An analysis of free-text queries for a multi-field web form

Kien Tjin-Kam-Jet
University of Twente
The Netherlands
tjinkamj@ewi.utwente.nl

Dolf Trieschnigg
University of Twente
The Netherlands
trieschn@ewi.utwente.nl

Djoerd Hiemstra
University of Twente
The Netherlands
hiemstra@ewi.utwente.nl

ABSTRACT

We report how users interact with an experimental system that transforms single-field textual input into a multi-field query for an existing travel planner system. The experimental system was made publicly available and we collected over 30,000 queries from almost 12,000 users. From the free-text query log, we examined how users formulated structured information needs into free-text queries. The query log analysis shows that there is great variety in query formulation, over 400 query templates were found that occurred at least 4 times. Furthermore, with over 100 respondents to our questionnaire, we provide both quantitative and qualitative evidence indicating that end-users significantly prefer a single field interface over a multi-field interface when performing structured search.

Categories and Subject Descriptors

H.5 [**Information Interfaces and Presentation**]: User Interfaces—*Natural language*

General Terms

Languages, Experimentation, Measurement

Keywords

Query translation, query reformulation, query log analysis, user study

1. INTRODUCTION

Many websites contain information that is stored in structured databases [6]. In order to gain access to this structured information, a user generally has to fill out and submit a complex web form consisting of multiple input fields and options. There are many potential benefits if one could search within such structured databases using a simpler interface consisting of a single text input field. For instance, in distributed IR [5], connecting structured databases to a general web search engine could potentially be simpler because the query could just be forwarded to the database without any query transformation. It could also be beneficial for Desktop search, which concerns semi-structured data collections and presents some challenges which are closely related to those in distributed IR [11]. Lastly, a single-field interface could be easier to use and could lead to better system usability. However, this leads to the following questions: *i*) do end-users prefer to use a single text field interface over a multi-field interface when they are performing structured search? *ii*) how do end-users phrase free-text queries in a single text field when they intend to perform a structured search? Finally, *iii*) how difficult is it to correctly interpret such free-text queries? The answers to these questions could: *i*) strengthen the motivation for the use of a single-field interface; *ii*) lead to a better understanding of the complexities involved in free-text queries with a structured search intent; and ultimately, *iii*) lead to solutions that improve the search experience for the end-user.

Our contributions are as follows. We report on the variety in free-text query formulations when end-users perform structured search. We provide both quantitative and qualitative evidence indicating that end-users prefer a single field interface over a multi-field interface when performing structured search. Furthermore, we categorize the type of errors made by both system and user, and show that many frequent errors concern spelling errors or out-of-dictionary terms. With regards to the query sessions, in the majority (68%) of the sessions, the very first query is already successful and returns results. In the remaining 32% of the sessions, 68% continued the session and obtained a successful result (needing 1.7 additional queries on average to successfully obtain results).

The remainder of this paper is structured as follows. In Sect. 2 we overview related work on query log analysis. In Sect. 3 we describe what data was used and how the data was obtained. We describe our research methodology in Sect. 4, and present our findings in Sect. 5. Finally, discuss our work in Sect. 6, and conclude our work in Sect. 7.

2. RELATED WORK

Transaction Log Analysis (TLA), also referred to as search log or query log analysis, is a methodology to "examine the characteristics of searching episodes in order to isolate trends and identify typical interactions between searchers and the system" [9]. Transaction logs are viewed as an unobtrusive way of collecting significant amounts of data on the search behavior of a large number of users. Many studies have

IIiX 2012, Nijmegen, The Netherlands

analyzed various aspects of large-scale search logs. In the end, the ultimate goal is to use this knowledge to improve the search experience for the user. Search log analysis can be used to improve, for example, retrieval functions [10], spelling corrections [2], and query segmentation [13, 8]. According to the large-scale Altavista search log study (which concerns about 1 billion entries), users tend to formulate short queries (2.3 words per query on average), and sessions are relatively short (2 queries on average) [15]. Similar findings regarding query length and query characteristics were reported in the Excite search log study (which concerns about 1 million entries) [16] and in an MSN search log study (concerning about 15 million entries) [3]. In the latter MSN study it was noted that, if we assumed direct relation between the reciprocal rank of the clicks and the effectiveness of the retrieval, the effectiveness decreases as the query length increases. Recent studies relating somewhat to the structured query intent concerned in our work, like [1, 12], have analyzed query logs to extract common patterns of search or to extract structured information from queries. Since the aim of these studies was the development of a suitable extraction algorithm, no figures that could indicate user behavior, like query and session length statistics, were reported. However, these studies provide evidence that many queries in a general web search log contain structured patterns, and that it is important to further study the complexity of free-text queries with a structured intent.

