Efficient Query Obfuscation with Keyqueries

Maik Fröbe, Eric Oliver Schmidt, and Matthias Hagen

Martin-Luther-Universität Halle-Wittenberg
<first name>.<last name>@informatik.uni-halle.de

Abstract. Search engine users who do not want a sensitive query to actually appear in a search engine's query log can use query obfuscation or scrambling techniques to keep their information need private. However, the practical applicability of the state-of-the-art obfuscation technique is rather limited since it compares hundreds of thousands of candidate queries on a local corpus to select the final obfuscated queries. We propose a new approach to query obfuscation combining an efficient enumeration algorithm with so-called keyqueries. Generating only hundreds of candidate queries, our approach is orders of magnitude faster and makes close to real time obfuscation of sensitive information needs feasible.
Our experiments in TREC scenarios on the ClueWeb corpora show that our approach achieves a retrieval effectiveness comparable to the previous exhaustive candidate generation at a run time of only seconds instead of hours. Overall, 75 % of the private information needs can be obfuscated by retrieving at least one relevant document of the original private query—that itself will not appear in the search engine logs. To further improve a user's privacy, the query obfuscation can easily be combined with other client-side tools like TOR routing, TrackMeNot or PEAS fake queries.

1 Introduction

Privacy and confidentiality are current challenges in the field of information retrieval [11]. One respective research direction addresses the anonymization of query logs [20, 25, 39] to avoid scenarios like the deanonymization of users when the AOL query log was released.[1] However, any such privacy techniques applied on the search engine side require users to trust the search engine. This trust might be unacceptable for users with some very sensitive information need, especially given the recent news that the police can access query logs;[2] unthinkable and in conflict with the United States Bill of Rights if someone used traditional libraries to obtain their information.

User-side query obfuscation techniques can improve privacy and confidentiality without any need to trust the search engine. Corresponding algorithms [3–6] derive alternative queries that are submitted to the search engine instead of the actual query. Hence, the sensitive query itself never appears in the search engine's log, which would happen for other privacy techniques like hiding the actual query in a stream of fake queries—an attacker would then still know that the sensitive query exists. The challenge of query obfuscation techniques is to derive alternative, less sensitive queries that still return results relevant to the

[1] https://www.nytimes.com/2006/08/09/technology/09aol.html
[2] https://www.cnet.com/news/google-is-giving-data-to-police-based-on-search-keywords-court-docs-show/

original query to improve a user's privacy and confidentiality in the sense of not directly revealing "secrets" that a user perceives as private.

The existing query obfuscation approaches [3–6] (cf. Section 2 for details) derive queries that retrieve similar results as the original private query on a local corpus. While those approaches are effective in coming up with good alternative queries that retrieve results similar to the private original query, they do so by running hundreds of thousands of candidate queries against the private corpus. To reduce the computational effort and to enable an actual practical applicability with close to real time efficiency, we propose a new approach (cf. Section 3) that generates only hundreds of candidates while maintaining the effectiveness of the previous approaches. Our new approach first retrieves the top results of the private query on the private corpus as target documents and from them extracts promising keywords and keyphrases as vocabulary of obfuscated queries. To formulate minimal query obfuscation candidates, we employ keyqueries [17, 18] that retrieve (some of) the target documents in the top ranks against the private corpus. Skipping non-minimal queries (in the sense of term sets) and using a practically efficient enumeration algorithm from the field of hypergraph transversal generation, our new approach is orders of magnitude more efficient than the previous obfuscation methods. In a final step, the candidate obfuscation queries are ranked and optionally presented to the user to select the queries that can be submitted to the public search engine. The results of these queries are then locally re-ranked and presented to the user.

We compare our new query obfuscation algorithm to the existing approach of Arampatzis et al. [4] on 63 TREC Web Track topics with sensitive information needs on the ClueWeb09 and ClueWeb12 crawls (cf. Section 4). In the evaluation, we simulate that users obfuscate queries on a small local corpus and then submit the obfuscated queries to a search engine indexing a disjoint and much larger corpus, and we also conduct a user study on candidate queries that should be removed to not reveal the private information need. Altogether, ignoring non-minimal queries as our efficient enumeration scheme does comes with a nice regularizing effect: often it even increases the retrieval effectiveness while at the same time reducing the number of generated candidates between 17 %–19 % and dramatically reducing the overall number of submitted queries by orders of magnitude. The approach with the best efficiency/effectiveness tradeoff obtains a Precision@10 of 0.38 on par with the state-of-the-art query obfuscation of Arampatzis et al. [4], while on average submitting only 784 candidate queries instead of hundreds of thousands.

2 Related Work

We reviews the existing tools for private web search that improve users privacy without requiring any trust in search engines, and existing approaches to query obfuscation as an orthogonal technique to current privacy tools.

Tools for Private Web Search. Popular search engines log user interactions to enable continuous improvements, e.g., by learning ranking models [22, 27], or creating user profiles [33]. Collecting the required query logs received negative attention [35], and, although not intended, server-side anonymization can fail [23].

Hence, many privacy techniques aim at impeding data collection by search engines, hiding the private query within additional cover queries [35, 2, 13, 38]. TrackMeNot submits cover queries sampled from popular queries [35]. Ensuring some semantic similarity allows selecting cover queries similar to the private query on unrelated topics, thus better hiding the private query [29]. However, users may experience suboptimal retrieval performance since cover queries can introduce noise to their profile [9], which motivates fine-grained adaptions to cover queries following user-defined personalization / privacy tradeoffs [1, 2]. Instead of submitting additional cover queries, PEAS combines the private query with multiple fake queries using a logical OR [30]. Other approaches use anonymous communication via TOR routing [16, 12], or intercept HTTP communication [31], filtering out identifying information such as cookies or timing attacks. Altogether, those tools improve the privacy of web search users, but they still submit the private query to the search engine, revealing the existence of the private information need that ends up as a query in the search log.

Query Obfuscation. Query obfuscation approaches, introduced by Arampatzis et al. [5], replace the private query with a set of less sensitive queries and combine the results at the searcher's end into a single ranking, preventing the private query from being submitted to the search engine at all. Query obfuscation works in two steps. In the first step, a candidate generation produces candidate queries. Second, a candidate selection chooses the candidate queries submitted to the public search engine. There are semantic and statistical approaches to query obfuscation, applying different methods to the candidate generation and selection.

