Anonymizing Query Logs by Differential Privacy

Sicong Zhang, Hui Yang, Lisa Singh
Department of Computer Science
Georgetown University
sz303@georgetown.edu, huiyang@cs.georgetown.edu, Lisa.Singh@georgetown.edu

ABSTRACT

Query logs are valuable resources for Information Retrieval (IR) research. However, because they are also rich in private and personal information, the huge concern of leaking user privacy prevents query logs from being shared from the search companies to the broad research community. Bothered by the lack of good research data for years, the authors of this paper are motivated to explore ways to generate anonymized query logs that can still be effectively used to support the search task. We introduce a framework to anonymize query logs by differential privacy, the latest development in privacy research. The framework is empirically evaluated against multiple search algorithms on their retrieval utility, measured in standard IR evaluation metrics, using the anonymized logs. The experiments show that our framework is able to achieve a good balance between retrieval utility and privacy.

Keywords

Privacy-Preserving Information Retrieval, Query Log Anonymization, Differential Privacy

1. INTRODUCTION

Query logs are essential research resources for Information Retrieval (IR), especially for the field of Web search. However, releasing query logs without proper anonymization may lead to serious violations of users' privacy. This was the case in 2006 when American Online (AOL) released an insufficiently anonymized version of their query log [1]. Table 1 shows a sample of this AOL query log.

Existing work on query log anonymization has attempted to protect the privacy of search logs in many ways. For instance, [1, 3] used clustering techniques and k-anonymity, which assumes each query to be issued by at least k different users, to anonymize query logs. The limitation of a k-anonymity approach is that its privacy guarantee can be easily broken when an adversary knows information about the users from an unexpected source. When an adversary knows about the user more than what the k-anonymity algorithm assumes, the adversary could join the unexpected source with existing ones and break the privacy guarantee. A stronger privacy notation is thus needed in query log anonymization.

Table 1: A Sample of the AOL Query Log during March 2006, from User 2178.

Query	Rank	Clicked URL
weatherchannel	1	http://www.weatherchannel.com.au
weatherchannel	5	http://www.weatherchannel.com.ru
honda accord check engine light...	1	http://www.automedia.com
fuel additives check engine light...	1	http://www.smogtips.com
...		

In this paper, we propose to use differential privacy [6, 8] to anonymize a query log. Differential privacy is the state-of-the-art approach which provides a strong privacy notion. It has been widely used in the database and data mining communities. Differential privacy provides guarantees which can be theoretically proved that every individual user in the datasets would not be identified. Unlike k-anonymity, differential privacy does not make assumptions about the amount and scope of an adversary's background knowledge.

A query log anonymization mechanism $A(Q)$ satisfies (ϵ, δ)-*differential privacy* if for all neighboring query logs Q_1 and Q_2, and for all possible outputs Q^* the following inequality holds: $Pr[A(Q_1) = Q^*] \leq e^{\epsilon} \times Pr[A(Q_2) = Q^*] + \delta$, where ϵ and δ are two model parameters related to the level of privacy guarantees. The smaller their values, the better the privacy guarantee. Specifically, a differentially private algorithm achieves ϵ-differential privacy if $\delta = 0$, which is even stronger than (ϵ, δ)-differential privacy.

In addition, most existing work in query log anonymization [8] measured the utility of the anonymization output in terms of the size of the remaining logs, without measuring a utility that is directly related to retrieval performance. It is thus difficult to tell how much utility is left in the query logs after anonymization in terms of how useful the logs is when we use them to retrieve relevant documents in a Web search algorithm. In this paper, we propose the retrieval utility function from the viewpoint of a search engine to report the actual usefulness of query logs after anonymization by differential privacy. Our approach achieves ϵ-differential privacy with $\delta = 0$.

To evaluate our approach, we experiment on the task of document retrieval with the anonymized query log using two Web search algorithms, a query-click model [4] and an implicit feedback model [2]. We then calculate the utility of the anonymized query log using retrieval effectiveness measures such as nDCG [7]. The results show that our framework is able to generate anonymized query logs that maintain a good level of retrieval utility. In another experiment on the

SIGIR '16, July 17–21, 2016, Pisa, Italy.

