Identifying Queries in Instant Search Logs

Markus Fischer[1] Kristof Komlossy[2] Benno Stein[2] Martin Potthast[3] Matthias Hagen[4]

[1]Friedrich-Schiller-Universität Jena [2]Bauhaus-Universität Weimar
[3]Leipzig University [4]Martin-Luther-Universität Halle-Wittenberg

ABSTRACT

Query logs of search engines with instant search functionality are challenging for log analysis, since the log entries represent interactions at the keystroke level, rather than at the query level. To enable log analyses at the query level, a user's logged sequence of keystroke-level interactions needs to be mapped to distinct queries. This problem bears strong parallels to session detection in "standard" query logs (i.e., forming groups of subsequent queries on the same topic), but there are salient differences. In this paper, we present a new approach to identifying interactions belonging to the same query in instant query logs. In an experimental comparison, our new approach achieves an F_2 score of 0.93 compared to only 0.83 of a state-of-the-art cascading method for query log session detection.

CCS CONCEPTS

• **Information systems → Query log analysis**.

KEYWORDS

Instant search; Query identification; Netspeak; Word search engine

ACM Reference Format:
Markus Fischer, Kristof Komlossy, Benno Stein, Martin Potthast, Matthias Hagen. 2021. Identifying Queries in Instant Search Logs. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3404835.3463025

1 INTRODUCTION

The traditional way to submit a query to a search box-based search engine is to explicitly hit enter or to click/tap on a submit button. With instant search, no such interaction is required from the user, but the query is submitted with every keystroke or once the user pauses typing for long enough (e.g., 300 ms). Instant search is as old as information systems and widespread in desktop search. Online, the most recognized deployment used to be Google's "Instant" functionality that has been discontinued in the wake of the mobile revolution. But there still are search engines that successfully employ instant search. From the available studies as well as Google's initial claims, instant search saves the user about 2–5 seconds per query and conveys a perceptible feeling of responsiveness.

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3463025

Table 1: Instant search log session of a Netspeak user with interactions merged into queries (dashed line as separator).

Time	Search box content	Time	Search box content
09:00:00	searc	*continued*	
09:00:02	searching for	09:00:51	look
09:00:06	searching for *	09:00:52	looking fo
09:00:15	looking f	09:00:53	looking for results
09:00:17	looking for results	09:00:59	for results
09:00:38	seraching for results	09:01:02	s for results
09:00:42	seching for results	09:01:04	searc for results
09:00:43	seaching for results	09:01:07	searching for results
09:00:44	searching for results	09:01:20	* for results

Table 1 exemplifies an instant query log of the Netspeak[1] word search engine [18, 22] that implements a wildcard search to support users looking for common formulations [17]. Unlike for traditional non-instant search engine logs, which log only the query strings that are intentionally submitted, the log entries of instant query logs also include "intermediate" interactions, loosely tracking a user's keystrokes. In the example in Table 1, the user seems unsure whether to write 'looking for results' or 'searching for results.' A manual analysis of Netspeak log excerpts revealed many such occasions of users switching back and forth between two (or more) formulation variants:[2] a *see-saw pattern*. In such see-saw situations (but also in general), the users may greatly benefit when their previous queries would be displayed in a clickable form. However, simply showing recently logged interactions would be too confusing. To only show the actually intended queries instead, the instant log needs to be split into groups of consecutive interactions belonging to the same query. Forming such groups also enables a general query-level analysis of the user–system interactions. However, a simple time-based splitting is insufficient, since, in the case of Netspeak, users typically submit different queries in rapid succession.

