Personalizing Web Search using Long Term Browsing History

Nicolaas Matthijs
University of Cambridge
15 JJ Thomson Avenue
Cambridge, UK
nm417@cam.ac.uk

Filip Radlinski
Microsoft
840 Cambie Street
Vancouver, BC, Canada
filiprad@microsoft.com

ABSTRACT

Personalizing web search results has long been recognized as an avenue to greatly improve the search experience. We present a personalization approach that builds a user interest profile using users' complete browsing behavior, then uses this model to rerank web results. We show that using a combination of content and previously visited websites provides effective personalization. We extend previous work by proposing a number of techniques for filtering previously viewed content that greatly improve the user model used for personalization. Our approaches are compared to previous work in offline experiments and are evaluated against unpersonalized web search in large scale online tests. Large improvements are found in both cases.

Categories and Subject Descriptors

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

General Terms: Algorithms, Experimentation

Keywords: AlterEgo, Browsing History, Evaluation, Personalized Web Search, Interleaving, Ranking, User Profile

1. INTRODUCTION

Although web search has become an essential part of our lives, there is still room for improvement. In particular, a major deficiency of current retrieval systems is that they are not adaptive enough to users' individual needs and interests (e.g. [27]). This can be illustrated with the search query "ajax". This query will return results about Ajax based web development, about the Dutch football team Ajax Amsterdam, and websites about the cleaning product Ajax. Clearly, different users would prefer different results. Additionally, previous research has noted that the vast majority of search queries are short [22, 9] and ambiguous [4, 19]. Often, different users consider the same query to mean different things [27, 9, 16, 20, 32]. Personalized search is a potential solution to all these problems.

WSDM'11, February 9–12, 2011, Hong Kong, China.

Personalizing web search has received a lot of attention by the research community (e.g. [1, 2, 6, 7, 8, 11, 13, 14, 15, 16, 20, 21, 23, 25, 26, 28]). We improve upon this work in two key ways: First, we build an improved user profile for personalizing web search results. Second, we improve upon the evaluation methodology, by performing the first large online comparative evaluation of personalization strategies.

To successfully personalize search results, it is essential to be able to identify what types of results are relevant to users. Two alternatives are: (1) ask users to label documents as more personally relevant or not, and (2) infer personal relevance automatically. As the former approach requires extra effort from users, we opt for the latter. In particular, the content of all the web pages visited by users, along with the users' particular behavior on web search results, is used to build a user model. This data was collected using a Firefox add-on created for this purpose. The profile constructed is then used to rerank the top search results returned by a non-personalized web search engine. The key difference from previous work in the profiles we construct is that we parse web page structure, using term extraction and part of speech tagging to extract noun phrases to refine the user model. We show that this yields significant retrieval improvements over web search and other personalization methods, without requiring any effort on the user's part, and without changing the user's search environment.

Second, most previous work on search personalization has involved an evaluation using either (1) a small number of users evaluating the relevance of documents for a small set of search queries not representative of a real workload, (2) the TREC query and document collection, and simulating a personalized search setting, or (3) an after-the-fact log based analysis. Improvements found using these methods do not necessarily translate to actual improvements in user search experience on a real query workload. In this work, we start by using document judgments obtained from a small number of users for 72 queries to assess potential approaches. We then select three methods for complete online evaluation, with our personalized search system being used by 41 users for two months to issue thousands of queries as part of their day to day web search activities. We use an interleaving evaluation approach, that has been shown to accurately reflect differences in ranking relevance [17, 18].

After reviewing related work next, we give an overview of the user profile generation and re-ranking strategies investigated in Section 3. We then describe our evaluation approach in detail, with results from our offline evaluation in Section 5, and online evaluation in Section 6.

2. RELATED WORK

Previous work on search personalization is typically characterized by the data source used to learn about the user on one hand, and the way in which a user is modeled on the other hand.

Observed User Interactions

A number of personalization approaches using previous user interactions with the search engine to describe the users have been proposed. This has the benefit that such usage data is easily collected by search engines.

Aiming for short-term personalization, Sriram et al. [24] describe a search engine that personalized based on the current user session. Although this approach is shown to improve retrieval quality, session data is often too sparse to personalize ideally, and does not allow personalization before the second query in each session. Similarly, [6] propose using such session-level personalization.

A longer term personalization click model can also be used, exploting clickthrough data collected over a long time period. For example, Speretta and Gauch [23] and Qiu and Cho [16] model users by classifying previously visited web pages into a topic hierarchy, using this model to rerank future search results. Similarly, Joachims [11] proposes using user click-through data as training data to learn a general search retrieval function, which can then be used to produce personalized rankings for individual users or groups of users. Other related approaches include [20, 25, 26].

Also, a particularly straightforward yet effective search interaction personalization approach is PClick, proposed by Dou et al. [7]. This method involves promoting URLs previously clicked on by the same user for the same query. We will compare our approach to PClick, and also extend it to all previously visited web pages similarly to [28].

In only using search interaction data, and often limited within the same search session, the methods described above suffer from data sparsity. As such, they often must re-rank results with only a limited amount of data about the user. Other methods have attempted to incorporate more information about the user by using the full browsing history (e.g. Sugiyama et al [25]).

The most promising profile based approach was proposed by Teevan et al. [28]. They use a rich model of user interests, built from search-related information, previously visited web pages, and other information about the user including documents on their hard drive, e-mails, and so forth. They then use this data to re-rank the top returned web search results, by giving more weight to terms deemed more personally relevant. In doing this, they obtain a significant improvement over default web ranking. We will compare our method to this term reweighting approach.

Representing the User

Irrespective of the source of data about users, a model must encode this data. A variety of such models have been used in the past. These include using a vector of weighted terms (e.g. [5, 28]), a set of concepts (e.g. [14]), an instance of a predefined ontology (e.g. [8, 15, 21]) or a hierarchical category tree based on ODP and corresponding keywords (e.g. [2, 13]). In this paper, we will focus on modeling users through a vector of weighted terms.

