Expert Systems with Applications 40 (2013) 6805–6812

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Collaborative pseudo-relevance feedback

Dong Zhou [a,*], Mark Truran [b], Jianxun Liu [a], Sanrong Zhang [a]

[a] Key Laboratory of Knowledge Processing and Networked Manufacturing, College of Hunan Province & School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, Hunan 411201, China
[b] School of Computing, Teesside University, UK

ARTICLE INFO

Keywords:
Pseudo-relevance feedback
Information retrieval
Collaborative filtering
Adaptive tuning

ABSTRACT

Pseudo-relevance feedback (PRF) is a technique commonly used in the field of information retrieval. The performance of PRF is heavily dependent upon parameter values. When relevance judgements are unavailable, these parameters are difficult to set. In the following paper, we introduce a novel approach to PRF inspired by collaborative filtering (CF). We also describe an adaptive tuning method which automatically sets algorithmic parameters. In a multi-stage evaluation using publicly available datasets, our technique consistently outperforms conventional PRF, regardless of the underlying retrieval model.

1. Introduction

In the field of information retrieval (IR), the 'vocabulary problem' occurs when a query and a document contain terms which are lexically distinct, but semantically similar (e.g., the query contains the term 'mountain', while the document uses the synonym 'alpine'). One solution to this problem is known as pseudo-relevance feedback (PRF). PRF is a query expansion technique that extracts highly-weighted terms from the top-ranked documents returned by an initial search (Baeza-Yates and Ribeiro-Neto, 1999; Manning et al., 2008). These terms, which are assumed to be relevant, are added to the source query, which is resubmitted to the search engine (Rocchio, 1971; Salton, 1971). In theory, this expanded query leads to higher quality search results.

The effectiveness of PRF is heavily dependent on two algorithmic parameters – the number of top-ranked 'pseudo-relevant' documents (k), and the number of expansion terms extracted from each document (j). Determining optimal values for these parameters can be extremely difficult. Researchers commonly use fixed values obtained by 'tuning' the parameters against a test collection (Cleverdon, 1966). This approach has two drawbacks. Firstly, uniform parameters will actually *decrease* the quality of the results for some of the queries (Billerbeck and Zobel, 2005; Carpineto et al., 2002). Secondly, although parameter tuning is possible on Cranfield-style test collections, it is less feasible for 'real world' search applications where relevance information is difficult to obtain (e.g., Web IR, intranet search, digital libraries etc.).

In this paper, we describe a new approach to PRF inspired by *collaborative filtering* (Su and Khoshgoftaar, 2009). We call this approach collaborative PRF (or CPRF). CPRF uses *adaptive tuning* to automatically set critical algorithmic parameters. This means it remains effective even if manual parameter tuning against a test collection is impractical. Note that our technique is not *parameter free*. There are two parameters which must be manually set. However, as we demonstrate in Section 6.4, these parameters have a very limited impact on retrieval effectiveness.

CPRF is easy to implement. Assume an IR system receives a test query q_a. Our approach invokes a conventional PRF algorithm using a *range* of values for k and j. This produces a *set* of expanded queries. After filtering, these queries are submitted to an IR engine, generating a set of result lists. We treat the top-ranked documents in each list as 'ratings' that similar queries have 'awarded'. We fuse this pseudo-collaborative feedback with the IR-generated result lists to obtain the final ranking.

The remainder of this paper is organized as follows. In Section 2, we summarise related work. In Section 3, we explain how CPRF works. In Section 4 we describe the process of adaptive parameter tuning. Section 5 describes our various experiments. Section 6 presents our results. Section 7 concludes the paper and proposes future work.

2. Related work

2.1. Query expansion

Early approaches to query expansion were *manual* in nature. Search engine users were asked to extract expansion terms from top-ranked documents and reformulate their own queries (Harter, 1986). Manual query expansion was not popular because it required user intervention and required some knowledge of the underlying retrieval system.

Researchers quickly migrated to *automatic* query expansion techniques. In automatic query expansion, additional terms

* Corresponding author. Tel.: +86 18711331970.
E-mail address: dongzhou1979@hotmail.com (D. Zhou).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.06.030

(extracted from machine-readable thesauri or document corpora) are added to the query without the user's intervention (Bhogal et al., 2007; Park and Ramamohanarao, 2007; Qiu and Frei, 1993; Voorhees, 1994). PRF is a technique for automatic query expansion in which expansion terms are extracted from highly ranked documents returned by an initial search. Various attempts have been made to improve the performance of PRF (Lee et al., 2008; Metzler and Croft, 2007; Tao and Zhai, 2006) but the issue of optimal parameter setting remains a problem.

Recent work in this area has concentrated on the development of tools for *selective query expansion*. These tools share a similar idea, that is, to disable query expansion if the expanded query is predicted to perform poorly (Amati et al., 2004; Cronen-Townsend et al., 2004; He and Ounis, 2007; Li et al., 2012; Xu et al., 2009; Yom-Tov et al., 2005). One of the best known predictive functions, known as 'clarity score', is the Kullback–Leibler divergence between the query model and the collection model (Cronen-Townsend et al., 2005). Unfortunately, practical application of this predictive function has not been straightforward.

Related work includes log-based methods for selective query expansion (Cui et al., 2003; Fonseca et al., 2005), approaches utilising external evidence (Kwok and Chan, 1998; Zhou et al., 2012) and methods exploiting web data (Arguello et al., 2008; Elsas et al., 2008; Li et al., 2007; Milne et al., 2007; Xu et al., 2009). Query expansion continues to be an active research discipline. He and Ounis (2007) have even proposed the creation of a test collection specifically for this field.

2.2. Collaborative filtering/ranking

Collaborative filtering is a technique commonly used by commercial recommender systems. Recommender systems make predictions about the likelihood that a user u will like an item i. A prerequisite for this operation is a matrix relating items to ratings (Shardanand and Maes, 1995). These ratings are awarded by u and his/her peers (i.e., the user community). Assuming the availability of this matrix, the recommendation process works like this:

- Find the subset of all users who have awarded ratings to *other* items that agree with the ratings awarded by u.
- Filter this subset to find users who have already rated i.
- Calculate the mean rating awarded to i by these users.
- If the mean rating is positive, recommend i to u.

Algorithm 1. Populates the rating matrix V.