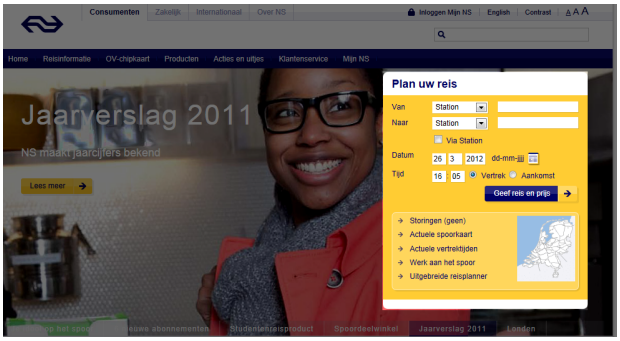
3. DATA ACQUISITION

We developed a website[1] that serves as an alternative single-field interface (see Fig. 1a) for the Dutch railways site[2] which has a multi-field search interface (see Fig. 1b). We will use the terms *NS* and *Treinplanner* to refer to the Dutch railways site and our alternative site, respectively. Treinplanner adopts a pattern-based approach to find the best interpretation of the queries. This approach has been described in [18]. A valid query results in one or more interpretations, i.e. ways in which the web form of the NS can be filled out. After receiving the list of interpretations, the client's browser automatically submits the top interpretation to the NS and displays the results from the NS to the user. Examples of an invalid and a valid query can be seen in Fig. 2a and Fig. 2b, respectively. The queries that were submitted to the Treinplanner were logged. It was also logged whether and how (e.g. using a mouse or the arrow keys on a keyboard) users selected the query suggestions that were shown while entering a query.

Visitors of the Treinplanner site can also provide feedback and participate in a usability study comparing the NS and the Treinplanner. They can provide feedback as either a general comment, or as specific feedback on the results for a particular query (i.e. giving a brief description of the error and annotating what they meant with the query). A visitor that wants to participate in the usability study must first enter some demographic information like his or her age, gender, and education level. The visitor must also indicate how much experience he or she has with both the NS and the Treinplanner systems. A visitor who has enough experience with both systems can continue to the System Usability Scale (SUS) questionnaire. The SUS questionnaire is a simple, ten-item *Likert* scale giving a global view of subjective assessments of usability [4]. The questionnaire is administered for each system. This results in two scores ranging from 0 to 100, a higher score indicates a better usability of the system.

[1] `http://treinplanner.info`
[2] `http://www.ns.nl`

(a) The Treinplanner interface: a single-field search box. Inside the search box, an example query is shown (translated as *search example: tomorrow at eleven departing from Amsterdam to Utrecht*).



(b) The NS interface: a multi-field search form. (We have emphasized the search form by darkening the picture surrounding of the form.)

Figure 1: Single- and multi-field interfaces.

We gathered participants for our experiment through announcements on social media and through a number of press releases. In particular, we issued a tweet which was also re-tweeted by the NS (which has over 30,000 followers). Furthermore, we crawled all tweets containing the keyword "Treinplanner" to gain qualitative data on the end-users' opinion (on Twitter) about using this single-field search interface to actually perform structured search.

The data used for our analysis (query log data, usability study, opinion, tweets) were collected this year (2012) from January the 23^{rd} to March the 25^{th}.

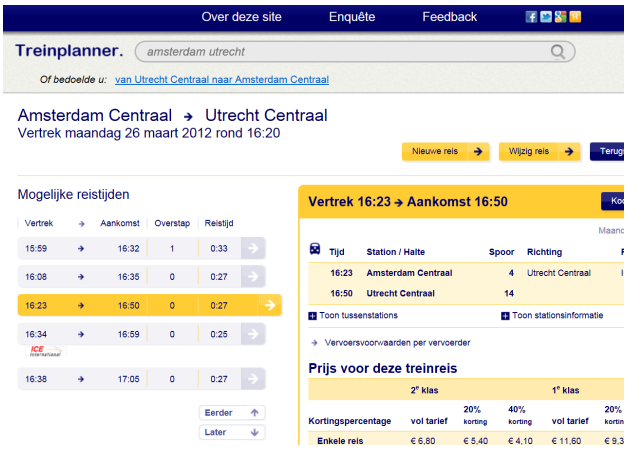
4. RESEARCH METHODOLOGY

In this section, we first describe how we analyzed the query log data. We next describe how we analyzed the quantitative data from the user study. Finally, we describe how we analyzed the qualitative data from the user comments and Twitter messages about the Treinplanner.

4.1 Query log analysis

We performed the following in our analysis of the query log: *i*) we cleaned the data and grouped the queries into sessions; *ii*) we manually analyzed a sample of the log; *iii*) we classified the queries into query types; *iv*) we extracted the query templates from each query; *v*) we analyzed if and how query suggestions were used.