In this paper, we present a cascading approach for identifying queries in the instant search logs of Netspeak. Based on a manually annotated sample log, we combine classification rules with a downstream logistic regression classifier, to achieve high precision and convincing recall, outperforming a state-of-the-art cascading method for "standard" query log session detection and the two previous methods for query identification in instant logs. For reproducibility, we publish the Webis Netspeak Instant Log 2021 (Webis-NIL-21),[3] a carefully anonymized sample of Netspeak's query log, as well as the code underlying our experiments.[4]

[1]https://netspeak.org
[2]As an aside, many users appear to be unaware of the corresponding browser buttons.
[3]https://webis.de/data.html#webis-nil-21
[4]https://github.com/webis-de/SIGIR-21

2 RELATED WORK

At first, search session detection relied on the time gaps between consecutive queries. Different thresholds were suggested to detect *physical sessions* of "uninterrupted" search: 5 min [4, 21], 10–15 min [8], 30 min [19], and even 60 or 120 min [1]. While easily implementable, any such time-based split will merge queries from different information needs. In fact, more than half of typical physical sessions contain queries for two or more information needs [15].

To further subdivide physical sessions, *logical session* detection asks to identify consecutive queries with the same information need. Time gaps alone only yield a rather low 70–80% accuracy [12], since, for instance, queries with no shared terms often belong to different logical sessions [10]. But query patterns like repetition, specialization, and generalization [11], as well as reformulation [9] are signals indicating the same logical session. Combining such lexical cues (also, Levenshtein distance [12] or Jaccard similarity [14]) with time gaps substantially improves the detection accuracy [5, 7, 12, 14]. In addition, also semantic features have been proposed: search result overlap [3, 16, 19, 20], explicit semantic analysis (ESA) [14], Linked Open Data connections [7], or fastText embeddings [6].

Instant search log studies are by far less common than studies of "traditional" logs. Google Instant came,[5] Google Instant went.[6] Introduced at the height of desktop/laptop computing, the feature was discontinued at around the time when mobile web usage reached 50% market share.[7] A reason for introducing the feature was a claimed average savings of 2–5 seconds per query on the user side,[8] while the ones for discontinuing it were the "very different input and interaction and screen constraints" on mobile.[6] However, instant search is still deployed at scale, for example, at LinkedIn [23].

To date, only Cetindil et al. [2] and Kim and Li [13] suggested approaches to identify queries in instant search logs—either by only using normalized Levenshtein distance [2] or by combining edit distance with time gap information [13]. Our approach combines and extends these ideas with the general cascading scheme of Hagen et al. [7] to effectively and efficiently detect queries in instant logs.

3 QUERY IDENTIFICATION APPROACH

To group consecutive entries of a user's instant search log into queries, we employ a five-step process: four "perfect precision" rules followed by a logistic regression classifier. Each step receives as input only those consecutive pairs of log entries that have not already been decided by a previous step to be split or merged. The idea of using cheap rules first is inspired by the approach of Hagen et al. [7] for logical session detection. But even though logical session detection in a traditional search log bears some similarities to identifying queries in an instant search log, query identification in instant search logs happens at a "finer" granularity, and features capturing lexical (dis)similarity play an even more important role.

Step 1: Time Gap-Based Physical Sessions. Following previous studies on query log splitting [7, 13, 21], we consider a sequence of log entries with "short" time gaps between consecutive entries to be a *physical session*. Our approach thus also first splits the log into physical sessions. Regarding Netspeak's use case, we set the gap to five minutes. For web search logs, a gap of 30 minutes is common, since users may read some of the result documents. Netspeak's use case differs, since only a table of phrases matching a query is returned that indicates "commonness" via occurrence frequencies. Users may request example sentences to see a matching phrase in context, but scanning the results and looking at examples takes only seconds up to a minute. A pause of five minutes or more can thus be interpreted as separating two physical sessions—a hypothesis also supported by analyses on our manually annotated training set.

