Personalizing Web Search Results Based on Subspace Projection

Jingfei Li[1], Dawei Song[1,2], Peng Zhang[1], Ji-Rong Wen[3], and Zhicheng Dou[4]

[1] School of Computer Sci & Tec, Tianjin University, Tianjin, China
[2] The Computing Department, The Open University, UK
[3] School of Information, Renmin University of China, Beijing, China
[4] Microsoft Research Asia
{dawei.song2010,darcyzzj,jirong.wen}@gmail.com
jingfl@foxmail.com,zhichdou@microsoft.com

Abstract. Personalized search has recently attracted increasing attention. This paper focuses on utilizing click-through data to personalize the web search results, from a novel perspective based on subspace projection. Specifically, we represent a user profile as a vector subspace spanned by a basis generated from a word-correlation matrix, which is able to capture the dependencies between words in the "satisfied click" (SAT Click) documents. A personalized score for each document in the original result list returned by a search engine is computed by projecting the document (represented as a vector or another word-correlation subspace) onto the user profile subspace. The personalized scores are then used to re-rank the documents through the Borda' ranking fusion method. Empirical evaluation is carried out on a real user log data set collected from a prominent search engine (Bing). Experimental results demonstrate the effectiveness of our methods, especially for the queries with high click entropy.

Keywords: Personalization, User Profile, Subspace Projection

1 Introduction

Over decades, modern search engines have transformed the way people access and interact with information. Users can easily search for relevant information by issuing simple queries to search engines. Despite the increasing popularity and convenience, search engines are facing some challenges. For example, given a query, a typical search engine usually returns a long list of URLs, usually displayed in a number of pages. We call the results list as *Original List*. However, the top ranked URLs may not always satisfy users' information needs well. Users may have to scroll down the current result page and even turn to the following pages to find desired information. This would affect the users' search experience and satisfaction. One way to tackle this problem is through search personalization based on an individual user's profile representing the user's personal preferences and interests.

Personalized search has recently attracted much interest. Many personalized search strategies [5][6][8][10][13] build on the users' click-through data, where it is assumed that the clicked URLs are relevant [8]. This assumption is not rigorous, because users often go back quickly after clicking an irrelevant result. Previous research indicated that the clicks with short dwell time ("quick backs") are unlikely to be relevant[10]. In this paper, we utilize the "SAT click" criteria [10] ([i] the user dwelled on the result page corresponding to the clicked URL for at least 30 seconds; [ii] the click was the last click in current query session) to judge the relevance of a clicked document. Only the "satisfied" click data (URLs and corresponding documents) are used to build a user profile. The "SAT click" data is also used as the ground truth when evaluating the proposed algorithms.

The classic Vector Space Model (VSM) has been a popular choice for user profile representation [21], in which the queries, documents and user profiles are all represented as vectors in a term space [12][21]. Generally, the weight of each term (or keyword) in a user profile vector is calculated by its $TF \times IDF$ weight. However, representing user profiles as weighted keyword vectors has several inherent limitations. As the number of keywords increases, the vector representation becomes ambiguous. Moreover, the traditional bag of words models in IR, such as VSM and unigram language model, are based on the *term independence assumption*. This assumption simplifies the development and implementation of retrieval models, but ignores the fact that some words are dependent on each other. Intuitively, two co-occurring words can convey more semantic information than the single words individually. For example, when "Obama" and "Romney" co-occur in a document, we may easily recognize that this document is about the American Election, but if we only observe one single word "Obama" or "Romney", the the topic of this document can be different. To address this issue, term dependencies need to be mined and incorporated into IR models to improve retrieval performance [1][9][21]. In this paper, we propose to represent a user profile as a vector subspace, and make use of term dependence information, in the form of a word-correlation matrix, to generate the user profile subspace.

Our method is inspired by the idea of using a vector space basis for modeling context, originally proposed by Melucci [15] , where each basis vector refers to a contextual property. In linear algebra, a vector can be generated by a basis. In this way, an information object (e.g., an information need) represented by a vector can be generated by the context modeled with the basis. Melucci [1] computed the probability that an information object has been materialized within a context. In this paper, we extend the idea to user profile representation. We systematically investigate and evaluate two novel algorithms based on the subspace projection for personalized re-ranking of Web search results. Specifically, we represent a user profile as a subspace spanned by a basis derived from a word-correlation matrix built from the user's SAT clicked web pages. The personalized score for a document can be computed by projecting the document (represented as a vector in the first algorithm or another word-correlation subspace in the second algorithm) onto the user profile subspace. Then we re-rank the original list returned by a prominent search engine (Bing) based on the Borda' rank fusion method [14].