Require: q_a A TEST QUERY
Require: C A DOCUMENT CORPUS
1: $Q \leftarrow$ QUERY-EXPANSION (q_a)
2: /*Select a subset of these queries using clarity score */
3: $Q_c \leftarrow$ QUERY-CLARITY (Q)
4: **for** $q_i \in Q_c$ **do**
5: $(d_1, \ldots, d_x) \leftarrow$ IR-RETRIEVE (q_i, C)
6: $D \leftarrow D \cup (d_1, \ldots, d_x)$
7: **end for**
8: $(d_1, \ldots, d_y) \leftarrow$ IR-RETRIEVE (q_a, C)
9: $D \leftarrow D \cup (d_1, \ldots, d_y)$
10: **for all** $q_i \in Q_c \cup q_a$ **do**
11: **for all** $d_j \in D$ **do**
12: $v_{ij} \leftarrow$ RATE (q_i, d_j)
13: $V \leftarrow v_{ij}$
14: **end for**
15: **end for**
16: **return** V

Given a large enough matrix, this calculation quickly becomes computationally expensive. There are a number of memory- and model-based algorithms designed to optimise the process (Billsus and Pazzani, 1998; Hofmann, 2004; Resnick et al., 1994a; Sarwar et al., 2001). All of them experience difficulties when the matrix is sparsely populated. Accurately recommending products to new users (the 'cold start' problem) can also be challenging (Cacheda et al., 2011).

In previous work, CF systems have exploited document ranking functions to improve system performance (Weimer et al., 2007; Liu and Yang, 2008). The use of click-through logs alongside CF algorithms has also been investigated (Cao et al., 2010). However, to date, there has been no concerted attempt to combine collaborative filtering with PRF to improve the query expansion process.

3. Collaborative PRF

In this section, we explain how collaborative PRF actually works. The best way to do this is via a worked example. Assume our system has a test query q_a. First, we get a set of results for this query using an IR ranking function. Let's assume this operation produces the following set of (hopefully relevant) documents:

$$IR = \{d_1, d_2, d_5, d_7, d_{11}, d_{12}, d_{13}\}$$

Next, we invoke a conventional PRF algorithm, using a range of values for parameters k (the number of pseudo-relevant documents) and j (the number of expansion terms). This will generate a number ($m = k* j$) of expanded queries similar to q_a such that $Q = \{q_1, q_2 \ \ldots \ q_m\}$. We use clarity score to filter this set of expanded queries (Cronen-Townsend et al., 2005). We keep queries with low clarity scores as they are likely to produce diversified results (Algorithm 1, line 3).

In the next step, we retrieve the top-ranked documents for each query in the set of filtered queries Q_c (Algorithm 1, lines 4–7). We pool these documents with the top-ranked documents retrieved for q_a, and we use this data to populate a matrix V (Algorithm 1, lines 10–14). In Algorithm 1, x is the number of top-ranked documents used for the expanded queries, y is the number of top-ranked documents used for the test query q_a, and the RATE() function returns the score from the IR ranking function. We discuss parameters x and y in greater detail in Section 4.

In the matrix below, q_1 has two entries indicating retrieved documents $\{d_2, d_3\}$, q_2 has 3 retrieved documents and q_3 has 6. We take the top-x results for three queries. Here, we assume that $x = 2$. The parameter m is set to 3. For later use, we denote the query mean rating as $\overline{v_{a\cdot}}$ and document mean rating as $\overline{v_{\cdot d}}$.

$$V = \begin{bmatrix} & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \\ q_a & (\times) & (\times) & - & - & - & - \\ q_1 & - & \times & \times & - & - & - \\ q_2 & - & - & \times & \times & \times & - \\ q_3 & \times & \times & \times & \times & \times & \times \end{bmatrix}$$

Having built the rating matrix from V, we predict the relevance p_{ad} of each document $d \in D$ to q_a using a suitable collaborative filtering algorithm (Cacheda et al., 2011). This procedure is described in Algorithm 2. In this algorithm, lines 2–5 calculate the CF scores for all documents in the pool, excluding the high-ranking documents retrieved by q_a. These documents $(d_1, \ldots, d_y)$ are scored using a standard IR function. Depending on the exact CF algorithm used, it may or may not be possible to assign a CF prediction to every document beyond d_y (Cacheda et al., 2011).

In our experiment, we trial 4 different CF algorithms:

- A *user-based* algorithm, which makes recommendations based on the similarity of users (Resnick et al., 1994b).
- An *item-based* algorithm, which looks for similar items (Sarwar et al., 2001).
- An algorithm which uses singular value decomposition (SVD) to reduce the dimensionality of the rating matrix (Sarwar et al., 2000).
- An algorithm, known as *SlopeOne*, that exploits 'popularity differentials' between ratings assigned to items. In this study we use the weighted version (Lemire and Maclachlan, 2005).

Whichever CF algorithm we select, the output of this operation is another ranked set of documents. Returning to our example, this set might look like this:

$CF = \{d_1, d_2, d_3, d_4, d_5\}$

Algorithm 2. Generates predictions p_{ad} for each document.

Require: q_a A TEST QUERY
Require: V THE RATING MATRIX
Require: D A SET OF DOCUMENTS
1: /*Calculate CF predictions for all documents except top ranked documents for q_a */
2: **for all** $d_i \in D \backslash (d_1, \ldots, d_y)$ **do**
3: $p_{ai} \leftarrow$ CF-CALCULATION (q_a, d_i)
4: $\overrightarrow{p_{a\cdot}} \leftarrow p_{ai}$
5: **end for**
6: **return** $\overrightarrow{p_{a\cdot}}$

In the final stage of our procedure, we fuse the IR-generated results with the CF-generated results to produce a final document ranking. This procedure is described in Algorithm 3. In this algorithm, we iterate through a subset of the documents returned by the test query q_a, *combining* the CF and IR scores. Parameter c (Algorithm 3, line 2) is usually greater than y, and typically set to 1000.