In particular, Teevan et al. [28] make use a rich keyword-based representation of users, utilizing a desktop index which indexes files on the user's hard drive, e-mails, visited web pages and so on. However, this approach treats web documents as common documents and does not take advantage of the characteristics and structure encapsulated within a web page. In this paper, we focus just on web documents, using users' complete browsing history. We also exploit the specific characteristics and structure of web pages, showing this yields substantial improvements. Additionally, consider a variety of different weighting schemes to improve retrieval quality.

Finally, some previous research suggests that such profile based personalization may lack effectiveness on unambiguous queries such as "london weather forecast", and therefore no personalization should be attempted in these cases [29]. However, if this or a related query has been issued by this user before, we could detect any preference for particular weather forecast websites by using the user's URL history as suggested by the PClick approach [7]. Hence, we find that a combination of user representations is important.

Commercial Personalization Systems

Recently, personalized search has also been made available in some mainstream web search engines including Google[1] and Yahoo!. These appear to use a combination of explicitly and implicitly collected information about the user. Many more companies are engaging in personalization both for search (e.g. surfcanyon.com) and for advertising based on user behavior. However, as the details of these approaches are not publicly available and because it is hard to programmatically get access to these personalized rankings, we only compare our approach to the default search engine ranking and not the personalized version.

3. PERSONALIZATION STRATEGIES

In this section, we describe our approach. The first step consists of constructing a user profile, that is then used in a second phase to re-rank search results.

3.1 User Profile Generation

A user is represented by a list of terms and weights associated with those terms, a list of visited URLs and the number of visits to each, and a list of past search queries and pages clicked for these search queries. This profile is generated as shown in Figure 1. First, a user's browsing history is collected and stored as (URL, HTML content) pairs. Next, this browsing history is processed into six different summaries consisting of term lists. Finally, the term weights are generated using three different weighting algorithms. We now describe each of these steps in detail.

3.1.1 Data Capture

To obtain user browsing histories, a Firefox add-on called AlterEgo was developed. To respect the users' privacy as much as possible, a random unique identifier is generated at installation time. This identifier is used for all data exchange between the add-on and the server recording the data[2].

[1]http://googleblog.blogspot.com/2009/12/personalized-search-for-everyone.html

[2]Note that it is necessary to collect this data server-side for research purposes, but our approach does not require the data to be centralized. Our entire method can execute client-side, avoiding the privacy concerns that arise with

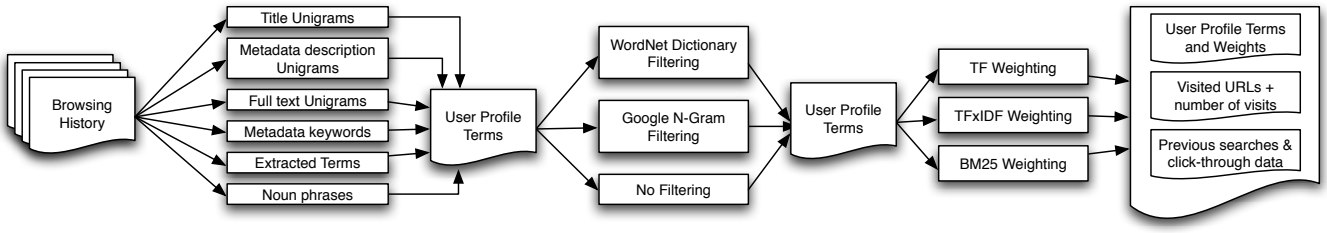

Figure 1: User Profile Generation Steps and Workflow

Table 1: Captured Data Statistics

Metric	Total	Min	Max	Mean
Page Visits	530,334	51	53,459	10,607
Unique Page Visits	218,228	36	26,756	4,365
Google Searches	39,838	0	4,203	797
Bing Searches	186	0	53	4
Yahoo Searches	87	0	29	2
Wikipedia Pages	1,728	0	235	35

Every time a user leaves a non-secure (non-https) web page, the add-on transmits the user's unique identifier, the page URL, the visit duration, the current date and time, and the length of the source HTML to the server. The server then attempts to fetch the source HTML of this page. This is performed server-side to ensure that only publicly-visible data is used. Once the source HTML is received, the server compares its length to the length received from AlterEgo. If the length difference is smaller than 50 characters, the HTML is accepted and saved along with the other page visit data. Otherwise, we assume the content probably came from a password protected but non-secure site (e.g. Facebook, Hotmail, etc.) and the record is discarded.

Participants for this study were recruited via a website explaining the purpose and consequences to potential users, publicized on various e-mail lists, resulting in 50 participants taking part. Whilst we expect that most of these participants are employed in the IT industry due to the recruitment process, a number of people outside of the IT industry without significant web search experience participated as well. The add-on captured data for three months from March to May 2010. As shown in Table 1, a total of 530,334 page visits (or an average of 10,607 page visits per user) were recorded. 58% of the visits were to unique pages. The add-on also recorded 39,838 Google searches, 186 Bing searches and 87 Yahoo! searches, indicating that our users were strongly biased towards Google as their search engine, hence Google was used as the baseline in our experiments. An average user issued 797 queries over the three months, indicating that at least 7.5% of all non-secure web requests were search related.

3.1.2 Data Extraction

We considered the following summaries of the content viewed by users in building the user profile:

Full Text Unigrams
The body text of each web page, stripped of html tags.

server-based approaches. The add-on was optimized to be not noticeably slower than the non-personalized web search.

Table 2: Extracted terms from the AlterEgo website and the Wikipedia page about Mallorca

AlterEgo	Mallorca
add-ons	majorca
Nicolaas	palma
Matthijs	island
CSTIT	spanish
Nicolaas Matthijs	balearic
Language Processing	cathedral
Cambridge	Palma de Mallorca
keyword extraction	port

Title Unigrams
The words inside any <*title*> tag on the html pages.

Metadata Description Unigrams
The content inside any <*meta name="description"*> tag.

Metadata Keywords Unigrams
The content inside any <*meta name="keywords"*> tag.

Extracted Terms
We implemented the Term Extraction algorithm as presented in [31], running it on the full text of each visited web page. It attempts to summarize the web page's text into a set of important keywords. This algorithm uses the C/NC method, which uses a combination of linguistic and statistical information to score each term. Term candidates are found using a number of linguistic patterns and are assigned a weight based on the frequency of the term and its subterms. This is supplemented with term re-extraction using the Viterbi algorithm. The outcome of this algorithm run on two sample web pages can be seen in Table 2.