2 Related Work

In this section, we briefly review the related work on two areas, including personalized information retrieval and the geometry underlying IR.

Personalized search aims to provide customized search results according to an individual user's interests. Various personalized search methods have been proposed in recent years [4][5][10]. They are based on either explicit relevance feedback or implicit feedback through various user interaction behaviors, such as clicks, scrollings, adding pages to favorites, and so on. For example, Bennett et al. [4] utilized the position information of users to influence search results. Sontag et al. [5] proposed a generative model to predict the relevance of a document for a specific user. Collins-Thompson et al. [20] took the reading level of the users into considerations to improve the effectiveness of retrieval. Xiang et al.[7] integrated various context information generated by user interaction into the learning to rank model to improve the IR performance. In [2], several personalization strategies were proposed. It came to a conclusion that personalized search can lead to a significant improvement on some queries but has little effect on other queries (e.g., queries with low click entropy).

In a seminal book about the geometry of IR [17], Hilbert's vector spaces were used to represent documents. Similarly, Melucci [15] proposed an idea of using a basis to model the context in IR. In [16], a geometric framework is proposed to utilize multiple sources of evidence presented in current interaction context (e.g., display time, document retention) to develop enhanced implicit feedback models personalized for each user and tailored for each search task. The models we develop in this paper are inspired by the subspace projection method investigated in [1], which, in our opinion, is a general and principled theoretical framework for incorporating word dependencies and provides a unified representation for both user profiles and documents.

3 The Subspace Theoretical Framework

3.1 Probability of a Vector Out of Subspace

Suppose $B = \{b_1, ..., b_k\}$ is a basis of a k-dimension subspace defined over R^n, where $b_i^T \cdot b_i = 1$ and b_i's are mutually orthogonal. $L(B)$ is the subspace spanned by B. $\mathbf{x}$ is a vector, and $L(\{\mathbf{x}\})$ is the set of vectors of the form $c\mathbf{x}$, where c is a scalar. The vector $\mathbf{x}$ may or may not be generated by B. If $\mathbf{x}$ is generated by B, there exists a set of weights $\{p_1, .., p_k\}$ such that $\mathbf{x} = p_1 b_1 + ... + p_k b_k$. Note that every vector generated by B is entirely contained in $L(B)$. The vectors that cannot be generated by B is not contained in $L(B)$, but these vectors may be more or less close to $L(B)$. Intuitively, if a vector $\mathbf{x}$ is close to $L(B)$, the information object (e.g., a document) represented by $\mathbf{x}$ is likely within the context spanned by B. Similarly, the information object represented by a vector being far from $L(B)$ is unlikely to be generated by the context spanned by B. Based on the notions just illustrated, we can model a user profile as a basis and documents as vectors, then compute the inner product between a document vector and the projection of the vector onto the user profile subspace as the probability that the corresponding user is interested in the document, which formalized as Equation 1.

$$Pr[L(B)|L(\{\mathbf{x}\})] = x^{\mathrm{T}} \cdot P_B \cdot x \tag{1}$$

where we restrict $x^{\mathrm{T}} \cdot x = 1$, P_B is the projector to $L(B)$, namely $P_B = B^{\mathrm{T}} \cdot B$, and $P_B \cdot x$ is the projection of x onto the subspace B. Each basis vector in the subspace can be considered as a concept of a user profile. The projection of a document vector onto the subspace can then be interpreted as the concept of user profile which is most related to the document. This formula is different from the traditional VSM modeling of user profile as a single vector, which may contain more irrelevant noises for current search topic.

3.2 Probability of a Subspace Out of another Subspace

In above theoretical framework, a document is represented as a normalized vector, which assumes that a document only contain one topic (corresponding to one basis vector). However the fact is that one document may have multiple topics, e.g., one document introduces both the beautiful scenery and the notability of one place. To address this gap, we extend the projection-based method in Section 3.1 to the projection from one subspace to another subspace. In the extension, we represent a document as a subspace instead of a vector, denoted as $L(O)$ spanned by a basis $O = \{x_1, ..., x_m\}$, where each dimension corresponds to a concept of the document. We then compute the probability of $L(O)$ out of $L(B)$ according to Luders's rule[1]:

$$Pr[L(B)|L(O)] = \frac{tr(P_O \cdot D \cdot P_O \cdot P_B)}{tr(D \cdot P_O)} \tag{2}$$

where P_B, P_O are the projectors to L_B, L_O respectively, $tr(\cdot)$ is the trace of a matrix, and D is a density matrix (symmetric, positive definite and has trace one). In our work, the word correlation matrix built for user profile is regarded as the density matrix.

