Inf Retrieval (2014) 17:471–491
DOI 10.1007/s10791-013-9236-1

Using multiple query representations in patent prior-art search

Dong Zhou · Mark Truran · Jianxun Liu · Sanrong Zhang

Received: 2 May 2013 / Accepted: 29 October 2013 / Published online: 21 November 2013

Abstract Before a patent application is made, it is important to search the appropriate databases for *prior-art* (i.e., pre-existing patents that may affect the validity of the application). Previous work on prior-art search has concentrated on single query representations of the patent application. In the following paper, we describe an approach which uses multiple query representations. We evaluate our technique using a well-known test collection (CLEF-IP 2011). Our results suggest that multiple query representations significantly outperform single query representations.

Keywords Patent search · Prior-art · Collaborative filtering

1 Introduction

Patent search is an active sub-domain of the research field known as information retrieval (IR; Tait 2008). A common task in patent IR is the *prior-art* search. This type of search is usually performed by individuals who need to ensure originality before applying for, or granting, a new patent. These individuals use IR systems to search databases containing previously filed patents. The entire patent application, or some subset of words extracted from it, is typically used as the query (Mahdabi et al. 2012; Piroi et al. 2011; Xue and Croft 2009).

At the time of writing, there are various state-of-the-art patent IR systems (e.g., Becks et al. 2011; Lopez and Romary 2010; Magdy and Jones 2010; Mahdabi et al. 2011). All of these systems use *single query* representations of the patent application. In this paper, we

D. Zhou (✉) · J. Liu · S. Zhang
Key Laboratory of Knowledge Processing and Networked Manufacturing & School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, Hunan, China
e-mail: dongzhou1979@hotmail.com

M. Truran
School of Computing, Teesside University, Middlesbrough, UK
e-mail: m.a.truran@tees.ac.uk

describe an approach to prior-art search that uses *multiple query* representations. Given a patent application, we generate a *set* of similar queries. Each of these queries is an *alternative representation* of the information contained in the application. This set of queries is submitted to an IR engine. We treat each batch of search results as a set of 'ratings'. We analyse these ratings using collaborative filtering (CF) algorithms. Subsequently, we merge this pseudo-collaborative feedback with a set of standard IR results to achieve a final document ranking.

The remainder of this paper is organized as follows. In Sect. 2, we summarise related work from the fields of patent IR, collaborative filtering and data fusion. In Sect. 3, we describe a CF-based implementation of patent prior-art search. In Sect. 4, we discuss an extension of our technique called *iterative refinement*. Section 5 documents the experiments we used to evaluate our algorithms. Section 6 presents our results. Section 7 concludes the paper and proposes future work.

2 Related work

2.1 Patent IR

One of the defining challenges in patent IR is the problem of representing a long, technical document as a query. Early systems mimicked the approach taken by professional patent examiners, who (at the time) valued high frequency words as query terms (Itoh et al. 2003; Iwayama et al. 2003). Recently, use of the entire patent application (or a large set of terms automatically extracted from it) has become popular. Automatically extracting appropriate query terms from a patent application is a difficult task. These documents usually contain a large volume of text sectioned into multiple fields (e.g. *Title, Abstract, Description, Claims*. etc.). This being the case, how do we extract the 'right' terms, and which field(s) do we extract those terms from?

Xue and Croft (2009) examined query terms taken from different fields of a patent application. In an experiment using data published by the United States Patent and Trademark Office (USPTO), they found the best performance was obtained using high frequency terms extracted from the raw text of the *Description* field. These results were subsequently confirmed by other research teams. Magdy et al. (2011) produced the second best run of CLEF-IP 2010[1] (Piroi 2010) using patent numbers extracted from the *Description* field, and Mahdabi et al. confirmed this finding when experimenting with Language Models (LM; Mahdabi et al. 2011, 2012).

The status of phrases in patent IR is somewhat uncertain. One study has suggested that retrieval performance can be improved by including noun phrases (obtained via a global analysis of the patent corpus) in the query (Mahdabi et al. 2012). Another study, using a different patent collection (CLEF 2011 rather than CLEF 2010), found quite the opposite (Becks et al. 2011).

There are several key differences between patent search and conventional IR. Patent queries, which typically contain several hundred terms, are obviously much longer than standard IR queries. This makes high precision retrieval very difficult, as the information need is quite diffuse. Techniques which work well in conventional search do not always translate gracefully to patent IR. Pseudo-relevance feedback (PRF), for example, performs very poorly in this particular context (Ganguly et al. 2011; Mahdabi et al. 2012). In patent

[1] http://www.clef-initiative.eu/.

