# Utilizing Re-finding for Personalized Information Retrieval

Sarah K Tyler
University of California, Santa
Cruz
1156 High St.
Santa Cruz, CA 95064
skt@soe.ucsc.edu

Jian Wang
University of California, Santa
Cruz
1156 High St.
Santa Cruz, CA 95064
jwang30@soe.ucsc.edu

Yi Zhang
University of California, Santa
Cruz
1156 High St.
Santa Cruz, CA 95064
yiz@soe.ucsc.edu

## ABSTRACT

Individuals often use search engines to return to web pages they have previously visited. This behaviour, called re-finding, accounts for about 38% of all queries. While researchers have shown how re-finding differs from traditionally studied new-findings, research on how to predict and utilize re-finding is limited. In this paper we explore re-finding for personalized search. We compared three machine learning algorithms (decision trees, Bayesian multinomial regression and support vector machines) to identify re-findings. We then propose several re-ranking methods to utilize the prediction, including promoting predicted re-finding URLs and combining re-finding prediction with relevance estimation. The experimental results demonstrate that using re-finding predictions can improve retrieval performance for personalized search.

## Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]:

## General Terms

Algorithms, Design, Experimentation

## Keywords

Re-finding, Query Log Analysis, Personalized Search

## 1. INTRODUCTION

In addition to seeking out new content, web users also use search engines to re-find websites which they have previously visited. Past research has shown that half of all webpages a person visits are pages the person has seen before[8, 11, 2] and about 38% of all search engine queries are used to re-find[12, 14]. Tyler and Teevan[14] characterized re-finding using large-scale log analysis, observing that the query used to re-find typically ranks the re-found URL higher than the previous re-finding. Nevertheless, for one

search engine query log (discussed in detail in section 4), we find that the average rank of re-finding URLs is 5.3 and only 56.6% of re-finding URLs are at rank one. By predicting re-finding queries (those that lead to clicks on previously visited URLs), we might improve retrieval quality and the search engine clickthrough rates.

Previous works for aiding re-finding has taken a tool centric approach. Web tools, such as the Web browser back button [10], bookmarks [1], and browser histories [6] have been used to study re-visitation. These tools have been built into the browser and do not incorporate the re-visitation in the search algorithms. Other tools store complex queries for future use [9] as well as using the saved search results list to improve re-finding.[13] The Re:Search Engine [13] and the SearchBar [7], are both designed to primarily support re-visitation. Re:Search uses the past result lists for a given query and merges it with the new list. SearchBar maintains query histories and browsing histories of the user, as well as allowing the user to enter notes about the pages, to aid the user when she wishes to revisit past sites.

In this paper, we use regression models to predict re-finding based on a user's search history. We compare three approaches to modify search results based on predictions about re-finding. The first approach boosts the predicted target re-finding URLs to the top of the results list. The second approach filters the predicted re-finding URLs, keeping only those that are relevant to the search query. The third approach ranks URLs based on a combination of re-finding score and relevance score.

We find that users often behave consistently with themselves, however not with each other. Thus we explore re-finding from both a global (across all users) and an individual (for each given user) level. The decision to re-rank is always based on each user's own history, thus the final retrieval results are personalized.

## 2. NOTATION AND DEFINITIONS

Define $Q_i$ as the query instance with encompassing information, and $q_i$ as the query string. Let $U_i$ be the set of clicked URLs associated with $Q_i$. Two queries $Q_i$ and $Q_j$ issued by the same user are considered to be re-finding if they share a clicked URL, ie. $U_i \cap U_j \neq \emptyset$. Queries need not be consecutive in order to be considered re-finding. That is, a user may issue multiple queries over multiple sessions and multiple days between $Q_i$ and $Q_j$. The rank order of URLs for $Q_i$ is defined as $\pi_i$. Thus $\pi_i(u) = r$ indicates that URL $u$ is ranked at position $r$ for the $ith$ query.