DOI: http://dx.doi.org/10.1145/2911451.2914732

Table 2: Samples of the anonymized AOL query log.

Part 1: Query Counts		Part 2: Click-Through Counts			Part 3: Query Transitions Counts		
Query	*Counts*	*Query*	*Clicked-through URL*	*Counts*	*Query*	*Next Query*	*Counts*
weather	13826	weather	http://www.weather.com	4190	weather	aol weather	44
weather channel	1175	weather	http://weather.yahoo.com	1035	weather	weather channel	25
aol weather	284	aol weather	http://weather.aol.com	30	weather	las vegas weather	9
las vegas weather	126	aol weather	http://aolsvc.weather.aol.com	16	aol weather	aol yellow pages	5
...		...			...		

tradeoff between privacy and utility, the framework is shown to be able to achieve a good balance between the two.

2. PROBLEM FORMULATION

This section presents our formulation of the query log anonymization problem.

Query Log Q: Query log Q is a textual document that records query data between the search engine and its users. Usually, it contains a record for each user including the user's id, the query, a ranked list of URLs that the search engine returns to the user, click-through information, and timestamps for all user actions.

Web Search Using Query Logs: Given a query q and a query log Q, the task of Web Search is to provide a ranked list of documents or URLs D that is relevant to q, from a set of documents or URLs that are built into a pre-indexed corpus C. Most Web search algorithms fit into this setting. User clicks, query reformulations, time spent examining the returned documents, and clicked documents on similar queries shared by multiple users are often the key elements used in a modern Web search algorithm.

The Task of Query Log Anonymization: Given an input query log Q, the task is to produce a version of the log in which the identifiable data is removed and the remaining data is adequately anonymized so as to reduce the likelihood of re-identification of users. The output of this task is an anonymized query log Q', with a guaranteed degree of privacy.

Privacy Function A: An anonymized query log Q' is generated by applying a privacy function A on the original query log Q. That is, $Q' = A(Q)$. Usually, A is parameterized to indicate the level of privacy that Q' can achieve. For example, in differential privacy, ϵ is the parameter in A, i.e., $Q' = A(\epsilon, Q)$. Smaller ϵ values indicate higher levels of privacy protection.

Utility Function U: In privacy-related research, the remaining utility of the data after applying a privacy function on it is an indispensable part of the research. Usually, a utility function U needs to be domain specific to be able to evaluate the usefulness of the data in a domain. The utility function can be applied on both the original data $U(Q)$ and the anonymized data $U(Q')$ to compare the utility deduction.

Utility Function for Web Search: In the context of information retrieval, a utility function U could be a two-step process – the first is to use the query log for document retrieval, i.e., to retrieve a set of ranked documents D for any $q \in Q$, where $D = R(q), q \in D$ and R is a retrieval algorithm. The second is to use IR evaluation metrics E to measure how good the retrieved document list D is with respect to each q being evaluated; that is $E : E(D)$. Therefore, the utility function of a query log Q can be represented as $U(Q) = E(R(Q))$, where E is a retrieval effectiveness measure for search results generated by R.

Goal: The goal of a successful query log anonymization algorithm is to have $|U(Q) - U(Q')| < \sigma$, where σ is kept small. At the same time, a successful query log anonymization algorithm should ensure that the privacy level $\epsilon : Q' = A(\epsilon, Q)$ is small enough to provide high privacy guarantee.

3. FORMAT OF ANONYMIZED LOG

In order to improve the privacy level of query logs to match the specifications of differential privacy, a search record might be removed or modified into a set of statistics. However, we need to be aware that such changes made on the original data only make sense if the remaining logs can provide enough information to be useful, in our case, to still be able to support Web Search. This section explains the output format of query log anonymization in our framework.