Step 2: String Containment. To cover the query typing process (e.g., from 09:00:51 to 09:00:53 in Table 1), this step checks whether $s_1 \subseteq s_2$ or $s_1 \supseteq s_2$ holds for the search box contents s_1 and s_2 of two consecutive log entries. However, users sometimes "reuse" a previous query when formulating a new one (in Table 1, at 09:00:59 the term 'looking' was deleted to type 'searching'). One reason may be that the search box is not emptied when the Netspeak browser tab is not closed. Such "reuse" behavior causes the subsequent search box content to be a substring or a superstring of the previous one. Only checking string containment would then introduce wrong merges. We avoid this by combining string containment with a time gap check. Analyzing our manually annotated training set shows that two consecutive log entries can safely be considered to belong to the same query when the time gap between them is less than 700 ms on a successful string containment check.

Step 3: Lexical Similarity Merging. This step invokes a more expensive lexical similarity check on the remaining log entry pairs that have not been decided by the two previous steps. To this end, we use the Jaccard similarity *sim* of the character trigrams of the search box strings and combine it with time gap information. Step 3 considers two consecutive log entries s_1 and s_2 to belong to the same query iff $sim(s_1, s_2) \geq 0.5$ and their time gap is below 3 seconds (parameter values determined on the pairs of the manually annotated training set that remained after Step 2).

Step 4: Lexical Dissimilarity Splitting. Consecutive log entry pairs with lexically dissimilar search box content are candidates for query splitting. However, sometimes users start their querying process with an example query from Netspeak's interface and then delete everything except the operator to start their new query. Splitting only based on lexical dissimilarity would introduce an undesired new query in such cases. Again, an additional time gap helps: Two consecutive log entries s_1 and s_2 are considered to belong to different queries iff $sim(s_1, s_2) \leq 0.05$ and their time gap is more than 30 seconds (parameter values determined on the training set). Since the lexical similarity is computed in the previous step, despite their conceptual difference, both steps are computed at once.

Step 5: Logistic Regression. The still undecided pairs elude the formulation of a basic decision rule and are thus handed off to a logistic regression classifier, employing 22 features: the time gap between consecutive log entries; the search box content length difference and ratio; string containment and lexical similarity measures; term and character overlap counts; an edit distance metric; features indicating whether an entry consists of one of Netspeak's example queries or if the operator showcased by one of them has been retained; and, to cover cases where users revisit searches to formulate new requests (e.g., from bookmarks), a feature that counts previous occurrences of the same search box content per user.

[5]https://googleblog.blogspot.com/2010/09/search-now-faster-than-speed-of-type.html
[6]https://searchengineland.com/google-dropped-google-instant-search-279674
[7]https://gs.statcounter.com/platform-market-share#quarterly-200901-202002
[8]https://web.archive.org/web/20100910003455/http://www.google.com/instant/

Table 2: (a) Evaluation of our approach on the test set. False positives (FP) are log entry pairs which were split but should belong together while false negatives (FN) are falsely omitted splits. (b) Characteristics of Netspeak's instant query log.

(a)

Approach			Decided entry pairs				Score	Run time
Step		Decision	Indiv.	Cumul.	FP	FN	F_2	per pair
1	Time gap	defer/split	9.1%	9.1%	0	0	0.68	0.0017 ms
2	Containment	defer/merge	15.9%	25.0%	0	0	0.51	0.0019 ms
3	Lexical similarity	defer/merge	38.7%	63.7%	0	1	0.70	0.0110 ms
4	Lexical dissimilarity	defer/split	1.0%	64.7%	0	0	0.75	(with Step 3)
5	Logistic Regression	merge/split	35.3%	100.0%	32	31	0.93	0.8106 ms
Our approach		merge/split	100%		**32**	31	**0.93**	0.8242 ms
Kim and Li [13]		merge/split	100%		299	**8**	0.88	0.0577 ms
Cetindil et al. [2]		merge/split	100%		299	77	0.77	0.0570 ms
Hagen et al. [7]		merge/split	100%		59	84	0.83	**0.0096 ms**

(b)

Characteristic	Value
Number of log entries	13,302,548
Number of queries	1,956,643
Number of queries with operator	635,301
Number of unique users	321,155
Number of revisiting users	45,825
Number of users active over more than one year	8,829
Average number of log entries per user	41.4
Average number of log entries per physical session	9.3
Average number of queries per user	6.1
Average number of log entries per query	6.7
Average length of query (characters)	14.4
Average length of query (terms)	2.7
Median query duration (seconds)	4.3