4 Personalized Web Search Re-ranking Algorithms

We now present our concrete algorithms that implement the subspace projection based theoretical framework described in the previous section. In this paper, a "query instance" refers to an information object that contains the user ID, query terms, query time stamp, original result list, clicked URL list, and so on. It is worth noting that different query instances may contain the same query terms. A user's search process is captured by the user's "query trace", which is a sequence of query instances sorted by query time stamp. A query trace is formalized as $q_1, ..., q_i, ..., q_{n-1}, q_n$, where $q_1, ..., q_i, ..., q_{n-1}$ are the historical query instances, and q_n is the current query instance (for which the original search results are to be re-ranked). For each q_i, we download the actual documents of the top 30 returned URLs from Bing search engine as the *original list.* When re-ranking the original list of q_n, we compute the personalized score for each URL using our personalized algorithms and obtain the *personalized ranked list* according to the personalized scores. After that, a *re-ranked list* is gained by combining the original list with personalized ranked list using the Borda' ranking fusion method [14].

4.1 Algorithm 1: Document Vector Projection onto User Profile Subspace Algorithm (V-S)

In this algorithm, we represent a user profile as a subspace that consists of the top K eigenvectors (corresponding to the top K eigenvalues) of a $N \times N$ word-correlation matrix, where N is the size of the vocabulary. Each eigenvector depicts a distribution of words corresponding to a concept of user's search interests. The top K eigenvectors constitute a basis of user profile subspace corresponding to the main aspects of user's search history. A document is represented as a N dimensional column vectors and can be generated from the user profile subspace with a probability. For example, if a document can be totally generated from this subspace, the probability is 1; conversely, the probability is 0. We can rank documents based on such probabilities to get a personalized ranked list for the current query instance.

Step1: Building the Word-Correlation Matrix for User Profile In this work, we build a document collection for each user, which are composed of all of the historical SAT clicked web pages. We preprocess each web page by segmenting it into a list of sentences. In this way, the document collection can be processed into a set of sentences, denoted as $S = \{s_k\}, k = 1, ..., M$. The word-correlation matrix for the user is built based on the sentence set. We denote the user profile matrix as M^P, each element of which is defined as:

$$M^P_{ij} = r(w_i, w_j) \times TFIDF(w_i) \times TFIDF(w_j) \tag{3}$$

where M^P_{ij} is an element of M^P corresponding to the i_{th} row and the j_{th} column. w_i and w_j are two words, and $r(w_i, w_j)$ reflects the correlation between them. Equation 3 aims to not only capture the dependency relationship between words, but also reflect the importance of each word. $TFIDF(w_i)$ is the product of term frequency (TF) of w_i in the sentence set and its inverse document frequency (IDF) in a global document collection with 273298 web pages (not the user profile document collection). The mutual information (See Equation 4) between two variables is used to define the correlation between two words. Indeed, any other correlation measures can be applied here. A more systematic study of different correlation measurements will be carried out as future work.

$$r(w_i, w_j) = \begin{cases} I(X;Y), & \text{if } i \neq j \\ H(X), & \text{if } i = j \end{cases} \tag{4}$$

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) log \frac{P(x,y)}{P(x)P(y)}; H(X) = -\sum_{x \in X} P(x) log P(x) \tag{5}$$

where X and Y are two random variables which indicate the existence of w_i and w_j respectively. $P(x) = P\{X = x\}$, $P(y) = P\{Y = y\}$, $P(x,y) = P\{X = x, Y =$

$y\}, x, y \in \{0, 1\}$. Here, $P\{X = 1\}$ is the probability that w_i occurs in the sentence set, and $P\{X = 0\}$ is the probability that w_i does not occur in the sentence set. $P\{Y = 0\}$ and $P\{Y = 1\}$ have similar definition corresponding to w_j. $P(x, y)$ is the joint probability of X and Y. Note that, the Dirichlet smoothing method[19] has been used while estimating the word probability to avoid zero probability. $I(X;Y)$ is the mutual information of X and Y, which indicates the dependency relationship of the words. The diagonal elements of the matrix contains a factor, the self-information of X, which indicates the amount of information of X.