Firstly, we need to keep queries, which are central in query logs. They are kept in the textual format as they are. Low-frequency queries are removed since they are too unique and greatly increase the chance to break privacy guarantee if they stay. Next, the click-throughs are also key data in a query log. However, they can only be released in a statistical format in order to apply differential privacy on the log. Therefore, we aggregate all the click-throughs and show them as summary counts (see Table 1). Furthermore, highly identifiable features such as the user ids are removed during anonymization. Therefore, they are not shown up in the output log. Finally, we also maintain query transitions in a query log, allowing researchers to develop more advanced web search algorithms for multiple search queries in sessions.

The following format is proposed for an anonymized query log (shown in Table 2). Each anonymized query log Q' consists of three parts. The first part contains the released search queries and their corresponding frequencies in Q. Notice that all queries in Q' are in plain text, allowing researchers the opportunity to develop full-text retrieval algorithms. The second part contains click through data for each of the released query-URL pairs. Each line shows a query, a clicked URL for this query, and the number of clicks for the query-URL pair. The last part of Q' contains information about the query transitions in Q. Each line shows a pair of adjacent queries and the frequency of this query transition. To achieve differential privacy [6], all of the statistics in Q' could be modified with Laplacian noise [10].

4. ANONYMIZING QUERY LOG

We care about both the privacy levels and the utility it remains after being private. On the privacy side, our query log anonymization algorithm is a significant improvement upon [8] where their approach provides an (ϵ, δ)-differential pri-

vacy while we can achieve ϵ-differential privacy. We achieve this by making use of an external stochastic query pool to expand the query set. On the utility side, we are probably the first to use the IR evaluation metrics to measure the utility of an anonymized query log.

The main steps of anonymizing a query log Q are:
1) Remove Sensitive Data. We remove unique queries (frequency less than 5) or queries containing unique terms. This is to filter out sensitive data such as SSN or bank account numbers. **2) Limit User Activities.** We only keep the first q_f number of queries and the first c_f number of URL clicks for a user who has been logged in Q. The remaining set of search records forms Q_{clean}. **3) Expand the Query Set.** We define a query pool Q_p as the collection of potential queries to add into a query log. Query log owners, such as search engine companies could extend a to-be-released query log with more queries sampled from search records outside of this to-be-released log. Theoretically, every sufficiently frequent query gets a chance to be included in the expanded set. In this paper, we simulate this process by using high-frequency n-grams in general English [5]. We refer the combined set of queries as Q^+, where $Q^+ = Q_{clean} + Q_p$. **4) Select Final Set of Queries.** We use $Lap(b)$ as the Laplacian noise with parameter b [6]. We define the fuzzed query counts as the original query counts plus the corresponding Laplacian noise. We choose to release a query q when its fuzzed query counts $(M(q, Q^+) + Lap(b))$ is greater than a threshold K. Here $M(q, Q^+)$ is the frequency of the query q in Q^+. Note that the added queries can also be released with fuzzed query counts if the counts are greater than the threshold. The final query set generated after this step is referred to as $Q_{reduced}$. **5) Generate Log Statistics.** As presented in section 3, we release the query counts and click counts for each URL. All counts are fuzzed with Laplacian noise. **6) Generate Query Transitions.** We also release the query transition information to preserve sequential information of the query logs. We release adjacent query transitions from Q_{clean} with fuzzed counts, if both queries are included in $Q_{reduced}$.

5. UTILITY OF WEB SEARCH

As we defined in Section 2, the utility function of a query log Q in the context of IR is $U(Q) = E(R(Q))$, which is a nested function of two parts, retrieval and IR evaluation.

Part 1: Retrieval The first part is to use the query log Q for document retrieval, i.e., to retrieve a set of ranked documents D for any $q \in Q$. This step produces ranking lists which we denote as $R(Q)$. In this preliminary work, we test our approach of generating Q' using two click-based Web search algorithms. One is a random walk algorithm based on the query-click graph, and the other uses clicks as implicit feedback. Both of them do not require access to document content.