We tried a number of different methods when combining IR and CF scores, from the simple (e.g. CombMAX, CombSUM) to the complex (e.g. CombRSV%, CORI, Z-score)(Savoy, 2004; Savoy, 2005). As shown in Fig. 1, the best performing combinatorial method in this context was CombRSV%. CombRSV% is usually defined in the following way:

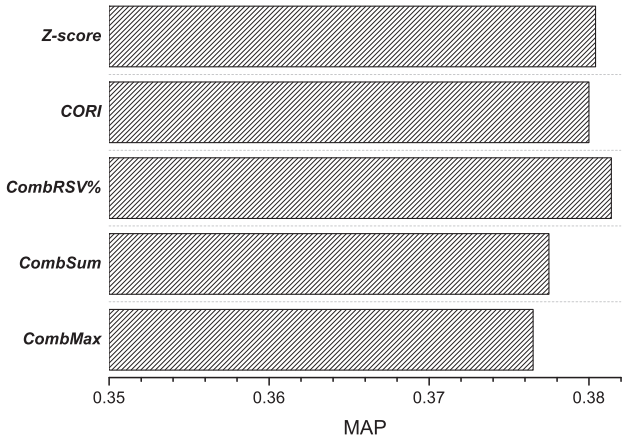

Fig. 1. Performance of combinatorial methods (CLEF-2006 dataset, see Section 5 for experimental settings).

$$SUM(RSV_i / MAX_{RSV})$$

where *RSV* denotes the retrieval status value (i.e., the score). In the context of CPRF, *RSV* refers to p_{ad} *or* the IR scores of the documents retrieved by the test query q_a (Algorithm 3, line 5).

Once we have calculated a combined score for $(d_1, \ldots, d_c)$, we aggregate the documents retrieved by the test query with the collaboratively found documents (Algorithm 3, lines 7–11). Now we have a set of documents with three sets of scores:

1. IR scores as retrieved by the IR engine.
2. CF scores, if $p_{ad} \notin \overrightarrow{p_{a\cdot}}$.
3. Combined scores (CombRSV%) if $d_i \in (d_1, \ldots, d_c)$.

We sort this set using all three scores (Algorithm 3, line 12). This produces a final ranked list, capped at 1000 documents, which we return to the user. Completing our example, we would expect to see something like this when we integrate the results:

$CombRSV = \{d_1, d_2, d_5, d_3, d_4, d_7, d_{11}, d_{12}, d_{13}\}$

Algorithm 3. Integrates IR-generated and CF-generated results to produce a final ranking list.

Require: q_a A TEST QUERY
Require: C A DOCUMENT CORPUS
Require: $\overrightarrow{p_{a\cdot}}$ CF PREDICTIONS
1: /*Combine the IR and CF scores for c documents*/
2: $(d_1, \ldots, d_c) \leftarrow$ IR-RETRIEVE (q, C)
3: **for all** $d_i \in (d_1, \ldots, d_c)$ **do**
4: **if** $p_{ad} \in \overrightarrow{p_{a\cdot}}$ **then**
5: COMBRSV-SCORE $(d_i) \leftarrow$ COMBRSV $(p_{ad_i}$, IR-SCORE $(d_i))$
6: **end if**
7: **end for**
8: /*Aggregate documents retrieved by q_a and expanded queries*/
9: **for all** d_j THAT $p_{aj} > 0$ **do**
10: $D_{final} \leftarrow (d_1, \ldots, d_c) \cup d_j$
11: **end for**
12: SORT BY IR SCORE, CF SCORE, COMBRSV-SCORE
13: **return** D_{final}

3.1. Possible weaknesses of CPRF

Collaborative filtering algorithms face the following problems: *sparsity* of the rating matrix and *cold start*. The 'sparsity problem' occurs because each user rates only a small subset of the available items, so most of the cells in the rating matrix are empty. The 'cold start' problem refers to the difficulties inherent in making recommendations for users recently introduced into the system. Neither problem reduces the viability of CPRF. For the IR-based pseudo-ratings, we can always enlarge the top-ranked list to prevent sparsity, and the CF ratings are used to calculate the final scores, *not* to predict accurate items. Furthermore, the 'cold start' scenario is impossible in CPRF, as a PRF algorithm will always produce 'similar' queries.

Another obvious weakness of CPRF is *computational complexity*. When parameters k and j are large, the number of expanded queries (and therefore the number of retrieval operations) becomes unmanageable. We have already discussed a partial solution to this problem (see Section 3). When populating the rating matrix, we filter all expanded queries using query clarity score (Cronen-Townsend et al., 2005). Further improvements to our algorithm might be obtained using a *heuristics-based approach*. This would involve the classification of positive query features and the selection of 'good'

expanded queries prior to document retrieval. This could work alongside, or instead of, the clarity score filter. Distributed computation, using an algorithm similar to *MapReduce*, is another option (Dean and Ghemawat, 2008).

4. Adaptive parameter tuning

There are two critical parameters listed in Algorithm 1. Parameter x is the number of top-ranked documents used for the expanded queries, and parameter y is the number of top-ranked documents used for the test query. Importantly, these parameters do *not* have to be tuned manually. They are tuned *automatically* using a adaptive method exploiting information gain (IG).

Algorithm 4. Calculates the best cut-off number.

Require: D_r POSITIVE EXAMPLES
Require: D_{nr} NEGATIVE EXAMPLES
1: $n \leftarrow |D_r \cup D_{nr}|$
2: $max_i \leftarrow max_1 \ldots n(IG(D_r,D_{nr},\{d_1, \ldots, d_i\},\{d_i + 1, \ldots, d_n\}))$
3: **return** i

Algorithm 5. Calculates the Information Gain (IG).

Require: D_r POSITIVE EXAMPLES
Require: D_{nr} NEGATIVE EXAMPLES
Require: S_+ PREDICTIVE RELEVANT
Require: S_- PREDICTIVE NON-RELEVANT
1: $IG \leftarrow 1 - |S_+|/|S_+ \cup S_-| \cdot Entropy\ (S_+) - |S_-|/|S_+ \cup S_-| \cdot Entropy(S_-)$
2: **if** $|D_r \cap S_+| < |D_{nr} \cap S_-|$ **then**
3: $IG \leftarrow -IG$
4: **end if**
5: **return** IG

Our adaptive method works by calculating the *utility* of the test query and all expanded queries. In this case, utility is measured by the information gained when separating a set of positive and negative examples. For a query q and a set S composed of positive and negative examples, the IG of q is calculated as the change in information entropy E when splitting S into subsets S_i according to the retrieval scores of q. We have:

$$IG(q,S) = E(S) - \sum_i E(S_i) \cdot \frac{|S_i|}{|S|}$$

where $E(S)$ represents the information entropy in a set S. The retrieval scores for q are the IR scores of documents in a particular run when q is used as a query. In theory, these scores should be continuous, but in our case they must be discretized in order to split them into subsets. Following (Egozi et al., 2011), the scores are discretized by calculating IG for every possible cut-off value, and using the best score as the query's IG. The cut-off selection process is described in Algorithm 4. In this algorithm, line 2 generates subsets S_i by treating $\{d_1, \ldots, d_i\}$ as predicted relevant documents and $\{d_i + 1, \ldots, d_n\}$ as predicted non-relevant documents. (These sets appear as S_+ and S_-, respectively, in Algorithm 5). We tried every possible i from 1 to n, where n represents the number of positive and negative examples (see the explanation to D_r and D_{nr} below).