# 3. MODELS AND METHODOLOGIES

We use three different regression models to predict re-finding: Decision Trees, Support Vector Machines (SVM) and Bayesian Multinomial Regression (BMR). Decision trees are easy to interoperate; by looking at the structure of the tree one can gain an understanding of what makes up re-finding. Each child branch can effectively be a different model, thus the tree can model intra vs inter session re-findings, or different user types. While decision trees illustrate the effectiveness of simple models, SVM and BMR give insight as to whether a more complex model can have a substantial gain. For SVM, SVMLight[1] is used since it has been highly optimized, and often used in the literature. We use the Rutger's implementation of BMR for the same reasons[2].

## 3.1 Model Feature

The models incorporate three types of information: *Query Change*, *Personalized* and *Shared* features.

### 3.1.1 Query Change Features

The *Query Change Features* capture the changes between two query instances, $Q_i$ and $Q_j$. The first three feature groups are designed to capture string similarities between $q_i$ and $q_j$. The last feature group captures the time between query instances.

- *MinimalChange* is a binary feature that indicates the difference between $q_i$ and $q_j$ does not alter query meaning. Subfeatures include *DomainVariant* (domains in the query strings are functionally identical), *WordOrder*, presence of *StopWords*, *Spelling* (two strings differ by an absolute edit distance less than 2 or a relative edit distance of 0.02) and the presence of *Alphanumerics*.

- *SubstantialChange* is a binary feature that indicates the query strings differ in a meaningful way. Subfeatures include *TermAdded*, *TermRemoved* and *Different*. Any two queries will either exhibit a minimal change or a substantial change between them.

- Query string overlap using *LongestCommonSubstring* and (since the first search terms are often the most important) *LongestCommonPrefix*.

- The time interval between the two queries which includes the real valued features *DeltaTime*, *DeltaSession* and *DeltaQueries* as well as the boolean feature *SameSession*. Additionally *SessionPosition* measures how far into the session each query occurs.

### 3.1.2 Personalized Features

The *Personalized Features* for a given user are designed to capture a user's propensity to re-find. A user may re-find a given webpage multiple times and may have developed a set of queries that she typically uses, thus "settling" on a set of queries. Additionally, if the user is struggling to re-find a given URL, she may click other familiar URLs, re-finding multiple times.

- The probability of re-finding for the query string $q_i$ (*PersonalizedProbQi*), query string $q_j$ (*PersonalizedProbQj*), URL $u_i$ (*PersonalizedProbUi*).

[1] http://svmlight.joachims.org/
[2] http://www.stat.rutgers.edu/ madigan/BMR/

- A boolean indication of the last query being re-finding (*LastRefinding*), at least one of the last three queries (*Last3Refinding*), or last five queries (*Last5Refinding*) or any query in the same session (*SessionRefinding*). These features are calculated for both $Q_i$ and $Q_j$

### 3.1.3 Shared Features

Popular websites, such as news sites and social networking sites, are likely to be popular re-findings across users. The *Shared Features* are features independent of the current user, calculated on a hold out set of approximately 6000 users.

- An integer feature rank (*URLRank*) for the URL associated with the first finding query instance $Q_1$.

- The probability of re-finding for the query string $q_i$ *SharedProbQi* and query string $q_j$ *SharedProbQj*. Probability of a URL being refound for $u_i$ (*SharedProbUi*) and the current URL in question (*SharedProbU*).

## 3.2 Training the Models

Approximately 3000 users are selected at random. The first forty query instances of each user are used for training and the next ten query instances for validation. These two sets correspond to an average of 44 days in the query logs. Remaining query instances by the user (an average of 66) are used for testing and evaluated against all previously seen URLs including those in the training and validation sets.

There may be multiple query instances in which the user clicked the same URL in the training set. Each of these instances may capture different characteristics of re-finding. All pairs of queries that have a result click in common are instances of re-finding. This biases the classifiers in favor of popular re-findings. If the same URL is predicted as a re-finding for a given test instance, the highest probability is kept. There may be multiple query instances in which the user clicked the same URL in the training set. Each of these instances may capture different characteristics of re-finding. All pairs of queries that have a result click in common are instances of re-finding. This biases the classifiers in favor of popular re-findings. If the same url is predicted as re-finding for a given test instance, the highest probability is kept.