(a) An invalid query, in this case due to a spelling error, Treinplanner returns an error message.



(b) A valid query shows the results from the NS site.

Figure 2: Results of the Treinplanner system.

4.1.1 *Cleaning and grouping the data*

First, we removed all queries issued by ourselves (based on our ip addresses). Second, we discarded all queries that did not contain a cookie ID (this could happen when client browsers did not accept cookies). By using cookies, we can easily discriminate the queries of one client from another. We grouped all queries by their cookie ID and then segmented those queries into *search episodes* and *search sessions* using the geometric session detection method proposed by Gayo-Avello [7]. A search episode denotes all actions performed by a single user within a search engine during, at most, one day. An episode comprises one or more sessions, and each session comprises one or more successive queries related to one single information need or goal.

4.1.2 *Manual sample analysis*

We first introduce some new terminology. The Treinplanner system only returns results if: *i*) it can detect at least a departure and an arrival station name in the query, so if the query contains *sufficient* information; and if *ii*) none of the departure, arrival, or via-stations are the same, so if the information is *non-conflicting*. Otherwise, the system either indicates that some information is missing or that there is conflicting information. We refer to queries that contain sufficient and non-conflicting information as *valid* queries. Conversely, we refer to queries that contain insufficient or conflicting information as *invalid* queries.

Treinplanner does not apply spelling corrections. Therefore, queries with spelling errors for which the search intent is obvious, like "Amst*re*dam to Utrecht", are invalid because the system could not detect at least two stations[3].

A valid query does not imply a *correctly interpreted* result for the user, e.g. the system might have extracted the wrong date and time, the wrong station names, or might have missed some important pieces of information. Therefore, we assessed the correctness of the interpretations by manually inspecting a random sample of the query log. We determined the true/false negatives (e.g. was the query really invalid or did the system fail to correctly interpret an otherwise valid query?), and we determined the true/false positives (e.g. was the query correctly interpreted, or were some meaningful parts incorrectly interpreted).

During the manual inspection, we used the following rules of thumb. For queries that are marked as invalid by the Treinplanner: if a human annotator could make sense from the query (by finding sufficient and non-conflicting information) we marked the query as a false negative. For queries that are marked as valid by the Treinplanner: if a human annotator could find meaningful pieces of information that were misinterpreted by the system, we marked the query as a false positive. In both cases, we noted what caused the error (e.g. a spelling mistake, a synonym that is not in the system's lexicon, some concatenated words).

Finally, we manually analyzed the explicit user feedback on wrong query interpretations such as when the system has mixed up the intended input fields of the stations, or has failed to recognize a relevant piece of information.

4.1.3 *Session analysis*

First, we analyzed the sessions in terms of their query validity. For example, how many sessions started with a valid query? In sessions that did not start with a valid query, how many successive queries did it take on average to reach a valid query? How many sessions did not have any valid queries at all? Second, to gain more insight in how users change their queries during a session, we classified successive queries within a session into one of the following query types:

Lexical repeat. A successive query is a lexical repetition of its previous query if the Levenshtein distance between the characters of both queries is less than some threshold[4].

Specialization. A successive query is a specialization if the set of terms of the successive query is a proper superset of the set of terms of the preceding query. For example, when one subsequently enters "Amsterdam" and "Amsterdam Utrecht", the latter query is a specialization query.

Generalization. A successive query is a generalization if the set of terms of the successive query is a proper subset of the set of terms of the preceding query. A generalization is the opposite of a specialization. For example, when one subsequently enters "Amsterdam Utrecht" and "Amsterdam", the latter query is a generalization query.

Mixture. A successive query is a mixture if both queries contain at least one term that is not in the other query,

[3]Amst*re*dam should be spelled as Amst*er*dam

[4]We used different thresholds depending on the length l (in number of characters) of the longest query: 0, if $1 \leq l \leq 3$; 1, if $4 \leq l \leq 14$; 2, if $15 \leq l \leq 24$; and 3 for $l \geq 25$

and if the intersection of the set of terms of both queries is not empty. For example, when one subsequently enters "Amsterdam Utrecht" and "Amsterdam Rotterdam", the latter query is a mixture query.

New. A successive query is new if the intersection of the set of terms of both queries is empty, while the queries themselves are not empty. For example, when one subsequently enters "Amsterdam Utrecht" and "Rotterdam Tilburg", the latter query is a new query.

Semantic repeat. A successive query is a semantic repetition of its previous query if the same set of key-value pairs can be extracted from both queries. For example, when one first enters "from Amsterdam to Utrecht" and then "to Utrecht from Amsterdam" (or just "Amsterdam Utrecht"), then the latter query is a semantic repetition.

