“I Know What You Did Last Summer” — Query Logs and User Privacy

Rosie Jones Ravi Kumar Bo Pang Andrew Tomkins
Yahoo! Research, 701 First Ave, Sunnyvale, CA 94089.
{jonesr,ravikumar,bopang,atomkins}@yahoo-inc.com

ABSTRACT

We investigate the subtle cues to user identity that may be exploited in attacks on the privacy of users in web search query logs. We study the application of simple classifiers to map a sequence of queries into the gender, age, and location of the user issuing the queries. We then show how these classifiers may be carefully combined at multiple granularities to map a sequence of queries into a set of candidate users that is 300-600 times smaller than random chance would allow. We show that this approach remains accurate even after removing personally identifiable information such as names/numbers or limiting the size of the query log.

We also present a new attack in which a real-world acquaintance of a user attempts to identify that user in a large query log, using personal information. We show that combinations of small pieces of information about terms a user would probably search for can be highly effective in identifying the sessions of that user.

We conclude that known schemes to release even heavily scrubbed query logs that contain session information have significant privacy risks.

Categories and Subject Descriptors: H.3.3 **[Information Storage and Retrieval]**: Information Search and Retrieval

Generall Terms: Algorithms, Experimentation, Measurements

Keywords: k-anonymity, query log analysis, privacy

1. INTRODUCTION

Privacy research is a young research field. We have a number of interesting approaches to guaranteeing privacy in certain limited domains, but at the same time we have practical problems that are beyond the scope of current models. One such important practical problem is the following: How can search engine query log information be released to the research community in such a way that meaningful research can be performed, but smart, dedicated, and unscrupulous individuals willing to expend significant effort cannot compromise the privacy of the logged users?

The AOL incident in June 2006 has taught us that even a seemingly innocuous release of query logs can lead to undesirable consequences. An understanding of the potential vulnerabilities of search engine query logs is an imperative first step before it becomes possible to design privacy schemes to address the vulnerabilities; this is the goal of this paper. We initiate the study of subtle cues to user identity that exist as vulnerabilities in web search query logs, which may be exploited in attacks on the privacy of users.

Privacy attack models. We begin with a characterization of two key forms of attack against which a query log privacy scheme must be resilient. The first is a *trace attack*, in which an attacker studies a privacy-enhanced version of a sequence of searches (*trace*) made by a particular user, and attempts to discover information about that user. In our study of this type of attack, we draw upon the framework of k-anonymity [13], and study the extent to which information about gender, age, and location may be guessed from queries in a web search search log, and how effectively uncertain information along these dimensions will allow us to identify a small number of users containing the true user who generated the trace.

The second is a *person attack*, in which an unscrupulous agent attempts to discover that traces in a search engine log correspond to a particular known user. This is possible if some personal/background information of the user is accessible by the agent. We show that the risks of this form of attack are significant: even if the logs have been scrubbed by removing information about names and places, a few pieces of independent information may quickly shatter the set of logs into a small set of candidates containing the user.

Query log vulnerabilities. We show (Section 4) that query logs can be used to identify a user’s gender, age, and zip code with reasonable accuracy, which may be ingredients for a privacy attack. We build essentially off-the-shelf classifiers to accomplish these. Our goal is *not* to construct the best possible classifiers for these problems, but to merely illustrate the existence and easy identification of privacy-revealing vulnerabilities in highly-noisy log.

Next, we show (Section 5) that we can use combinations of these classifiers to shatter a large population of users into many smaller bins, which could then be exhaustively checked by a diligent attacker. Specifically, we show that combined classifier output, with no regard to other information, maps a small number of users (61) into a bin containing between 1 and 10 candidates including the correct one, and maps a somewhat larger set of users (1428) into a bin containing between 10 and 100 users including the correct one. As we show later, this is several orders of magnitude more revealing than random chance would allow.

We then turn our attention to person attacks. We show (Section 6) that it is possible for an attacker who can make reasonable guesses about likely queries a user might make, to identify a particular user known to them, even if they do not guess unique queries. With unique queries, it is almost certain they can identify the user.

