a query followed by a landing page. For each of those pairs, we examined the suggestion list recommended by the learnt ACO model.

5 Results and Discussion

After running the evaluation framework on both datasets using the setup described in the previous sections, we plot the obtained MRR scores in Figures 5 and 6.

By examining those figures we can observe the following:

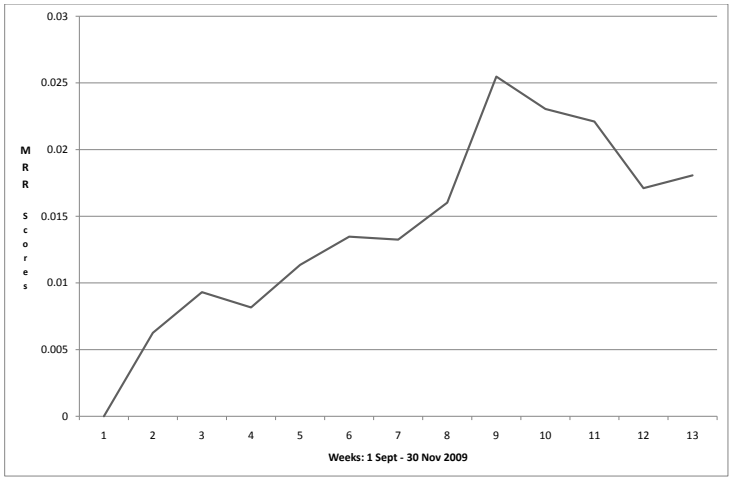

Fig. 5. AutoEval run on DBS logs.

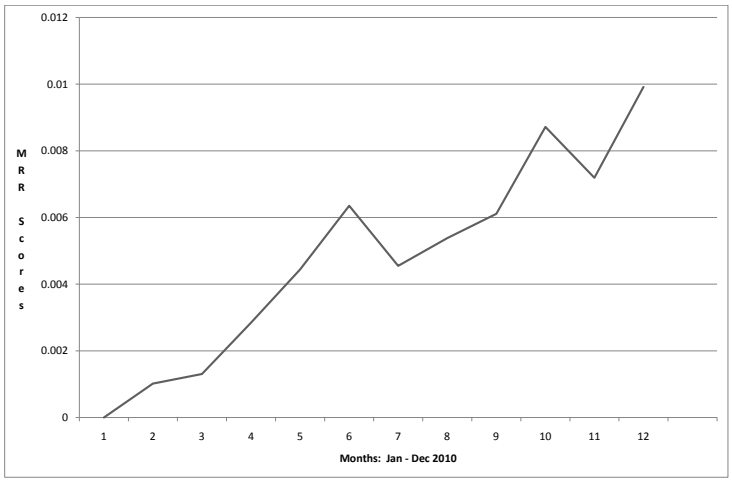

Fig. 6. AutoEval run on TEL logs.

– The ACO adaptive learning model is capable of learning useful relations from the TEL logs and the DBS logs. In both cases, despite the spike, ACO

is achieving higher scores over time in both logs. This confirms findings of a previous study on TEL logs [4].
– The MRR scores in the TEL logs are lower than those in the DBS logs, despite using a much longer period. It is not only the higher traffic on the DBS search engine that is having the impact on the ACO performance but also the higher sparsity in the TEL queries as discussed in the analysis section.

The results of the second sets of experiments are shown in table 5. This is suggesting that in about 7% of the cases the query recommendation system is able to suggest the last query to the users. The percentage is low but it is not surprising due to the sparsity and low traffic of the query logs which we discuss in the analysis section. However, a search engine with this recommendation system in place is better than the one that does not provide query suggestions as it is sometimes capable of assisting the user in reducing the path to their search goal by suggesting useful queries.

	all sessions (3600)	single click sessions (1259)
correct recommendation ratio	251/3600	86/1259
correct recommendation ratio @ 10	226/3600	78/1259

Table 5. Correct query recommendation ratio.

Acknowledgements

This research is part of the AutoAdapt research project. AutoAdapt is funded by EPSRC grants EP/F035357/1 and EP/F035705/1.

References

1. M.-D. Albakour, U. Kruschwitz, N. Nanas, Y. Kim, D. Song, M. Fasli, and A. De Roeck. Autoeval: An evaluation methodology for evaluating query suggestions using query logs. In *Proceedings of ECIR 2011*, volume 6611 of *Lecture Notes in Computer Science*, pages 605–610. Springer Berlin / Heidelberg, 2011.
2. M.-D. Albakour, U. Kruschwitz, N. Nanas, D. Song, M. Fasli, and A. De Roeck. Exploring ant colony optimsation for adaptive interactive search. In *Proceedings ICTIR 2011, forthcoming.*, 2011.
3. M. R. Ghorab, J. Leveling, D. Zhou, G. J. F. Jones, and V. Wade. Identifying common user behaviour in multilingual search logs. In *CLEF 2009, Corfu, Greece, September 30 - October 2, 2009, Revised Selected Papers*, volume 6241 of *Lecture Notes in Computer Science (LNCS)*, pages 518–525. Springer, 2010.
4. U. Kruschwitz, M.-D. Albakour, J. Niu, J. Leveling, N. Nanas, Y. Kim, D. Song, M. Fasli, and A. De Roeck. Moving towards Adaptive Search in Digital Libraries. In *Advanced Language Technologies for Digital Libraries*. Springer, 2011. Forthcoming.