Successive queries within a session were classified according to the specified order of the classes. That is, if a query could not be classified as the first class, *lexical repeat*, we tried the second class, *specialization*. If it could not be classified as specialization, we tried the third, and so on, until the query was classified. This way, we ensured that a query belonged to, *at most*, one class. Note the asymmetric output of this procedure: queries from the class lexical repeat could in theory also belong to the class semantic repeat; however, queries from the class semantic repeat could never belong to the class lexical repeat. Related work on query log analysis (see Sect. 2) normally concerns keyword search queries; however, due to our experimental setup, our query log contains free-text queries that contain structured key-value pairs. Therefore, we can classify a successive query as a *semantic* repetition with its previous query if the set of key-value pairs of both queries are the same.

4.1.4 Template extraction

A query consists of a sequence of keywords and domain attributes, e.g. like station, date, and time. By abstracting away the specific instances of each attribute, we can represent a query by its query *template* [1]. The templates were automatically extracted by the Treinplanner system. Pieces of text that it recognized as stations were marked as `station`; words indicating what kind of station (departure, destination, or intermediate) were marked as *from*, *to*, or *via*, respectively; dates, times, or words indicating time of departure or time of arrival, were marked as `date`, `time`, or `arr/dep`, respectively. Finally, words that were not recognized were marked as `oov` (out-of-vocabulary). For instance, the query "Amsterdam, destination: Utrecht, arrival time 10 pm" is represented by the template: `station` *to* `station arr/dep time`.

4.1.5 Query suggestion usage

The Treinplanner interface shows query suggestions while typing, but are these suggestions used? And does it pay to use the suggestions, i.e. do users who use query suggestions issue more valid queries than users not using suggestions? We compared the queries where no suggestions were selected to those where a suggestion was used once or multiple times. We applied the Pearson's chi-square test (χ^2) to find out whether or not using suggestions and entering valid queries are correlated [14]. The χ^2 tests for dependence and does not assume a normal probability distribution.

4.2 Usability study – quantitative analysis

We compared the usability scores for the Treinplanner to those for the NS. We checked whether the scores differed significantly using the paired T-test, with $p < 0.05$. Since the T-test assumes a normal distribution on the data and since we do not know for certain that our data follows a normal distribution, we also applied the statistically weaker sign test which does not assume a normal distribution. The tests were performed over all participants and over groups of participants to check whether or not the difference is significant across all groups.

4.3 User opinions – qualitative analysis

We performed a qualitative analysis of the user's opinions about Treinplanner, also by manual inspection. The user opinions were obtained from comments that were given at the Treinplanner site, and from tweets that mentioned "Treinplanner". From the comments, we counted how many mentioned positive aspects, negative aspects, bugs, or possible improvements. From the tweets, we counted how many mentioned positive aspects, and how many mentioned negative aspects of the Treinplanner system. Tweets that were otherwise neutral were discarded.

5. RESULTS

In this section, we adhere to the structure of the previous section about our methodology. We first present our findings on the query log analysis, then on the usability study, and end with our findings on the user opinion analysis.

5.1 Query log results

5.1.1 Descriptive statistics

We collected a total of 36,271 queries, which, after cleaning, resulted in 30,472 queries. These queries were issued by 11,933 different clients, based on the client's cookie ID. The distributions of query lengths over valid, invalid, and all queries is depicted in Fig. 3. Valid queries are slightly longer than invalid queries on average. More detailed query length statistics are given in Table 1. After grouping the queries by client and applying the geometric query segmentation method [7], we found 13,058 search episodes and 14,541 search sessions. This data is summarized in Table 2. The distribution of the number of queries within a session is depicted on a log-log plot in Fig. 4. It can be seen from the figure that the session length follows a Zipfian distribution (the dots are approximately linearly aligned), which is consistent with findings in related literature [16].

5.1.2 Manual sample results

For the manual query log inspection, we randomly selected a total of 1,500 queries, or 5% of the query log: 750 valid queries, and 750 invalid queries.

False positives. Out of the 750 valid queries, 46 (6.1%) were incorrectly interpreted. In all false positives, the system failed to interpret certain meaningful parts of the query, such as: "1800", "ten past 2", "next month", and "around rush hour". Other frequent mistakes were the lack of spaces between two meaningful parts (e.g. "Wednesday10 am") and spelling errors (e.g. "Amsterdam Utrecht elevn o'clock").