The first retrieval algorithm is based on a random walk model, a variation of a popular web search algorithm proposed by Craswell et al. [4]. In the graph, nodes are queries and documents (URLs), while the transitions include clicked documents from a query, adjacent queries in the original log, and self-loops to a node itself. The transition probability $P(k|j)$ from a query node j to a document node or another query node k is calculated by:

$$P(k|j) = \begin{cases} (1-s)C_{jk}/\Sigma_i C_{ji} & , \forall k \neq j \\ s & , k = j \end{cases} \quad (1)$$

where, C_{jk} is the weight between node j and k given by Q', and s is the self-transition probability. If both nodes j and k are query nodes, weight C_{jk} is defined as the query transition counts from j to k in Q'; otherwise, if j is a query node while k is a document node, weight C_{jk} is defined as the click-through counts for this query-document pair in Q'. In our approach, we set the self loop probability $s = 0.1$. Considering the cost of computation, each time step we start from a test query node, random walk three steps before stopping. After that, we can rank documents in descending order by the probability of being the stopping node.

The second retrieval algorithm we implemented is based on implicit feedback from user clicks, which is a variant of [2]. Given a query q, the relevance score $S(d)$ for each document d is calculated as:

$$S(d) = \begin{cases} \lambda \frac{1}{I_d+1} + (1-\lambda)\frac{1}{O_d+1}, & \text{if implicit feedback exists for d} \\ \frac{1}{O_d+1}, & \text{otherwise} \end{cases} \quad (2)$$

where λ is a parameter to weight the importance of user click. O_d is the original rank which is ranked using the order of click-through counts of document d with the query q, according to Q'. I_d is the rank of d from Q_{Test} when the user made the click. In our approach, we empirically set $\lambda = 0.6$. Finally, the documents are ranked in the descending order of $S(d)$ scores for each individual query q.

Part 2: IR Evaluation The second part is to apply IR evaluation metrics on those retrieved documents and to generate a final set of numerical scores to indicate the level of retrieval utility. We apply the evaluation metrics E to the ranked list R. We evaluate the search results using standard IR evaluation metrics such as nDCG [7] and MAP (Mean Average Precision) [9].

Table 3: Statistics of the AOL query log.

Statistics	Counts
Total number of records	36,389,567
Log size (GB)	2.2
# of unique user IDs	657,426
# of unique queries	10,154,742
# of clicks	19,442,629
Avg. clicks per user	29.57

Table 4: Utilities with Random Walk. Two-tailed t-tests ($p < 0.01$) show that no significant difference of utility scores before and after anonymization.

Query Log	nDCG@10	P@5	P@10	MAP
Original	0.6658	0.1484	0.0779	0.6395
Anonymized	0.6675	0.1486	0.0777	0.6424

6. EXPERIMENTS

We use the AOL query log [1] for our experiments. Table 3 presents the major statistics of AOL query log. For each parameter setting, the experiments are conducted in the following order: (1) A 5-fold cross validation is used to partition the data. In each run, we use 80% of the data as the training set Q and the remaining as the test set Q_{Test}; (2)

Table 5: Utilities with Implicit Feedback. Two-tailed t-tests ($p < 0.01$) show that no significant difference of utility scores before and after anonymization.

Query Log	nDCG@10	P@5	P@10	MAP
Original	0.6919	0.1535	0.0796	0.6725
Anonymized	0.6897	0.1527	0.0790	0.6711

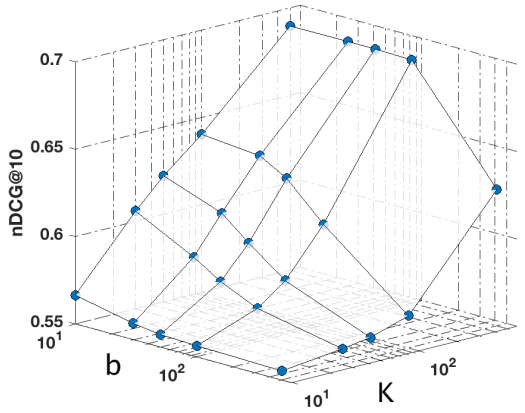

Figure 1: Relationships between noise b, query cutoff K and utility score $nDCG$@10.