IR, precision is relatively low even in the top-ranking results. Expanding a query using terms extracted from top-ranked patents tends to produce additional noise, rather than focus the information need.

2.2 Collaborative filtering

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 as follows:

- Find the subset of all users who have awarded ratings to *other* items that agree with the ratings awarded by *u*
- Use ratings awarded by like-minded users to predict items for *u*

Given a large enough matrix, this process quickly becomes computationally expensive. There are a number of memory- and model-based algorithms designed to optimise the process. Memory-based algorithms [e.g., item-based and user-based systems (Resnick et al. 1994; Sarwar et al. 2001)] exploit the whole matrix when computing predictions. Generally, these predictions are calculated from the ratings of neighbours (i.e. users or items that are similar to the active user/item). In contrast, model-driven techniques make predictions based on user behaviour models. The parameters of the models are estimated offline. Techniques exploiting singular value decomposition (SVD; Billsus and Pazzani 1998) and probabilistic methods [e.g., latent class models (Hofmann 2004)] are common in this context.

A number of CF algorithms use graph-based analysis to calculate item predictions. A common approach involves modelling the users as nodes in an undirected weighted graph, wherein edges represent the degree of similarity between users based on rating activity (Aggarwal et al. 1999; Luo et al. 2008). There are a number of variations from this basic pattern. For example, Wang et al. proposed a recommendation scheme based on *item graphs* (Wang et al. 2006). In this scheme, items are nodes and edges represent pairwise item relationships. Huang et al. advanced this idea, proposing a bipartite graph comprising of item nodes and users nodes (Huang et al. 2004). In this scheme, ratings are modelled as links connecting nodes from the disjoint sets. Transitive associations between the nodes are subsequently used to generate item predictions.

It is worth noting that memory- and model-based algorithms both experience difficulties when the ratings matrix is sparsely populated. Accurately recommending products to new users (i.e., the ‘cold start’ problem) is also challenging (Cacheda et al. 2011). In the past, collaborative feedback algorithms have been combined effectively with conventional IR models (Zhou et al. 2013). Researchers have exploited IR rankings and click-through logs to improve the performance of CF algorithms (Cao et al. 2010; Liu and Yang 2008; Weimer et al. 2007). However, to date, there has been no attempt to combine CF algorithms and IR models in the field of patent search.

2.3 Data fusion algorithms

Our work combines multiple results sets retrieved using alternate query representations (i.e., data fusion). There are two general approaches when fusing search results. The first approach is *unsupervised*. Shaw and Fox have proposed a number of successful algorithms in this context, including CombSUM and CombMNZ (Shaw and Fox 1994). Other unsupervised algorithms, developed for monolingual and multilingual search, include CombRSV, CombRSVNorm (Powell et al. 2000) and CORI (Callan et al. 1995; see also Savoy 2004, 2005). The second approach to data fusion is *supervised* (Sheldon et al. 2011; Si and Callan 2005; Tsai et al. 2008). The supervised technique involves two steps. In step one, the quality of various result sets is 'learnt' from relevance judgements. In step two, unseen results sets are merged using predictions based on step one. Optional pre-processing (e.g., systemic bias, query 'gating') may be applied during this stage (Sheldon et al. 2011; Si and Callan 2005; Tsai et al. 2008).

3 Collaborative patent prior-art search (CPAC)

In this section, we explain our approach to patent prior-art search. We begin with an explanation of our notation. Our technique deals with a finite set of queries, $Q = \{q_a, q_1, q_2 \ldots q_n\}$, and a finite set of documents $D = \{d_1, d_2 \ldots d_m\}$ aggregated from documents retrieved by Q. Each query $q \in Q$ is associated with a *profile*, which consists of a set of documents retrieved by submitting that query to a standard IR engine, $D_q \subseteq D$, and the corresponding retrieval scores. Note that we treat these retrieval scores as CF *ratings*. These ratings, denoted R, will always correspond to real numbers. The first query we send to the IR system is denoted q_a. The subset of queries that have retrieved a certain document d is defined as $Q_d \subseteq Q$. Note that q and d (used in the subscript) vary over the sets $\{q_a, q_1, q_2 \ldots q_n\}$ and $\{d_1, d_2 \ldots d_m\}$.