Remember that we are not just trying to find the *best* IG value, as in (Egozi et al., 2011). We are actually trying to find the optimal values for x and y, which is a slightly different operation. To do this, we note the cut-off number when the best value appears (Algorithm 4, line 2) and use this number (i.e., the value of i that maximize the IG scores) for both parameters (i.e., $i = x = y$). For a particular result list returned by a query q, the top z of the documents are tagged as positive examples (i.e. D_r), and the bottom z are tagged as negative examples (D_{nr}). We applied Algorithms 4 and 5 to all of the queries to automatically determine x and y. Parameter z was set to the same value for both the test query q_a and the expanded queries.

In addition to the use of IG described above, we also applied our IG function to the list of expanded queries Q (Algorithm 1, line 3). We sorted this list in descending order of $IG(q)$ and selected the first α expanded queries to populate the matrix V. Parameters z and α are discussed in greater detail in Section 6.4.

Table 1
Summary of the experimental datasets.

Collection	Contents	Documents	Queries	Language
CLEF-2006	Glasgow Herald 1995, L.A. Times 1994	169,477	50	English
CLEF-2007	L.A. Times 2002	135,153	50	English

5. Evaluation

In the following section, we describe a series of experiments designed to answer the following questions:

1. Does CPRF outperform conventional PRF?
2. Which CF algorithm is best suited to CPRF?
3. What is the impact of changing the retrieval model?
4. What is the impact of changing parameters α and z?

5.1. Evaluation data

The text corpus used in our evaluation was built using components of the CLEF-2006[1] and CLEF-2007 test collections (see Table 1). We also used the CLEF-2006 and CLEF-2007 query sets. Each query set contains 50 topics. All 50 topics were used in the experiment. Each topic is composed of several fields (e.g., *Title, Description, Narrative* etc.). We selected the *Title* field as this closely resembles real world queries. We used relevance judgements produced by previous CLEF workshops. Prior to indexing, a suffix stemmer (Porter, 1997) and a stop-word list[2] were applied to all documents and queries.

5.2. Evaluation metrics

We used the following metrics when evaluating the performance of our technique:

- Normalized Discounted Cumulative Gain (NDCG) (Järvelin and Kekäläinen, 2000).
- Mean average precision (MAP).

Statistically-significant differences in performance were determined using a paired t-test at a confidence level of 95%.

5.3. Retrieval and baseline systems

All information retrieval functions in our experiments were handled by the Terrier open source platform (Ounis et al., 2006).[3] The first baseline we used was BM25, a popular and robust probabilistic retrieval model (Jones et al., 2000; Robertson et al.,

[1] http://www.clef-initiative.eu/
[2] ftp://ftp.cs.cornell.edu/pub/smart/
[3] http://terrier.org/

Table 2
Retrieval effectiveness on CLEF-2006 dataset.

Method	NDCG	Method	MAP
CPRF	0.6319*	CPRF	0.3814*
BM25-PRF-2-11	0.6271*	BM25-PRF-6-20	0.3767*
BM25-PRF-2-9	0.6262*	BM25-PRF-6-16	0.3760*
BM25-PRF-2-10	0.6261*	BM25-PRF-6-17	0.3757*
BM25-PRF-6-16	0.6252*	BM25-PRF-6-19	0.3748*
BM25-PRF-2-8	0.6245*	BM25-PRF-6-18	0.3740*
BM25-PRF-6-17	0.6245*	BM25-PRF-6-15	0.3726*
BM25-PRF-6-20	0.6245*	BM25-PRF-2-11	0.3721*
BM25-PRF-6-19	0.6243*	BM25-PRF-2-9	0.3717*
BM25-PRF-2-7	0.6238*	BM25-PRF-6-14	0.3717*
BM25-PRF-6-18	0.6238*	BM25-PRF-2-10	0.3715*
BM25	0.5959	BM25	0.3392

* Statistically significant improvements over the BM25 baseline are marked with with an asterisk.

Table 3
Retrieval effectiveness on CLEF-2007 dataset.

Method	NDCG	Method	MAP
CPRF	0.7285*	CPRF	0.4565*
BM25-PRF-10-17	0.7223*	BM25-PRF-4-19	0.4519*
BM25-PRF-4-14	0.7222*	BM25-PRF-4-16	0.4516*
BM25-PRF-10-18	0.7219*	BM25-PRF-4-17	0.4514*
BM25-PRF-4-19	0.7218*	BM25-PRF-4-11	0.4513*
BM25-PRF-4-16	0.7216*	BM25-PRF-4-18	0.4513*
BM25-PRF-4-17	0.7216*	BM25-PRF-10-15	0.4511*
BM25-PRF-4-20	0.7215*	BM25-PRF-4-10	0.4510*
BM25-PRF-4-9	0.7211*	BM25-PRF-10-9	0.4506*
BM25-PRF-10-6	0.7200*	BM25-PRF-4-8	0.4498*
BM25-PRF-4-8	0.7198*	BM25-PRF-10-10	0.4497*
BM25	0.6702	BM25	0.3856

* Statistically significant improvements over the BM25 baseline are marked with an asterisk.

1994). We also used BM25-PRF, a probabilistic retrieval function utilising PRF based on Divergence From Randomness (DFR) theory (Amati and Van Rijsbergen, 2002; Harter, 1975). We tried a range of parameters for the BM25-PRF retrieval function during set-up, settling on $k = 10$ (pseudo-relevant documents) and $j = 20$ (expansion terms). There was a significant performance drop-off beyond these values.