Scrubbing personally identifiable information. An approach that has received some public interest is to “scrub” logs by removing certain forms of personally identifiable information (PII), in order to reduce the chance of linking a session to a user [10]. However,

CIKM’07, November 6–8, 2007, Lisboa, Portugal.

all earlier results of which we are aware are one-sided in the sense that they show PII as a vulnerability, but do not study if removing PII makes logs safe. We address this question and show that a wide range of information beyond the usual culprits of proper names, place names, social security numbers, and the like, is amenable to longitudinal analysis that can still compromise user privacy (Section 7). We conclude that there are real concerns in any scheme based on the approach of scrubbing a broad set of potentially identifying queries from the log, and releasing the remainder. We also argue that limiting the size of query logs is not a viable solution either; even a small amount of query log leakage is a grave matter.

2. RELATED WORK

Novak et al [12] use content similarity to disambiguate and anti-alias users who are using multiple pseudonyms. Frankowski et al [5] show that even when a user's data is anonymized, their public statements about rare interests can be joined with anonymized data to reveal their identity. Recently, Kumar et al [10] show that anonymizing query logs by hashing individual tokens does not prevent the decryption of that hash by an attacker with access to another external source of web logs. Even more recently, Adar [1] discusses specific schemes for anonymizing query logs sessions, by removing unique queries, hashing rare queries, and fragmenting into shorter sessions, as well as fragmenting users into topic profiles. Backstrom et al [4] look at the problem of identifying users in an anonymized social network.

The privacy notion behind k-anonymity [13] is that the user can be distinguished from no more than $k-1$ other users. Inspired by the medical record analysis done in the k-anonymity literature, we will look at the ability to automatically classify user session query logs into the correct *gender*, *age*, and *location* bins, since conjunctions of these may compromise privacy.

Argamon et al [3] classify texts according to the gender of the author, achieving an accuracy of 80% using part-of-speech tags and distribution of prepositions. Similar techniques have been applied to author identification [11], genre identification [2], and native-language identification [15]. Hu et al [7] use click-through and browsing behavior to classify user logs into age and gender.

3. DATA SOURCES

We use two kinds of data for experiments. The first is a collection of anonymized subset of user profiles from Yahoo!. The second is a subset of anonymized query logs from the Yahoo! web search.

First we describe the profile data, which is a subset of all the registered Yahoo! users. Each user profile has an anonymized Yahoo! user id along with its demographic information including age, zip code, and gender. We first pre-filter the profiles to retain only those with non-empty age, gender, and US zip code. This resulted in a population of around 100M profiles.

Unlike gender, both age and and zip code can be studied at different granularities. We consider age at either the fine-grained level of YEAR or at the coarser level of one of ten pre-specified BUCKETS.[1] We study three increasingly coarse versions of the zip code digits: we use ZIP5 to denote the regular five-digit zip code and ZIP2 (resp. ZIP3, ZIP4) to denote the first two (resp. three, four) digits of ZIP5. Note that each fixed zip version along with the age bucket and gender defines a collection of cells in three dimensions.

From a collection of 79 days of Yahoo! web search query logs, we identify sessions issued by users whose ids are in the profile data, with all of age, gender and US zip code defined. We select a sample of the users who issued a non-trivial number of queries (more than one hundred) over this time period. Queries issued by these users form our query log data. For each of the 744K users in this dataset, all queries issued by that user during this period are extracted, uniqued by day, and lower-cased. We refer to this collection of queries as the *trace* for that user.

Analysis of profiles. We can obtain an overall picture of the vulnerability of our profile data in terms of user age bucket, gender, and zip code, by considering how many users fall into each group when we distinguish using these three attributes. Using the terminology of Samarati and Sweeney [13], we call a cell for a particular conjunction of age, gender, and zip code *k-vulnerable* if there are k users in that cell. If k is generally small, identifying the age bucket, gender, and zip code of a user from their query trace is likely to lead to identification of the user. After mapping a user trace to a cell of size k, the attacker could use other information such as hobbies and names from the queries to narrow down the individual.

Suppose our database of users (filtered further to remove ages below 13 and above 80) is broken into cells that share the same gender and age bucket, and that share either three, four or five digits of zipcode. We have significant number of users falling into cells with relatively small size. For instance, for $k = 100$, which considers only users whose gender, age bucket, and first three digits of zipcode are jointly shared by 99 or fewer other users, there are almost 1000 such combinations (one of which, for instance, might be males from 25–29 years old living in zipcode 950xx) and almost 100K people out of our users who inhabit a cell of size 100 users or fewer.