The ability to re-find information may depend on how the user typically interacts with the search engine, including his or her skill level in formulating a query and whether he or she tends to reformulate, or keep searching through the result lists. A personalized model, not just a model using personalized features, may perform better than a single model. We considering two basic set-ups over all features: (1) personalized, designated $Model_P$, where one model per user is trained and (2) non-personalized, designated $Model_G$ for global, where a single model over all users is trained.

## 3.3 Re-Ranking Methods

There are several methods for re-ranking the search results to incorporate re-finding. Normalized Discounted Cumulative Gain (NDCG)[4] with the assumption only clicked URLs are relevant is used for evaluating the re-rankings. NDCG measures the relative improvement of one ranking to another.

### 3.3.1 Rank Re-finding URLs Higher

A straight forward approach to re-ranking is to move predicted re-found URLs to the top of the results page. Multiple

predicted re-findings are ranked according to the probability of re-finding. Since the average number of clicks during re-finding is 1.54, a variation of this approach where only the most likely re-finding URL is promoted is considered.

### 3.3.2 Rank Relevant Re-Finding URLs Higher

The above approach only considers the probability of re-finding for a URL and not the relevance of the URL to the query. In this re-ranking method, a predicted re-finding URL is re-ranked only if the URL is also returned in the top 100 search result URLs (thus relevant). Again, all relevant re-finds are ranked first.

### 3.3.3 Combining Re-Finding with Relevance

A URL may have a small probability of re-finding. In this case it is desirable to promote it, but not necessary to the top of the results list. Re-findings URLs are re-ranked based on a linear combination of the original relevance score and the re-finding prediction

$$rank = \alpha * p_{refind} * \pi_{refind} u_i + (1-\alpha) * (1-p_{refind}) \hat{\pi_{logs}}(u_i)$$

$\pi_{refind}(u_i)$ is the ranking of URLs by the probability that they are being re-found in decreasing order. Estimating $\hat{\pi_{logs}}(u_i)$ is discussed in section 4.2. The optimal $\alpha$ of 0.256 is found using stochastic search on the validation set.

## 4. DATASET

The experiments are evaluated using static logs and by querying a live search engine.

### 4.1 AOL Static Log

The AOL log contains 20M distinct queries that were submitted by 650k users from the AOL Search Engine[3], from March 2006 to May 2006. The log contains domain information instead of the full URL for each user click. We, however, feel this is an acceptable approximation because domains often group similar information, either by topic (i.e. the CDC website) or task (i.e. Wikipedia or a news site.) In either case if the click on a repeated domain is intentional, the user may be able to find the intended information by navigating the resulting website. In some cases the same domain appears at different ranks in the search results. Since the full URL to differentiate between domains is absent, repeat occurrences of the same domains in a given result list are omitted. The current query is predicted to be a re-finding relative to all previous queries for a given user, creating a quadratic number of query pairs to test. In order to keep the problem tractable, only queries issued within 100 clicked queries of each other are possible re-finding candidates. This encompasses 88.4% of all re-findings.

### 4.2 AOL Live Crawl

Each query from the static log is issued to the live search engine on May 2010. The top 100 results are used to estimate the rank of a non-clicked result from the static log. Since the static log was collected in 2006, there are some potential compatibility issues, including changes to the cyber landscape as well as improvements to the underlying search engine. For example, the static log contains the query "continental 757-200". As of May 21st, 2010, the second result