Noun Phrases
Noun phrases were extracted by taking the text from each web page and splitting it into sentences using a sentence splitter from the OpenNLP Tools[3]. The OpenNLP tokenization script was then run on each sentence. The tokenized sentences were tagged using the Clark & Curran Statistical Language Parser[4] [3], which assigns a constituent tree to the sentence and part of speech tags to each word. Noun phrases were then extracted from this constituent tree.

3.1.3 Term List Filtering

To reduce the number of noisy terms in our user representation, we also tried filtering terms by removing infrequent words or words not in WordNet. However, neither of these were found to be beneficial. Therefore we do not discuss term list filtering further.

[3]http://opennlp.sourceforge.net/
[4]http://svn.ask.it.usyd.edu.au/trac/candc/wiki

3.1.4 Term Weighting

After the list of terms has been obtained, we compute weights for each term in three ways.

TF Weighting

The most straightforward implementation we consider is Term Frequency (TF) weighting. We define a frequency vector $\vec{F}$ that contains the frequency counts of a given term t_i for all of the input data sources, as shown in Equation (1). For example, f_{title} is the number of times a given term t_i occurs in all of the titles in the user's browsing history. We calculate a term weight based on the dot product of these frequencies with a weight vector $\vec{\alpha}$:

$$\vec{F}_{t_i} = \begin{bmatrix} f_{title_{t_i}} \\ f_{mdesc_{t_i}} \\ f_{text_{t_i}} \\ f_{mkeyw_{t_i}} \\ f_{terms_{t_i}} \\ f_{nphrases_{t_i}} \end{bmatrix} \quad (1)$$

$$w_{TF}(t_i) = \vec{F}_{t_i} \cdot \vec{\alpha} \quad (2)$$

For simplicity, we limit ourselves to three possible values for each weight α_i: 0, ignoring the particular field, 1, including the particular field, and $\frac{1}{N_i}$, where N_i is the total number of terms in field i. This gives more weight to terms in shorter fields (such as the meta keywords or title fields). We call the last *relative weighting*.

TF-IDF Weighting

The second option we consider is TF-IDF (or Term Frequency, Inverse Document Frequency) weighting. Here, words appearing in many documents are down-weighted by the inverse document frequency of the term:

$$w_{TFIDF}(t_i) = \frac{1}{log(DF_{t_i})} \times w_{TF}(t_i) \quad (3)$$

To obtain IDF estimates for each term, we use the inverse document frequency of the term on all web pages using the Google N-Gram corpus[5].

Personalized BM25 Weighting

The final weight method we consider was proposed by Teevan et al. [28], which is a modification to BM25 term weighting:

$$w_{pBM25}(t_i) = log\frac{(r_{t_i}+0.5)(N-n_{t_i}+0.5)}{(n_{t_i}+0.5)(R-r_{t_i}+0.5)}, \quad (4)$$

where N represents the number of documents on the web (estimated from the Google N-Gram corpus, 220,680,773), n_{t_i} is the number of documents in the corpus that contain the term t_i (estimated using the Google N-Gram corpus), R is the number of documents in the user's browsing history and r_{t_i} is the number of documents in the browsing history that contains this term within the selected input data source.

While this method allows us to compare our results against the approach proposed by Teevan et al., note that we do not have access to users' full Desktop index, and are limited to their browsing history, making our implementation of their approach potentially less effective.

[5]http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html

3.2 Re-ranking Strategies

Like previous work, we use the user profile to re-rank the top results returned by a search engine to bring up results that are more relevant to the user. This allows us to take advantage of the data search engines use to obtain their initial ranking, by starting with a small set of results that can then be personalized. In particular, [28] noted that chances are high that even for an ambiguous query the search engine will be quite successful in returning pages for the different meanings of the query. We opt to retrieve and re-rank the first 50 results retrieved for each query.

3.2.1 Scoring Methods

When reranking, each candidate document can either be scored, or just the snippets shown on the search engine result page can be scored. We focus on assigning scores to the search snippets as it was found to be more effective for re-ranking search results by Teevan et al. [28]. Also, using search snippets allows a straightforward client-side implementation of search personalization. We implemented the following four different scoring methods:

Matching

For each word in the search snippet's title and summary that is also in the user's profile, the weight associated with that term will be added to the snippet's score:

$$score_M(s_i) = \sum_{z=1}^{N_{s_i}} f_{t_z} \times w(t_z) \quad (5)$$

where N_{s_i} represents the total number of unique words within the snippet's title and summary, and f_{t_i} represents the number of occurrences of t_i within the snippet. Words in the snippet title or summary but not in the user's profile do not contribute towards the final score. This method is equivalent to taking the dot product between the user profile vector and the snippet vector.

Unique Matching

A second search snippet scoring option we consider involves counting each unique word just once:

$$score_{UM}(s_i) = \sum_{z=1}^{N_{s_i}} w(t_z) \quad (6)$$

Language Model

The third score calculation method generates a unigram language model from the user profile in which the weights associated with the terms are used as the frequency counts for the language model:

$$score_{LM}(s_i) = \sum_{z=0}^{N_{s_i}} \log\left(\frac{w(t_z)+1}{w_{total}}\right) \quad (7)$$

where N_{s_i} is the total number of words in the snippet's title and summary, and w_{total} stands for the sum of all the weights within the user profile. The language model estimates the probability of a snippet given a user's profile. To avoid a zero probability for snippets that contain words not in the user's profile, we use add-1 smoothing.

PClick

As a final snippet scoring method, we use the PClick algorithm proposed by Dou et al. [7]. It assumes that for a query

q submitted by a user u, the web pages frequently clicked by u in the past are more relevant to u. The personalized score for a snippet is:

$$score_{PC}(s_i) = \frac{|Clicks(q,p,u)|}{|Clicks(q,\bullet,u)| + \beta} \quad (8)$$

where $|Clicks(q,p,u)|$ is the number of clicks on web page p by user u for query q in the past, $|Clicks(q,\bullet,u)|$ is the total click number on query q by u, and β is a smoothing factor set to 0.5. Note that PClick makes no use of the terms and weights associated to the user's profile and is solely based on click-through data for a given query. As such, it only affects repeated queries.