4. DEMOGRAPHIC CLASSIFICATION

We describe experiments to automatically classify query logs to identify a user's self-identified *age*, *gender*, and *zip code*. These attributes have been shown to facilitate identifying individuals [13].

For gender and age, we used bag-of-words classifiers, which have the property of using a large number of features (words) for classification. We set aside one tenth of the query log data with aligned age and gender labels as training data. In pilot studies with the training set, we compare classifiers by training on half of the training set and test on the other half. We use support vector machines (SVMs) for gender and age classification; they outperformed Naive Bayes in our pilot study. We used the svm_{light} package[2] with all parameters set to their default values, after first length-normalizing the bag-of-words vectors representing all terms in the trace with zero-one values indicating presence of features. As we will see, performances of these classifiers reflect the rich information encoded in simple bag-of-words representations of queries that can potentially be harvested by a resourceful attacker.

Gender and age. Our gender classifier achieved 83.8% accuracy on the test set, outperforming the naive baseline scheme of always predicting the majority class (57% males). Many of the top indicators in the model are in accordance with stereotypical images of male and female interests: *fanfiction*, *bridal*, *makeup*, *women's*, *knitting*, *hair*, *ecards*, *glitter*, *yoga*, and *diet* are good indicators for the female class, while *nfl*, *poker*, *espn*, *ufc*, *railroad*, *prostate*, *football*, *golf*, *male*, *wrestling*, *compusa*, as well as a variety of adult terms are good indicators for the male class.

A user's age may be detected from their interests. We could use bins of age ranges and perform multi-class classification. However, if there are shift of interests in queries as users age, we expect this shift to occur gradually rather than abruptly at the borders of predefined buckets, making it preferable to predict age in birth years

[1] The starting age for each of the ten buckets is given by this list: 13, 18, 21, 25, 30, 35, 45, 55, 65.

[2] `http://svmlight.joachims.org`

instead. Thus we use SVM regression (with a tube-width of 1). Average of absolute error ($\epsilon = |\mathrm{age}_{\mathrm{predicted}} - \mathrm{age}_{\mathrm{true}}|$) on the test set is 7.0, outperforming a baseline of always guessing the middle point.

δ	1	3	7	10
% users with $\epsilon < \delta$	14.7	33.4	63.9	79.0

Again, when we analyze the weights assigned to tokens by the model, what we find as the top indicative terms are not surprising. Among the indicative terms for relative youth, we have: *myspace*, *pregnancy*, *wikipedia*, *lyrics*, *quotes*, *apartments*, *torrent*, *baby*, *wedding*, *mall*, *soundtrack*; among the indicative terms for older age: *aarp*, *telephone*, *lottery*, *amazon.com*, *retirement*, *funeral*, *senior*, *mapquest*, *medicare*, *newspapers*, *repair*.

Location: Zip code. A user's zip code (US postal code) or other identifier of location may be detectable from place names used in their queries. We use a black-box classifier based on the internet locality product, Whereonearth (WOE), which is now a part of Yahoo! Given a query, WOE determines if this query has a location component and if so, outputs a list of locations at the best guessed granularity (i.e., city, county, state, country) along with the confidence. It also outputs an aggregated confidence that captures how location-specific is the query. For a given user, we consider all the queries in the trace. For each query, we run the WOE classifier. If the classifier returns an aggregated confidence of below 0.5, we ignore the query. Otherwise, we accumulate the zip code and labels for each US city. After processing all the queries for the user, we aggregate further and output the top three candidates corresponding to each of ZIP3, ZIP4, and ZIP5. The results are below.

Zip	ZIP5	ZIP4	ZIP3
Accuracy top guess (%)	6.27	13.7	34.9
Accuracy top-3 guesses (%)	13.1	25.1	54.1

Improved accuracy could be obtained by retrieving web pages for the queries and identifying locations from those, as well as looking at the query context for the placenames; see [6, 9].