---

---

**Table 1: The overall performance of each classifier with optimal classification thresholds.**

| Classifier | Precision | Recall | F-Measure |
|---|---|---|---|
| $Tree_G$ | 0.918 | 0.724 | 0.810 |
| $BMR_G$ | 0.673 | 0.771 | 0.719 |
| $SVM_G$ | 0.697 | 0.833 | 0.759 |
| $Tree_P$ | 0.817 | 0.770 | 0.793 |
| $BMR_P$ | 0.833 | 0.823 | 0.828 |
| $SVM_P$ | 0.870 | 0.816 | 0.843 |

for this query is seatguru.com, which was registered in August 2007. The third result is seatexpert.com, which was registered in May 2008. Neither of these two websites existed in 2006, so neither can be in our initial data set. In general 87.2% of the domains returned in the live crawl were registered before the static logs were collected. Additionally, 53.4% of all domains and 74.0% of all re-findings from the static logs also occur in the live crawl. The relative similarity between the URL ordering of the two lists can be evaluated using the Kendall Rank Correlation Coefficient (KRCC). KRCC measures the pairwise agreements of items, ranging from -1 to 1. The average KRCC over each query was 0.26 which shows general agreement. The average difference $\pi_{logs}(u_i) - \pi_{crawl}(u_i) = 8.6$, however 27.9% of the clicked domains from the static logs were at the same rank as in the live crawl, and 42.0% were within 2. Since the two sets appear similar, we estimate $\hat{\pi_{logs}}(u_i)$ as $\pi_{crawl}(u_i)$, the rank position from the live crawl.

## 5. EVALUATION

We gain insight into the effectiveness of our features by examining the structure of the decision trees. $Tree_G$ first branches on MinimalChange, matching past intution that MinimalChange and SubstantialChange re-findings are fundamentally different[14]. Yet few of the personalized trees used minimal change information (7.1% of all branches), with the most common type of minimal change being exact match (3.8% of all branches). Other common features were SessionPosition (11% of all branches), the probability of the query strings having been used for re-finding in the past (15.1% of all branches) and substring matching (9.8% of all branches). Another important set of features was our "burstiness" features that indicated whether recent queries were re-finding (7.1% of all branches). Overall, the average number of branches in our personalized decision tree models is 24.3. For approximately 21% of our users, however, the model simply predicted no re-finding.

The conditional probability by re-finding for a given feature also varies greatly between users. For each of the binary features, $f$, there is a user with a conditional probability of re-finding, $p(r|f) \geq 95\%$ and another with $p(r|f) \leq 5\%$ for the same feature. Over all users, the standard deviation of the conditional probability of re-finding for minimal change is 33.2%. This suggests that a single global prediction model may be insufficient.

### 5.1 Predicting Re-Finding

Predicting re-findings at ranks near the top of the search results list do not offer much improvement during the re-

**Table 2: The performance of each classifier broken down by rank of the re-finding URL.**

| | Rank | | | | | Ave. |
|---|---|---|---|---|---|---|
| | 1 | 2 - 3 | 4 - 6 | 7 - 9 | 10+ | Rank |
| $Tree_G$ | 28894 | 1,233 | 196 | 34 | 41 | 1.12 |
| $Tree_P$ | 30126 | 3,137 | 753 | 223 | 290 | 1.43 |
| $BMR_G$ | 37029 | 5,818 | 2,254 | 978 | 1960 | 2.68 |
| $BMR_P$ | 31905 | 4,189 | 1,222 | 337 | 313 | 1.50 |
| $SVM_G$ | 36659 | 5,594 | 2,113 | 932 | 1840 | 2.87 |
| $SVM_P$ | 27069 | 3,368 | 869 | 225 | 220 | 1.44 |

**Table 3: The NDCG for each re-ranking method**

| | None | All Re-finds | Top Re-find | Relevant Re-finds | Combo. |
|---|---|---|---|---|---|
| $Tree_G$ | 0.610 | 0.558 | 0.565 | 0.599 | 0.600 |
| $Tree_P$ | 0.541 | 0.484 | 0.504 | 0.523 | 0.523 |
| $BMR_G$ | 0.281 | 0.268 | 0.276 | 0.281 | 0.281 |
| $BMR_P$ | 0.499 | 0.489 | 0.495 | 0.494 | 0.496 |
| $SVM_G$ | 0.383 | 0.369 | 0.379 | 0.383 | 0.383 |
| $SVM_P$ | 0.581 | 0.577 | 0.581 | 0.582 | 0.582 |

ranking. Table 5.1 shows the number of identified re-findings at each rank. Although the personalized models had higher precision and f-measure, the global models are better at identifying re-findings at the higher ranks. 95.1% of the re-found URLs for $Tree_G$ are at rank 1. For $SVM_P$, only 85.3% of the predicted re-finds are at rank 1. Even though the f-measure is similar, this difference in rank indicates that the $SVM_P$ might be the better choice.