Using the query profiles, we construct a rating matrix V. V will contain $|Q|$ rows and $|D|$ columns. Each element of V, $v_{qd} \in R \cup \varnothing$, denotes the rating given by query $q \in Q$ to document $d \in D$. A value of $\varnothing$ for v_{qd} indicates that the query q has not retrieved the document d yet. We process this matrix using a CF algorithm (see Algorithm 2). The goal of this algorithm is to predict the value v for documents which have not been retrieved. Let us denote the prediction for $d \in D$ by query $q \in Q$ as $p_{qd} \in R \cup \varnothing$ (p_{ad} for q_a). If our CF algorithm is not able to make this prediction, then we set $p_{qd} = \varnothing$. For later use, we define the subset of document ratings for the query q as $v_{q\cdot} = \{v_{qd} \in V / d \in D_q\}$, and the subset of query ratings for the document d as $v_{\cdot d} = \{v_{qd} \in V / q \in Q_d\}$. We also denote the document mean rating for a query q as $\overline{v_{q\cdot}}$($\overline{v_{a\cdot}}$ for q_a) and query mean rating for a document d as $\overline{v_{\cdot d}}$.

Now, assume our system receives a single query q_a. First, we obtain a set of results for this query from a standard IR engine. Next, we generate a set of queries $Q' = \{q_1, q_2 \ldots q_n\}$ similar to q_a. Each of these auto-generated queries is an *alternative representation* of the information contained in q_a (see Sect. 3.2 for the query representations used in this study). We retrieve the top-ranked documents for each query in $Q (Q \leftarrow q_a \cup Q')$, using this data to construct our ratings matrix. This process is described in Algorithm 1. In this algorithm, x is the number of top-ranked documents we retrieve for each query q, and *RATE*() returns the score from the IR engine. Note that the top ranked documents $\{d_1, \ldots, d_x\}$ are likely to be different for each $q_i \in Q$. We cache the documents returned by IR-RETRIEVE (), together with the scores, for later use.

Algorithm 1 Populates the rating matrix V

```
Require: $q_a$ THE ORIGINAL QUERY
Require: $Q'$ A SET OF QUERIES SEEDED BY $q_a$
Require: $C$ A DOCUMENT CORPUS

  $Q \leftarrow q_a \cup Q'$
  for all $q_i \in Q$ do
     $\{d_1, \ldots, d_x\} \leftarrow$ IR-RETRIEVE $(q_i, C)$
     $D \leftarrow D \cup \{d_1, \ldots, d_x\}$
  end for
  for all $q_i \in Q$ do
     for all $d_j \in D$ do
        $v_{ij} \leftarrow$ RATE $(q_i, d_j)$
        $V \leftarrow v_{ij}$
     end for
  end for
  return $V$
```

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

An example of a populated matrix is shown above. In this matrix, d_i denotes the sequence number of the document in the entire corpus. Here, we assume that $x = 2$. The *RATE*() function in Algorithm 1 returns real numbers, but we have replaced all real numbers with a symbol ($\times$) to simplify the diagram. A ($\diamond$) symbol indicates that a query q_i has not retrieved that specific document d_i.

Having built the rating matrix V, we predict the relevance p_{ad} of each document $d \in D$ to q_a. This procedure is described in Algorithm 2. In this algorithm, we iterate through all documents in D (excluding those documents which were retrieved using the original query q_a) to produce a vector of predictions $\overrightarrow{p_{a\cdot}}$. These predictions are calculated using a collaborative filtering algorithm (Cacheda et al. 2011). In our experiment, we try four different CF algorithms, as follows:

User-based

$$p_{ad} = \overline{v_{a\cdot}} + \sigma_a \frac{\sum_{q \in neigh_a} \left[\left(\frac{v_{ad} - \overline{v_{a\cdot}}}{\sigma_q} \right) s(a, q) \right]}{\sum_{q \in neigh_a} s(a, q)}$$

Item-based

$$p_{ad} = \frac{\sum_{d'} (s(d', d) v_{ad})}{\sum_{d'} |s(d', d)|}$$

SVD

$$p_{ad} = \overline{V_{a\cdot}} + U_r \cdot \sqrt{S_r^T}(u) \cdot \sqrt{S_r} \cdot R_r^T(d)$$

SlopeOne

$$p_{ad} = \frac{\sum_{d' \in D_q - \{d\}} \left(\sum_{x \in S_{dd'}} \frac{v_{xd} - v_{xd'}}{|S_{dd'}|} + v_{qd}\right) |S_{dd'}|}{\sum_{d' \in D_q - \{d\}} |S_{dd'}|}$$

Algorithm 2 Generates predictions p_{ad} for each document

```
Require: $q_a$ THE ORIGINAL QUERY
Require: $V$ THE RATING MATRIX
Require: $D$ A SET OF DOCUMENTS
  for all $d_i \in D \setminus \{d_1, \ldots, d_y\}$ do
    $p_{ai} \leftarrow$ CF-CALCULATION $(q_a, d_i)$
    $\overrightarrow{p_{a\cdot}}$.APPEND $(p_{ai})$
  end for
  return $\overrightarrow{p_{a\cdot}}$
```