Semantic query obfuscation [5, 6] leverages semantic relations between terms, replacing the private query with sets of queries representing more general concepts. The WordNet taxonomy [28] is used to generate candidate queries by extracting hyponyms, i.e., more general terms with an "is-kind-of" relationship to a query term. Query candidates are selected with a user-defined similarity threshold, where semantic similarity is measured using the distance in WordNet [32], discarding candidate queries above the threshold. Semantic query obfuscation comes with two drawbacks. First, obfuscated queries have a semantic relationship to the private query, introducing the risk of revealing the private information need. Second, the extraction of terms from WordNet does not take into account that some terms are more effective in retrieving relevant documents than others.

Statistical query obfuscation [3, 4] addresses both problems by using a private search engine for the generation and selection of candidate queries, yielding a better retrieval effectiveness. The private query is submitted to the private search engine, using the top-ranked documents as target documents for candidate generation and candidate selection. During candidate generation, a sliding window covering 16 consecutive terms moves over the target documents, emitting all one-, two-, and three-term combinations occurring in the same sliding window as candidates. Afterwards, the candidate selection removes candidate queries that do not meet the user specified privacy level by removing queries that retrieve too few documents on the private search engine. Finally, the remaining candidate queries with the highest pointwise mutual information for the target documents are submitted to the public search engine.

Arampatzis et al. [5, 6] conducted a recall-oriented evaluation of query obfuscation on the ClueWeb09B corpus using 95 sensitive queries extracted from the

AOL query log. The private search engine indexes 50,000 documents sampled from the ClueWeb09B. Statistical query obfuscation retrieves 56–76 % of the top-50 documents retrieved for the private query, compared to 22–25 % retrieved by semantic query obfuscation. However, those excellent results come at non-negligible costs: first, the number of candidate queries submitted to the private search engine is huge, and second, Arampatzis et al. apply only a lightweight privacy level that leads to, for instance, obfuscating the private query `gun rack` by the not really less sensitive queries `gun` or `gun light`.

Our new approach to query obfuscation drastically reduces the number of candidates examined by statistical query obfuscation while maintaining the good retrieval effectiveness. Additionally, we increase the privacy level (e.g., prohibiting that the query `gun rack` is obfuscated by `gun`). Considering the popular hypothesis that information needs are non-specifiable [7], we believe that only users can assess if an obfuscated query reveals the private information need or not. Hence, we include a step where the user must confirm the obfuscated queries before they are submitted to the public search engine. We evaluate this setup in a user study, supporting the participants with automatic filtering inspired by semantic query obfuscation.

3 Approach

We describe one existing and two new exhaustive candidate generation approaches, followed by two candidate selection approaches. Finally, we present our new efficient approach that reduces the number of candidates based on the concept of keyqueries and an efficient enumeration algorithm.

3.1 Vocabulary Extraction and Exhaustive Candidate Generation

The goal of the vocabulary extraction is to extract vocabulary that produces candidate queries being able to replace a given private query with the help of a private search engine. Therefore, vocabulary extraction approaches submit the private query to the private search engine, using the top-n ranked documents (we set $n = 10$) as target documents. Additionally, each approach uses a filter list to remove vocabulary items that would reveal the private information need. We describe three approaches that extract the vocabulary for query obfuscation from the target documents, and exhaustive candidate generation approaches combining this vocabulary into unordered combinations as candidate queries.

To automatically remove candidate queries that would likely reveal the private information need, we create a filter list specific to the private query. This filter list contains all terms from the private query and, inspired by semantic query obfuscation [5, 6], synonyms, hyponyms, and hypernyms of private query terms extracted with WordNet [28] since such terms would quickly reveal the private information need. We normalize the filter list with Lucene's default tokenization, lower casing, and stemming terms with the Porter stemmer [36], covering inflected forms of revealing terms. Using this filter list, we employ a more restrictive level of privacy as Arampatzis et al. [4], which only ensured that the original private query is not submitted, e.g., `gun` or `gun light` were valid obfuscated queries for a private query `gun rack`. Hence, our filter list ensures that such revealing relationships are automatically removed, and we review the remaining queries to further remove sensitive candidate queries in a user study.

Sliding Window As proposed by Arampatzis et al. [3, 4], we employ the sliding window approach to generate multi-term candidate queries. Therefore, we extract the main content of the target documents with Boilerpipe [24] and follow Arampatzis et al. by moving a sliding window, covering 16 subsequent terms as vocabulary, over each target document's main content. From the vocabulary, we exclude stop words using Lucene's default stop word list for English, and items occuring in the filter list since those terms may reveal the private information need as all words from the filter list have a direct semantic relationship to the private query. Given the vocabulary of a window, we emit all unique unordered one-, two-, and three-term combinations. Arampatzis et al. choose 16 terms to assure that extracted terms are related to each other [34], ensuring in combination with the maximum length of three terms per candidate query that the number of candidate queries does not explode.

Still, the candidate generation with the sliding window is arguably a brute force method, having advantages and disadvantages. On the one hand, the exhaustive nature of the generated candidates may serve as a valuable empirical upper bound to the retrieval effectiveness. On the other hand, the number of candidate queries generated by the sliding window easily becomes difficult to handle in practice because the candidate selection submits all candidate queries to the private search engine. A single window, containing a vocabulary of 16 unique terms, emits $\binom{16}{1} + \binom{16}{2} + \binom{16}{3} = 696$ unordered one-, two-, and three-term combinations. Additionally, the number of generated queries is not limited, increasing with the length of the extracted unique main content of the target documents.

Noun Phrase Given a set of target documents, we extract noun phrases from the main content of each target document using the top-p noun phrases (we set $p = 10$) as vocabulary to generate candidate queries. We choose noun phrases as they mimic typical keyword searches. In contrast to sliding window candidate generation, the noun phrase approach provides exact estimates of the number of generated candidates, since p unique noun phrases as vocabulary produce $2^p - 1$ unique unordered non-empty combinations as candidate queries (i.e., the power set of the query vocabulary minus the empty query). Given a vocabulary of top-p noun phrases, the noun phrase candidate generation emits all possible $2^p - 1$ unordered combinations as candidate queries.