Step2: Computing the Personalized Score for each URL The words-correlation matrix built in Step 1 is a $N \times N$ symmetric matrix. We can decompose it through the Singular Value Decomposition (SVD)[11] and get the top K eigenvectors as the basis $B_P = (v_1, ..., v_i, ..., v_K)_P$, where v_i is the i_{th} eigenvector of the user profile matrix corresponding to the i_{th} eigenvalue of the all eigenvalues in a descending order. Then the projector for the user profile (P_P) can be gained by the product of corresponding basis and its transposition, i.e., $P_P = B_P \cdot B_P^{\mathrm{T}}$. In order to obtain the personalized score for each URL, we represent each document (URL) as a N dimensional vector (V^d) based on the vocabulary. We utilize the $TF \times IDF$ (denoted as *TFIDF* here), after normalization, as the weight of each element in the document vector.

$$V_d = \left(\frac{TFIDF_1}{\sqrt{\sum_i TFIDF_i^2}}, ..., \frac{TFIDF_i}{\sqrt{\sum_i TFIDF_i^2}}, ..., \frac{TFIDF_N}{\sqrt{\sum_i TFIDF_i^2}}\right)^{\mathrm{T}} \quad (6)$$

The personalized score of each URL can be obtained naturally by projecting the document vector onto the user profile subspace (also see Equation 1):

$$PScore(u) = V_d^{\mathrm{T}} \cdot P_P \cdot V_d \quad (7)$$

where $PScore(u)$ is the personalized score for a URL. After this step, we can get the personalized rank list according to $PScore(u)$.

Step 3: Re-Ranking the Query Instance Since we cannot get the actual relevance score from the Bing search engine, we use the rank-based fusion method for re-ranking the original result list with the personalized ranked list. We denote the original result list of a query instance as τ_1 and the personalized ranked list gained in step 2 as τ_2. Then we combine the rankings in τ_1 and τ_2 using the Borda' ranking fusion method and sort the web pages with the combined rankings. Let u be one URL of the original result list of one query instance. Borda's method first assigns a score $B_i(u)$ = "the number of the URLs ranked below u in the rank τ_i", and then the total Borda' score $B(u)$ is defined as $\sum_{i=1}^{2} B_i(u)$ [14]. Finally, we re-rank the result list according to the Borda score $B(u)$ to get a re-ranked list τ. It should be noted that the different URLs may have the same Borda' score in the actual experiment, which may lead to the uncertainty of ranking in the re-ranked list. To avoid this problem, we sort the URLs according to the relative order in τ_1 when the same Borda score occurs.

4.2 Algorithm 2: Document Subspace Projection onto User Profile Subspace (S-S)

This algorithm shares the same framework with the first algorithm described in Section 4.1, while the only difference is the method used for obtaining the personalized score for a URL. For this reason, we leave out the common parts of the two algorithm, and focus on how to get the personalized score. In this algorithm, the user profile subspace construction is the same as in Algorithm 1. Each document is also represented as a words-correlation matrix in the same way to build a user profile matrix. The document matrix is decomposed through SVD, so that the basis of the document subspace (B_d) is generated (also the selection of top K eigenvectors as the basis). From the basis, we get the projector for the document subspace $P_d = B_d \cdot B_d^{\mathrm{T}}$. The personalized score is derived by projecting the document subspace onto the user profile subspace as introduced in Section 3:

$$PScore(u) = tr(P_d \cdot M^P \cdot P_d \cdot P_P)/tr(D \cdot P_d) \tag{8}$$

where M^P is the word-correlation matrix for user profile as a density matrix, P_P is the projector corresponding to the user profile subspace. The advantages of this algorithm compared with the first algorithm are that (i) the correlation between words is taken into account in the document representation; (ii) key concepts of a document with multiple topics are captured through SVD and considered in the document representation.