To further supply the logistic regression with information about the user's query formulation process, not only the feature values for a to-be-decided pair are considered, but potentially also those for preceding pairs of log entries. Since a static lookback to a fixed number of $n \geq 1$ previous log entries would often include information from a preceding different query that might confuse the classifier, we use a dynamic lookback including only the feature values of the "chain" of pairs of log entries that have been previously decided to belong to the first log entry of the now-to-be-decided pair, including the logistic regression classifier's decisions on previous pairs.

4 EVALUATION

To compare our query identification approach to the state of the art, we extract a training set and a test set from a 7-year interval of Netspeak's instant log, comprising 384,707 users (14,578,906 entries).

Preprocessing. An in-depth manual log analysis of "outlier" behavior resulted in the creation of bot removal rules. Any user who matches at least one of the following three conditions is omitted from the training and test sets: (1) more than 10 log entries in a 1-second window, (2) more than 300 log entries in a 15-minute window, (3) average search box content length exceeds 50 characters. The first rule results in some false positive removals, since connection problems may lead to some bursts of log entries when delayed requests are sent in bulk by the browser. Still, our manual analysis showed that more than 99% of the removed users can safely be viewed as bots due to their "non-human" behavior. After the removal of 1,509 bots, 13,538,048 interactions from 383,198 users remained. As a further cleansing step, we removed all non-ASCII characters from the logged queries, since Netspeak primarily served English-language requests during the period of our log excerpt. When an interaction only contained non-ASCII characters, it was removed completely. Finally, the 62,043 users with at most one remaining interaction after non-ASCII removal were also removed, since query identification does not make sense for them.

Training/Test Sampling. To create a training set and a test set, we sampled from the preprocessed data the complete logs of 520 users (38,384 log entries), thus ensuring that at least 500 users remain in case further users had to be manually removed as previously undiscovered bot users. Our sample represents the distribution of the number of interactions per user of the entire log, comprising some users with very many interactions, some with a mid-range number, etc. As a kappa test, our three human annotators each manually annotated the "query boundaries" in the log entries of three users. Since the annotators achieved a near-perfect inter-annotator agreement, each of them then independently annotated about one third of the remaining user logs. During annotation, the annotators indeed found potential bot users or developers who tested the service with few different, repeated queries, or queries that resembled a sliding window over some text. These users were removed, so that the manually annotated logs of 513 users remained (37,209 entries).

The test set is a sample of 52 users (10%) of all annotated logs (3,315 entries with 473 splits), the training set the remainder. During the evaluation, the parameter tuning of the decision rules, and the training of the logistic regression classifier, log entry pairs count as false positives when they are split but actually belong together as per the ground truth. Likewise, falsely omitted query splits count as false negatives. As effectiveness measure, we employ the F_2 score to emphasize wrong merges (i.e., false negatives) as a bigger problem than wrong splits (i.e., false positives). Both the training and the test set are released as Webis-NIL-21 alongside the paper.

Step-by-step Evaluation. Table 2a shows a step-by-step evaluation of our approach on the test set. The first step (time gap to split physical sessions) decides 9.1% of all log entry pairs without false positives, which is not surprising, since the time gap was conservatively determined on the training set. If one would stop after this step, splitting only into physical sessions, the overall F_2 score would be 0.68 with about 63% of all query borders found. Additionally invoking the second step (string containment) then correctly decides another 15.9% of the log entry pairs that they actually belong to the same query without false negatives. However, the overall F_2 score after this step drops, since stopping after this step would now imply to split every undecided pair, whereas, after the first step, all undecided pairs would be assumed to belong together.