3.2.2 Rank and Visit Scoring

Finally, we consider two adjustments to the snippet scores. First, in the re-ranking framework discussed so far, the original ranking is not taken into account. The original rank can be incorporated into the final snippet score by multiplying the snippet weight by the inverse log of the snippet's original rank r_{s_i}:

$$\text{finalScore}(s_i) = score(s_i) \times \frac{1}{1 + log(r_{s_i})} \quad (9)$$

Second, we consider giving additional weight to URLs that have been visited previously. This extends PClick in that it boosts *all* URLs that have previously been visited, while PClick only boosts URLs that have directly been clicked for the current search query. The snippet score will be boosted by the number of previous visits to that web page (n_i) times a factor v:

$$\text{finalScore}(s_i) = score(s_i) * (1 + v \times n_i) \quad (10)$$

4. EVALUATION APPROACH

We now consider potential evaluations for personalized search strategies. On the one hand, offline approaches allow the creation of a standard dataset that can be used to optimize personalization parameters. On the other hand, only an online test with actual users can truly reflect how changes to rankings affect user behavior. We now explore the available alternatives, and describe our final strategy.

Relevance judgements

The first possible offline evaluation approach (e.g. used by Teevan et al. [28]) is based on assembling a group of people that judge the relevance of the top k documents or search snippets for a set of queries. Given these relevance judgements, a metric such as (N)DCG or (Normalized) Discounted Cumulative Gain [10] can be calculated for a given query and ranking, reflecting the quality of the presented ranking for that user. This approach has the advantage that once the relevance judgements are made, it allows for testing many different user profile and re-ranking parameter configurations. However, due to the long time it takes to judge k documents, this can only be done for a small number of search queries. As volunteers need to be found to sit through this slow and tedious evaluation process, it is also hard to gather a large group of evaluators. The evaluation process also does not reflect a user's normal browsing and searching behavior, which might influence the final results. Moreover, this approach assumes that (N)DCG is the right way to combine a set of relevance judgements into a rank quality score. Finally, the queries evaluated must be representative of a true query load, or offline results may not reflect perhaps poorer performance for non-personalizable queries.

Side-by-side evaluation

An alternative offline evaluation method, previously used for example by [30], consists of presenting users with two alternative rankings side-by-side and asking which they consider best. The advantage of this method is that a judgement is made of which ranking is the best one, evaluating the entire presented ranking. However, in real life situations users might only look at the first couple of results, potentially biasing the evaluation. Judging two rankings next to each other is considerably faster than judging k documents per query, but it still requires a long offline evaluation exercise. Additionally, an evaluator has to provide a new assessment for each distinct ordering of documents that is investigated. This makes it hard to use such judgments to tune reranking parameters.

Clickthrough-based evaluation

One common online evaluation approach involves looking at the query and click logs from a large search engine (e.g. used by [7]). The logs record which search results were clicked for each query, thus allowing the evaluator to check if the clicked result would be positioned higher in a personalized ranking. This allows for testing many parameter configurations and also does not require any additional user effort as such logs reflect natural user behavior. However, the method can have difficulties in assessing whether a search personalization strategy actually works. First, users are more likely to click a search result presented at a high rank, although these are not necessarily most or more relevant [12]. It is also unsuccessful in assessing whether lower results would have been clicked had they been shown at a higher rank. Finally, we have no access to such large scale usage and user profile data for this experiment.

Alternatively, both personalized and non-personalized rankings can be shown online to users, with metrics such as mean clickthrough rates and positions being computed. However, [18] showed that such an approach is not sensitive enough to detect relatively small differences in relevance with thousands of queries as we could obtain in an online experiment.

Interleaved evaluation

The final online evaluation we consider, which to our knowledge has not been used before for evaluating personalized search, is interleaved evaluation [11, 18]. Interleaved evaluation combines the results of two search rankings by alternating between results from the two rankings while omitting duplicates, and the user is presented with this interleaved ranking. The ranking that contributed the most clicks over many queries and users is considered better. Radlinski et al. [18] showed that this approach is much more sensitive to changes in ranking quality than other click-based metrics. It has also shown to correlate highly with offline evaluations with large numbers of queries [17]. On top of that, this method does not require any additional effort from the user, reflecting normal search engine usage. However, one evaluation only provides an assessment for one particular ranking, and thus an evaluation is required for each parameter configuration being investigated.

In general, online metrics are harder to improve than offline metrics. First, bringing a relevant result to a position where it is not clicked has no effect: For example, a result moved up from rank 8 to rank 3 will have no effect if the user only selects the rank 1 result. Second, it measures performance on complete query workload, avoiding placing an emphasis on a small sample of typical queries. In consequence, such online methods provide a more reliable measurement as to whether personalization yields a real improvement.

4.1 Evaluation Design

This last online approach, interleaved evaluation, is most sensitive and reliable, and best reflects real user experience. It would thus be preferred for evaluating a personalized search system. However, our user profile generation and re-ranking steps both have a large number of parameters, and it is infeasible to perform an online evaluation for all of them. Hence, we start with an offline NDCG based evaluation to pick the optimal parameter configurations, that we then evaluate with the more realistic and harder online interleaved evaluation.

5. OFFLINE EVALUATION

We now describe how we collected relevance judgments for offline evaluation, and how these were used to identify the most promising personalization strategies to evaluate online.

5.1 Relevance Judgements

Six participants who had installed the AlterEgo plugin were recruited for an offline evaluation session. At that point, two months of browsing history had been recorded and stored for each. Mirroring the approach in [28], each participant was asked to judge the relevance of the top 50 web pages returned by Google for 12 queries according to the criteria in Table 3. The documents were presented in a random order, and required the users to look at the full web pages rather than the result snippets.

Participants were first asked to judge their own name (Firstname Lastname) as a warm-up exercise. Next, each participant was presented with 25 general queries in a random order, consisting of sixteen taken from the TREC 2009 Web Search track and nine other UK focused queries such as "football" and "cambridge". Each participant was asked to judge 6 of these. Next, each participant was presented with their most recent 40 search queries (from their browsing history) and were asked to judge 5 for which they remembered the returned results could have been better. Examples of selected queries of both types are shown in Table 4.