5. TRACE ATTACKS

We begin with a high-level characterization of the vulnerability. We consider an attacker who wishes to map a particular search trace to an actual user profile. We assume the attacker attempts to classify the trace to produce a gender, age, and zip code; the resulting information may be generalized in many ways. After running the classifiers, the attacker must therefore state which profiles are candidates for the author of the trace. If the true author is one of the candidates, we evaluate the attack based on the total number of candidates, where a smaller number corresponds to a more successful attack. But if the true author is not one of the candidates, we say the attack failed.

The attacker must walk a fine line between including too many candidates, and hence leaking only a small amount of information about the author of the trace, to including too few candidates, and perhaps missing the true author entirely. Given the results of the classifiers, we consider 60 approaches to generating a set of candidate profiles, each of which corresponds to a different amount of generalization of the classifier output, as follows.

(i) Age. (5 cases) In the EXACT-YEAR matching scheme, we consider only candidate profiles with the birth year indicated by the classifier. In the LOOSE-YEAR scheme, we allow all profiles whose birth year is within three of that indicated by the classifier. In the EXACT-BUCKET, we identify ten buckets of age ranges, and ask that candidate profiles should exactly match the bucket output by the classifier. In LOOSE-BUCKET, we ask that the candidate profile's birth year be within an adjacent bucket to the birth year output by the classifier. Finally, in the NO-AGE condition, we do not remove any candidates based on age.

(ii) Zip digits. (4 cases) We consider ZIP5, ZIP4, ZIP3, and ZIP2.

(iii) Zip count. (3 cases) We consider three more conditions, based on the observation that the classifier outputs multiple candidate zip codes at each level (ZIP2-5). In COUNT1, we consider only the first such candidate, with whatever number of digits is given by the zip digits parameter. Likewise for COUNT2 and COUNT3.

These give us the sixty possible levels of granularity. As the gender classifier is quite accurate, we employ it in all cases.

Our basic experiment is: fix one of the 60 levels of granularity, classify each trace, and generate a set of candidate profiles based on that level. The true author of the trace is either present in the candidate set or not. If present, we say that the scheme produced a hit in a cell whose size k is the number of candidates. If not present, we say the attack failed at the given level of granularity. We will plot for each of the sixty levels of granularity the number of traces that are successfully attacked with a cell size of k or less, for varying values of k. Figure 1(a) shows the results for a specific range of k. Here, k is shown on the x axis, and the number of user traces is shown on the y axis.

The results should be read as follows. We extracted 750K traces, and around 100M user profiles. Thus, a naive attack might successfully attack all profiles, with a cell size of 100M. Another naive attack for a particular value of k might always generate the same set of k candidates; of those k candidates, we expect roughly $k \times 750K/100M = k/133$ to be part of the original set of 750K, and thus, the naive attack would generate the point $(k, k/133)$ on the graph, representing a line with slope 1/133. The graphs in the figure over a wide range of values of k typically show a slope of roughly 3-6, indicating that the attack is 300–600 times more likely than random chance to indicate the correct user.

