Open Research Online

The Open University's repository of research publications
and other research outputs

Personalised Query Suggestion for Intranet Search with Temporal User Profiling

Conference or Workshop Item

For guidance on citations see FAQs.

Version: Accepted Manuscript

oro.open.ac.uk

engines cannot access and index those Intranet documents which require authorised logins. The searcher, therefore, may need to use an Intranet search engine to locate relevant documents.

Using collective user interaction data (e.g., query logs) for query suggestions has been shown useful for Intranet search [1, 2]. Existing Intranet search approaches appear to follow a "one size fits all" strategy. That is different users who submit the same query receive the same query suggestion list. However, different users may have different topics of interest. Consequently, the users who have submitted the same query may have different search intentions. For example, a sociology student submitting the query "lecture notes" is likely to be more interested in sociology classes than maths classes. Moreover, users' interests and search intentions may be dynamically evolving depending on their interactions with the system (e.g., clicks on documents), and when the interactions are made, during a search session [3].

To address these issues, we propose a unified framework to personalise query suggestions for Intranet search. Specifically, we use the interaction data of each user during a search session to build *user profiles*, which represent the user's topics of interests and may change over time in response to the user's interaction with the system.

It is worth noting that search personalisation (e.g., search result re-ranking, query suggestion, query auto-completion, etc.) has been studied extensively in the context of web search engines [3, 8, 9, 10, 11, 12, 13]. However, little attention has been paid to the same task for Intranet search. Moreover, personalisation methods on web search engines typically construct users' profiles using their click information [3, 10, 11, 12], but less account has been taken of how users modify their queries for building the profiles.

In our proposed framework, we construct two temporal topic-based user profiles for each search session. The first is a *click user profile* based on the clicked documents. The second is a *query user profile* based on the user's query modification history within the search session. We then use the two profiles within a learning-to-rank framework to re-rank suggested queries generated by a non-personalised method for query suggestion on Intranet search [1]. Experimental results show that our approach helps to significantly improve the query suggestion performance.

2. QUERY SUGGESTION FRAMEWORK

2.1 Building Temporal User Profiles

Each search session contains two types of event (i.e., queries and clicks). Given a search session, we propose to build two

---

Personalised Query Suggestion for Intranet Search with Temporal User Profiling

Thanh Vu[1], Alistair Willis[1], Udo Kruschwitz[2], and Dawei Song[1,3]
[1] The Open University, Milton Keynes, United Kingdom
[2] University of Essex, Essex, United Kingdom
[3] Tianjin University, Tianjin, P.R.China
{thanh.vu, alistair.willis, dawei.song}@open.ac.uk, udo@essex.ac.uk

ABSTRACT

Recent research has shown the usefulness of using collective user interaction data (e.g., query logs) to recommend query modification suggestions for Intranet search. However, most of the query suggestion approaches for Intranet search follow an "one size fits all" strategy, whereby different users who submit an identical query would get the same query suggestion list. This is problematic, as even with the same query, different users may have different topics of interest, which may change over time in response to the user's interaction with the system.

In this paper, we address the problem by proposing a personalised query suggestion framework for Intranet search. For each search session, we construct two temporal user profiles: a *click user profile* using the user's clicked documents and a *query user profile* using the user's submitted queries. We then use the two profiles to re-rank the non-personalised query suggestion list returned by a state-of-the-art query suggestion method for Intranet search. Experimental results on a large-scale query logs collection show that our personalised framework significantly improves the quality of suggested queries.

Keywords: Interactive IR; Intranet Search; Personalised Query Suggestion; Temporal User Profiles; Learning to Rank;

1. INTRODUCTION

Query suggestion is an important feature in web search engines (e.g., Bing, Google) as well as in domain-specific search engines (e.g., Intranet search) [1]. Query suggestions help users quickly refine the input query to better meet the user's information need by recommending possible terms to modify the original input query.

In this paper, we focus on query suggestion for Intranet search, which is different from web search [6]. Specifically, Intranet search (e.g., university intranets) is *domain-specific* and built to satisfy the user's information need related to a specific domain (e.g., the university's document corpus). Moreover, the Intranet may not be fully indexed and accessible by web search engines. For example, web search

temporal profiles for the specific user. These are a click user profile (denoted as $profile(C)$), built using the user's clicked documents, and a query user profile (denoted as $profile(Q)$), built using the user's submitted queries within the session. Click profile have been extensively used in other search personalisation methods [3]; we expect that the query profile will enrich the representation of the user's search interests.