On average, each participant took about 2.5 hours to complete this exercise. Particularly interestingly, *all* participants mentioned that during the exercise they came across useful websites of which they were previously unaware, indicating that there is a potential for search personalization to improve the search experience.

5.2 Results and Discussion

The following parameters were investigated for profile generation, representing the different steps shown in Figure 1:

- All combinations of the six different input data sources with three possible values for α (0, 1 and normalized)
- Three term weighting methods: TF, TF-IDF and pBM25

Table 3: Offline relevance judgement guidelines

(a) Select *Not Relevant* if the document is not useful and not interesting to you.
(b) Select *Relevant* if the document is interesting to you, but is not directly about what you were hoping to find or if the document is somewhat useful to you, meaning that it touches on what you were hoping to find (maximum 1 paragraph), but not very extensively.
(c) Select *Very Relevant* if the document is useful or very interesting to you, i.e. it is what you were hoping to find.

Table 4: Some queries selected by study participants

Prepared Queries	History-Based Queries
Cambridge	Abbey Pool
GPS	BBC
Website design hosting	Titanium iPhone
Volvo	Vero Moda

The following reranking parameters were investigated:

- The four snippet weighting methods: Matching, Unique Matching, Language Model and PClick
- Whether or not to consider the original Google rank
- Whether or not to give extra weight to previously visited URLs. A weight of $v = 10$ was used because this appeared to give the best performance in early tests

For every profile and ranker combination, the mean personalized NDCG@10 was measured as follows:

$$NDCG@10 = \frac{1}{Z}\sum_{i=1}^{10}\frac{2^{rel_i}-1}{\log_2(1+i)} \quad (11)$$

where rel_i is the relevance judgement of the document (non-relevant = 0, relevant = 1 and very relevant = 2) and Z is such that the maximum NDCG for each query is 1. In all of these following results, we compare NDCG scores that have been averaged across all queries and all users.

5.2.1 Reranking Performance

We selected four user profile and re-ranking parameter settings to evaluate further, summarized in Table 5. We compared these with several baselines. Our results are summarized in Table 6. All reported significance figures were obtained using a two-tailed t-test.

We compare the performance of our approach against are the default (non-personalized) Google ranking, term reweighting as proposed by Teevan et al. [28] and the PClick method [7]. These results agree with previously published results. However, we note that our implementation of the Teevan algorithm only uses of the user browsing history as input data, as we do not have access to the user's files or e-mails, which may disadvantage it.

The strategies we evaluate in depth are (1) `MaxNDCG`, which yielded the highest average NDCG score on the offline dataset; (2) `MaxQuer`, which improved the most queries; (3) `MaxNoRank`, the method with highest NDCG that does not take the original Google ranking into account; and (4) `MaxBestPar`, obtained by greedily selecting each parameter sequentially

Table 5: Selected personalization strategies. *Rel* indicates relative weighting, $v = 10$ indicates setting parameter v to 10 in Equation 10. For parameter descriptions, see Section 3.

Strategy	Profile Parameters							Ranking Parameters		
	Full Text	Title	Meta Keywords	Meta Descr.	Extracted Terms	Noun Phrases	Term Weights	Snippet Scoring	Google Rank	Urls Visited
`MaxNDCG`	–	Rel	Rel	–	–	Rel	TF-IDF	LM	1/log	v=10
`MaxQuer`	–	–	–	–	Rel	Rel	TF	LM	1/log	v=10
`MaxNoRank`	–	–	Rel	–	–	–	TF	LM	–	v=10
`MaxBestPar`	–	Rel	Rel	–	Rel	–	pBM25	LM	1/log	v=10

Table 6: Summary of offline evaluation results

Method	Average NDCG	+/=/− Queries
Google	0.502 ± 0.067	–
Teevan	0.518 ± 0.062	44/0/28
PClick	0.533 ± 0.057	13/58/1
`MaxNDCG`	**0.573 ± 0.042**	48/1/23
`MaxQuer`	0.567 ± 0.045	**52/2/18**
`MaxNoRank`	0.520 ± 0.060	13/52/7
`MaxBestPar`	0.566 ± 0.044	45/5/22

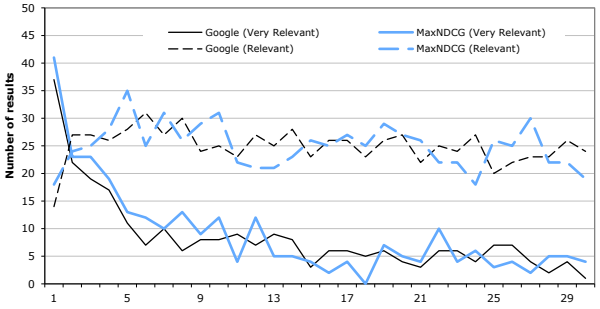

Figure 2: Distribution of relevance at rank for the Google and `MaxNDCG` rankings

in the order Title, Full Text, Meta Keywords, Meta Description, Extracted Terms, Noun Phrases, Term Weights, Snippet Scoring, Urls Visited, Snippet Scoring.

`MaxNDCG` and `MaxQuer` are both significantly ($p < 0.01$) better than default Google, Teevan and PClick. `MaxNDCG`, with an average NDCG of 0.573, yields a 14.1% improvement over Google, and `MaxQuer`, with an average NDCG of 0.567, yields a 12.9% improvement over Google.

Interestingly, despite `MaxNoRank` ignoring the Google rank, it obtains an NDCG score that is significantly ($p < 0.05$) better than Google, and better than Teevan. While this may be a result of overfitting the parameters given our small offline dataset, we observed many such parameter combinations, hence we do not believe this to be the case.

An alternative metric for comparing the personalization methods is the number of queries for which the NDCG score improved, was unchanged, or got worse – as shown on the right of Table 6. Interestingly, while PClick improves fewest, it performed better than Teevan in terms of NDCG. This is because the PClick method only works on repeated queries, but makes bigger improvements on average. Also, the Teevan approach has a negative effect on many queries.