To extract noun phrases from the target documents, we derive the main content with the help of Boilerpipe [24]. Afterwards, we use the OpenNLP Part-of-Speech-Tagger[3] to extract term groups of two to five subsequent terms that contain at least one noun and otherwise only adjectives, removing all noun phrases with a stemmed term occuring in the filter list of the private query since such noun phrases might reveal the private information need. Moreover, we remove (near-)duplicated noun phrases by normalizing noun phrases with lowercasing and stemming, retaining the most frequent unprocessed noun phrase of a group of duplicates, counting the duplicates for subsequent steps. We sort extracted noun phrases descending by their normalized frequency (number of target documents that contain the normalized noun phrase), breaking ties by the total number of occurences in the target documents. This strategy worked well in first pilot

[3] opennlp.apache.org

experiments, and is motivated by the idea that more common phrases tend to be less specific and therefore less sensitive or even insensitive to the private query.

TF-IDF Our new TF-IDF approach allows combining terms located at distant positions while maintaining full control over the number of generated candidate queries. Opposite to the previous vocabulary extraction approaches that a vocabulary with terms close to each other (e.g., within a 16 term sliding window), effective queries might combine terms not co-occurring close to each other.

To maintain a small and controllable number of generated candidates for our new TF-IDF vocabulary while discarding locality restrictions, we exploit the idea that terms occuring often in a target document but rarely in other documents retrieve this target document at high positions. Therefore, the private search engine must expose read access to the indexed document vectors to enable fast lookup of the document frequency and term frequency to calculate TF-IDF scores of terms. Since standard baselines in retrieval experiments, such as BM25 with RM3 [26], require document vectors, it is reasonable to assume that the private search engine provides access to document vectors as well.

Given n target documents (we set $n = 10$) and access to document vectors, our new TF-IDF approach ranks the terms of a target document descending by their TF-IDF score, discarding terms that occur on the filter list specific to the private query. The terms with the highest TF-IDF score are promising for retrieving the target document at the top positions. Hence, we select the t terms (we set $t = 7$) with the highest TF-IDF scores for a target document as vocabulary. Since terms in the index are already normalized, typically via stemming or lemmatization, we replace the top index terms with the corresponding raw terms extracted from the documents, emitting all unordered combinations of the t terms as candidates for the target document. The maximum number of candidate terms generated by the TF-IDF candidate generation with n target documents using t terms per document is $n \cdot (2^t - 1)$, resulting in 1270 candidates for our parameter choices.

3.2 Candidate Selection

Candidate selection approaches score candidate queries generated for target documents of a private query with the help of the private search engine, inducing a ranking of candidate queries from which the top candidates are selected. Therefore, a candidate query is submitted to the private search engine. We use the pointwise mutual information proposed by Arampatzis et al. [4], and the normalized discounted cumulative gain as candidate selection approaches.

Arampatzis et al. [4] conduct pilot experiments using a comprehensive collection of measures for the selection of candidate queries (precision, recall, F-measure, pointwise mutual information, normalized pointwise mutual information), finding that the pointwise mutual information works best for the candidate selection. Additionally, we employ the normalized discounted cumulative gain (nDCG) [21] that performed well during our pilot experiments. Other measures, like MAP, would be suited as well, but we prefer nDCG because MAP faces some critics nowadays because of a rather unrealistic user model [15]. Given a set of target documents D, we assign each target document an information gain of 1, and all other documents 0. This information gain ensures that all candidate queries retrieving the target documents at the top positions obtain a perfect nDCG of 1.

Algorithm 1 Keyquery generation using the HBC enumeration scheme

Input: Target document set D; keyquery parameters k, l, and m; query building vocabulary V; maximal query complexity c; candidate selection s
Output: Keyqueries Q and keyquery candidates $C_1 \cup \ldots \cup C_c$ ranked by S

$Q \leftarrow \emptyset \quad S \leftarrow \emptyset \quad C_1 \leftarrow V \quad C_n \leftarrow \emptyset \; \forall \; n \in \{2, \ldots, c\}$
for all $v \in V$ **do**
  $\pi \leftarrow$ search results for v against private search engine
  if π has at least l search results **then**
    $S[v] \leftarrow s(\pi)$
    $D' \leftarrow \{$ top-k search results from π $\}$
    if v is keyquery, i.e., $|D' \cap D| \geq m$ **then**
      $Q \leftarrow Q \cup \{v\}$
      $C_1 \leftarrow C_1 \setminus \{v\}$
$i \leftarrow 1$
while $C_i \neq \emptyset \wedge i < c$ **do**
  for all $c, c' \in C_i$ with $|c \cap c'| = i - 1$ **do**
    $\hat{c} \leftarrow c \cup c'$
    if $\hat{c}$ has not been generated yet **then**
      if $\hat{c} \setminus \{v\} \in C_i$ for all $v \in \hat{c}$ **then**
        $\pi \leftarrow$ search results for $\hat{c}$ against private search engine
        if π has at least l search results **then**
          $S[\hat{c}] \leftarrow s(\pi)$
          $D' \leftarrow \{$ top-k search results from π $\}$
          if $\hat{c}$ is keyquery, i.e., $|D' \cap D| \geq m$ **then**
            $Q \leftarrow Q \cup \{\hat{c}\}$
          else
            $C_{i+1} \leftarrow C_{i+1} \cup \{\hat{c}\}$
  $i \leftarrow i + 1$

3.3 Keyquery-Based Candidate Query Generation

Our approach constructs keyqueries [17, 18] for the target documents of the private query from a given query vocabulary, reducing the number of generated candidates by exploiting the minimality constraint of keyqueries. Following the definition of Hagen et al. [18], we view a query q as a *keyquery* for a set of target documents D against a private search engine P, iff q fulfills the following conditions: (1) every $d \in D$ is in the top-k results returned by P for q, (2) q has at least l results, and (3) no $q' \subset q$ fulfills the first two conditions.