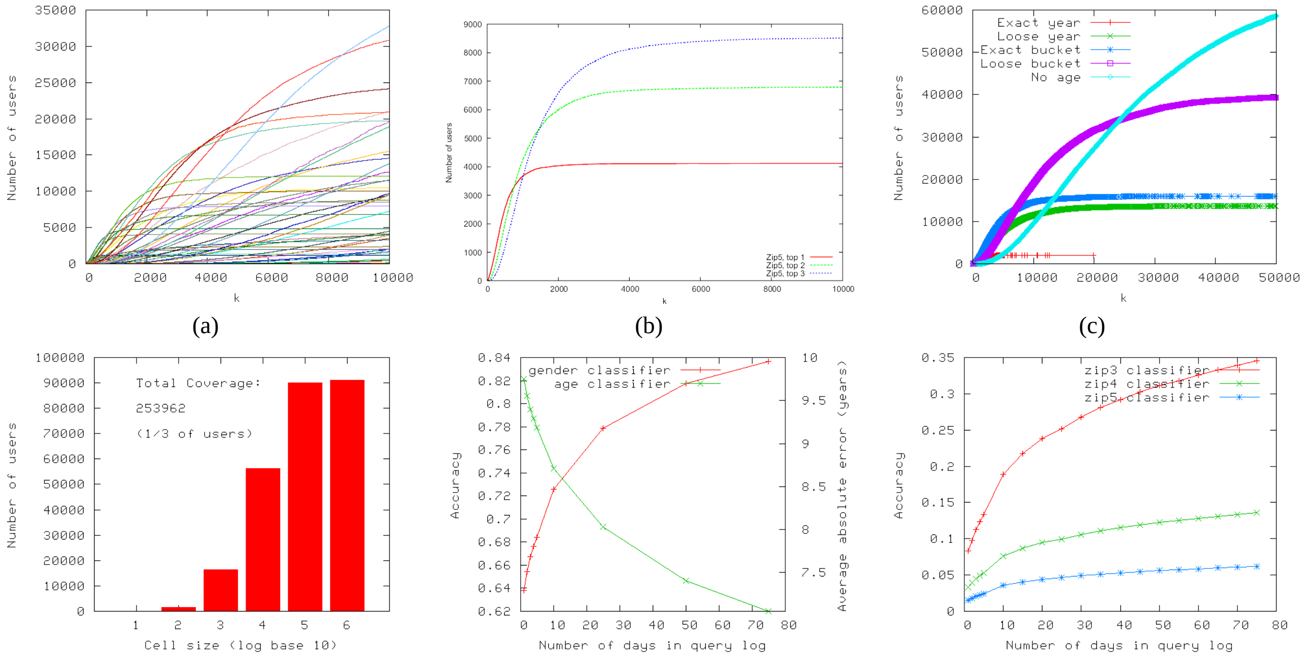
Notice that the charts show many different lines, one corresponding to each of the sixty levels of granularity. The reader should trace the upper curve at all points as the region of interest. The attacker may select a value of k of interest, and potentially choose the level of granularity most appropriate for that value of k.

Understanding classifier performance. We now provide a slightly more detailed view of performance for various different classifiers. Figure 1(b) shows the results for all three zip counts, using gender match, LOOSE-YEAR age match, and the ZIP5 zip digit conditions.

Notice that for small values of k, the COUNT1 condition dominates, as the first zip code output by the classifier is more likely to be correct. But as k grows, the additional zip candidates give significant improvement in coverage without any significant loss in accuracy. This pattern recurs in all other cases, suggesting that systems to scrub logs should be resilient to attackers that use any available cues to produce a large number of candidate zip codes.

Figure 1(c) compares the five different age conditions for the ZIP4 and COUNT2 geographic granularity, along with gender. LOOSE-YEAR and EXACT-BUCKET perform quite similarly, and the asymptotic trends are as expected. Overall coverage differs by a factor of 36 from NO-AGE to EXACT-YEAR, but the eleven users in cells of size 56 of less in the EXACT-YEAR condition are broadened to cells of over 1000 in some cases for the NO-AGE condition. Thus if we could prevent classification of user age from query traces, we could make attacks much more difficult.

Characterizing privacy vulnerabilities. We now consider an experiment to combine information from all levels of granularity into a single algorithm to match user traces to user profiles. The experiment captures the potential leakage inherent in the data using the classifiers we have on hand, but does not represent a potential attack. It proceeds as follows. For each user trace, a particular gran-



(a) (b) (c)

(d) (e) (f)

Figure 1: List of figures

ularity of attack (for instance, an attack that extracts the two most likely ZIP4s, and the most likely year of birth for the trace, and then extracts all matching profiles) will produce a set of candidate users of a certain size, which may or may not contain the actual user who produced the trace. The goal of an attack is therefore to find the smallest size of cell that contains the actual user. A natural upper bound is simply to consider all possible granularities of attack, and with omniscience, select the granularity that produces the smallest cell containing the actual user. We perform this experiment for all traces, with the results shown in Figure 1(d). We can narrow 1428 users down into bins of size 10-100, and 61 users into bins of size 1-10. A diligent attacker could exhaustively investigate these smaller groups of users, to identify the specific trace log.

The following table shows the amount of information leaked for various users, in the same experiment. To specify a user from a set of 100M users, 26.5 bits of information must be specified. The table shows how many users are vulnerable to leaks of certain amounts of identify information. For about 2/3 of the users, these techniques as developed do not leak any bits of information; simple modifications could allow small leaks for almost all of these users. From a privacy perspective, however, we must focus instead on the 33% of users for whom much more substantial leaks are possible.