5 Empirical Evaluation

5.1 Baseline: Vector Space Model (VSM)

In this paper we set the VSM as a baseline algorithm to compare with our algorithms described above. In SVM, both user profile and documents are represented as vectors. The personalized score for a URL is the cosine similarity between the user profile vector V_P and the document vector V_d constructed with the same method as described in the first algorithm..

$$PScore(u) = \frac{V_P \cdot V_d}{\sqrt{V_P^{\mathrm{T}} \cdot V_P} \times \sqrt{V_d^{\mathrm{T}} \cdot V_d}} \tag{9}$$

5.2 Experiment Settings

To test our personalized re-ranking algorithms, we conduct experiments on a real query log collection. In the experiments, we randomly sampled 107 users' query logs as the training and testing data from a global query log with 1166 users over a certain period of time. Table 1 shows the detailed information of the global query log and sampled query log, which indicates that they have some similar statistical properties. In addition, we store a global document collection with 273,298 web pages downloaded from the Internet based on the URLs in the selected query log. We have preprocessed the web pages by extracting the content data, segmenting them into sentences, removing stop words and stemming the words with Porter Stemmer [18].

We build a vocabulary for each user by selecting the top N words(we set $N = 1000$ in this paper) according to their TF-IDF weights in the SAT clicked document collection in the user's search history. The vocabulary is updated dynamically as user issuing new queries into the search engine. In the representation of user profile, K, the number of selected eigenvectors in the basis of the user profile subspace, is an important parameter which determines the number of the topics in the user profile used to personalize the web search results of the current query. We conduct systematic experiments to test the influence of different K on the algorithms' performance.

Table 1. Detailed information about the global query log and the selected query logs.

Items	#users	#query	#distinct query	#Clicks	#SATClicks	#AverageActiveDays
Global Logs	1,166	541,065	221,165	475,624	357,279	20.963
Selected Logs	107	55,486	25,618	54,766	36,761	20.444

The Click Entropy is a concept proposed in [2], which is a direct indication of query click variation. It is computed based on all of the clicks for a distinct query (i.e., which is unique in the query log).

$$ClickEntropy(q) = \sum_{u \in U(q)} -P(u|q) \log_2 P(u|q) \tag{10}$$

where $U(q)$ is the collection of URLs that are clicked for the distinct query q, and $P(u|q)$ is the percentage of the clicks on the URL u among all the clicks for q. Dou et al.[2] pointed out that the smaller click entropy means that the majorities of users agree with each other on a small number of web pages for a query. It has been shown in the Literatures[2][3] that personalized search algorithms have different performance on query instances with different click entropies: generally speaking, the queries with low click entropy tend to have a less potential to benefit from personalization [2]. In this paper, we report the distribution of our experimental results over various different click entropy ranges. Note that, for statistical significance, we compute the click entropy for each distinct query based on a large scale global query log (see Table 1).

In the real scenario of searching, users may skip the first SERP (Search Engine Results Page) and turn to following pages. Intuitively, this phenomenon indicates that a user may dissatisfy the search results returned by the search engine. From this view, the percentage of turning pages (= number of query instances that users turn to next pages / total number of queries) can reflect the users' satisfaction with the search results to some extent. The larger the percentage is, the less the user's satisfaction tend to be. Fig.1 (A) shows that users have relatively high satisfaction with the search results for queries with lower click entropy and there is less need of re-ranking results for these queries. With this consideration, we focus on re-ranking the query instances with *relatively high* click entropy. This is a typical long tail task. Fig.1 (B) shows the distribution of the numbers queries over different click entropies for the test data.

5.3 Evaluation Metrics

We utilize the evaluation metric introduced in Dou et al. [2] to evaluate the quality of a ranked URLs list for query instances. It is called *Rank Scoring*,

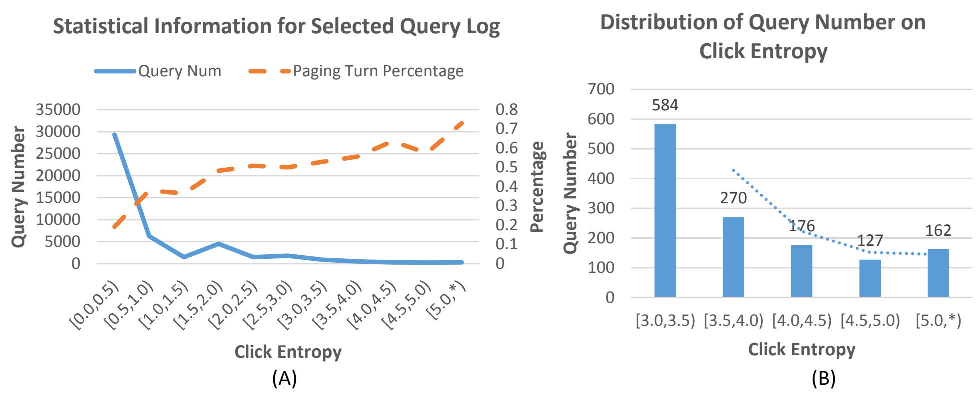


Fig. 1. (A) is the statistical information for selected query log; (B) is the histogram of query number distribution on click entropy for testing data set.