The pairs deferred by the first two steps run through the third and fourth step (lexical (dis)similarity + time), which introduce one false negative, but still achieve an almost perfect precision. The four steps together already reliably decide 64.7% of all pairs. However, if the process stopped after these steps, the resulting F_2 score of 0.75 is not convincing. Hence, the less specific but more sensitive logistic regression step is invoked on the remaining 35.3% of the pairs.

Combining the rules with the logistic regression yields a final overall F_2 score of 0.93 (precision: 0.93, recall: 0.93). However, the 22 features to be computed for the logistic regression have a comparably high run time of about 0.8 ms per pair, dominating the overall run time of our approach. In an online scenario, its throughput on a standard machine is about 3500 entry pairs per second (2300 pairs decided at comparably "no" cost, and 1200 need be fed to the logistic regression classifier). Considering our use case of displaying only the most recent queries to a user instead of all logged instant interactions, the run time and throughput are very convincing. They are also sufficient for possible further use cases like generating suggestions in Netspeak's web interface (e.g., queries with operators to educate users exerting a see-saw behavior).

Comparative Evaluation. To put the performance of our approach into perspective, we compare it to the instant query identification methods of Cetindil et al. [2] and Kim and Li [13], and to the cascade for session detection by Hagen et al. [7]. Among the competitors, the approach of Kim and Li [13] achieves the best F_2 score of 0.88 at a run time of 0.0577 ms per pair. Since our logs do not include clicks, we omitted this feature in our reimplementation of Kim and Li's approach. One could argue that the run time investment needed to run our five-step approach still pays off due to the substantially higher effectiveness and a good throughput of 3500 pairs per second on a single standard machine.

The other two approaches achieve even lower F_2 scores at run times comparable to the approach of Kim and Li [13]. In the cascade of Hagen et al. [7], we excluded the final semantic decision steps, since they reduced the F_2 score to 0.78. This is not surprising, since merging interactions that are semantically similar but lexically different is reasonable when search sessions are to be identified in general search logs, but detrimental for instant query identification: two semantically similar entries like 'looking for results' and 'searching for results' are not part of the same instant query.

Error Analysis. During development and evaluation, we carried out extensive error analyses, identifying special cases in the logs that cause the query identification to fail: (1) Even for the users that remained after bot removal, some interaction sequences were logged without time gaps (likely due to connection problems), resulting in a random rather than the intended interaction order. Query identification cannot handle such problems. (2) Other errors were traced back to the annotations themselves. Sometimes an annotator simply missed a query split or placed the split "off-by-one" (e.g., by a twitchy mouse movement when clicking on the desired query border in our annotation tool). We corrected annotations whenever we encountered such cases, but some errors may still remain. To take such issues into account, it would make sense to assign different degrees of error severity when evaluating effectiveness—a topic not yet addressed in our study.

5 INSIGHTS FROM A FIRST LOG ANALYSIS

Our query identification approach was run on 13,302,548 Netspeak log entries of 321,155 unique users that remained after preprocessing. Some preliminary results are shown in Table 2b. About 14.3% of the users used Netspeak on more than a single day. However, this is an underestimation, since a user with dynamic IP addresses and/or deleted cookies will not be recognized as the same user; there are no other means of user tracking for privacy reasons. Among the returning users, 8,829 are "active users" who make use of Netspeak fairly regularly over at least one year after their first interaction.

The log has been split into 1,956,643 queries (6.09 queries on average per user). An average physical session contains 9.27 log entries and an average query consists of 6.65 entries with 4.34 seconds as the median time between the first and the last log entry. For a detected query, we consider the last entry as the actually intended query / search intent. These queries contain, on average, 14.38 characters and 2.69 terms (word 5-grams are the longest phrases for which Netspeak can retrieve commonness scores). An operator is used in 32.5% of the queries; 106,916 users (33.3%) use operators.