False negatives. Out of the 750 invalid queries, 66 (8.8%) were wrongly marked as invalid. The most frequent mistake, causing just over half of the false negatives, could

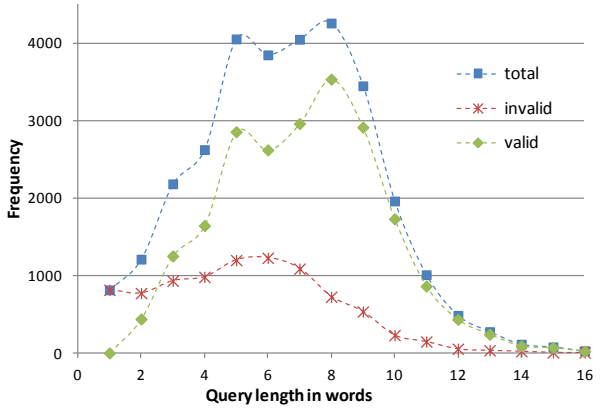


Figure 3: Distribution of query lengths.

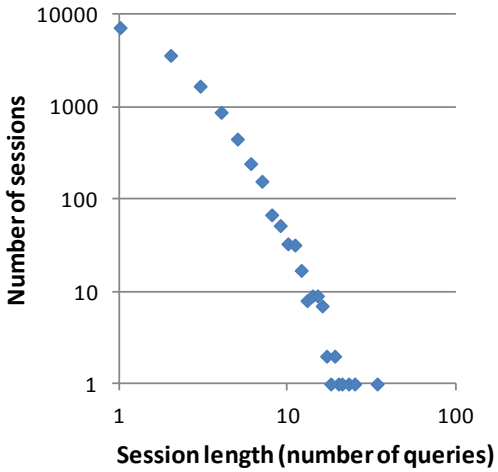


Figure 4: Distribution of session lengths.

Table 1: Query length in characters (c), words (w).

	Total Queries	avg. c (w)	min. c (w)	max. c (w)	std.dev. c (w)
Invalid	29%	33 (5)	1 (1)	100 (43)	16 (3)
Valid	71%	45 (7)	9 (2)	141 (34)	14 (3)
All	100%	42 (7)	1 (1)	141 (43)	16 (3)

Table 2: Query log entries by granularity level.

	Number
Queries	30,472
Sessions	14,541
Episodes	13,058
Clients	11,933

be attributed to the lack of certain synonyms for some station names. Therefore, given a query that is perfectly interpretable by a human annotator, the system would indicate that there was no departure, or arrival, station. The second most frequent mistake was the use of dashes as a separator between meaningful parts, like "Amsterdam-Utrecht". The third reason, causing 7 false negatives, was that if a query was phrased in a particular way[5], there would be no result. Further, we observed many spelling errors and a frequent lack of spaces between two meaningful parts in a query. However, even if those mistakes could be corrected, most of those queries would still be invalid as they only mentioned one station or something that is not in the scope of the system (e.g. like street or place names).

Explicit feedback. There were 31 queries reported by users for which the system gave a misinterpreted result. Reasons for the most frequently reported mistakes could be attributed to: an incomplete lexicon (26%); usage of particular terms by which users indicate arrival time instead of departure time (23%); and, issues relating to either the recognition of times or the interpretation of times — e.g. five o'clock, but is that in the evening or morning? (31%).

[5]A date (consisting of a day and a month), followed by a two-digit number, followed by some indication of a time (e.g. "am", "pm", "hour")

Table 3: Sessions grouped by session length and by the validity of the session's initial query.

	First query is	
Session length	Valid	Invalid
Single-query	6,367	889
Multi-query	3,490	3,795

5.1.3 Session results

Out of the 14,541 sessions, 7,256 sessions consisted of just one query, and 7,285 consisted of two or more queries. Table 3 shows how many sessions started with a valid query and how many started with an invalid query. It also shows how many were single-query or multi-query sessions. In the majority (68%) of the sessions, the very first query is already successful and returns results. From the 3,795 multi-query sessions that started with an invalid query, 3,192 (84%) contained at least one valid query, and 603 (16%) contained no valid queries. After an initial invalid query, it took 1.7 queries on average to reach a valid query.

As described in Sect. 4, we classified successive queries within sessions into one of several query types. Some queries could not be assigned to any class and they were inspected manually. Almost all unclassifiable pairs of queries looked like this: "tomorrow from Utrecht to The Hague" and "tomorrow from The Hague to Utrecht". That is, the queries contained the same terms, which is why they could not be classified as any of the specialization, generalization, mixture, or new query types. Further, the queries were neither lexical nor semantic repetitions of each other. Whether these queries should still be considered as mixture queries or as a new query type is open for discussion. For now, we will use the query type *other* to refer to these queries.