At this point, it is important to remember that CPRF generates a set of expanded queries using *all possible permutations* of k and j below these maximum values (i.e., 10 x 20 = 200 permutations of k and j). Each of these permutations can also be considered as a baseline. The goal of the first experiment was to demonstrate that CPRF can outperform all of them, with zero tuning.

In Section 6.3 we discuss the relationship between the performance of CPRF and the underlying retrieval model. In this part of the experiment, we used retrieval models exploiting statistical language models (LM) (Ponte and Croft, 1998) and term frequency-inverse document frequency (TFIDF) (Jones, 1972). These retrieval models are bundled with the standard Terrier distribution.

6. Results

6.1. CPRF vs. PRF

In our first experiment, we evaluated the performance of CPRF against conventional PRF. We used the *SlopeOne* algorithm to calculate the CF rankings (see Section 3). The results are shown in Table 2 (CLEF-2006) and Table 3 (CLEF-2007), which compare CPRF to the top scoring BM25-PRF-*k*-*j* variants.

There are three interesting points to note here. First, CPRF achieved the highest scores on both datasets, beating all 200 variations of BM25-PRF-*k*-*j* (with statistical significance). Second, CPRF achieved a statistically significant increase in performance over the BM25 baseline (no relevance feedback). Only 10% (CLEF-2006 dataset) and 12.5% (CLEF-2007 dataset) of the BM25-PRF-*k*-*j* variants managed the same feat. Third, performance of the BM25-PRF-*k*-*j* retrieval model was extremely unstable, with some scores actually *below* the BM25 baseline.

Fig. 2 illustrates the change in performance we observed when varying BM25-PRF parameters k and j. To confirm these findings, we repeated our experiment using the CLEF-2007 dataset. The results are shown in Figs. 3 and 4. Fig. 3 shows the change in MAP we recorded when parameter k was steadily increased with j held constant. Fig. 4 shows the change in MAP we observed when parameter j was increased with k held constant. These tests confirm that BM25-PRF is heavily dependent on parameter values. Finding the optimal values for these parameters is clearly a difficult task, especially when manual tuning (via a test collection) is not an option.

Fig. 5 plots the precision-recall (PR) curves for BM25, CPRF and the top performing BM25-PRF-*k*-*j* variant on the CLEF-2006 dataset. The performance gains achieved using CPRF are consistent across the full range of values.

6.2. Alternative CF algorithms

In our second experiment, we repeated the procedures described in Section 6.1, trying three alternative collaborative feedback algorithms in place of *SlopeOne*. The results are shown in Fig. 6 (CLEF-2006 dataset). Note that there is little difference

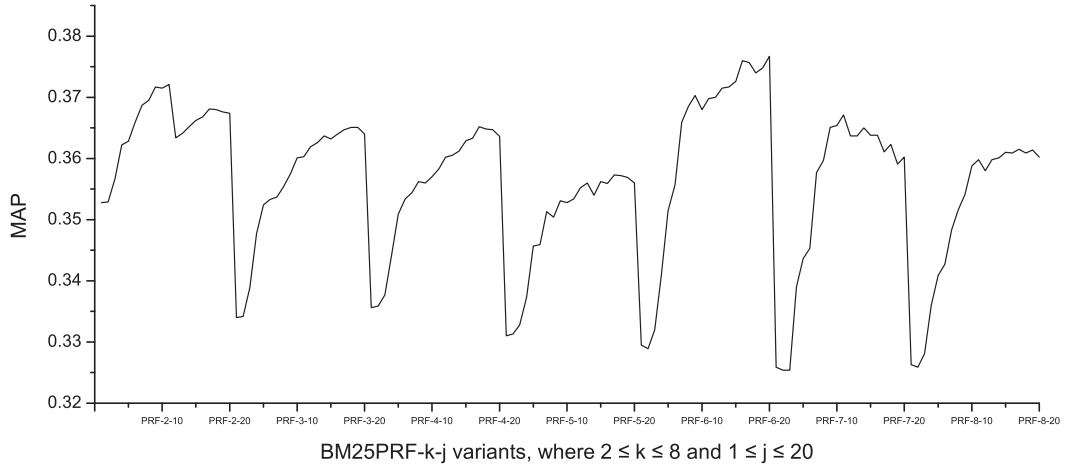

Fig. 2. Retrieval effectiveness of the BM25-PRF-*k*-*y* variants (CLEF-2006 dataset).

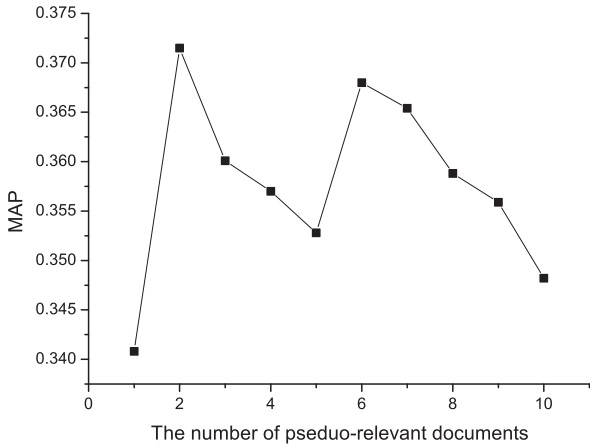

Fig. 3. The impact of parameter *k* (number of pseudo-relevant documents) on the performance of BM25-PRF-*k*-10 (CLEF-2007 dataset).

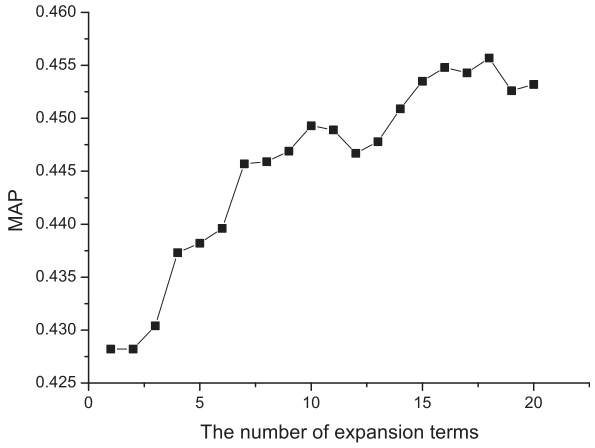

Fig. 4. The impact of parameter *j* (number of expansion terms) on the performance of BM25-PRF-5-*j* (CLEF-2007 dataset).