The first two conditions define the specificity and the generality of a keyquery with parameters k and l. We set $k = 10$ and $l = 100$ to ensure that a keyquery does not reveal the private information need by retrieving exactly the target documents. The third condition is a minimality constraint allowing us to skip non-minimal query candidates. We argue that minimality is important since the private search engine differs from the public search engine in the indexed documents and the retrieval model. Hence, minimality avoids constructing too complex queries when a query is already "good enough", thus mitigating overfitting.

Given query vocabulary V for a private query obtained from some extraction approach, the power set $\mathcal{Q} = 2^V$ forms all possible obfuscated queries. It will not always be possible for $\mathcal{Q}$ to contain queries that retrieve all target documents D within the top-k results (even for large k). Hence, we relax the first keyquery con-

dition and require that a keyquery retrieves at least m of the target documents D within the top-k results (we set $m = 3$ determined in pilot experiments).

Algorithm 1 gives the pseudocode of our approach to generate obfuscation candidates. The algorithm efficiently enumerates all queries from $\mathcal{Q}$ while skipping queries that violate the minimality constraint, thus reducing the number of generated candidates and mitigating overfitting of queries on the private search engine. The algorithm is a modified version of the HBC algorithm [19] (named after the authors Hébert, Bretto and Crémilleux) originally proposed for the problem of transversal hypergraph generation. Even though not being output-polynomial for that general generation problem [14], the basic algorithmic idea helps us to efficiently enumerate candidate queries up to a specified fixed length of c items from the vocabulary in c stages (we set $c = 3$ for sliding window, $c = 7$ for TF-IDF, and $c = 5$ for noun phrases since this worked best in pilot experiments). A vocabulary extraction approach may produce multiple vocabulary sets, such as the sliding window scheme creating a vocabulary set for each sliding window. In this case, we run the HBC algorithm per vocabulary set, caching results for duplicated query candidates constructed from different sliding windows. In the first stage (lines 2–9), our approach identifies all items from the vocabulary that are keyqueries, ensuring that no superset of those vocabulary items is enumerated (line 9). In the i-th stage (lines 12–23), all valid candidates of length $i + 1$ (consisting of $i + 1$ vocabulary items) are generated. By combining only valid candidates from the previous stage (line 12), the HBC enumeration scheme ensures that the minimality criterion is not violated and in our case to reduce the risk of revealing the private information need by retrieving less than l results (such candidates are pruned in previous rounds in lines 17 and 20).

4 Evaluation

To experimentally compare the algorithms, we obfuscate 63 TREC Web Track topics with sensitive information needs on a trusted private search engine with 50 m documents. After a user study to ensure that obfuscated queries do not reveal the sensitive information need, we submit obfuscated queries to an public search engine, measuring retrieval effectiveness in Cranfield-style experiments.

4.1 Experimental Design

Topics with sensitive information needs We use 63 sensitive information needs originating from the TREC web tracks labeled for the ClueWeb09 and the ClueWeb12 in our evaluation since they come with many relevance judgments. We review the originally 300 web track topics and find that 63 of them specify information needs that users might want to obfuscate (primarily health-related). Table 1a provides an overview over the 4 categories of private information needs from our 63 topics with examples and short descriptions. For each category, we describe why users might want to obfuscate an associated information need.

Experimental Setup Our setup simulates users with access to a relatively small private search engine who want to obfuscate private information needs from a much larger public search engine. We ensure that the private and the public

Table 1. Overview over the private information needs (a) and the corresponding search engines used for topics from the ClueWeb09 (CW09; private search engine indexes CW12b13, public search engine indexes CW09) and the ClueWeb12 (CW12; private search engine indexes CW09b, public search engine indexes CW12).

(a) Private Information Needs

	Example Topics	
Category	Nr.	Query
Health (35 Topics)	26	lower heart rate
A user wants health advice	88	forearm pain
while hiding a potential disease.	266	symptoms heart attack
Personal (22 Topics)	11	gmat prep classes
A user considers a personal change and	18	wedding budget calculator
wants to hide this before it is definitive.	57	ct jobs
Law/Crime (3 Topics)	59	how to build a fence
A user wants legal advice while being	61	worm
able to reject any criminal intend.	62	texas border patrol
Family Pets (3 Topics)	38	dogs for adoption
A user has fears regarding a pet and	50	dog heat
wants to hide them until a solution.	111	lymphoma in dogs

(b) Search Engines

		Topics	
		CW09	CW12
Private	Corpus	CW12b13	CW09b
	Documents	52.3 m	50.2 m
	Size	28.9 GB	27.9 GB
	with vectors	126.3 GB	126.1 GB
	Retrieval	Anserini [37] in 2 separate variants: BM25 vs. QLD.	
Public	Corpus	CW09	CW12
	Documents	1.0 b	733.0 m
	Size	3.6 TB	3.1 TB
	Retrieval	ChatNoir [8] with BM25F and multiple fields.	

search engine differ in all essential parts. Since we are using topics annotated for the ClueWeb09 respectively the ClueWeb12, we use ChatNoir [8] as the public search engine from which users want to hide their private information need. ChatNoir is a publicly available research search engine indexing the complete ClueWeb09 and ClueWeb12 using main content extraction, language detection, and metadata extraction (keywords, headings, hostnames, etc.). ChatNoir ranks documents by combining BM25 scores of multiple fields (title, keywords, main content, and the full document) and uses the documents' SpamRank [10] to remove spam. As the private search engine, we use two Anserini [37] based search engines, one with BM25 and one with QLD as the retrieval model (with and without document vectors required for the TF-IDF candidate generation), to simulate larger differences in the retrieval model between the private and public search engine. Anserini employs a completely different document processing as ChatNoir, using the full document text, without main content extraction, for retrieval. Table 1b gives an overview of the search engines used per topic. For a ClueWeb09 topic, we use ChatNoir's ClueWeb09 index as public search engine, and Anserini with BM25 (respectively QLD) on the ClueWeb12b13 as the private search engine (vice versa for ClueWeb12 topics). With this setup, the private search engine has a substantially smaller document corpus as the public search engine originating from a different crawl with considerably various rankings.