In the user-based algorithm, we use Pearson's correlation coefficient to measure the similarity between q_a and $q \in Q$ (denoted as $s(a, q)$) as follows:

$$s(a,q) = \frac{\sum_{d \in D_a \cap D_q} (v_{ad} - \overline{v_{a\cdot}})(v_{qd} - \overline{v_{q\cdot}})}{\sqrt{\sum_{d \in D_a \cap D_q} (v_{ad} - \overline{v_{a\cdot}})^2 \sum_{d \in D_a \cap D_q} (v_{qd} - \overline{v_{q\cdot}})^2}}$$

where D_a denotes documents retrieved for q_a and D_q denotes documents retrieved for $q \in Q$. After calculating the the similarity between different queries, we calculate predictions by considering the contribution of each neighbour in the matrix, weighted by its similarity to q_a ($neigh_a$). We use the technique suggested by Herlocker et al. (2002), taking into account the mean $\overline{v_{a\cdot}}$, as well as the standard deviation σ_a and σ_q of the meaning ratings for the queries q_a and q in Q. Similarly, we define the similarity between different documents (denoted as $s(d', d)$) for the item-based algorithm as:

$$s(d',d) = \frac{\sum_{q \in Q} (v_{qd} - \overline{v_{q\cdot}})(v_{qd'} - \overline{v_{q\cdot}})}{\sum_{q \in Q} (v_{qd} - \overline{v_{q\cdot}})^2 \sum_{q \in Q} (v_{qd'} - \overline{v_{q\cdot}})^2}$$

In the weighted *SlopeOne* algorithm, $S_{dd'}$ is the set of queries that have 'rated' both documents d and d'. In the SVD algorithm [referred to as 'LSI/SVD' in Cacheda et al. (2011)], we use a matrix factorization technique that converts V into three matrices:

$$V = U \cdot S \cdot R^T$$

where U and R are orthogonal matrices, and S is a diagonal matrix of size $k \times k$ (where k is the rank of V). This matrix is iteratively reduced by discarding the smallest values, to produce a matrix S_γ with $\gamma < k$. The reconstructed matrix, $V_\gamma = U_\gamma \cdot S_\gamma \cdot R_\gamma^T$ is the best rank-γ approximation of the rating matrix V.We calculate CF predictions from this (reduced dimension) matrix using the formula stated above.

We choose these four CF algorithms because they are very popular and have produced good results (Cacheda et al. 2011). Other CF algorithms could be used instead. Whichever algorithm is used, the output of this stage will be another ranked set of documents. In the final stage of our procedure, we fuse a set of IR-generated results with the CF-generated results. This procedure is described in Algorithm 3, where we combine the documents returned for q_a by the IR engine with the vector of predictions produced in Algorithm 2. We tried a number of combinatorial methods. CombRSVNorm seems to work best in this context (see further Sect. 6.1). It is usually defined in the following way:

$$COMBRSVNORM = SUM[(RSV_i - MIN_{RSV})/(MAX_{RSV} - MIN_{RSV})]$$

where *RSV* denotes the retrieval status value (i.e., the score). Now we have *three* sets of document rankings (IR scores, CF scores and *COMBRSVNORM* scores). We sort the documents using all three scores (sort precedence as listed above, descending order) to produce a final ranking.

Algorithm 3 Fuses IR-generated and CF-generated results

```
Require: $q_a$ THE ORIGINAL QUERY
Require: $D$ A DOCUMENT SET RETRIEVED BY $q_a$
Require: $\overrightarrow{p_{a\cdot}}$ CF PREDICTIONS
  for all $d_i \in D$ do
    if $p_{ad} \in \overrightarrow{p_{a\cdot}}$ then
      COMBRSVNORM-SCORE $(d_i) \leftarrow$ COMBRSVNORM-SCORE $(p_{ad_i}$, IR-SCORE $(d_i))$
    end if
  end for
  for all $d_j$ THAT $p_{aj} > 0$ do
    $D_{final} \leftarrow D \cup d_j$
  end for
  SORT BY IR SCORE, CF SCORE, COMBRSVNORM-SCORE
  return $D_{final}$
```

3.1 Possible weaknesses of our technique

As mentioned above, CF algorithms have two known weaknesses:

[1] *Sparsity*—In a typical recommender system, most users will rate only a small subset of the available items. This means that most of the cells in the rating matrix will be empty.

[2] *Cold start*—CF algorithms struggle to generate recommendations for users recently introduced into the system.

Neither problem affects our technique. We can ensure that the matrix is sufficiently populated by manipulating the size of the top-ranked results lists (via parameter *x*). And ‘cold start’ does not occur because ‘similar’ queries are auto-generated (see below).