Regarding see-saw behavior, i.e., users switching back and forth between different queries to compare the results, we look for the query pattern `A(BA)+B?`: Query A, followed by Query B and then A again (and potentially also some more back and forth). In such cases, switching could be eased by showing Queries A and B in a clickable form in the web interface, or even by suggesting a query with an appropriate operator that combines the individual results (e.g., the asterisk operator at the end of the session in Table 1). Omitting Netspeak's example queries from the analysis, 34,789 query sequences actually match the pattern. At first glance, this is a non-negligible but still small amount, whereas, when viewed at the user level, it becomes clear that quite many of the more frequent users exert see-saw behavior: Among the users that used Netspeak on at least two different days, 21% (9,607 users) engaged at least once in a see-saw search. This ratio raises to 24.8% among the active users.

6 CONCLUSION AND OUTLOOK

Query log analyses are key to understand the users of a search engine. Although instant search is not a widespread feature among (online) search engines, some large-scale deployments exist, and the development of tools to analyze their logs is crucial for their improvement. In this paper, we report on a query identification approach for instant query logs. By combining near-perfect-precision steps with a subsequent logistic regression classifier, our approach achieves a high accuracy, outperforming the state of the art, while maintaining a reasonable run time that enable real-time analyses.

Our high-precision approach processed the cleaned Netspeak instant log. An analysis at the query level revealed that some users revisit previous queries within a short time frame, which suggests that they are comparing results, apparently being unaware of Netspeak's more advanced query operators. Since about 25% of the active users, i.e., the ones using Netspeak regularly over a longer time, exert this see-saw search behavior, an immediate application of our approach is a deployment as part of the user interface. Showing a user's recent, locally logged queries in an overview could help save some time when switching back and forth. With further effort, a wildcard query could be suggested that returns the union of the search results of the queries in question, and educates the users.

Another potential next step is to further investigate our hypothesis that the last interaction of a detected query represents the information need. Maybe already some intermediate interaction was the user's original need but by paying "attention" to the shown instant results they directly realized that they needed to type more—saving time compared to needing to click "search" every time.

REFERENCES

[1] Nikolai Buzikashvili and Bernard J. Jansen. 2006. Limits of the web log analysis artifacts. In *Workshop on Logging Traces of Web Activity: The Mechanics of Data Collection at the Fifteenth International World Wide Web Conference (WWW 2006), May 22–26, 2006, Edinburgh, Scotland.*

[2] Inci Cetindil, Jamshid Esmaelnezhad, Chen Li, and David Newman. 2012. Analysis of instant search query logs. In *Proceedings of the 15th International Workshop on the Web and Databases 2012, WebDB 2012, Scottsdale, AZ, USA, May 20, 2012*, Zachary G. Ives and Yannis Velegrakis (Eds.). 7–12. http://db.disi.unitn.eu/pages/WebDB2012/papers/p3.pdf

[3] Shui-Lung Chuang and Lee-Feng Chien. 2004. A practical web-based approach to generating topic hierarchy for text segments. In *Proceedings of the 2004 ACM CIKM International Conference on Information and Knowledge Management, CIKM 2014, Washington, DC, USA, November 8–13, 2004*, David A. Grossman, Luis Gravano, ChengXiang Zhai, Otthein Herzog, and David A. Evans (Eds.). ACM, 127–136. https://doi.org/10.1145/1031171.1031193

[4] Doug Downey, Susan T. Dumais, and Eric Horvitz. 2007. Models of searching and browsing: Languages, studies, and application. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI 2007, Hyderabad, India, January 6–12, 2007*, Manuela M. Veloso (Ed.). 2740–2747. http://ijcai.org/Proceedings/07/Papers/440.pdf

[5] Daniel Gayo-Avello. 2009. A survey on session detection methods in query logs and a proposal for future evaluation. *Inf. Sci.* 179, 12 (2009), 1822–1843. https://doi.org/10.1016/j.ins.2009.01.026