# users	61	1428	16k	56k	90k	90k	496k
bits leaked	23-26	20-23	17-20	13-17	10-13	< 10	0

6. PERSON ATTACKS

In the person attack we assume that the adversary seeks to identify the query stream of a particular user known to the attacker. In this scenario the attacker can bring additional information to bear, such as queries that the user is likely to have issued. Our assumption is that there are multiple mechanisms by which the attacker might have insight into the queries performed by the given user.

The first mechanism involves making guesses about queries the user might have performed, based on the following types of information: the attacker might be able to (1) exploit obvious knowledge, such as the gender, zip code, and age of the given user; (2) exploit conversations with the given user, eg., knowing the user is planning a vacation to Tahiti; or (3) observe lifestyle changes in the the given user, such as the purchase of a new car.

The second mechanism involves gaining access to the user's browser. Specifically, the attacker may be able to check the browser history or request use of the given user's browser on some pretext, and might enter a unique query that can later be used as an anchor for a person attack. Notice that this mechanism is extremely difficult to guard against using approaches based on scrubbing of query logs.

We focus on the vulnerabilities of the first mechanism of attack. First, we have developed a set of characteristic searches that an attacker might be able to guess about a neighbor or friend. The table below shows these queries, along with the number of users from our sample of 744K users who issued a search containing the term. Note that we chose them without reference to a query log, and so did not select any terms unique to any individual. As we will discuss below, most queries are unique, so this represents a relatively weaker form of attack.

	Common	Rare
Cars	volkswagen beetle (478) honda odyssey (1504) toyota prius (1070)	triumph tr3 (23) e-type jaguar (5)
Sports	skiing (9618) football (123802)	bassmaster (388) skulling (17)
Food	pizza (104,888) italian restaurant (4998) brie (39,325)	assam (747)
Books	harry potter (27,838) danielle steele (238) freakonomics (574)	holly lisle (20) elizabeth moon (27)

Using these somewhat arbitrarily chosen terms, we looked for user logs containing combinations of these terms. Knowing that someone queried for "honda odyssey" places them in a bin of size just 1504. Knowing also gender, zip code, and approximate age, the attacker can narrow the search using the ideas from Section 5.

When we look for users searching for two or more of these terms, we found that 99 bins were of size 1: knowing two or three things likely to be sought by the user can lead to unique identification, despite the fact that none of the terms themselves is unique. The table below gives examples of query combinations that lead to both unique bins and bins containing more users.

Query set	Bin size
harry potter, pizza	4855
football, skiing	2430
italian restaurant, pizza	1441
harry potter, volkswagen beetle	27
honda odyssey, italian restaurant	20
football, skiing, toyota prius	9
football, triumph tr3	4
football, harry potter, volkswagen beetle	3
pizza, triumph tr3	2
danielle steele, volkswagen beetle	1
brie, holly lisle, pizza	1

The table below shows that there are 99 combinations of terms leading to unique bins, and 31 combinations of terms that lead to bins containing just two users. Combined there are 320 combinations of non-singleton query terms that lead to grouping users into bins of less than 100 users. Clearly knowledge of combinations of small numbers of query terms can compromise k-anonymity for small k.

# users in bin	100+	51-99	26-50	6-25	3-5	2	1	< 100
# bins	51	13	17	65	44	31	99	320

In general, many queries in a log are singletons [14, 8, 1] Thus, the risk of finding a uniquely-identifying query is very high, even without accounting for combinations of queries.

7. REDUCING INFORMATION

We study the impact of reducing the amount of information that is present in a query log. We consider two forms of reduction. The first is filtering specific personally identifying information from the query log. The second is to provide query logs with fewer days worth of queries. For these forms of reduction, we analyze the performance of our gender, age, and zip code classifiers.

Removing PII. We remove the following personally identifying information from the query log, on a term by term basis. We are deliberately aggressive in this stripping, preferring to remove too much information than too little. The following information was removed: numbers and numbers that are part of alphanumeric strings, names of all US cities, US states, and US state abbreviations, and first and last names, which was obtained from census records.