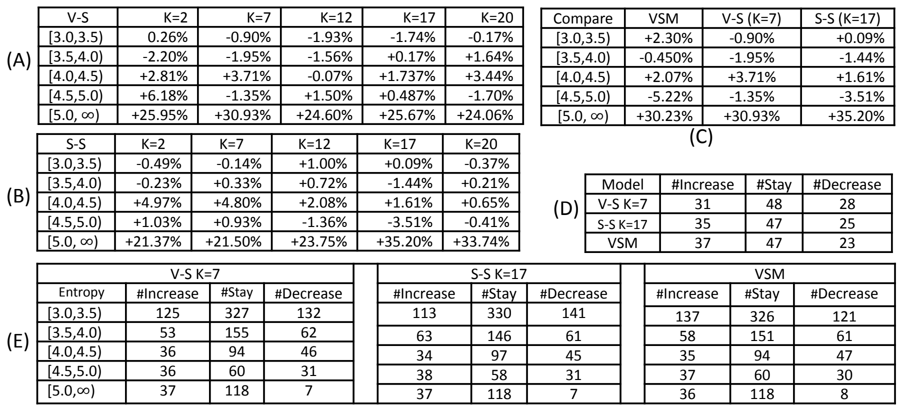
denoted as R_q for a query instance q. The average rank scoring for a set of query instances is denoted as $R_{average}$.

$$R_q = \sum_j \frac{\delta(q,j)}{2^{(j-1)/(\alpha-1)}}; R_{average} = 100\frac{\sum_q R_q}{\sum_q R_q^{Max}} \tag{11}$$

where j is the rank of a URL in the list; $\delta(q,j)$ is 1 if URL j is relevant to user's information need in query instance q and 0 otherwise; and α is set to 5, which follows the setting in[2]. The R_q^{Max} is the obtained maximum possible rank scoring for a query instance when all relevant URLs appearing at the top of the ranked list. A larger rank scoring value indicates a better quality of the ranked URLs list. Moreover, the "SAT click" is used for relevance judgement of a URL. In our experiments, we evaluate the original result list given by Bing and the re-ranked list given by our algorithms in the same way. The performance of proposed algorithms can be measured by the *improvement percentage* of the re-ranked rank scoring compared with the Bing's original rank scoring. The positive value (improvement percentage>0) means that the performance is increased after re-ranking, while zero value means performance staying unchanged and the negative value means the performance decreased.

5.4 Experimental Results and Discussions

Fig.2 shows the re-ranking performance of our algorithms in comparison with the baseline algorithm (VSM). (A) and (B) show the improvement percentage of our algorithms with different parameter K distributed on different click entropy intervals. The results show that the re-ranking performance of both V-S and S-S reach their peaks at a specific K value. A too small K value, e.g., $K = 2$, implies that too few topics are selected to personalize the web search which may leave out some important information for the current query. A too large K value, e.g., $K = 20$, may introduce too much noise. Only a proper K value can result in



(A)

V-S	K=2	K=7	K=12	K=17	K=20
[3.0,3.5)	0.26%	-0.90%	-1.93%	-1.74%	-0.17%
[3.5,4.0)	-2.20%	-1.95%	-1.56%	+0.17%	+1.64%
[4.0,4.5)	+2.81%	+3.71%	-0.07%	+1.737%	+3.44%
[4.5,5.0)	+6.18%	-1.35%	+1.50%	+0.487%	-1.70%
[5.0, ∞)	+25.95%	+30.93%	+24.60%	+25.67%	+24.06%

(B)

S-S	K=2	K=7	K=12	K=17	K=20
[3.0,3.5)	-0.49%	-0.14%	+1.00%	+0.09%	-0.37%
[3.5,4.0)	-0.23%	+0.33%	+0.72%	-1.44%	+0.21%
[4.0,4.5)	+4.97%	+4.80%	+2.08%	+1.61%	+0.65%
[4.5,5.0)	+1.03%	+0.93%	-1.36%	-3.51%	-0.41%
[5.0, ∞)	+21.37%	+21.50%	+23.75%	+35.20%	+33.74%