Since a user's interests and search intentions may change over time, the more recently clicked documents and submitted queries could better represent the user's current interests. In this paper, we propose to use a *decay function* to capture this characteristic as in [3, 12].

2.1.1 Extracting Topics from Clicked Documents

We consider that click information (e.g., clicked documents) is a good indicator of those documents' relevance to the user's interests [7]. To build the user profile, we use the topics discussed in the documents. We first extract clicked documents from the Intranet search's query logs. After that, we employ Latent Dirichlet Allocation (LDA) [4] to automatically extract latent topics (denoted as Z) from the clicked documents (denoted as D). After training an LDA model using the clicked documents, we apply the model to extract topics for the remaining documents in the collection. Finally, each document is described as a multinomial distribution over the topics (denoted as $P(Z|D)$), in which each topic is represented as a multinomial distribution over the entire vocabulary.

2.1.2 Building a Click User Profile

We represent the temporal click user profile as a multinomial distribution over the topics as in [3]. Specifically, the user set is denoted as U. Let u be an instance of U. Let $D_c = \{d_{c_1}, d_{c_2}, ..., d_{c_n}\}$ be the set of clicked documents of the user u in the current search session, we define the click user profile of the user u (given the clicked document set D_c) as a distribution over topics Z (denoted as $P_C(Z|U)$). The probability $p_C(z|u)$ indicates how much the user u is interested in topic $z \in Z$. $p_C(z|u)$ is defined as a mixture of probabilities of z given $d_{c_i} \in D_c$ as follows

$$p_C(z|u) = \sum_{d_{c_i} \in D_c} \lambda_i p(z|d_{c_i}) \tag{1}$$

$\lambda_i = \frac{1}{N}\alpha^{t_{d_{c_i}}-1}$ is the exponential decay function of $t_{d_{c_i}}$, which is the order of the document d_{c_i} clicked by the user u in the search session. $t_{d_{c_i}} = 1$ indicates that d_{c_i} is the most recently clicked document; N is the normalisation factor. α is the decay parameter ($0 \leq \alpha \leq 1$).

2.1.3 Building a Query User Profile

Let $Q = \{q_1, q_2,, q_m\}$ be the submitted query set of u in the search session. Because the number of Intranet documents is smaller and can be assumed to change less frequently than web search engines', we make the simplifying assumption of describing each query by the set of documents that contain all the query words, denoted as $D_{q_i} = \{d_{i_1}, d_{i_2}, ..., d_{i_k}\}$. Then, each search query q_i (given the document set D_{q_i}) is modelled as a distribution over topics Z (denoted as $P(Z|q_i)$). The probability of a topic $z \in Z$ given $q_i \in Q$ (i.e., $p(z|q_i)$) is defined as a mixture of probabilities of z given a document $d_{i_j} \in D_{q_i}$ as follows

$$p(z|q_i) = \sum_{d_{i_j} \in D_{q_i}} \frac{1}{|D_{q_i}|} p(z|d_{i_j}) \tag{2}$$

$|D_{q_i}|$ is the size of the document set D_{q_i}.

We then model the query user profile of the user u (given the query set Q) as a distribution over topics Z (denoted

as $P_Q(Z|u)$). The probability of a topic z given u (i.e., $p_Q(z|u)$) is defined as a mixture of probabilities of z given query $q_i \in Q$ as follows

$$p_Q(z|u) = \sum_{q_i \in Q} \lambda_i p(z|q_i) \tag{3}$$

$p(z|q_i)$ is defined in Equation 2. Similar to the click user profile, $\lambda_i = \frac{1}{M}\alpha^{t_{q_i}-1}$ is the exponential decay function of t_{q_i}, which is the order of the query q_i submitted by the user u in the search session. $t_{q_i} = 1$ indicates that q_i is the most recent query; M is the normalisation factor. α is the decay parameter ($0 \leq \alpha \leq 1$).

2.2 Re-ranking Suggested Queries

We use the two user profiles in a learning-to-rank mechanism to re-rank the query suggestion list returned by a non-personalised query suggestion method proposed by Adeyanju *et al.* [1], denoted as *Adeyanju's*. Specifically, *Adeyanju's* first constructs a domain knowledge structure in the form of a concept subsumption hierarchy using both the Intranet document collection and collective users' query logs. Next, the suggestion list is generated using the top n terms most relevant to the query in the hierarchy.

For each input query, our re-ranking method is detailed as follows

(1) We generate the top n ranked suggested queries using *Adeyanju's* method. We denote a suggested query as q_s.