Figure 5 depicts the query type distribution of successive queries within a session. The types are split into valid and invalid queries. From this figure we can see that, apart from new and semantic repetition queries, the majority of successive queries within a session are valid queries. Specialization and mixture queries perform particularly well. A plausible explanation is that a users naturally submit a specialization

query after an invalid query when, for instance, the preceding invalid query did not contain both a departure and arrival station.

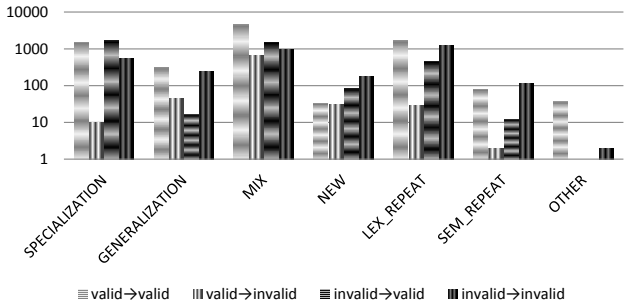


Figure 5: Distribution of query modification types in sessions with at least 2 queries.

5.1.4 Query template results

Table 4 lists the 12 most frequent templates. The first and second templates denote the type of error in invalid queries as perceived by the system. For example, if a users enters "to Amsterdam via Utrecht" or just "to Amsterdam", then the system would indicate that no departure station was given. There were over 400 templates that occurred at least 4 times, and in total there were almost 1,500 templates. This shows that there is great variety in how users formulate structured information needs.

Table 4: Most frequent query templates.

Template	frequency
`[INVALID] no destination given`	15.8%
`[INVALID] no departure given`	12.6%
`date time` *`from`* `station` *`to`* `station`	7.7%
`date` *`from`* `station` *`to`* `station`	6.8%
`station` *`to`* `station`	3.4%
`station station`	2.8%
`from` `station` *`to`* `station`	2.3%
`date` *`from`* `station` *`to`* `station time`	2.1%
`date time arr/dep` *`from`* `station` *`to`* `station`	2.0%
`oov` *`from`* `station` *`to`* `station`	1.9%
`date` *`from`* `station` *`to`* `station arr/dep time`	1.1%
`date station station`	1.0%
...	40.5%

5.1.5 Suggestion usage results

The Treinplanner system shows query suggestions while the user is typing and formulating a query. Query suggestions are generally perceived as helpful. From Table 5, which shows how users select the query suggestions, we can see that most queries are typed in completely without making use of the query suggestions. We can also see that query suggestions are more often selected with the mouse rather than with the arrow keys on the keyboard. On the one hand, this is surprising since we assumed that the ease of formulating a query using only a keyboard would outweigh the effort of grabbing the mouse to make a selection from query suggestions. On the other hand, most queries were typed in completely, which may indicate that users do prefer to just use the keyboard. Nevertheless, the results suggest that one should select the query suggestions more often since the ratio between valid and invalid queries, when selecting and making use of the query suggestion (ratio of 3.1 : 1.0), is significantly higher ($p < 0.001$) than when not making use of the query suggestions (ratio of 2.3 : 1.0).

Table 5: Statistics on the usage of query suggestions.

	Total	Valid	Invalid
No use of suggestions	22,308	15,517	6,791
Use of suggestions	8,164	6,171	1,993
Suggestion selection method			
Mouse only	6,447	4,902	1,545
Keyboard only	1,651	1,209	442
Both mouse and keyboard	66	60	6

5.2 Usability analysis – quantitative results

There were 116 participants (99 male and 17 female) that opted in and completed our online survey. The modal, average, and standard deviation of the ages of the participants were 25, 40, and 19 years, respectively. The education background of the participants is summarized in Table 6. The Simple Usability Scale (SUS) scores for the NS and the Treinplanner systems are shown in Table 7. Overall, the participants significantly prefer the Treinplanner system (first row in the table). By grouping the participants based on their level of search engine familiarity, we can see that as the familiarity increases, the difference between interface preference grows larger. The groups 'frequent' and 'often' significantly prefer the single-field interface, whereas in the group 'rare' there is almost no difference between the two systems.

Table 6: Participant's education background

Education level	Studying	Completed
Elementary or middle school	0	1
High or junior high school	4	19
College or university	32	60

Table 7: Survey scores grouped by participants' experience with search engines (like Google and Bing). Frequent means daily usage over 20 times, often means 5 to 20 times a day, and rare less than 5 times a day. Numbers in bold are statistically significant.