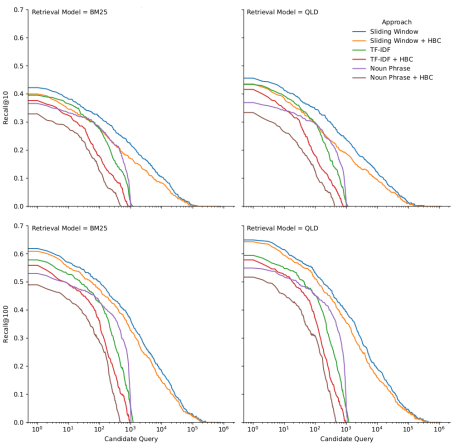
Candidate Generation We execute our three candidate generation approaches with and without HBC algorithm on the 63 sensitive topics using a private search engine with BM25 (respectively QLD). Table 2a shows the number of candidate queries generated on average per topic. Unsurprisingly, the sliding window generates huge numbers of candidates, over 250 000 for BM25 (even more for QLD), all of which must be submitted to the private search engine during candidate selection, causing runtimes in the range of multiple hours, even with high parallelization. The TF-IDF and noun phrase approach generate easily manageable numbers of candidates (less than 0.5 % of sliding window). Applying the HBC algorithm reduces the number of generated candidates between

Table 2. Overview of the number (a) and the estimated quality (b) of the sliding window (SW) [4], TF-IDF, and noun phrase (NP) candidate generation approaches without and with HBC. The quality of candidates is estimated by measuring Recall in retrieving the target documents for the private query on the private search engine. We report for a private search engine using BM25 and one using QLD as retrieval models.

(a) Number of Generated Candidates

		Candidate Generation		
		SW	TF-IDF	NP
BM25	Candidates	253560.1	1075.8	1023.0
	with HBC	212409.8	801.8	323.9
	Query Length	2.7	3.6	12.6
	with HBC	2.6	3.1	8.2
	Recall@10 > 0.5	507.7	51.8	67.5
	with HBC	35.4	8.1	3.6
	Recall@100 > 0.5	1851.6	144.1	246.4
	with HBC	979.4	64.6	41.8
	Vocabulary	1117.7	41.5	21.6
QLD	Candidates	318088.1	1044.0	1023.0
	with HBC	266533.7	766.8	316.1
	Query Length	2.7	3.6	12.5
	with HBC	2.6	3.1	8.0
	Recall@10 > 0.5	763.5	57.0	77.5
	with HBC	87.4	10.6	3.0
	Recall@100 > 0.5	2719.8	160.4	260.8
	with HBC	1483.5	74.1	43.7
	Vocabulary	1302.1	40.5	21.0

(b) Estimated Quality of Generated Candidates



17 %–19 %, skipping more than 40 000 queries for the sliding window approach, substantially reducing the number of queries to be submitted to the private search engine. Table 2b provides an overview over the Recall@10 (and Recall@100) of the candidate generation approaches in retrieving the target documents on the private search engine. The Recall of candidate queries is averaged over topics, sorting candidate queries by their per-topic Recall on the X-axis (starting with the best candidate of a topic at X=1). The sliding window approach generates candidates with the highest Recall, which comes at no surprise since this approach is arguably a brute force enumeration. TF-IDF candidates follow closely, providing excellent candidates considering they generate less than 0.5 % candidates as the sliding window. In all cases, generating candidates using QLD achieves slightly better Recall. Applying the HBC algorithm reduces the Recall of the generated candidate queries slightly. However, this is intended as HBC does not expand "already good" queries, hence mitigating overfitting on the private search engine.

User Study We conduct a user study with computer science students and scholars to identify candidate queries that reveal the private information need and remove those queries from subsequent evaluations. Our study intends users of query obfuscation to confirm all candidate queries before submitting them to the public search engine. This manual process is important because users with sensitive information needs may not trust automatic approaches removing sensitive queries.

The study itself consists of two steps. First, we collect candidate queries for all topics by selecting the top 25 candidate queries from all candidate generation and selection combinations (query terms, synonyms, hyponyms, or hypernyms of a query term are already removed). In the second step, annotators judge whether a candidate query might reveal the private information need. The annotators started by familiarizing themselves with the topic, reading the query, description,

etc. After having understood the topic and potential privacy threats described by the privacy category, an annotator judges candidate queries in random order.

The annotators assess the candidate queries on a 3-point scale to identify garbage queries (-1), queries that reveal the private information need (0), and queries they would submit to the public search engine (1). The garbage label (-1) indicates low-quality queries, consisting of many spelling mistakes or nonsense words that would make such a query conspicuous for the receiving search engine. An example labeled as garbage within our user study is `lickspringscasino` (instead of `lick springs casino`). Annotators were not allowed to consult additional material (like searching the web) to assess whether a query reveals the private information need since real users would not be able to consult the web without submitting the query to the public search engine. Still, we ensure a higher privacy level than previous work (cf. Section 2).

During our user study, 21 216 candidate queries were labeled by two annotators. The Fleiss' κ of 0.67 indicates a good agreement between the annotators. Overall, 252 candidate queries were judged as garbage (label -1), 1 553 as revealing (label 0), and the remaining 19 350 as non-revealing (label 1). We remove candidate queries labeled as garbage or revealing (label 0 or -1) from subsequent evaluations.

4.2 Experimental Results

Table 3 shows the retrieval performance of all approaches, submitting 5 (respectively 20) obfuscated queries to the public search engine, retrieving the top-10 or top-100 results per query. All retrieved results are re-ranked on the user-side with Anserini's BM25 retrieval model, measuring the average number of relevant documents retrieved and Precision@10 in the user's ranking. Overall, we observe that selecting candidate queries with the pointwise mutual information achieves lower effectiveness than normalized discounted cumulative gain. Confirming our expectations, submitting more obfuscated queries, and retrieving more documents from the public search engine per query, improves the effectiveness (the right column of Table 3b shows the best results). In most cases, the Recall-observations on the private search engine (Table 2b) transfer to the effectiveness that we observe on the obfuscated queries on the public search engine, i.e., the sliding window approach outperforms TF-IDF, and the noun phrase approach obtains the lowest retrieval effectiveness.