[6] Pedro Gomes, Bruno Martins, and Luís Cruz. 2019. Segmenting user sessions in search engine query logs leveraging word embeddings. In *Digital Libraries for Open Knowledge - 23rd International Conference on Theory and Practice of Digital Libraries, TPDL 2019, Oslo, Norway, September 9–12, 2019, Proceedings (Lecture Notes in Computer Science, Vol. 11799)*, Antoine Doucet, Antoine Isaac, Koraljka Golub, Trond Aalberg, and Adam Jatowt (Eds.). Springer, 185–199. https://doi.org/10.1007/978-3-030-30760-8_17

[7] Matthias Hagen, Jakob Gomoll, Anna Beyer, and Benno Stein. 2013. From search session detection to search mission detection. In *Open research Areas in Information Retrieval, OAIR '13, Lisbon, Portugal, May 15–17, 2013*, João Ferreira, João Magalhães, and Pável Calado (Eds.). ACM, 85–92. http://dl.acm.org/citation.cfm?id=2491769

[8] Daqing He and Ayse Göker. 2000. Detecting session boundaries from web user logs. In *Proceedings of the BCS-IRSG 22nd annual colloquium on information retrieval research, Cambridge, UK*. 57–66.

[9] Jeff Huang and Efthimis N. Efthimiadis. 2009. Analyzing and evaluating query reformulation strategies in web search logs. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2–6, 2009*, David Wai-Lok Cheung, Il-Yeol Song, Wesley W. Chu, Xiaohua Hu, and Jimmy J. Lin (Eds.). ACM, 77–86. https://doi.org/10.1145/1645953.1645966

[10] Bernard J. Jansen, Amanda Spink, Chris Blakely, and Sherry Koshman. 2007. Defining a session on web search engines. *J. Assoc. Inf. Sci. Technol.* 58, 6 (2007), 862–871. https://doi.org/10.1002/asi.20564

[11] Bernard J. Jansen, Amanda Spink, and Bhuva Narayan. 2007. Query modifications patterns during web searching. In *Fourth International Conference on Information Technology: New Generations, ITNG 2007, 2–4 April 2007, Las Vegas, Nevada, USA*, Shahram Latifi (Ed.). IEEE Computer Society, 439–444. https://doi.org/10.1109/ITNG.2007.164

[12] Rosie Jones and Kristina Lisa Klinkner. 2008. Beyond the session timeout: Automatic hierarchical segmentation of search topics in query logs. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26–30, 2008*, James G. Shanahan, Sihem Amer-Yahia, Ioana Manolescu, Yi Zhang, David A. Evans, Aleksander Kolcz, Key-Sun Choi, and Abdur Chowdhury (Eds.). ACM, 699–708. https://doi.org/10.1145/1458082.1458176

[13] Taewoo Kim and Chen Li. 2015. RILCA: Collecting and analyzing user-behavior information in instant search using relational DBMS. In *Real-Time Business Intelligence and Analytics - International Workshops, BIRTE 2015, Kohala Coast, HI, USA, August 31, 2015, BIRTE 2016, New Delhi, India, September 5, 2016, BIRTE 2017, Munich, Germany, August 28, 2017, Revised Selected Papers (Lecture Notes in Business Information Processing, Vol. 337)*, Malú Castellanos, Panos K. Chrysanthis, and Konstantinos Pelechrinis (Eds.). Springer, 3–18. https://doi.org/10.1007/978-3-030-24124-7_1

[14] Claudio Lucchese, Salvatore Orlando, Raffaele Perego, Fabrizio Silvestri, and Gabriele Tolomei. 2011. Identifying task-based sessions in search engine query logs. In *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9–12, 2011*, Irwin King, Wolfgang Nejdl, and Hang Li (Eds.). ACM, 277–286. https://doi.org/10.1145/1935826.1935875

[15] Rishabh Mehrotra and Emine Yilmaz. 2016. Query log mining for inferring user tasks and needs. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19–23, 2016, Proceedings, Part III (Lecture Notes in Computer Science, Vol. 9853)*, Bettina Berendt, Björn Bringmann, Élisa Fromont, Gemma C. Garriga, Pauli Miettinen, Nikolaj Tatti, and Volker Tresp (Eds.). Springer, 284–288. https://doi.org/10.1007/978-3-319-46131-1_36