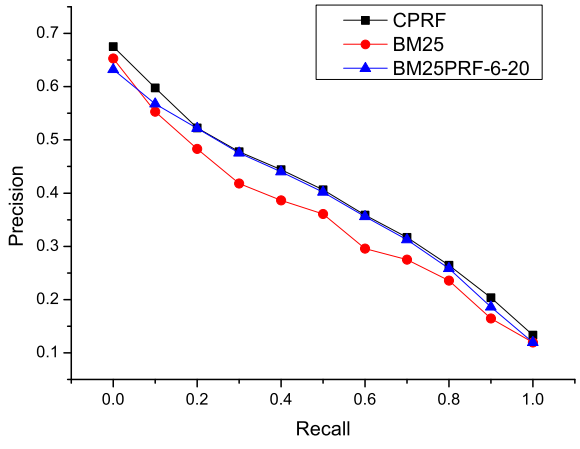

Fig. 5. PR curves for BM25, CPRF and the top performing BMPRF-*k*-*j* variant (CLEF-2006 dataset).

between the model-based CF algorithms (SVD and Weighted *Slope-One*) and the memory-based alternatives (User-and Item-based). This is significant. Memory-based CF algorithms work well when the ratings matrix is dense, but their accuracy decreases under sparse conditions. In contrast, model-based algorithms are less

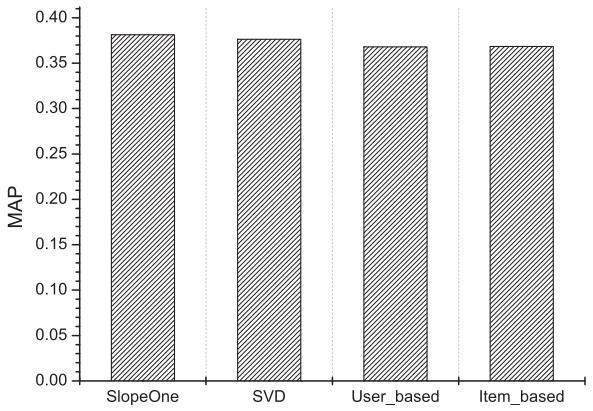

Fig. 6. The effect of changing the collaborative feedback algorithm (CLEF-2006 dataset).

Table 4
Retrieval effectiveness on CLEF-2006 dataset.

Method	MAP	Method	MAP
CPRF	0.3275*	CPRF	0.3768*
LM-PRF-9-1	0.3237*	TFIDF-PRF-6-16	0.3746*
LM-PRF-9-5	0.3217*	TFIDF-PRF-6-20	0.3745*
LM-PRF-10-6	0.3211*	TFIDF-PRF-6-19	0.3744*
LM-PRF-9-2	0.3210*	TFIDF-PRF-2-12	0.3735*
LM-PRF-8-1	0.3206*	TFIDF-PRF-6-17	0.3734*
LM-PRF-9-3	0.3204*	TFIDF-PRF-6-15	0.3731*
LM-PRF-10-4	0.3203*	TFIDF-PRF-6-18	0.3728*
LM-PRF-5-16	0.3187*	TFIDF-PRF-2-11	0.3724*
LM-PRF-9-6	0.3180*	TFIDF-PRF-6-14	0.3718*
LM-PRF-5-15	0.3178*	TFIDF-PRF-2-9	0.3713*
LM	0.3003	TFIDF	0.3461

* Statistically significant improvements over the LM and TFIDF baselines are marked with an asterisk.

Table 5
Retrieval effectiveness on CLEF-2007 dataset.

Method	MAP	Method	MAP
CPRF	0.4086*	CPRF	0.4556*
LM-PRF-8-14	0.4022*	TFIDF-PRF-4-16	0.4517*
LM-PRF-8-15	0.3998*	TFIDF-PRF-4-15	0.4516*
LM-PRF-8-17	0.3998*	TFIDF-PRF-4-19	0.4514*
LM-PRF-8-13	0.3982*	TFIDF-PRF-10-18	0.4514*
LM-PRF-8-16	0.3979*	TFIDF-PRF-4-17	0.4511*
LM-PRF-6-7	0.3979*	TFIDF-PRF-4-14	0.4510*
LM-PRF-4-9	0.3974*	TFIDF-PRF-10-10	0.4509*
LM-PRF-6-8	0.3972*	TFIDF-PRF-9-20	0.4507*
LM-PRF-9-15	0.3969*	TFIDF-PRF-4-8	0.4504*
LM-PRF-8-19	0.3951*	TFIDF-PRF-10-9	0.4498*
LM	0.3365	TFIDF	0.3859

* Statistically significant improvements over the LM and TFIDF baselines are marked with an asterisk.

sensitive to density changes. In this study, the SVD-based algorithm and the weighted *SlopeOne* algorithm produce similar results. This suggests that the 'sparsity problem' is not a critical issue for CPRF.

6.3. Alternative retrieval models

In our third experiment, we repeated the procedures described in Section 6.1, trying two alternative retrieval models in place of BM25. The results, shown in Table 4 (CLEF-2006) and Table 5 (CLEF-2007), suggest that CPRF is agnostic w.r.t. retrieval model. CPRF outperformed all of the language model (LM-PRF-*k*-*j*) and term frequency-inverse document frequency model (TFIDF-*k*-*j*) baselines.

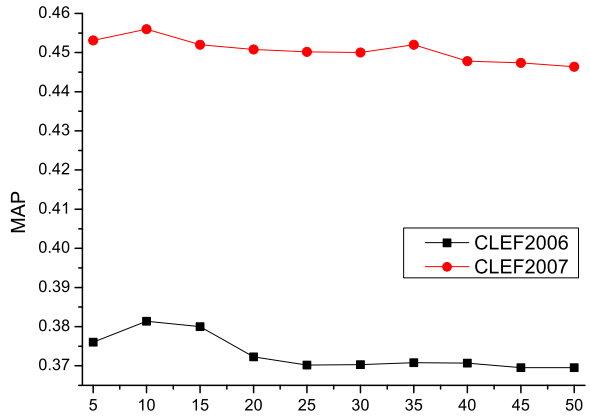

Fig. 7. The effect of varying the z parameter (CLEF-2006 and CLEF-2007 datasets).