For gender and age, we retrain the classifiers on the training data after removing the personally identifying information according to the above protocol. We observe that the performances of both gender and age classification on the test data remain largely unchanged: accuracy for gender is 83.6% and average absolute error of age prediction is 7.1.

For zip code, the results are different. After removing personally identifying information (especially, the geographic component), the classifier accuracy reduced significantly. The table below shows the classification accuracy of the zip code classifier.

Zip	Zip5	Zip4	Zip3	Zip2
Accuracy (%)	0.59	0.99	2.81	6.14

However, since the gender and age classifiers were not adversely affected by this operation, it seems plausible that removing the above personally identifying information may not be sufficient to preserve anonymity.

Limiting query log history. Intuitively, we are more likely to correctly identify a given user if we have access to longer history of the user's query log. A query log containing 79 days' queries potentially reveals more information than a query log with one day's worth of queries. To verify this quantitatively, we keep the gender, age, and zip code classifiers unchanged, but restrict the aggregated queries for each user in the test set to the first n days in the query log. To measure accuracy in this case, we only count the users who have issued at least one query during the first n days.

For gender, age, and zip, Figure 1(e-f) shows that performance does go down as we include fewer days in the query log. But even with one day's worth of queries, there is enough information that enables the classifiers to outperform baselines (of both random guess and always predicting the majority class). Therefore, releasing query logs for a few number of days is not a bullet-proof solution to preserving anonymity.

8. CONCLUSIONS

We have shown that the release of user query log data with session information can lead to compromises of user privacy. Removing classes of identifying terms such as names, digits and places is not sufficient to prevent attacks which use a combination of techniques.

9. REFERENCES

[1] E. Adar. User 4XXXXX9: Anonymizing query logs. In *Query Logs Workshop at the 16th WWW*, 2007.

[2] S. Argamon, M. Koppel, and G. Avneri. Routing documents according to style. In *Proc. 1st Workshop on Innovative Information Systems*, 1998.

[3] S. Argamon, M. Koppel, J. Fine, and A. R. Shimoni. Gender, genre, and writing style in formal written texts. *Text*, 23(3):321–346, 2003.

[4] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou R3579X? Anonymized social networks, hidden patterns, and structural steganography. In *Proc. 16th WWW*, pages 181–190, 2007.

[5] D. Frankowski, D. Cosley, S. Sen, L. Terveen, and J. Riedl. You are what you say: Privacy risks of public mentions. In *Proc. 29th SIGIR*, pages 565–572, 2006.

[6] L. Gravano, V. Hatzivassiloglou, and R. Lichtenstein. Categorizing web queries according to geographical locality. In *Proc. 12th CIKM*, pages 325–333, 2003.

[7] J. Hu, H.-J. Zeng, H. Li, C. Niu, and Z. Chen. Demographic prediction based on user's browsing behavior. In *Proc. 16th WWW*, pages 151–160, 2007.

[8] B. J. Jansen, A. Spink, and T. Saracevic. Real life, real users, and real needs: A study and analysis of user queries on the web. *IPM*, 36(2):207–227, 2000.

[9] R. Jones, W. V. Zhang, P. Jhala, and B. Rey. Geographic intention and modification in web search. *International Journal of Geographical Information Science*, 2007.

[10] R. Kumar, J. Novak, B. Pang, and A. Tomkins. On anonymizing query logs via token-based hashing. In *Proc. 16th WWW*, pages 629–638, 2007.

[11] F. Mosteller and D. Wallace. *Inference and Disputed Authorship: The Federalist Papers*. Addison-Wesley, 1964.

[12] J. Novak, P. Raghavan, and A. Tomkins. Anti-aliasing on the web. In *Proc. 13th WWW*, pages 30–39, 2004.

[13] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *Proc. 17th PODS*, page 188, 1998.

[14] C. Silverstein, M. Henzinger, H. Marais, and M. Moricz. Analysis of a very large altavista query log. Technical Report 1998-014, Digital SRC, 1998.

[15] L. M. Tomokiyo and R. Jones. You're not from 'round here, are you? Naive Bayes detection of non-native utterance text. In *Proc. 2nd NAACL*, 2001.