[16] Donald Metzler, Susan T. Dumais, and Christopher Meek. 2007. Similarity measures for short segments of text. In *Advances in Information Retrieval, 29th European Conference on IR Research, ECIR 2007, Rome, Italy, April 2–5, 2007, Proceedings (Lecture Notes in Computer Science, Vol. 4425)*, Giambattista Amati, Claudio Carpineto, and Giovanni Romano (Eds.). Springer, 16–27. https://doi.org/10.1007/978-3-540-71496-5_5

[17] Martin Potthast, Matthias Hagen, Anna Beyer, and Benno Stein. 2014. Improving cloze test performance of language learners using web n-grams. In *Proceedings of the 25th International Conference on Computational Linguistics, COLING 2014, August 23–29, 2014, Dublin, Ireland*, Junichi Tsujii, and Jan Hajic (Eds.). Association for Computational Linguistics, 962–973. https://www.aclweb.org/anthology/C14-1091/

[18] Martin Potthast, Martin Trenkmann, and Benno Stein. 2010. Netspeak: Assisting writers in choosing words. In *Advances in Information Retrieval. 32nd European Conference on Information Retrieval, ECIR 2010, Milton Keynes, UK, March 28–31, 2010, Proceedings (Lecture Notes in Computer Science, Vol. 5993)*, Cathal Gurrin, Yulan He, Gabriella Kazai, Udo Kruschwitz, Suzanne Little, Thomas Roelleke, Stefan M. Rüger, and Keith van Rijsbergen (Eds.). Springer, Berlin Heidelberg New York, 672. https://doi.org/10.1007/978-3-642-12275-0_75

[19] Filip Radlinski and Thorsten Joachims. 2005. Query chains: Learning to rank from implicit feedback. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2005, Chicago, Illinois, USA, August 21–24, 2005*, Robert Grossman, Roberto J. Bayardo, and Kristin P. Bennett (Eds.). ACM, 239–248. https://doi.org/10.1145/1081870.1081899

[20] Xuehua Shen, Bin Tan, and ChengXiang Zhai. 2005. Implicit user modeling for personalized search. In *Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management, CIKM 2005, Bremen, Germany, October 31 – November 5, 2005*, Otthein Herzog, Hans-Jörg Schek, Norbert Fuhr, Abdur Chowdhury, and Wilfried Teiken (Eds.). ACM, 824–831. https://doi.org/10.1145/1099554.1099747

[21] Craig Silverstein, Monika Rauch Henzinger, Hannes Marais, and Michael Moricz. 1999. Analysis of a very large web aearch engine query log. *SIGIR Forum* 33, 1 (1999), 6–12. https://doi.org/10.1145/331403.331405

[22] Benno Stein, Martin Potthast, and Martin Trenkmann. 2010. Retrieving customary web language to assist writers. In *Advances in Information Retrieval. 32nd European Conference on Information Retrieval, ECIR 2010, Milton Keynes, UK, March 28–31, 2010, Proceedings (Lecture Notes in Computer Science, Vol. 5993)*, Cathal Gurrin, Yulan He, Gabriella Kazai, Udo Kruschwitz, Suzanne Little, Thomas Roelleke, Stefan M. Rüger, and Keith van Rijsbergen (Eds.). Springer, Berlin Heidelberg New York, 631–635. https://doi.org/10.1007/978-3-642-12275-0_64

[23] Ganesh Venkataraman, Abhimanyu Lad, Viet Ha-Thuc, and Dhruv Arya. 2016. Instant search: A hands-on tutorial. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17–21, 2016*, Raffaele Perego, Fabrizio Sebastiani, Javed A. Aslam, Ian Ruthven, and Justin Zobel (Eds.). ACM, 1211–1214. https://doi.org/10.1145/2911451.2914806