Table 1 shows the accuracy for our algorithms at the optimal thresholds of classification. $Tree_G$ far outperforms $SVM_G$ and $BMR_G$ models, likely due to the fact that as the decision tree branches, it is effectively creating different models, and users do not appear to be consistent with each other. Surprisingly, $Tree_G$ with optimal threshold performance is not too different from the more sophisticated models of $SVM_P$ and $BMR_P$.

## 5.2 Re-Ranking

Results of the re-ranking improvements in terms of NDCG are shown in Table 3. NDCG is computed for queries only when a re-finding is predicted, therefore, each classifier is calculated over a different set of queries.

Adding relevance to re-finding helps improve the NDCG over using just re-finding predictions alone, despite the limitations in augmenting a 2006 static log file with query results from 2010. The "Top Re-find First" method outperformed "Re-finds First", which is not surprising given the average re-finding only has one click. For our tree models, re-ranking appears to hurt classification likely because the average rank position for re-finding in our tree models was 1.12 for $Tree_G$ and 1.43 for $Tree_P$. Since the re-findings found by $Tree_G$ model were already at a low rank, there is little room for improvement. The NDCG of re-ranking was worse than the baseline, likely due to the click position bias[5] adversely affecting results.

## 6. CONCLUSION AND FUTURE WORK

In this paper we use re-finding behavior to personalize search. We found Bayesian multinomial regression, decision trees and support vector machines all predicted re-findings well, but that SVMs were better apt at identifying those at lower ranks. We also provided several methods for re-ranking search results: including a direct boosting, and combining relevance with re-finding. A next step for this work is to test our models in a user study. Our evaluation of re-finding to improve the search results is adversely affected by the click bias. By testing our methods out in a live environment, we will have a better indication of how much re-finding can be used to improve ranking.

## 7. ACKNOWLEDGEMENTS

## 8. REFERENCES

[1] D. Abrams, R. Baecker, and M. Chignell. Information archiving with bookmarks: personal web space construction and organization. In *CHI*, 1998.

[2] A. Cockburn, S. Greenberg, S. Jones, B. Mckenzie, and M. Moyle. Improving web page revisitation: analysis, design and evaluation. *IT & Society*, 2003.

[3] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *TOIS*, 20(4), 2002.

[4] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Trans. Inf. Syst.*, 25(2):7, 2007.

[5] A. Komlodi, D. Soergel, and G. Marchionini. Search histories for user support in user interfaces. *J. Am. Soc. Inf. Sci. Technol.*, 57(6):803–807, 2006.

[6] D. Morris, M. Ringel Morris, and G. Venolia. Searchbar: a search-centric web history for task resumption and information re-finding. In *CHI*, 2008.

[7] H. Obendorf, H. Weinreich, E. Herder, and M. Mayer. Web page revisitation revisited: implications of a long-term click-stream study of browser usage. In *CHI*, 2007.

[8] V. V. Raghavan and H. Sever. On the reuse of past optimal queries. In *SIGIR*, 1995.

[9] M. S and S. Dumais. Examining repetition in user search behavior. In *ECIR*, 2007.

[10] L. Tauscher and S. Greenberg. How people revisit web pages: empirical findings and implications for the design of history systems. *IJHCS*, 1997.

[11] J. Teevan, E. Adar, R. Jones, and M. A. S. Potts. Information re-retrieval: repeat queries in yahoo's logs. In *SIGIR*, 2007.

[12] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR*, 2005.

[13] S. K. Tyler and J. Teevan. Large scale query log analysis of re-finding. In *WSDM*, 2010.