3.2 Query representations

In this section, we introduce the various query representations used in our technique. The first query representation, denoted *ALL*, is the *Description* field of a full-length patent application (stop words and numbers removed, terms stemmed). Note that the stop word list used is not patent-specific (unlike Becks et al. 2011; Mahdabi et al. 2011) and phrases are ignored.

The second query representation, denoted *LM*, adopts the unigram model proposed by Mahdabi et al. (2012). Applying the unigram model involves estimating the importance of each term in the patent application according to a weighted log-likelihood approach, as follows:

$$P(t|q_{LM}) = Z_t P(t|\Theta_q) \log \frac{P(t|\Theta_q)}{P(t|\Theta_C)}$$

where $Z_t = 1/\sum_t P(t|q)$ is the normalization factor. Θ_q is a model describing the query language. Θ_C is a model describing the language used in the corpus. These models are defined in the following way:

$$P(t|\Theta_q) = (1-\lambda)\cdot P_{ML}(t|d) + \lambda \cdot P_{ML}(t|C)$$

where the maximum likelihood estimation of a term t in a document d, $P_{ML}(t|d)$, is defined as $\frac{n(t,d)}{\sum_{t'} n(t',d)}$. $0<\lambda<1$ is a parameter used to control the influence of each estimation, and $n(t, d)$ is the term frequency of term t in document d.

The third query representation, denoted *LMIPC* considers *International Patent Classifications* (IPC) (Mahdabi et al. 2012). We build a relevance model Θ_{LMIPC} specifically for this purpose. The result model is defined as:

$$P(t|q_{LMIPC}) = (1-\lambda)\cdot P(t|\Theta_{LMIPC}) + \lambda \cdot P(t|q_{LM})$$

where $P(t|\Theta_{LMIPC})$ is calculated using:

$$P(t|\Theta_{LMIPC}) = \sum_{d \in LMIPC} P(t|d)\cdot P(d|\Theta_{LMIPC})$$

and

$$P(D|\Theta_{LMIPC}) = Z_d \sum_t P(t|\Theta_d) \log \frac{P(t|\Theta_{LMIPC})}{P(t|\Theta_C)}$$

where $Z_d = 1/\sum_{D \in LMIPC} P(D|\Theta_{LMIPC})$ is a document specific normalization factor. Next, we have three query representations exploiting standard IR weighting schemes, denoted *TF*, *TFIDF*, and *BM*25 respectively:

$$P(t|q_{TF}) = \frac{n(t,d)}{\max_{t'} n(t',d)}$$

$$P(t|q_{TFIDF}) = \frac{n(t,d)}{\max_{t'} n(t',d)} \cdot \log \frac{|D|}{df_t}$$

$$P(t|q_{BM25}) = \sum_t w_t \frac{(k_1+1)n(t,d)}{K+n(t,d)} \frac{(k_3+1)n(t,d)}{k_3+n(t,d)}$$

where $|D|$ is the total number of documents, df is document frequency, $w_t = \log \frac{|D|-df_t+0.5}{df_t+0.5}$ is the inverse document frequency weight of term t and $K = k_1 \cdot ((1-w_b) + w_b \cdot \frac{|d|}{avg|d|})$ includes the impact of the average document length. k_1, k_3 and w_b are free parameters (see Sect. 5.5 for the values used in this experiment). The final query representation, denoted *UFT*, is the raw text of the *Description* field minus unit frequency terms (i.e., terms which occur only once in the patent query).

We chose these query representations because they are popular and produce good results. Other query representations could be substituted. Throughout, we followed the normal procedures when calculating the top weighted terms in the patent (Becks et al. 2011, Ganguly et al. 2011; Mahdabi et al. 2012; Piroi 2010).

4 Iterative refinement

In this section, we describe a method which improves the basic performance of CPAC. This method assumes that CF-generated results and IR-generated results *mutually reinforce* one another. Updating one set of scores should iteratively propagate to the other set of scores via *pairwise document relationships* (i.e., associations created when the same document is rated by different query representations). To exploit these relationships, we adjust the CF-generated scores and IR-generated scores using a function which regularizes the smoothness of document associations over a connected graph. These document associations are easy to model within our CF framework. We construct an undirected weighted graph describing documents that are 'rated' by queries. In this graph, nodes represent documents and edges represent pairwise document relationships.