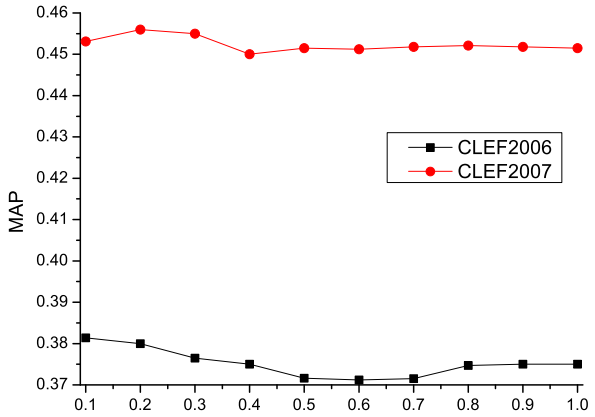

Fig. 8. The effect of varying the α parameter (CLEF-2006 and CLEF-2007 datasets).

6.4. Impact of parameters

Two parameters in our algorithm have to be manually set. The first, z, sets the number of positive and negative examples to use during adaptive parameter tuning (see Section 4). The second, α, dictates the number of expanded queries to use when populating the CF matrix (see Section 3). In our fourth experiment, we wanted to examine the impact of these parameters on retrieval performance. We re-ran the experiment described in Section 6.1, varying both parameters. We then repeated this operation using the CLEF-2007 dataset. Figs. 7 and 8 illustrate the results.

As illustrated, these parameters have *minimal* impact on retrieval performance. The best performance on CLEF-2006 was obtained using $z = 10$ and $\alpha = 10\%$, but even the worst performance was superior to the top ranked BM25-PRF-*k*-*j* variant. Optimal parameter values did change imperceptibly on CLEF-2007 ($z = 10$ and $\alpha = 20\%$), but the overall trend was stability. Parameter z provides the primary mechanism to determine x and y (see Section 3), and this remains constant for both test collections. We consider this to be a significant improvement over conventional PRF.

7. Conclusion and further work

In this paper, we have introduced a novel approach to pseudo-relevance feedback inspired by collaborative filtering. We have also described an adaptive tuning method which automatically sets critical algorithmic parameters. In a multi-stage evaluation using publicly available datasets, our method consistently outperformed conventional PRF. This result was duplicated on two test collections, using three different retrieval models.

In future work, we aim to explore the usefulness of alternative collaborative filtering algorithms. We also hope to examine the use of latent semantic analysis (LSA) when generating 'similar' queries. Heuristic-based approaches to query selection offer a particularly promising avenue for future development.

Acknowledgements

The authors thank the anonymous reviewers who greatly improved the quality of this manuscript during preparation. The work described in this paper was supported by the National Natural Science Foundation of China under Grant No. 61272063 and 61100054, Excellent Youth Foundation of Hunan Scientific Committee No. 11JJ1011, Hunan Provincial Natural Science Foundation of China under Grant No. 12JJ6064 and 12JJB009, Scientific Research Fund of Hunan Provincial Education Department of China under Grant No. 11B048 and 12K105.

References

Amati, G., Carpineto, C., & Romano, G. (2004). Query difficulty, robustness, and selective application of query expansion. In S. McDonald & J. Tait (Eds.), *Advances in information retrieval. Lecture notes in computer science* (vol. 2997, pp. 127–137). Berlin Heidelberg: Springer.

Amati, G., & Van Rijsbergen, C. J. (2002). Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transaction on Information System, 20*(4), 357–389.

Arguello, J., Elsas, J., Callan, J., & Carbonell, J.G. (2008). Document representation and query expansion models for blog recommendation. In *Proceedings of the second international conference on weblogs and social media*. (pp. 10–18).

Baeza-Yates, R. A., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. Boston, MA, USA.: Addison-Wesley Longman Publishing Co., Inc..

Bhogal, J., Macfarlane, A., & Smith, P. (2007). A review of ontology based query expansion. *Information Processing and Management, 43*(4), 866–886.

Billerbeck, B., & Zobel, J. (2005). Document expansion versus query expansion for ad-hoc retrieval. In *Proceedings of the 10th Australasian document computing symposium* (pp. 34C41). Australian Computer Society.

Billsus, D., & Pazzani, M. J. (1998). Learning collaborative information filters. In *Proceedings of the fifteenth international conference on machine learning. ICML '98* (pp. 46–54). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc..

Cacheda, F., Carneiro, V., Fernández, D., & Formoso, V. (2011). Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transaction on the Web, 5*(1), 2:1–2:33.

Cao, B., Shen, D., Wang, K., & Yang, Q. (2010). Clickthrough log analysis by collaborative ranking. In *The twenty-fourth aaai conference on artificial intelligence, AAAI 2010* (pp. 224–229). Atlanta, Georgia, USA: AAAI Press.

Carpineto, C., Romano, G., & GIANNINI, V. (2002). Improving retrieval feedback with multiple term-ranking function combination. *ACM Transaction on Information System, 20*(3), 259–290.

Cleverdon, Mills, J. (1966). Factors determining the performance of indexing systems. Volume I – Design, Volume II – Test results, ASLIB Cranfield Project, Reprinted in Sparck Jones & Willett, Readings in Information Retrieval.

Cronen-Townsend, S., Zhou, Y., & Croft, W.B. (2004). A framework for selective query expansion. In *Proceedings of CIKM 2004* (pp. 236–237).

Cronen-Townsend, S., Zhou, Y., & Croft, W. B. (2005). Predicting query performance. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 299–306). ACM. 564429.

Cui, H., Wen, J.-R., Nie, J.-Y., & Ma, W.-Y. (2003). Query expansion by mining user logs. *IEEE Transactions on Knowledge and Data Engineering, 15*(4), 829–839.

Dean, J., & Ghemawat, S. (2008). Mapreduce: Simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107–113.

Egozi, O., Markovitch, S., & Gabrilovich, E. (2011). Concept-based information retrieval using explicit semantic analysis. *ACM Transactions on Information Systems, 29*(2), 1–34<http://dl.acm.org/citation.cfm?id=1961209.1961211>.

Elsas, J. L., Arguello, J., Callan, J., & Carbonell, J. G. (2008). Retrieval and feedback models for blog feed search. In *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval SIGIR'08* (pp. 347–354). New York, NY, USA: ACM.