However, there is one noticeable difference: applying the HBC algorithm often improves the effectiveness, both in terms of retrieved relevant documents and Precision@10. Consequently, the best obfuscation approach using BM25 as private search engine is obtained by the sliding window approach with efficient enumeration of candidate queries by the HBC algorithm (retrieving 11.85 relevant documents on average at a Precision@10 of 0.39). The sliding window approach is closely followed by the TF-IDF enumeration with HBC (Precision@10 of 0.38). Using QLD as retrieval model of the private search engine shows the same figure, yielding only a slightly better Precision@10 of 0.40. We further inspect the number of topics for which at least one relevant document can be retrieved, finding a maximum of 75 % for TF-IDF and sliding window (both using HBC). This is expected since some topics in our pool have only a few relevant documents, and

Table 3. Overview of the average number of retrieved relevant documents (Rel.) and the Precision@10 (Prec@10) in case 5 (respectively 20) obfuscated queries are submitted to the public search engine, retrieving the top 10 (respectively top 100) documents from the public search engine per query. Results are reported for 3 candidate generation approaches (without and with HBC), nDCG respectively PMI as candidate selection approaches, and BM25 respectively QLD as retrieval model of the private search engine.

(a) Top 10 Results for Queries Selected with nDCG

		5 Queries		20 Queries	
		Rel.	Prec@10	Rel.	Prec@10
BM25	Sliding Window	1.31	0.12	3.02	0.21
	+ HBC	1.16	0.11	2.69	0.22
	TF-IDF	0.87	0.09	2.00	0.19
	+ HBC	0.92	0.09	2.11	0.19
	Noun Phrase	0.72	0.07	1.33	0.11
	+ HBC	0.95	0.09	1.64	0.14
QLD	Sliding Window	1.59	0.15	3.31	0.23
	+ HBC	1.18	0.12	3.03	0.22
	TF-IDF	1.18	0.12	2.21	0.21
	+ HBC	1.08	0.11	2.56	0.22
	Noun Phrase	0.74	0.07	1.15	0.12
	+ HBC	0.62	0.06	1.38	0.11

(b) Top 100 Results for Queries Selected with nDCG

		5 Queries		20 Queries	
		Rel.	Prec@10	Rel.	Prec@10
BM25	Sliding Window	6.39	0.28	10.70	0.34
	+ HBC	6.00	0.29	11.85	0.39
	TF-IDF	3.26	0.21	6.77	0.35
	+ HBC	3.11	0.23	7.69	0.38
	Noun Phrase	1.84	0.12	2.97	0.18
	+ HBC	2.31	0.16	4.02	0.25
QLD	Sliding Window	5.57	0.29	9.39	0.40
	+ HBC	5.49	0.29	10.97	0.40
	TF-IDF	4.28	0.24	7.93	0.37
	+ HBC	4.30	0.25	8.66	0.39
	Noun Phrase	1.75	0.16	2.74	0.18
	+ HBC	2.44	0.17	4.34	0.23

(c) Top 10 Results for Queries Selected with PMI

		5 Queries		20 Queries	
		Rel.	Prec@10	Rel.	Prec@10
BM25	Sliding Window	0.15	0.02	0.52	0.05
	+ HBC	0.20	0.02	1.41	0.13
	TF-IDF	0.84	0.08	1.97	0.16
	+ HBC	0.80	0.08	1.90	0.15
	Noun Phrase	0.89	0.08	1.61	0.15
	+ HBC	1.08	0.10	1.89	0.17
QLD	Sliding Window	0.11	0.01	0.49	0.05
	+ HBC	0.20	0.02	0.93	0.09
	TF-IDF	0.54	0.05	1.62	0.14
	+ HBC	0.49	0.05	1.54	0.13
	Noun Phrase	0.64	0.06	1.26	0.12
	+ HBC	0.62	0.06	1.11	0.10

(d) Top 100 Results for Queries Selected with PMI

		5 Queries		20 Queries	
		Rel.	Prec@10	Rel.	Prec@10
BM25	Sliding Window	0.34	0.03	1.79	0.13
	+ HBC	0.69	0.06	5.89	0.29
	TF-IDF	3.52	0.22	7.31	0.34
	+ HBC	3.54	0.23	8.67	0.35
	Noun Phrase	2.49	0.16	3.77	0.23
	+ HBC	3.30	0.20	4.49	0.24
QLD	Sliding Window	0.36	0.03	1.23	0.09
	+ HBC	0.56	0.05	3.33	0.19
	TF-IDF	2.75	0.16	6.23	0.28
	+ HBC	2.56	0.16	6.51	0.30
	Noun Phrase	2.66	0.16	4.11	0.22
	+ HBC	2.70	0.17	4.28	0.22

we apply a rigorous privacy level, making it unrealistic that all information needs can be obfuscated at such a high privacy. Overall, our results show that our new approach, especially combining TF-IDF candidates with the HBC algorithm, achieves state-of-the-art retrieval performance while drastically reducing the number of generated candidates.

5 Conclusion and Future Work

We have presented an approach inspired by previous query obfuscation techniques. Still, instead of prohibitively low efficiency, the application of an enumeration scheme from the transversal hypergraph field allows for a close to real time query obfuscation since orders of magnitude fewer candidate queries are explored. At the same time, our approach is on a par with respect to retrieval effectiveness with the previous state-of-the-art. While still being kind of "slow search" in the sense that the obfuscation needs a couple of seconds or with user feedback even some minutes, one can argue that in case of a really sensitive information need, users are willing to invest that time—and minutes are really affordable compared to an hour or more the previous state-of-the-art did require.

As interesting directions for future work, one could try to further speed up the process by just taking snippets of the local search into account instead of full documents for the query vocabulary generation. We also plan to implement the approach in an app that can directly be used when wanting to obfuscate queries against any of the big commercial search engines.