Let $G = (D, E)$ be a connected graph, wherein nodes D correspond to the $|D|$ documents rated/retrieved by different queries, and edges E correspond to the pairwise document relationships between documents. The weights on these edges are calculated using the 'ratings' assigned by queries, derived by multiplying a transpose of the ratings matrix V (V^T) with itself (V^TV). Further, assume an $n \times n$ symmetric weight matrix B on the edges of the graph, so that b_{ij} denotes the weight between documents d_i and d_j. We further define M as a diagonal matrix with entries

$$M_{ii} = \sum_j b_{ij}$$

We also define a $n \times 2$ matrix F with

$$F = \left[\overrightarrow{p_a\cdot}\overrightarrow{IR_a\cdot}\right]$$

where IR_a is a vector of IR scores retrieved by q_a and $f(a, d)$ denotes q_a's ratings assigned to d.

Thereafter, we develop a regularization framework for adjusting the CF-generated and IR-generated scores. Formally, the cost function $\Re(F, a, G)$ in a joint regularization framework is defined as:

$$\Re(F,a,G) = \frac{1}{2}\sum_{i,j=1}^{n} b_{ij}\left\|\frac{f(a,d_i)}{\sqrt{M_{ii}}} - \frac{f(a,d_j)}{\sqrt{M_{jj}}}\right\|^2 + \mu\sum_{i=1}^{n}\left\|f(a,d_i) - f^0(a,d_i)\right\|^2$$

where $\mu > 0$ is the regularisation parameter, $f^0(a,d_i)$ is the initial rating of the test query q_a and the document d_i. F and F^0 are the refined matrix and the initial matrix, respectively. The first term on the right-hand side of the cost function is the *global consistency constraint*. This constraint ensures that the weighting function does not change too much between nearby points. In this experiment, 'nearby points' are the refined rating scores reflecting the initial relationships between documents and the nearby documents. The second term on the right hand side of the cost function is the *fitting constraint*. This constraint ensures that the ratings assigned to documents fit the initial ratings. The trade-off between these two variables is controlled by the parameter μ.

Given the above, the final weighting function is defined as:

$$F^* = arg \min_{F \in \mathcal{F}} \Re(F, a, G)$$

where *arg* min stands for the argument of the minimum,[2] $\mathcal{F}$ denotes the set of $n \times 2$ matrices and $F \in \mathcal{F}$. After simplification, we can derive the following closed form solution:

$$F^* = \mu_2 (I - \mu_1 S)^{-1} F^0$$

where:

$$\mu_1 = \frac{1}{1+\mu}$$

$$\mu_2 = \frac{\mu}{1+\mu}$$

$$S = M^{-\frac{1}{2}} B M^{\frac{1}{2}}$$

and *I* is an identity matrix (see further Zhou et al. 2004; Zhu et al. 2003). Note that *S* is a normalized graph Laplacian matrix. Given the refined weighting matrix *F*, we can extract the refined $\overrightarrow{p_{a\cdot}}$ and $\overrightarrow{IR_{a\cdot}}$ scores.[3] This refinement method is described in Algorithm 4. In the iteration step of this algorithm, each node receives information from its neighbours while retaining its initial information. When *F*(*s*) converges, it is equivalent to the close form solution of F^* [refer to Zhou et al. (2004) for proof].

Algorithm 4 The iterative refinement method

```
Require: $\overrightarrow{p_{a\cdot}}$ CF PREDICTIONS
Require: $\overrightarrow{IR_{a\cdot}}$ THE IR SCORE VECTOR
Require: $V$ THE RATING MATRIX
Require: $z$ THE NUMBER OF ITERATIONS REQUIRED
  $B \leftarrow V^T V$
  SET $M$ AS DIAGONAL MATRIX WITH ENTRIES $M_{ii} = \sum_j b_{ij}$
  $S \leftarrow M^{-\frac{1}{2}} B M^{\frac{1}{2}}$
  $F^0 \leftarrow \left[\overrightarrow{p_{a\cdot}}^0 \overrightarrow{IR_{a\cdot}}^0\right]$
  for all $s = 1$ TO $z$ do
    $F(s+1) = \mu S F(s) + (1-\mu) F^0$
  end for
  return $\overrightarrow{p_{a\cdot}}^{s+1}, \overrightarrow{IR_{a\cdot}}^{s+1}$
```

5 Evaluation

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

[2] The set of points of the given argument for which the given function attains its minimum value.

[3] It is worth noting that μ_2 could be eliminated as it does not change the ranking.

[1] Does our technique outperform state-of-art patent IR systems?
[2] How effective is the refinement method described in Sect. 4?
[3] Which query representation is most effective?
[4] Which collaborative filtering algorithm performs the best?