Fonseca, B. M., Golgher, P., Póssas, B., Ribeiro-Neto, B., & Ziviani, N. (2005). Concept-based interactive query expansion. In *Proceedings of the 14th ACM international conference on Information and knowledge management CIKM'05* (pp. 696–703). New York, NY, USA: ACM.

Harter, S. P. (1975). A probabilistic approach to automatic keyword indexing. part i. on the distribution of specialty words in a technical literature. *Journal of the American Society for Information Science, 26*(4), 197–206.

Harter, S. P. (1986). *Online information retrieval: Concepts, principles, and techniques.*. Academic Press Professional, Inc..

He, B., & Ounis, I. (2007). Combining fields for query expansion and adaptive query expansion. *Information Processing and Management*.

Hofmann, T. (2004). Latent semantic models for collaborative filtering. *ACM Transaction on Information and System, 22*(1), 89–115.

Järvelin, K., & Kekäläinen, J. (2000). Ir evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval SIGIR'00* (pp. 41–48). New York, NY, USA: ACM.

Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation, 28*(1), 11–21.

Jones, K. S., Walker, S., & Robertson, S. E. (2000). A probabilistic model of information retrieval: Development and comparative experiments. *Information Processing and Management, 36*(6), 779–808.

Kwok, K. L., & Chan, M. (1998). Improving two-stage ad-hoc retrieval for short queries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval SIGIR'98* (pp. 250–256). New York, NY, USA: ACM.

Lee, K.S., Croft, W.B., & Allan, J. (2008). A cluster-based resampling method for pseudo-relevance feedback. In *Proceedings of SIGIR 2008* (pp. 235–242).

Lemire, D., Maclachlan, A. (2005). Slope one predictors for online rating-based collaborative filtering. In *Proceedings of SIAM data mining* (pp. 471–180).

Li, L., Zhong, L., Xu, G., & Kitsuregawa, M. (2012). A feature-free search query classification approach using semantic distance. *Expert Systems with Applications, 39*(12), 10739–10748.

Li, Y., Luk, W. P. R., Ho, K. S. E., & Chung, F. L. K. (2007). Improving weak ad-hoc queries using wikipedia asexternal corpus. In *Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval SIGIR'07* (pp. 797–798). New York, NY, USA: ACM.

Liu, N. N., & Yang, Q. (2008). Eigenrank: A ranking-oriented approach to collaborative filtering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval SIGIR'08* (pp. 83–90). New York, NY, USA: ACM.

Manning, C. D., Raghavan, P., & Schtze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.

Metzler, D., & Croft, W.B. (2007). Latent concept expansion using markov random fields. In *Proceedings of SIGIR 2007* (pp. 311–318).

Milne, D. N., Witten, I. H., & Nichols, D. M. (2007). A knowledge-based search engine powered by wikipedia. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management CIKM'07* (pp. 445–454). New York, NY, USA: ACM.

Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., & Lioma, C. (2006). Terrier: A high performance and scalable information retrieval platform. In *Proceedings of ACM SIGIR'06 workshop on open source information retrieval (OSIR 2006)*.

Park, L., & Ramamohanarao, K., (2007). Query expansion using a collection dependent probabilistic latent semantic thesaurus. In *Proceedings of the 11th Pacific-Asia conference on knowledge discovery and data mining (PAKDD07)* (pp. 224–235).

Ponte, J. M., & Croft, W. B. (1998). A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval SIGIR'98* (pp. 275–281). New York, NY, USA: ACM.

Porter, M. F. (1997). *Readings in information retrieval. Ch. An algorithm for suffix stripping*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. (pp. 313–316). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc..

Qiu, Y., & Frei, H.-P. (1993). Concept based query expansion. In *Proceedings of the 16th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 160–169). ACM (160713).

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994a). Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on computer supported cooperative work CSCW'94* (pp. 175–186). New York, NY, USA: ACM.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994b). Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work CSCW'94* (pp. 175–186). New York, NY, USA: ACM.

Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.,& Gatford, M., (1994). Okapi at trec-3. In *TREC* (pp. 109-126).

Rocchio, J. (1971). Relevance feedback in information retrieval (pp. 313–323).

Salton, G. (1971). *The SMART retrieval system; experiments in automatic document processing*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web WWW'01* (pp. 285–295). New York, NY, USA: ACM.

Sarwar, B.M., Karypis, G., Konstan, J.A., & Riedl, J.T. (2000). Application of dimensionality reduction in recommender systemsàca case study. In *Proceedings of the ACM WebKDD Workshop*.

Savoy, J. (2004). Combining multiple strategies for effective monolingual and cross-language retrieval. *Information Retrieval, 7*(1-2), 121–148.

Savoy, J. (2005). Comparative study of monolingual and multilingual search models for use with asian languages 4(2) (pp. 163–189).

Shardanand, U., & Maes, P. (1995). Social information filtering: Algorithms for automating 'word of mouth'. In *Proceedings of the SIGCHI conference on human factors in computing systems CHI'95* (pp. 210–217). New York, NY, USA: ACM Press/Addison-Wesley Publishing Co..

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence, 2009*, 4:2–4:2.

Tao, T., Zhai, C. (2006). Regularized estimation of mixture models for Robust pseudo-relevance feedback. In *Proceedings of SIGIR 2006* (pp. 162–169).

Voorhees, E. M. (1994). Query expansion using lexical-semantic relations. In *Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 61–69). New York, Inc.: Springer-Verlag. 188508.

Weimer, M., Karatzoglou, A., Le, Q., & Smola, A. (2007). Cofirank maximum margin matrix factorization for collaborative ranking. *Advances in Neural Information Processing Systems, 20*, 1593–1600.

Xu, Y., Jones, G., & Wang, B. (2009). Query dependent pseudo-relevance feedback based on wikipedia. In *Proceedings of the 32nd annual international ACM SIGIR conference on research and development in information retrieval* (pp. 59–66). ACM.

Yom-Tov, E., Fine, S., Carmel, D., & Darlow, A. (2005). Learning to estimate query difficulty: Including applications to missing content detection and distributed information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval SIGIR'05* (pp. 512–519). New York, NY, USA: ACM.

Zhou, D., Lawless, S., & Wade, V. (2012). Improving search via personalized query expansion using social media. *Information Retrieval, 15*, 218–242.