5.2.2 *Relevance Judgment Distribution*

Of the 3,600 offline relevance judgements collected, 9% were Very Relevant, 32% Relevant and 58% Non-Relevant. The relevance judgement distribution for the Google ranking and `MaxNDCG` re-ranking are shown in Figure 2. We see that the Google ranking manages to place many Very Relevant results in the top 5 results. While `MaxNDCG` adds more Very Relevant results into the top 5 positions, it mainly succeeds in adding Very Relevant results between rank 5 and 10. This is expected as the personalization strategy considers the Google rank and is less aggressive at high ranks.

5.2.3 *Parameter Effects*

To study the effect of each parameter, we now look at how they affect the overall strategy performance. In Figure 3, for each parameter p (e.g. term weighting, etc), we count how often the different possible weights of p obtained the highest NDCG across all other parameter combinations. While this ignores interaction effects, it provides a good insight into the effectiveness of each parameter. Note that most parameters individually only make a small difference in NDCG, even if they are preferred in most cases.

Profile Parameters

When generating users' profiles, we see that all data sources are individually helpful in improving personalization performance, except for the full web page text. This indicates that treating web pages like a normal document or a bag of words does not work, presumably due to their noisy nature. Using metadata keywords, extracted terms and the page title all yield significant ($p < 0.01$) improvements. Metadata description and extracted noun phrases give a significant ($p < 0.05$) but smaller improvement. The different input data sources can be ranked in terms of helpfulness for personalization: metadata keywords, extracted terms, title, metadata description and extracted noun phrases. However, a combination of the most helpful data sources does not necessarily achieve the best performance, as strong interactions exist.

It is worth noting that both term extraction and shallow parsing, which are used for the first time for search personalization in this paper, provide an improvement in search personalization. It can also be seen that using relative weighting consistently performs better than giving each data source the same weight. This can be explained by the fact that the most helpful data sources are those that contain the fewest terms.

When computing term weights, Term Frequency performs significantly worse than both TF-IDF and pBM25 ($p < 0.01$). pBM25 performs on average significantly better ($p < 0.05$) than TF-IDF. However, given all other parameter combinations, TF-IDF and pBM25 are best roughly equally often. pBM25 seems to work better in general when the input data is richer and thus noisier.

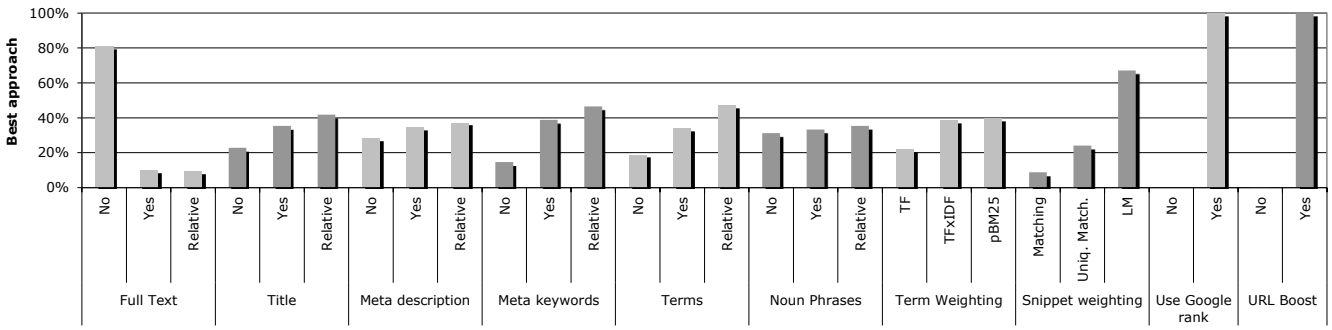

Figure 3: Fraction of parameter combinations on which each investigated parameter performs best.

Ranking Parameters

It is clear that one approach significantly outperforms all others. Using Matching for snippet scoring is significantly worse ($p < 0.05$) than Unique Matching, and both are significantly worse ($p < 0.01$) than using a Language Model based on the user profile. Unique Matching occasionally performs better than a Language Model when the user profile is only keyword based. Except for a single case, multiplying the snippet weight by 10 for URLs that have previously been visited helps. This is as expected given that previous visits to webpages are an indication of relevance. Including the original result rank from Google always helps as well.

6. ONLINE EVALUATION

Our final evaluation is a large scale online interleaved evaluation. It is crucial that the personalization system is evaluated by users performing regular day-to-day searches with real information needs. This allows the evaluation to assess whether the personalization yields an actual improvement in users' search experience. Also, given the limited size of our offline relevance data, if we did not perform an online test then our results may be overfitting to the dataset used to select the best strategies.

Based on the offline evaluation results, we selected the three most promising parameter settings to evaluate online, namely `MaxNDCG`, `MaxQuer` and `MaxBestPar`. We now describe the details of the online evaluation, then discuss our results.

6.1 Interleaving Implementation

An updated version of the Firefox add-on was developed and all volunteers who installed the initial version were asked to upgrade. This version detected Google web searches, and sent the search query, the unique user identifier and the current page number to the server. The first 50 search results were requested from Google, and one of the three personalization strategies was picked at random. The selected strategy was then used to rerank the search results.

Given the personalized and original ranking, interleaving is used to produce a combined ranking. Essentially interspersing the results from the two rankings, so that a click at random would be equally likely to be on a result from either ranking, interleaving allows clicks to provide an unbiased within-user test as to whether the original ranking or the personalized ranking is better. This interleaved ranking is presented to the user, indistinguishably from a normal Google ranking, and any clicks are recorded. If results from the personalized ranking are clicked more often, this is a strong indicator that the personalization was successful.