The proposed ϵ-differentially private query log anonymization framework is applied to anonymize the query log. (3) Documents are retrieved using the algorithms as described in Section 5 for queries in Q_{Test}; (4) These retrieval algorithms are also run on the original log Q to compare their performance against that when the anonymized log Q' is used. We report the utility scores in nDCG@10 [7], Precision@5,@10, and MAP.

6.1 Utility by Retrieval Effectiveness

We first compare retrieval results using the original query log (Q) and the anonymized query log (Q'). Table 4 compares the performances of the Random Walk algorithm on query logs before and after anonymization while Table 5 compares the performances of the Implicit Feedback algorithm on query logs before and after anonymization. The anonymized query log used in both tables is the same and was generated with the settings $\epsilon = 29.99$, query counts threshold $K = 500$, and noise scale $b = 10$. Within each of the two tables, statistical significance tests (two-tailed t-tests, $p < 0.01$) show that the performances on query logs before and after anonymization are not significantly different from each other. This confirms that the retrieval effectiveness of our anonymized query log is comparable to the retrieval effectiveness of the un-anonymized version.

6.2 Privacy-Utility Tradeoff

More privacy guarantee would consume more utility of an anonymized query log. Here we also study the privacy-utility tradeoff with different parameter settings. Figure 1 shows the utility score (measured in nDCG@10) for the Implicit Feedback algorithms using Q' with different values for the query cutoff threshold K and the noise level b. Fixing all the other parameters including the log size, we range both K and b from 10 to 500. Figure 1 plots the trends between the utility scores and the parameter values. Each data point represents the average from a 5-fold cross-validation experiments. We observe that as the noise level b increases, the utility scores $nDCG$@10 decreases. We also observe that the utility score is less sensitive to b when b is much smaller than K. This matches our intuition that larger noise (comparing with K) will reduce retrieval performance and cause decreased utility.

7. CONCLUSIONS

In this paper, we introduce a framework for anonymizing search query logs and evaluating their Web search utility. We apply differential privacy, which is a strong privacy notation, to anonymize the logs. The experiments show that the Web search algorithms using the anonymized logs do not perform significantly different from those using the original logs. Since the high-level privacy has been guaranteed by our ϵ-differentially private anonymization algorithm, we suggest that search engine companies might be able to use less strict parameter settings and still maintain high utility. By proposing a new query log anonymization algorithm and a novel utility evaluation framework, our work makes an important step towards releasing Web query logs.

8. ACKNOWLEDGMENTS

This research was supported by NSF grant CNS-1223825, NSF grant IIS-145374, and DARPA grant FA8750-14-2-0226. Any opinions, findings, conclusions, or recommendations expressed in this paper are of the authors, and do not necessarily reflect those of the sponsor.

References

[1] E. Adar. User 4xxxxx9: Anonymizing query logs. In *Query Logs Workshop at the WWW'07*.
[2] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *SIGIR '06*.
[3] C. Carpineto and G. Romano. Semantic search log k-anonymization with generalized k-cores of query concept graph. In *ECIR'13*.
[4] N. Craswell and M. Szummer. Random walks on the click graph. In *SIGIR '07*.
[5] M. Davies. N-grams data from the corpus of contemporary american english (coca). *Downloaded from http://www.ngrams.info*, 23:2012, 2011.
[6] C. Dwork. Differential privacy: A survey of results. In *Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008.
[7] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, Oct. 2002.
[8] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In *WWW '09*.
[9] S. Robertson. A new interpretation of average precision. In *SIGIR '08*.
[10] R. Sarathy and K. Muralidhar. Evaluating laplace noise addition to satisfy differential privacy for numeric data. *Trans. Data Privacy*, 4(1):1–17, Apr. 2011.