Experience with search engines	Group size	NS	Treinplanner
All groups	116	71	**84**
Frequent	54	68	**85**
Often	53	72	**84**
Rare	9	78	80

5.3 Opinion analysis – qualitative results

During the period of January the 25^{th} to February the 29^{th}, we collected over 150 opinions on the Treinplanner site

and over 300 tweets mentioning the Treinplanner. Out of all 150 opinions, 64% were positive, expressing statements like "great initiative", "nice!","works great and fast"; 21% gave suggestions on for improving the system; 6% were negative or skeptical of the system; and, 9% indicated mistakes like how a query was misinterpreted. It was even uttered 9 times, asking if such an interface would become available for another major travel planning site. From the Twitter messages, only 6 tweets mentioned a bug or expressed doubts about the system (e.g. "is the Treinplanner interface an improvement?", or "the site does not work"), the rest were positive (re)tweets, like "forget the NS travel planner, Treinplanner is better".

6. DISCUSSION

6.1 Methodology

There are two issues regarding our data collection method that should be addressed. First, while our collected queries may reflect the expected types of queries in a production system, it is possible that our sample contains an over-estimate of test-queries. Some users may have been inclined to explore the limits of the system and issued many different queries not necessarily related to a real information need. Second, we note that the results presented in Sect. 5.2 and Sect. 5.3 were based on not-so-random sampling. It is possible that only those users who would like to say something were analysed and not the silent majority (if any).

6.2 Results

One remarkable observation is the relatively high number of invalid queries. Search log studies typically deal with keyword queries since the search engine in question provides a keyword search service to its users. In contrast, Treinplanner allows its users to enter anything ranging from keyword queries up to complete natural language sentences. Therefore, the question arises whether the users' expectations will match the actual capabilities of the system. That is, will the users' queries be such that they make good use of the system's query understanding capabilities, will they be too simple, or will they be too complex? This has also been referred to as the *habitability problem* [17]. After manually inspecting a sample of the query log, we noticed that some users had a dialog system in mind. For instance, after submitting a query, they submit a subsequent query like "no, from amsterdam". Another wrongful expectation was that the system could interpret more than Dutch train station names, e.g. like street names or station names in foreign countries. This is partly reflected by the large proportion (over 55%) of invalid to invalid semantic repetitions. In other words, users would re-phrase their information needs, which contained unrecognized places, hoping or expecting that the system would then understand their query if it was formulated differently. Finally, many queries contained just one station name. This could indicate that users expected the system to know their current location (similar to how modern mobile devices know their geographic location) and assume it as their departure location. This was explicitly given as feedback by some users. While the habitability problem might account for some of these invalid queries, we again point out the possibility that some users were plainly biased towards testing the limits of the system. There were more than 1k sessions that contained 5 or more queries, which is strongly indicative of people testing the system.

Another remarkable observation is that the most likely valid-query template (`date time` *`from`* `station` *`to`* `station`) corresponds to the template of the example query shown in the Treinplanner interface. In contrast, in an earlier laboratory experiment where users were given different search tasks in various descriptions, the majority of the users did not follow the templates of the descriptions of the search tasks [18]. One way to mitigate this effect is to refrain from showing any example query in the first place, but that would increase the effect of the habitability problem. Another way to mitigate this effect is to randomly show different examples with different templates. However, from a production perspective, one can argue that the end user should formulate the question in a way that is expected: the more we can guide the user into using some pattern, the better.

7. CONCLUSION

We have described an in-depth analysis of how structured information needs are formulated as free-text queries. In addition, we have conducted a user study and analyzed user opinions to find out whether or not users prefer a single-field interface or a multi-field interface for formulating structured queries. Our research questions can be answered as follows.

i) *Do end-users prefer to use a single text field interface over a multi-field interface when they are performing structured search?* In general, users significantly prefer the single field-interface. From our questionnaire we can conclude that, the more experience a user has with general web search engines, the more pronounced the preference is for the single-field interface. Also, looking at the qualitative data obtained from the tweets and user opinions, the many positive responses towards the single-field interface are most pertinent, indicating that the single-field interface is preferred.

ii) *How do end-users phrase free-text queries in a single text field when they intend to perform a structured search?* We have extracted almost 1,500 unique templates describing how users formulate their queries. With over 400 query templates occurring at least 4 times, we can say that there is great variation in how queries are formulated. Furthermore, users make use of their own abbreviations for station names, or make many mistakes like spelling errors, or forget to separate words with a white space.

iii) *How difficult is it to correctly interpret such queries?* In this experiment we used a rule-based system to interpret and translate the user queries. Our manual analysis of 5% of the query log showed that 7.3% of the queries were incorrectly interpreted. The reasons for most mistakes could be attributed to the query containing spelling errors, or the system not containing enough synonyms. Certain types of errors might be challenging for a rule-based approach; but overall, the system correctly interpreted most queries with an accuracy of over 92%.