(2) We then compute similarity scores between q_s and $profile(C)$, and between q_s and $profile(Q)$. Both the suggested query q_s and a user profile (denoted as pf which is either $profile(C)$ or $profile(Q)$) are modelled as distributions over topics Z (Section 2.1). To measure the similarity between q_s and the user profile pf, we use Jensen-Shannon divergence ($D_{JS}\lfloor.||.\rfloor$), which is a popular method of measuring the divergence (similarity) between two distributions, to measure the similarity between q_s and pf

$$Sim(q_s|pf) = -D_{JS}\lfloor Q||P\rfloor = -\left(\frac{1}{2}D_{KL}\lfloor Q||M\rfloor + \frac{1}{2}D_{KL}\lfloor P||M\rfloor\right) \tag{4}$$

Here, Q and P are distributions over topics of q_s and pf, respectively. $D_{KL}\lfloor.||.\rfloor$ is the Kullback-Leibler divergence and $M = \frac{1}{2}(Q + P)$. We consider the scores as the *personalised features*. We also extract other *non-personalised features* of the input query q and the suggested query q_s. Table 1 shows the features extracted for re-ranking the suggestion list.

Table 1: The personalised query suggestion features

Feature	Description
Personalised Features	
ClickPersonalisedScore	The similarity score between the suggested query and the user click profile
QueryPersonalisedScore	The similarity score between the suggested query and the user query profile
Non-personalised Features	
QueryRank	Rank of the suggested query on the original list
QuerySim	The cosine similarity score between the current query and the previous query
QueryNo	Total number of queries that have been submitted to the Search Engine
SuggestedQueryCosine	The cosine similarity score between the current query and the suggested query
SuggestedQueryJaccard	The Jaccard distance score between the current query and the suggested query
SuggestedQueryEdit	The edit distance between the current query and the suggested query
SuggestedQueryLevenshtein	The Levenshtein distance between the current query and the suggested query
SuggestedQueryPreUsed	Whether the suggested query was used by the user in the same search session?

(3) After extracting the query features, to re-rank the top n suggested queries, we employ LambdaMART [5] to train ranking models. Among many learning-to-rank algorithms, LambdaMART is regarded as one of the best-performing algorithms and has been chosen as the base learning algorithm in various recent approaches to search personalisation [3].

3. EXPERIMENTAL METHODOLOGY

3.1 Evaluation Methodology and Dataset

Evaluation methodology For evaluation, we use *Auto-Eval*, an automated evaluation framework, which measures the performance of query suggestions automatically based on the actual query logs of an Intranet search [2]. For each query *suggestion list*, we assign a positive label for a suggestion if it is an actual *refinement*, which is the next submitted query in the search session, and there is at least one user click on retrieved results after the refinement. In other words, we interpret the user click after a reformulation as the criterion of a relevant suggestion. The remainder of the suggestion list is assigned negative (irrelevant) labels. We use the rank positions of the positively labelled queries as an approximation of the ground truth to evaluate the performance of query suggestions before and after re-ranking.

We also follow the experimental methodology in [1], that is, the model is evaluated continuously at periodic intervals. Specifically, we use the logs in week i for training the re-ranking model and the following week $i + 1$ for testing the trained model; where, in our experiments, $1 \leq i \leq w$, the number of weeks in the test period.

Dataset The dataset used in our experiments contains large-scale query logs collected from the search engine installed at the Web site of the University of Essex during the two years covering 1 January 2012 - 31 December 2013. Each log sample contains a session identifier, the event type (i.e., a query or a click), an auto-increment id, the event content (i.e., query text, click URL), and the event time-stamp.

We apply a simple pre-processing step to remove single event search sessions because, with those sessions, it is not possible to determine whether the user found the required information. We also remove those queries whose positive label set is empty from the dataset. We then analyse the remainder of the query logs. Table 2 shows basic statistics.

Table 2: Basic statistics of the evaluation search logs

Item	2012	2013	Total
#search sessions	397,461	338,391	735,804
#events	1,263,179	1,083,992	2,347,171
#events/session	**3.22**	**3.25**	**3.23**
#queries	757,645	659,284	1,416,929
#query/session	**1.91**	**1.95**	**1.93**
#clicked url	505,534	424,708	930,242
#clicks/session	**1.27**	**1.26**	**1.26**

3.2 Experimental Settings

Our personalisation method and baselines We name our proposed re-ranking model as *Ours*. We choose two baselines to compare our work against. The *first baseline* is *Adeyanju's* method [1], which we reimplemented to generate the original suggestion list for *re-ranking*.