Algorithm 1 Team-Draft Interleaving [18]

Input: Rankings $A = (a_1, a_2, \dots)$ and $B = (b_1, b_2, \dots)$
Init: $I \leftarrow (); TeamA \leftarrow \emptyset; TeamB \leftarrow \emptyset;$
while $(\exists i : A[i] \notin I) \wedge (\exists j : B[j] \notin I)$ **do**
  if $(|TeamA| < |TeamB|) \vee$
    $((|TeamA| = |TeamB|) \wedge (RandBit() = 1))$ **then**
    $k \leftarrow \min_i\{i : A[i] \notin I\}$ ... *top result in A not yet in I*
    $I \leftarrow I + A[k]$; ..... *append it to I*
    $TeamA \leftarrow TeamA \cup \{A[k]\}$ ..... *clicks credited to A*
  else
    $k \leftarrow \min_i\{i : B[i] \notin I\}$ ... *top result in B not yet in I*
    $I \leftarrow I + B[k]$ ..... *append it to I*
    $TeamB \leftarrow TeamB \cup \{B[k]\}$ .... *clicks credited to B*
  end if
end while
Output: Interleaved ranking I, $TeamA$, $TeamB$

In our experiments, we use the Team-Draft interleaving algorithm, as described in Algorithm 1 [18]. Intuitively, this algorithm is motivated by how sports teams are often assigned in friendly games: Given a pool of available players (all the results present in rankings A and B), two captains (one for TeamA and one for TeamB) take turns picking their next preferred player from the set of remaining players, subject to a coin toss every turn that determines which captain gets to pick first. The selection order determines the order of results shown to users, and player team indicates which "team" a click on this result counts for.

The user gives one vote per query impression to one of these two rankings. Suppose the user clicked on a results from ranking A and b results from ranking B. If a > b, we can say that the user has a preference for ranking A and gives his vote to ranking A. If b > a, the user votes for ranking B. When the user clicks equally often on ranking A and B, there is a tie and no vote is given. Due to space constraints, we refer the reader to [18] for further details.

To avoid presenting slightly different rankings every time a search page is refreshed, both the random personalization strategy selection and the random bits inside the interleaving algorithm were seeded with a combination of the unique user identifier, query and the current hour.

6.2 Results and Discussion

We performed an interleaved evaluation over two months, for the 41 users who updated the plugin. A total of 7,997 individual queries and 6,033 query impressions were observed. Of these, a total of 6,534 individual queries and 5,335 query

Table 7: Results of online interleaving test

Method	Queries	Google Vote	Re-ranked Vote
MaxNDCG	2,090	624 (39.5%)	955 (60.5%)
MaxQuer	2,273	812 (47.3%)	905 (52.7%)
MaxBestPar	2,171	734 (44.8%)	906 (55.2%)

Table 8: Queries impacted by search personalization

Method	Unchanged	Improved	Deteriorated
MaxNDCG	1,419 (67.9%)	500 (23.9%)	171 (8.2%)
MaxQuer	1,639 (72.1%)	423 (18.6%)	211 (9.3%)
MaxBestPar	1,485 (68.4%)	467 (21.5%)	219 (10.1%)

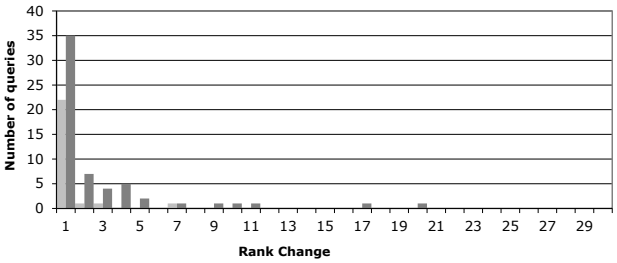

Figure 4: Rank differences for deteriorated (light grey) and improved queries (dark grey) for MaxNDCG

impressions received a click on a search result. This data set is larger than what was found necessary in [18] to assess the performance of different systems. About 24% of the queries were repeated queries.

Each query impression run by a user provides a vote for either the original Google ranking or the personalized ranking. In about 8% of the cases, the number of clicks for the original and the re-ranked version were the same and a tie was obtained. For the non-tie cases, the total number of votes for each strategy is shown in Table 7. We see that all three personalization approaches yield an improvement over the default Google ranking. MaxNDCG significantly outperforms ($p < 0.001$) the web ranking. MaxQuer and MaxBestPar outperform web ranking as well, although the improvements are smaller (but still significant, with $p < 0.01$ and $p < 0.05$ respectively). This suggests that MaxNDCG is best, matching our offline findings.

As an alternative summary of the results, the effect of personalization on search queries is shown in Table 8, counting how many queries were improved, unchanged or hurt by personalization: The *Unchanged* column indicates the number of queries for which the clicked result was at the same rank for both the non-personalized and the personalized ranking. The *Improved* column shows how often the clicked result was brought up, while the *Deteriorated* column shows the number of queries for which the clicked result was pushed down. These numbers are consistent with the interleaving results. On average, about 70% of the time the rank of the clicked result does not change[6], 20% of the queries are improved and 10% became worse. MaxNDCG is again the most successful personalization approach, having the highest change rate, and improving 2.7 times more queries than it harms.

[6]This is larger than the fraction of queries with ties in the interleaving analysis since Team-Draft interleaving always makes a preference when there is one click. This does not bias the summary results [17].

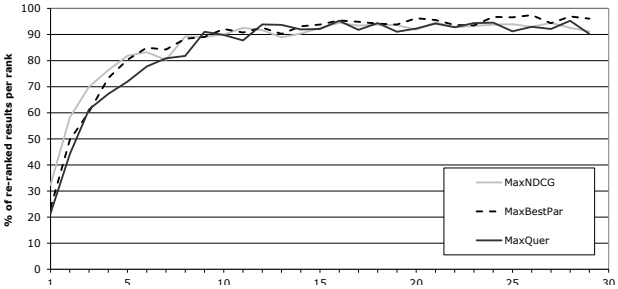

Figure 5: Degree of personalization per rank

Figure 4 shows the distribution of rank changes for all queries that were improved or became worse by MaxNDCG. The equivalent plots for the other strategies are very similar. We see that for a large majority of the deteriorated queries, the clicked result only loses 1 rank position compared to the original ranking. The majority of clicked results that improved a query gain 1 rank as well, however there are many clicked results that are pushed up 2 or more ranks. For all personalization strategies, the average rank deterioration is about 1.38 and the average rank improvement is around 3.5, indicating that the gains from personalization are on average more than double the losses.

Finally, Figure 5 shows how often the personalized and original rankings differ at a particular position for the three approaches. We see that most re-ranking is done after rank 10, having little or no influence on users' search experience, which explains why non-personalizable queries aren't hurt.