5.1 Experimental data

The text corpus used in our evaluation was built using the CLEF-IP 2011 test collection. This collection contains 3.5 million XML-encoded patent documents, relating to approximately 1.5 million individual patents.[4] These documents were extracted from the MAREC data corpus.[5] We used the CLEF-IP 2011 query set, which contains 1,351 topics (English subset). Each topic is a patent application comprising several fields. We built all queries using the *Description* field. Prior to indexing and retrieval, a suffix stemmer (Porter 1997) and a stop word list[6] were applied to all documents and queries. We also removed all numbers. Citation information was ignored. Relevance judgements were produced by CLEF campaign organizers. Judgements were extracted from published search reports.

5.2 Evaluation metrics

We used the following evaluation metrics in this experiment:

- The precision computed after 10, 50 and 100 documents were retrieved (P@10, P@50 and P@100)
- Normalized discounted cumulative gain (NDCG; Järvelin and Kekäläinen 2000)
- The recall computed after 10, 50 and 100 documents were retrieved (R@10, R@50 and R@100)
- Mean average precision (MAP).

Unless otherwise stated, results indicate *average performance* across all topics. Statistically-significant differences in performance were determined using a paired *t* test at a confidence level of 95 %.

5.3 Retrieval systems

All information retrieval functions in our experiment were handled by the Terrier open source platform (Ounis et al. 2006).[7] We used the BM25 retrieval model as it achieved (slightly) better results during set-up.

5.4 Baseline systems

We used a number of baseline systems to evaluate our technique. The first seven baseline systems relate to the seven query representations described in Sect. 3.2 (i.e., we used each query representation *in isolation* as a performance baseline). We also used the phase-based

4 A patent document can be a patent application, a search report, or a patent grant.

5 MAREC is a collection of 19 million patent documents available from http://ifs.tuwien.ac.at/~clef-ip/marec.shtml.

6 ftp://ftp.cs.cornell.edu/pub/smart/.

7 http://terrier.org/.

model described in Mahdabi et al. (2012), denoted *LMIPCNP*, and the query reduction method presented in Ganguly et al. (2011), denoted *QR*. *LMIPCNP* extends the *LMIPC* method, adding key phrases with similar semantics to the patent query. These phrases are extracted using the noun phrase patterns defined in Mahdabi et al. (2012). *QR* reduces a patent query by comparing segments of that query to top ranked documents using Language Models. The least similar segments are subsequently removed (Ganguly et al. 2011; Mahdabi et al. 2012). To measure the effectiveness of PRF in this context, we also carried out two retrieval runs using pseudo-relevance feedback (denoted *ALLPRF* and *UFTPRF*). We used the implementations provided by the Terrier platform for these two baselines (see further Robertson 1991).

5.5 Parameter settings

We used the training topics provided by the CLEF-IP 2011 organizers to empirically set all of the parameters used in this experiment, including those used by the baseline systems.[8] First, we set the parameters for our own method. We conducted a number of runs with different values for x (i.e., the number of top-ranked documents used when generating ratings). As shown in Fig. 1, there were no significant changes in MAP scores between 15 and 50 documents. The optimal value was obtained when $x = 10$. This is a relatively low figure, but it is consistent with the search environment. In patent search, precision is typically much lower than web search, often dropping off fairly quickly. Too many documents (e.g., $x = 100$) and the noise disrupts our technique. Too few (e.g., $x = 5$) and we miss relevant documents. The parameter μ was set to 0.99, consistent with prior work (Zhou et al. 2004, 2012).

The parameters for the baseline systems were set as described in Mahdabi et al. (2012) and Ganguly et al. (2011). Parameters $k_1,\ w_b,\ k_3$ (part of the *BM*25 model) were set to 1.2, 0.75 and 7 respectively. The number of phrases (used in *LMIPCNP*) was set to 10. The number of pseudo-relevant documents (used in *QR*) was set to 20. We used the training set to fix the number of query terms for *TF*, *TFIDF*, *BM*25, *LM* and *LMIPC*. We tried every number in the range $\{5, 10, 15, 20, \ldots 100\}$ [following a study placing the effective upper bound at 100 (Xue and Croft 2009)]. We found that 50 query terms was the most effective.

6 Results

In our first evaluation, we compared the performance of our technique with the baseline systems listed in Sect. 5.4. The results are shown in Table 1. Our multiple query technique, *CPAC*, performed extremely well. It achieved statistically significant improvements over the top performing baselines, including *ALL*, *QR* and *LMIPCNP*. Notably, it scored a 19.32 % improvement in MAP over *LM* (Mahdabi et al. 2012). These results support our assertion that multiple query representations are more effective than single query representations in patent search.

As shown in Table 1, our refinement method (*CPACRegu*) recorded statistically significant improvements over *CPAC*. In terms of MAP, *CPACRegu* scored 4.18 % higher

[8] CLEF-IP 2011 training data consists of 100 English topics distinct from the test topics.