We use the session-based approach proposed by Bennett *et al.* [3] as the *second baseline*. Specifically, in the baseline, we use only the click user profile (i.e., $profile(C)$) together with the non-personalised features detailed in Table 1 to re-rank the suggestion list. We name the baseline as *Click*.

It is worth noting that *Adeyanju's* is a non-personalised approach and achieved a good performance of query suggestion on Intranet search [1]. *Click* is a personalised approach and achieved good performances in web search personalisation [3]. Moreover, instead of the session-based approach by Bennett *et al.* [3] as our personalised baseline we could alternatively have used Shokouhi *et al.* [9] or Shokouhi [8].

LDA & LambdaMART We train an LDA model on the clicked documents extracted from the query logs, as de-tailed in Section 2.1. The number of topics (i.e., 300 in our experiments) is decided by using a held-out validation set which consists of 10% of all clicked documents. The selected number of topics is the one that gives the lowest perplexity value. The decay parameter α for the two user profiles is set to 0.95 as in [3]. The ranking function is learned using LambdaMART [5]. We used the default setting for LambdaMART's prior parameters[1].

Evaluation metrics The evaluation is based on the comparison between our personalised approach and the baselines. We use four evaluation metrics: Mean Average Precision (MAP), Precision (P@k), Mean Reciprocal Rank (MMR), and Normalized Discounted Cumulative Gain (nDCG@k).

4. RESULTS

4.1 Overall Performance

Table 3 shows promising results when user profiles are used to personalise the query suggestion list. We can see that, even using only the click user profile, the *Click* method has led to an improvement of 7.22% on MAP over *Adeyanju's* method. The combination of query and click profiles (i.e., *Ours*) achieves the highest improvement of 10.97% over *Adeyanju's* in terms of MAP score. The improvements indicate that personalisation helps improve the query suggestion performance. The improvements over *Adeyanju's* are all significant (*paired t-test*, p < 0.01).

Table 3: Overall performance of the methods. %rel denotes the relative improvement over *Adeyanju's*.

Model	MAP	P@1	P@5	MRR@10	nDCG@5	nDCG@10
Adeyanju's	0.5440	0.4113	0.1823	0.5447	0.5714	0.6000
Click	0.5833	0.4271	0.1981	0.5839	0.6193	0.6583
%rel	*+7.22%*	*+3.84%*	*+8.67%*	*+7.22%*	*+8.38%*	*+9.73%*
Ours	**0.6037**	**0.4526**	**0.2026**	**0.6043**	**0.6413**	**0.678**
%rel	*+10.97%*	*+10.04%*	*+11.14%*	*+10.94%*	*+12.23%*	*+13.02%*

In the comparison between personalisation methods (i.e., *Ours* and *Click*), Table 3 shows that using both the click and query user profiles (i.e., *Ours*) significantly improves the suggestion quality over the *Click* baseline ($p < 0.01$). Interestingly, our method produces a significantly better quality of the first query in the suggestion list with the improvement of 5.97% on P@1 over *Click*. The improvements of *Ours* over *Click* also indicate that the query user profile is important in the query suggestion task, especially for the quality of the first suggested query.

4.2 Performance on Different Query Positions

With more submitted queries and clicked documents, we are able to build richer user profiles. In this experiment, we aim to study whether the position of a query in a search session has any effect on the performance of personalised query suggestion. For each search session, we label queries by their positions during the session. Because there are few sessions containing more than three queries (i.e., 7.65% of sessions in the query logs), we label the first three queries from one to three according to the order of submission in the search session; the remaining queries are labelled as ≥ 4.

It is worth noting that for the first query, we cannot build the click user profile because there is no previously clicked document. However, we can still build the query user profile for the first query. We show the improvement in performance of the personalised methods over *Adeyanju's* in term of MAP

---

[1]Number of leaves = 10, minimum documents per leaf = 200, number of trees = 100 and learning rate = 0.15

Figure 1: Relative MAP improvements over *Adeyanju's* with different query positions.

Figure 2: Relative MAP improvements over *Adeyanju's* with different query lengths.

metric with different query positions in Figure 1. Here the statistical significance is verified by t-test ($p < 0.01$). For the first query within a search session, our method, which can use only the query user profile, significantly improves the query suggestion performance over *Adeyanju's*. It again confirms the effectiveness of the query information on personalised query suggestion for Intranet search.