In summary, MaxNDCG is the most effective personalization strategy according to both our online and offline tests. It performs significantly better than previous best approaches and outperforms the other parameter combinations in the relevance judgement evaluation. MaxNDCG is also significantly better than the other investigated approaches in the large scale online evaluation.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we investigated personalized web search, first learning users' long-term interests, and then reranking the first 50 search results from a search engine based on this profile. Our proposed personalization techniques significantly outperform both default Google ranking and the best previous personalization methods, which are also directly compared to each other for the first time. This is also the first large scale personalized search and online evaluation work for general web search that was not carried out at a search company.

We discover that the key to using web pages to model users is to not treat them as flat documents, rather as structured documents from which several types of data can be extracted. We also find that term extraction and document parsing can be beneficial for search personalization. The suggested methods can be implemented straightforwardly at large scale, and can be used without violating users' privacy.

The Firefox add-on used is available for download, allowing for personalization without altering the user's browsing experience. The source code is also available for download for the research community[7].

There are a number of natural extensions for this work.

[7]http://github.com/nicolaasmatthijs/AlterEgo

First, the set of parameters can still be expanded: (1) learning the parameter weights and (2) using other fields, such as headings in HTML, and learning the weights for each field, may yield further substantial improvements. Also, temporal information could be incorporated: (1) investigating how much browsing history should be used, (2) whether decaying the weight of older items is beneficial and (3) study how page visit duration can be usefully incorporated into the personalization algorithm. Additionally, a browser add-on has access to other behavioral information, such as time spent on a page, amount of scrolling, text selection and mouse activity, that we do not explore. Similarly to [28], we could also make use of more personal data such as user's files and e-mails.

Finally, one could also consider using the extracted profiles for purposes other than personalized search. After the experiments described had passed, all participants were presented with one of their keyword based profiles. Most users indicated that they were stunned by how well these profiles described them and that they would use the same set of keywords to describe themselves if asked. This indicates that there is a potential of using such profiles in different areas, such as personalizing advertisements, suggesting news articles to read or perhaps even interesting social networking groups to join.

8. REFERENCES

[1] P.-A. Chirita, C. S. Firan, and W. Nejdl. Summarizing local context to personalize global web search. In *Proc. of CIKM*, 2006.

[2] P. A. Chirita, W. Nejdl, R. Paiu, and C. Kohlschütter. Using ODP metadata to personalize search. In *Proc. of SIGIR*, 2005.

[3] S. Clark and J. R. Curran. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Comput. Linguist.*, 33(4):493–552, 2007.

[4] S. Cronen-Townsend and W. B. Croft. Quantifying query ambiguity. In *Proc. of HLT*, 2002.

[5] M. Daoud, L. Tamine-Lechani, and M. Boughanem. Learning user interests for a session-based personalized search. In *Proc. of Symposium on Information Interaction in Context*, 2008.

[6] M. Daoud, L. Tamine-Lechani, M. Boughanem, and B. Chebaro. A session based personalized search using an ontological user profile. In *Proc. of Symposium on Applied Computing*, 2009.

[7] Z. Dou, R. Song, and J.-R. Wen. A large-scale evaluation and analysis of personalized search strategies. In *Proc. of WWW*, 2007.

[8] S. Gauch, J. Chaffee, and A. Pretschner. Ontology-based personalized search and browsing. *Web Intelli. and Agent Sys.*, 1(3-4):219–234, 2003.

[9] 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. *Inf. Process. Manage.*, 36(2):207–227, 2000.

[10] K. Järvelin and J. Kekäläinen. IR Evaluation Methods for Retrieving Highly Relevant Documents. In *Proc. of SIGIR*, 2000.

[11] T. Joachims. Optimizing search engines using clickthrough data. In *Proc. of KDD*, 2002.

[12] 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. on Info. Sys. (TOIS)*, 25(2), April 2007.

[13] F. Liu, C. Yu, and W. Meng. Personalized web search by mapping user queries to categories. In *Proc. of CIKM*, 2002.

[14] F. Liu, C. Yu, and W. Meng. Personalized web search for improving retrieval effectiveness. *IEEE Trans. on Knowl. and Data Eng.*, 16(1):28–40, 2004.

[15] A. Pretschner and S. Gauch. Ontology based personalized search. In *Proc. of Int'l Conf on Tools with Artificial Intelligence*, 1999.

[16] F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *Proc. of WWW*, 2006.

[17] F. Radlinski and N. Craswell. Comparing the sensitivity of information retrieval metrics. In *Proc. of SIGIR*, 2010.

[18] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *Proc. of CIKM*, 2008.

[19] M. Sanderson. Ambiguous queries: test collections need more sense. In *Proc. of SIGIR*, 2008.

[20] X. Shen, B. Tan, and C. Zhai. Implicit user modeling for personalized search. In *Proc. of CIKM*, 2005.

[21] A. Sieg, B. Mobasher, and R. Burke. Web search personalization with ontological user profiles. In *Proc. of CIKM*, 2007.

[22] C. Silverstein, H. Marais, M. Henzinger, and M. Moricz. Analysis of a very large web search engine query log. *SIGIR Forum*, 33(1):6–12, 1999.

[23] M. Speretta and S. Gauch. Personalized search based on user search histories. In *Proc. of Int'l Conf on Web Intelligence*, 2005.

[24] S. Sriram, X. Shen, and C. Zhai. A session-based search engine. In *Proc. of SIGIR*, 2004.

[25] K. Sugiyama, K. Hatano, and M. Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *Proc. of WWW*, 2004.

[26] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen. CubeSVD: a novel approach to personalized web search. In *Proc. of WWW*, 2005.

[27] J. Teevan, S. Dumais, and E. Horvitz. Potential for personalization. *ACM Trans. Comput.-Hum. Interact.*, 17(1):31, March 2010.

[28] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proc. of SIGIR*, pages 449–456, 2005.

[29] J. Teevan, S. T. Dumais, and D. J. Liebling. To personalize or not to personalize: modeling queries with variation in user intent. In *Proc. of SIGIR*, 2008.

[30] P. Thomas and D. Hawking. Evaluation by comparing result sets in context. In *Proc. of CIKM*, 2006.

[31] A. T. A. Thuy Vu and M. Zhang. Term extraction through unithood and termhood unification. In *Proc. of Int'l Joint Conf on Natural Language Proc.*, 2008.

[32] R. W. White and S. M. Drucker. Investigating behavioral variability in web search. In *Proc. of WWW*, 2007.