Compare	VSM	V-S (K=7)	S-S (K=17)
[3.0,3.5)	+2.30%	-0.90%	+0.09%
[3.5,4.0)	-0.450%	-1.95%	-1.44%
[4.0,4.5)	+2.07%	+3.71%	+1.61%
[4.5,5.0)	-5.22%	-1.35%	-3.51%
[5.0, ∞)	+30.23%	+30.93%	+35.20%

(C)

(D)

Model	#Increase	#Stay	#Decrease
V-S K=7	31	48	28
S-S K=17	35	47	25
VSM	37	47	23

(E)

	V-S K=7			S-S K=17			VSM		
Entropy	#Increase	#Stay	#Decrease	#Increase	#Stay	#Decrease	#Increase	#Stay	#Decrease
[3.0,3.5)	125	327	132	113	330	141	137	326	121
[3.5,4.0)	53	155	62	63	146	61	58	151	61
[4.0,4.5)	36	94	46	34	97	45	35	94	47
[4.5,5.0)	36	60	31	38	58	31	37	60	30
[5.0,∞)	37	118	7	37	118	7	36	118	8

Fig. 2. The re-ranking performance evaluated from different angles. (A) and (B) show the improvement percentage of our algorithms with different parameter K distributed on click entropy intervals; (C) is the comparative results among VSM, S-S (K=17) and V-S (K=7); (D) is the distribution of user number on different re-ranking performance ('increase', 'stay' and 'decrease') for different algorithms; (E) is the distribution of query number on different click entropy intervals and different re-ranking performance ('increase', 'stay' and 'decrease') for different algorithms.

the most improvement of re-ranking performance. The results also show that the re-ranking performance of our algorithms (V-S and S-S) is relative poor for the queries with lower click entropy (less than 5.0), and is relatively good for queries with higher click entropy. One reason is that the Bing search engine has returned relative good results to users in the former case and thus there is little potential to improve it. There may even be a risk to harm the users' search experience when we personalize the queries with lower click entropy.

The average best performance of algorithms V-S and S-S appears in $K = 7$ and $K = 17$ respectively. Table (C) compares the performance between VSM and our algorithms, namely S-S (K=17) and V-S (K=7). We observe that VSM demonstrates a better performance for lower click entropy queries ([3.0,3.5)); however, our two algorithms outperform VSM when the click entropy is large (> 4.0); especially, the S-S gain the best performance of a 35.20% improvement in click entropy interval [5.0,∞). Table (D) in Fig.2 shows the distribution of user numbers over different re-ranking performance ('increase', 'stay' and 'decrease') for the 3 algorithms. This table indicates that the VSM helped slightly more users to improve the search results with the fewest harm (with the fewest number of users whose re-ranking performance is decreased) to user's search quality. (E) gives another statistical analysis of the experimental result, i.e., the distribution of query numbers over different click entropy intervals and different re-ranking performance ('increase', 'stay' and 'decrease') for different algorithms, showing that the VSM is more robust in lower click entropy intervals (less than 4.0) and the robustness in higher click entropy intervals for different algorithms is similar.

Overall, we find that our proposed algorithms are effective, especially, for the queries with higher click entropy (which are queries worthwhile to personalize [2]). The superiority of our methods are gained for four reasons: (i) we build word-correlation matrixes for user profile and documents, which not only captures the importance of single words, but also takes the correlation between words into consideration; (ii) we decompose the word-correlation matrix through SVD, and the dimensionality is reduced to a small value, so that the main aspects of the user search history can be used to personalize the new query; (iii) a document is represented as subspace spanned by the top K eigenvectors of the document word-correlation matrix, which captures the main topics of the documents and could serve as a denoising algorithm to some extent; (iv) the unified representation of the user profile and document as subspaces (or document as vector) well capture the geometrical features of the user profile and documents, and based on this representation, we incorporate the well-principled subspace projection theory into our personalization framework.