Overall, we can conclude that a very flexible system is needed to handle the large variety in query formulations. Also, spelling errors are an important problem and are partially solved by query suggestions. Finally, users suffer from the habitability problem which partially explains why they enter invalid queries. However, users are able to rephrase their query into a valid query, requiring less than two reformulations on average.

For future work we plan to analyse queries over a longer

period of time, particularly focussing on queries of returning users. Other interesting directions for future work include analyzing query formulations in more complex domains, such as, for example: product search sites, library catalogs, or travel-planning sites that include all kinds of transportation means and not just train travel.

Acknowledgment

We thank the NS, and in particular Han Mooiman for his valuable feedback and for his significant contributions regarding the setup of this experiment. Also, we thank the anonymous reviewers of IIiX 2012 for their helpful comments. This research was supported by the Netherlands Organization for Scientific Research, NWO, grant 639.022.809.

References

[1] G. Agarwal, G. Kabra, and K. C.-C. Chang. Towards rich query interpretation: walking back and forth for mining query templates. In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 1–10, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-799-8. doi: 10.1145/1772690.1772692.

[2] F. Ahmad and G. Kondrak. Learning a spelling error model from search query logs. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 955–962, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1220575.1220695.

[3] M. Bendersky and W. B. Croft. Analysis of long queries in a large scale search log. In *Proceedings of the 2009 workshop on Web Search Click Data*, WSCD '09, pages 8–14, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-434-8. doi: 10.1145/1507509.1507511.

[4] J. Brooke. Sus: A quick and dirty usability scale. In P. W. Jordan, B. Weerdmeester, A. Thomas, and I. L. Mclelland, editors, *Usability evaluation in industry*. Taylor and Francis, London, 1996.

[5] J. Callan. Distributed information retrieval. In *Advances in Information Retrieval*, pages 127–150. Kluwer Academic Publishers, 2000.

[6] K. C.-C. Chang, B. He, C. Li, M. Patel, and Z. Zhang. Structured databases on the web: observations and implications. *SIGMOD Rec.*, 33(3):61–70, 2004. ISSN 0163-5808. doi: 10.1145/1031570.1031584.

[7] D. Gayo-Avello. A survey on session detection methods in query logs and a proposal for future evaluation. *Inf. Sci.*, 179:1822–1843, May 2009. ISSN 0020-0255. doi: 10.1016/j.ins.2009.01.026. URL `http://dl.acm.org/citation.cfm?id=1523512.1523556`.

[8] M. Hagen, M. Potthast, B. Stein, and C. Braeutigam. Query segmentation revisited. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 97–106, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0632-4. doi: 10.1145/1963405.1963423.

[9] B. J. Jansen. Search log analysis: What it is, what's been done, how to do it. *Library & Information Science Research*, 28(3):407 – 432, 2006. ISSN 0740-8188. doi: 10.1016/j.lisr.2006.06.005.

[10] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 133–142, New York, NY, USA, 2002. ACM. ISBN 1-58113-567-X. doi: 10.1145/775047.775067.

[11] J. Kim and W. B. Croft. Ranking using multiple document types in desktop search. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 50–57, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0153-4. doi: 10.1145/1835449.1835461.

[12] X. Li, Y.-Y. Wang, and A. Acero. Extracting structured information from user queries with semi-supervised conditional random fields. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 572–579, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-483-6. doi: 10.1145/1571941.1572039.

[13] Y. Li, B.-J. P. Hsu, C. Zhai, and K. Wang. Unsupervised query segmentation using clickthrough for information retrieval. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information*, SIGIR '11, pages 285–294, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0757-4. doi: 10.1145/2009916.2009957.

[14] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts, 1st edition, May 1999. ISBN 978-0-262-13360-9.

[15] C. Silverstein, H. Marais, M. Henzinger, and M. Moricz. Analysis of a very large web search engine query log. *SIGIR Forum*, 33(1):6–12, Sept. 1999. ISSN 0163-5840. doi: 10.1145/331403.331405.

[16] A. Spink, D. Wolfram, M. B. J. Jansen, and T. Saracevic. Searching the web: The public and their queries. *Journal of the American society for information science and technology*, 52(3):226–234, 2001.

[17] C. W. Thompson, P. Pazandak, and H. R. Tennant. Talk to your semantic web. *IEEE Internet Computing*, 9(6):75–78, 2005. ISSN 1089-7801. doi: 10.1109/mic.2005.135.

[18] K. Tjin-Kam-Jet, D. Trieschnigg, and D. Hiemstra. Free-text search over complex web forms. In *Multidisciplinary Information Retrieval*, volume 6653 of *Lecture Notes in Computer Science*, page 14. Springer Berlin / Heidelberg, 2011. doi: 10.1007/978-3-642-21353-3_8.