From the second query, we can build both the query and click user profiles. One can see that the higher position of a query is, the larger improvement in performance the personalised query suggestion can be achieved. Specifically, from the query with high positions (i.e., ≥ 4), the improvements of *Click* and *Ours* are 11.37% and 18.22%, respectively. Figure 1 also shows that *Ours* outperforms *Click* significantly with the improvements of at least 3.45% ($p < 0.01$). It indicates that richer user profiles (by observing more clicked documents and submitted queries during the search session) help achieve better query suggestion performances. The findings offer future research directions that use user profiles which go beyond single sessions

4.3 Performance on Different Query Lengths

The query length is defined by the number of words in the query (e.g., the query "University webmail" has the length of 2). The length of a query might give an indication as to how specific the information need of an individual user is (i.e., a longer query can typically be assumed to reflect a more specific information need). In this experiment, we aim to show the impact of personalisation on query suggestion with different query lengths. We label each query by its length, which is the number of words in the query (i.e., from one to three and ≥ 4 words because there are few queries containing more than three words (i.e., 5.7% of queries in the query logs)).

Figure 2 shows the improvement in performance of *Click* and *Ours* over *Adeyanju's* in term of MAP with different query lengths. We see that personalisation methods achieve significantly better performances than the non-personalised method does ($p < 0.01$). Even for short queries (length 1 and 2) which tend to be more generic, the *Click* and *Ours* methods outperform *Adeyanju's* method with the improvement of more than 6.11% and 9.09%, respectively. We see that the longer a query is, the higher improvement personalised methods can achieve. Specifically, with a longer query (i.e., with length ≥ 4), the *Click* and *Ours* methods yield the highest improvements, i.e., 29.82% and 53.85%, respectively. This indicates that a longer query would also get more benefit from personalisation.

Figure 2 also indicates that by combining the query user profile with the click user profile, *Ours* significantly improves the query suggestion performance over *Click* ($p < 0.01$). Moreover, the improvements are larger with the longer queries (i.e., with length ≥ 3). In particular, the improvements of
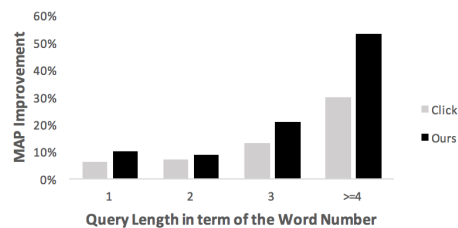
Ours over *Click* are 6.7% and 18.2% on the query length 3 and the query length ≥ 4, respectively.

5. CONCLUSIONS

In this paper, we proposed a personalised query suggestion framework and showed how it performed on Intranet search. We built two session-specific temporal user profiles, a query user profile using the submitted queries, and a click user profile using the clicked documents. We then extracted the personalised features using the two profiles and combined them with non-personalised features to learn a ranking model using LamdaMART. Finally, we used the ranking model to re-rank the query suggestion list returned by a state-of-the-art query suggestion approach for Intranet search.

Experimental results on a large-scale query log dataset collected from a university intranet search engine show that personalisation significantly improved the query suggestion performance. Using both the click user profile and query user profile achieved the highest performance indicating that personalised query suggestion for Intranet search should take into account both click and query information. Moreover, the positive impact of personalised query suggestions is more pronounced with longer queries and queries submitted later within a session.

6. REFERENCES

[1] I. A. Adeyanju et al. Adaptation of the concept hierarchy model with search logs for query recommendation on intranets. In *SIGIR'12*, 2012.

[2] M.-D. Albakour et al. Autoeval: An evaluation methodology for evaluating query suggestions using query logs. In *ECIR'11*, 2011.

[3] P. N. Bennett et al. Modeling the impact of short- and long-term behavior on search personalization. In *SIGIR'12*, 2012.

[4] D. M. Blei et al. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 2003.

[5] C. J. C. Burges et al. Learning to rank with non-smooth cost functions. In *NIPS'06*, 2006.

[6] D. Hawking. Enterprise search. In *Modern Information Retrieval, 2nd Ed.* 2010.

[7] T. Joachims et al. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR'05*, 2005.

[8] M. Shokouhi. Learning to personalize query auto-completion. In *SIGIR'13*, 2013.

[9] M. Shokouhi et al. Fighting search engine amnesia: Reranking repeated results. In *SIGIR'13*, 2013.

[10] T. Vu et al. Improving search personalisation with dynamic group formation. In *SIGIR'14*, 2014.

[11] T. Vu et al. Modelling time-aware search tasks for search personalisation. In *WWW'15 Companion*, 2015.

[12] T. Vu et al. Temporal latent topic user profiles for search personalisation. In *ECIR'15*, 2015.

[13] T. Vu et al. Search personalization with embeddings. In *ECIR'17*, 2017.