6 Conclusions and Future Work

In this paper we propose two novel personalized re-ranking algorithms, based on subspace projection, to re-rank the original web search results which outperform the traditional VSM model especially for the queries with higher click entropy. It is noting that, in our work, we did not select the most relevant historical queries for building the user profile, since we have utilized the subspace projection theoretical framework that can automatically detect the most relevant concepts (topics) when computing the personalized scores. More specifically, the selected top K eigenvectors can be seen as some important and different topics of user's search interests, and the projection from the document vector (subspace) to user profile subspace can map the most relevant topics to the retrieved documents. In the future, we will incorporate more information, such as similar queries and similar users, to further improve our model and algorithms.

Acknowledgements. This work is funded in part by the Chinese National Program on Key Basic Research Project (973 Program, grant no. 2013CB329304 and 2014CB744604), the Natural Science Foundation of China (grant no. 61272265), and the European Union Framework 7 Marie-Curie International Research Staff Exchange Programme (grant no. 247590).

References

1. Melucci, M. (2008). A basis for information retrieval in context. ACM Transactions on Information Systems (TOIS), 26(3), 14.
2. Dou, Z., Song, R., & Wen, J. R. (2007, May). A large-scale evaluation and analysis of personalized search strategies. In *WWW*, pp. 581-590. ACM.
3. Teevan, J., Dumais, S. T., & Horvitz, E. (2010). Potential for personalization. ACM Transactions on Computer-Human Interaction (TOCHI), 17(1), 4.
4. Bennett, P. N., Radlinski, F., White, R. W., & Yilmaz, E. (2011, July). Inferring and using location metadata to personalize web search. In *SIGIR*, pp. 135-144. ACM.
5. Sontag, D., Collins-Thompson, K., Bennett, P. N., White, R. W., Dumais, S., & Billerbeck, B. (2012, February). Probabilistic models for personalizing web search. In *WSDM*, pp. 433-442. ACM.

6. White, R. W., Bennett, P. N., & Dumais, S. T. (2010, October). Predicting short-term interests using activity-based search context. In *CIKM*, pp. 1009-1018. ACM.
7. Xiang, B., Jiang, D., Pei, J., Sun, X., Chen, E., & Li, H. (2010, July). Context-aware ranking in web search. In *SIGIR*, pp. 451-458. ACM.
8. Agichtein, E., Brill, E., & Dumais, S. (2006, August). Improving web search ranking by incorporating user behavior information. In *SIGIR*, pp. 19-26. ACM.
9. Zhang, S., & Dong, N. (2003). An effective combination of different order n-grams. Proceedings of O-COCOSDA, 251-256.
10. Bennett, P. N., White, R. W., Chu, W., Dumais, S. T., Bailey, P., Borisyuk, F., & Cui, X. (2012, August). Modeling the impact of short-and long-term behavior on search personalization. In *SIGIR*, pp. 185-194. ACM.
11. G. Golub and C. V. Loan. Matrix Computation, 2nd edition. The Johns Hopkins University Press, Baltimore, Maryland, 1989.
12. Xu, S., Bao, S., Fei, B., Su, Z., & Yu, Y. (2008, July). Exploring folksonomy for personalized search. In *SIGIR*, pp. 155-162. ACM.
13. Sun, J. T., Zeng, H. J., Liu, H., Lu, Y., & Chen, Z. (2005, May). CubeSVD: a novel approach to personalized Web search. In *WWW*, pp. 382-390. ACM.
14. Dwork, Cynthia and Kumar, Ravi and Naor, Moni and Sivakumar, Dandapani. (2001). Rank aggregation methods for the web. In *WWW*, pp. 613-622. ACM.
15. Melucci, Massimo. (2005).Context modeling and discovery using vector space bases. In *CIKM*, pp.808-815. ACM.
16. Melucci, Massimo and White, Ryen W. (2007). Utilizing a geometry of context for enhanced implicit feedback. In *CIKM*, pp.273-282. ACM
17. van Rijsbergen, Cornelis Joost. (2004). The geometry of information retrieval. The Cambridge University Press.
18. M.F. Porter. (1980) An algorithm for suffix stripping, Program, 14(3) pp.130-137.
19. Zhai, Chengxiang and Lafferty, John. (2001) A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR*, pp. 334-342. ACM.
20. Collins-Thompson, K., Bennett, P. N., White, R. W., de la Chica, S., & Sontag, D. (2011, October). Personalizing web search results by reading level. In *CIKM*, pp. 403-412. ACM.
21. Nanas, N., Vavalis, M., & De Roeck, A. N. (2010, July). A network-based model for high-dimensional information filtering. In *SIGIR* pp. 202-209. ACM.