References

1. Ahmad, W.U., Chang, K., Wang, H.: Intent-aware query obfuscation for privacy protection in personalized web search. In: Collins-Thompson, K., Mei, Q., Davison, B.D., Liu, Y., Yilmaz, E. (eds.) The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018. pp. 285–294. ACM (2018). https://doi.org/10.1145/3209978.3209983, https://doi.org/10.1145/3209978.3209983
2. Ahmad, W.U., Rahman, M.M., Wang, H.: Topic model based privacy protection in personalized web search. In: Perego, R., Sebastiani, F., Aslam, J.A., Ruthven, I., Zobel, J. (eds.) Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016. pp. 1025–1028. ACM (2016). https://doi.org/10.1145/2911451.2914753, https://doi.org/10.1145/2911451.2914753
3. Arampatzis, A., Drosatos, G., Efraimidis, P.S.: A versatile tool for privacy-enhanced web search. In: Serdyukov, P., Braslavski, P., Kuznetsov, S.O., Kamps, J., Rüger, S.M., Agichtein, E., Segalovich, I., Yilmaz, E. (eds.) Advances in Information Retrieval - 35th European Conference on IR Research, ECIR 2013, Moscow, Russia, March 24-27, 2013. Proceedings. Lecture Notes in Computer Science, vol. 7814, pp. 368–379. Springer (2013). https://doi.org/10.1007/978-3-642-36973-5_31, https://doi.org/10.1007/978-3-642-36973-5_31
4. Arampatzis, A., Drosatos, G., Efraimidis, P.S.: Versatile query scrambling for private web search. Inf. Retr. J. **18**(4), 331–358 (2015). https://doi.org/10.1007/s10791-015-9256-0, https://doi.org/10.1007/s10791-015-9256-0
5. Arampatzis, A., Efraimidis, P.S., Drosatos, G.: Enhancing deniability against query-logs. In: Advances in Information Retrieval - 33rd European Conference on IR Research, ECIR 2011, Dublin, Ireland, April 18-21, 2011. Proceedings. pp. 117–128 (2011). https://doi.org/10.1007/978-3-642-20161-5_13, https://doi.org/10.1007/978-3-642-20161-5_13
6. Arampatzis, A., Efraimidis, P.S., Drosatos, G.: A query scrambler for search privacy on the internet. Inf. Retr. **16**(6), 657–679 (2013). https://doi.org/10.1007/s10791-012-9212-1, https://doi.org/10.1007/s10791-012-9212-1
7. Belkin, N.J.: Anomalous states of knowledge as a basis for information retrieval. Canadian Journal of Information Science **5**(1), 133–143 (1980)
8. Bevendorff, J., Stein, B., Hagen, M., Potthast, M.: Elastic ChatNoir: Search Engine for the ClueWeb and the Common Crawl. In: Azzopardi, L., Hanbury, A., Pasi, G., Piwowarski, B. (eds.) Advances in Information Retrieval. 40th European Conference on IR Research (ECIR 2018). Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York (Mar 2018)
9. Biega, J.A., Roy, R.S., Weikum, G.: Privacy through solidarity: A user-utility-preserving framework to counter profiling. In: Kando, N., Sakai, T., Joho, H., Li, H., de Vries, A.P., White, R.W. (eds.) Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017. pp. 675–684. ACM (2017). https://doi.org/10.1145/3077136.3080830, https://doi.org/10.1145/3077136.3080830

10. Cormack, G.V., Smucker, M.D., Clarke, C.L.A.: Efficient and effective spam filtering and re-ranking for large web datasets. Inf. Retr. **14**(5), 441–465 (2011). https://doi.org/10.1007/s10791-011-9162-z, https://doi.org/10.1007/s10791-011-9162-z
11. Culpepper, J.S., Diaz, F., Smucker, M.D.: Research frontiers in information retrieval: Report from the third strategic workshop on information retrieval in lorne (SWIRL 2018). SIGIR Forum **52**(1), 34–90 (2018). https://doi.org/10.1145/3274784.3274788, https://doi.org/10.1145/3274784.3274788
12. Dingledine, R., Mathewson, N., Syverson, P.F.: Tor: The second-generation onion router. In: Blaze, M. (ed.) Proceedings of the 13th USENIX Security Symposium, August 9-13, 2004, San Diego, CA, USA. pp. 303–320. USENIX (2004), http://www.usenix.org/publications/library/proceedings/sec04/tech/dingledine.html
13. Domingo-Ferrer, J., Solanas, A., Castellà-Roca, J.: h(k)-private information retrieval from privacy-uncooperative queryable databases.">h(k)-private information retrieval from privacy-uncooperative queryable databases. Online Inf. Rev. **33**(4), 720–744 (2009). https://doi.org/10.1108/14684520910985693, https://doi.org/10.1108/14684520910985693
14. Elbassioni, K.M., Hagen, M., Rauf, I.: A lower bound for the HBC transversal hypergraph generation. Fundam. Informaticae **130**(4), 409–414 (2014). https://doi.org/10.3233/FI-2014-997, https://doi.org/10.3233/FI-2014-997
15. Fuhr, N.: Some common mistakes in IR evaluation, and how they can be avoided. SIGIR Forum **51**(3), 32–41 (2017). https://doi.org/10.1145/3190580.3190586, https://doi.org/10.1145/3190580.3190586
16. Goldschlag, D.M., Reed, M.G., Syverson, P.F.: Onion routing. Commun. ACM **42**(2), 39–41 (1999). https://doi.org/10.1145/293411.293443, https://doi.org/10.1145/293411.293443
17. Gollub, T., Hagen, M., Michel, M., Stein, B.: From keywords to keyqueries: content descriptors for the web. In: Jones, G.J.F., Sheridan, P., Kelly, D., de Rijke, M., Sakai, T. (eds.) The 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR '13, Dublin, Ireland - July 28 - August 01, 2013. pp. 981–984. ACM (2013). https://doi.org/10.1145/2484028.2484181, https://doi.org/10.1145/2484028.2484181
18. Hagen, M., Beyer, A., Gollub, T., Komlossy, K., Stein, B.: Supporting Scholarly Search with Keyqueries. In: Ferro, N., Crestani, F., Moens, M.F., Mothe, J., Silvestri, F., Di Nunzio, G., Hauff, C., Silvello, G. (eds.) Advances in Information Retrieval. 38th European Conference on IR Research (ECIR 2016). Lecture Notes in Computer Science, vol. 9626, pp. 507–520. Springer, Berlin Heidelberg New York (Mar 2016). https://doi.org/10.1007/978-3-319-30671-1_37
19. Hébert, C., Bretto, A., Crémilleux, B.: A data mining formalization to improve hypergraph minimal transversal computation. Fundam. Informaticae **80**(4), 415–433 (2007), http://content.iospress.com/articles/fundamenta-informaticae/fi80-4-04
20. Hong, Y., He, X., Vaidya, J., Adam, N.R., Atluri, V.: Effective anonymization of query logs. In: Cheung, D.W., Song, I., Chu, W.W., Hu, X., Lin, J.J. (eds.) Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009. pp. 1465–1468.

ACM (2009). https://doi.org/10.1145/1645953.1646146, https://doi.org/10.1145/1645953.1646146
21. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of IR techniques. ACM Trans. Inf. Syst. **20**(4), 422–446 (2002). https://doi.org/10.1145/582415.582418, http://doi.acm.org/10.1145/582415.582418
22. Joachims, T., Radlinski, F.: Search engines that learn from implicit feedback. Computer **40**(8), 34–40 (2007). https://doi.org/10.1109/MC.2007.289, https://doi.org/10.1109/MC.2007.289
23. Jones, R., Kumar, R., Pang, B., Tomkins, A.: Vanity fair: privacy in querylog bundles. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM 2008, Napa Valley, California, USA, October 26-30, 2008. pp. 853–862 (2008). https://doi.org/10.1145/1458082.1458195, https://doi.org/10.1145/1458082.1458195
24. Kohlschütter, C., Fankhauser, P., Nejdl, W.: Boilerplate detection using shallow text features. In: Davison, B.D., Suel, T., Craswell, N., Liu, B. (eds.) Proceedings of the Third International Conference on Web Search and Web Data Mining, WSDM 2010, New York, NY, USA, February 4-6, 2010. pp. 441–450. ACM (2010). https://doi.org/10.1145/1718487.1718542, https://doi.org/10.1145/1718487.1718542
25. Kumar, R., Novak, J., Pang, B., Tomkins, A.: On anonymizing query logs via token-based hashing. In: Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007. pp. 629–638 (2007). https://doi.org/10.1145/1242572.1242657, https://doi.org/10.1145/1242572.1242657
26. Lin, J.: The neural hype and comparisons against weak baselines. SIGIR Forum **52**(2), 40–51 (2018). https://doi.org/10.1145/3308774.3308781, https://doi.org/10.1145/3308774.3308781
27. Liu, T.: Learning to Rank for Information Retrieval. Springer (2011). https://doi.org/10.1007/978-3-642-14267-3, https://doi.org/10.1007/978-3-642-14267-3
28. Miller, G.A.: Wordnet: A lexical database for english. Commun. ACM **38**(11), 39–41 (1995). https://doi.org/10.1145/219717.219748, http://doi.acm.org/10.1145/219717.219748
29. Murugesan, M., Clifton, C.: Providing privacy through plausibly deniable search. In: Proceedings of the SIAM International Conference on Data Mining, SDM 2009, April 30 - May 2, 2009, Sparks, Nevada, USA. pp. 768–779 (2009). https://doi.org/10.1137/1.9781611972795.66, https://doi.org/10.1137/1.9781611972795.66
30. Petit, A., Cerqueus, T., Mokhtar, S.B., Brunie, L., Kosch, H.: PEAS: private, efficient and accurate web search. In: 2015 IEEE TrustCom/BigDataSE/ISPA, Helsinki, Finland, August 20-22, 2015, Volume 1. pp. 571–580. IEEE (2015). https://doi.org/10.1109/Trustcom.2015.421, https://doi.org/10.1109/Trustcom.2015.421
31. Saint-Jean, F., Johnson, A., Boneh, D., Feigenbaum, J.: Private web search. In: Proceedings of the 2007 ACM Workshop on Privacy in the Electronic Society, WPES 2007, Alexandria, VA, USA, October 29, 2007. pp. 84–90 (2007). https://doi.org/10.1145/1314333.1314351, https://doi.org/10.1145/1314333.1314351

32. Strube, M., Ponzetto, S.P.: Wikirelate! computing semantic relatedness using wikipedia. In: Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA. pp. 1419–1424. AAAI Press (2006), http://www.aaai.org/Library/AAAI/2006/aaai06-223.php
33. Sugiyama, K., Hatano, K., Yoshikawa, M.: Adaptive web search based on user profile constructed without any effort from users. In: Feldman, S.I., Uretsky, M., Najork, M., Wills, C.E. (eds.) Proceedings of the 13th international conference on World Wide Web, WWW 2004, New York, NY, USA, May 17-20, 2004. pp. 675–684. ACM (2004). https://doi.org/10.1145/988672.988764, https://doi.org/10.1145/988672.988764
34. Terra, E.L., Clarke, C.L.A.: Frequency estimates for statistical word similarity measures. In: Hearst, M.A., Ostendorf, M. (eds.) Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003. The Association for Computational Linguistics (2003), https://www.aclweb.org/anthology/N03-1032/
35. Toubiana, V., Subramanian, L., Nissenbaum, H.: Trackmenot: Enhancing the privacy of web search. CoRR **abs/1109.4677** (2011), http://arxiv.org/abs/1109.4677
36. Willett, P.: The porter stemming algorithm: then and now. Program **40**(3), 219–223 (2006). https://doi.org/10.1108/00330330610681295, https://doi.org/10.1108/00330330610681295
37. Yang, P., Fang, H., Lin, J.: Anserini: Enabling the use of lucene for information retrieval research. In: Kando, N., Sakai, T., Joho, H., Li, H., de Vries, A.P., White, R.W. (eds.) Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017. pp. 1253–1256. ACM (2017). https://doi.org/10.1145/3077136.3080721, https://doi.org/10.1145/3077136.3080721
38. Yu, P., Ahmad, W.U., Wang, H.: Hide-n-seek: An intent-aware privacy protection plugin for personalized web search. In: Collins-Thompson, K., Mei, Q., Davison, B.D., Liu, Y., Yilmaz, E. (eds.) The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018. pp. 1333–1336. ACM (2018). https://doi.org/10.1145/3209978.3210180, https://doi.org/10.1145/3209978.3210180
39. Zhang, S., Yang, G.H., Singh, L.: Anonymizing query logs by differential privacy. In: Perego, R., Sebastiani, F., Aslam, J.A., Ruthven, I., Zobel, J. (eds.) Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016. pp. 753–756. ACM (2016). https://doi.org/10.1145/2911451.2914732, https://doi.org/10.1145/2911451.2914732