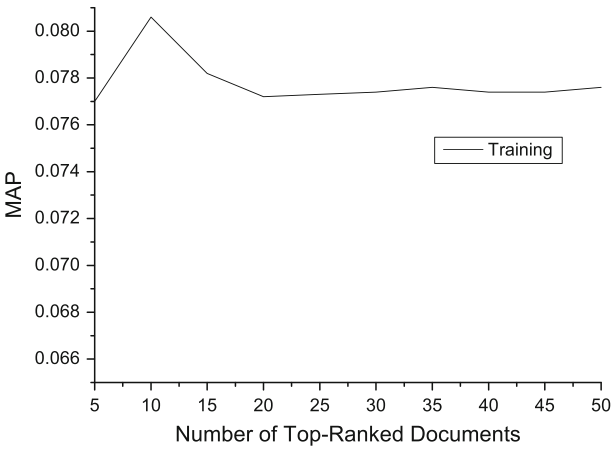


Fig. 1 Varying the number of *top* ranked documents (parameter *x*)

Table 1 Precision of collaborative patent search and various baselines

	MAP	NDCG	P@10	P@50	P@100
ALL	0.0925	0.2385	0.0856	0.0348	0.0213
*BM*25	0.0729	0.1967	0.0705	0.0266	0.0168
LM	0.0823	0.2228	0.0806	0.0304	0.0195
LMIPC	0.0847	0.2264	0.0788	0.0318	0.0201
TF	0.0645	0.1831	0.0600	0.0243	0.0156
TFIDF	0.0821	0.2243	0.0813	0.0311	0.0197
UFT	0.0833	0.2230	0.0797	0.0313	0.0198
LMIPCNP	0.0851	0.2283	0.0798	0.0316	0.0199
QR	0.0913	0.2388	0.0877	0.0356	0.0217
CPAC	0.0982	0.2476	0.0919	0.0361	0.0219
CPACRegu	0.1023	0.2635	0.103	0.0388	0.0239

than *CPAC*, and improved *ALL* (the highest scoring baseline) by 10.59 % . A similar trend emerged in terms of NDCG, where *CPACRegu* exceeded *CPAC* by 6.42 % and *ALL* by 10.48 %. The performance of the *CPACRegu* method measured by P@10, P@50 and P@100 was particularly strong, showing improvements w.r.t *ALL* of 20.33, 11.49 and 12.21 % respectively. These results support our earlier claim that CF-generated results and IR-generated results mutually reinforce each other.

Figure 2, which plots the precision-recall curves[9] for the various systems, suggests that the gains achieved using our methods are consistent. Interestingly, the refined version of our technique outscores the baselines on almost all of the evaluation metrics, despite being specifically tuned for MAP. Comparing our system to the results published for CLEF-IP 2011, we note that *CPAC* is only fractionally lower than the best performing run, while *CPACRegu* outperforms it completely (Piroi et al. 2011). To summarise, the results described above indicate that our technique is extremely suitable for patent prior-art search, and that it is capable of state-of-the-art performance.

[9] This figure shows the interpolated precision at a certain recall level, see further (Manning et al. 2008).

6.1 Comparison with standard data fusion

In this section, by way of comparison, we examine an alternative approach to multiple query patent search that does *not* use CF-based analysis. In this approach, we create multiple query representations of each patent application as described above. Then we submit these queries to a standard IR engine, combining the search results using conventional data fusion algorithms (see Sect. 3). We wanted to know if we could outperform CPAC using this simpler technique.

The first task was to determine which *type* of data fusion algorithm to use (i.e., supervised or unsupervised). We evaluated 5 unsupervised methods and one supervised method (see Tables 2, 3) using a subset of the CLEF-IP 2011 English query set (675 topics). We calculated *CombMNZ* (an unsupervised method) by multiplying the sum of the scores for a document by the number of lists that contained that document. To apply the supervised technique (*LAMBDAMERGE*) we split the query set into two subsets (training and testing), selecting gating features appropriate to our query representations (Sheldon et al. 2011).

Figure 3 compares the retrieval performance of the algorithms. *CombSUM* and *CombMNZ* were the lowest scoring techniques. Interestingly, the supervised technique, *LAMBDAMERGE*, was outperformed by two unsupervised methods (*CombRSVNorm* and *CombRSV*). This unexpected result may be due to the diverse query representations used in our study, which produced highly dissimilar result sets. This was a challenging environment for data fusion algorithms, one which clearly did not suit *LAMBDAMERGE*. Figures 4 and 5 report the results when we compared our CF-based technique to the

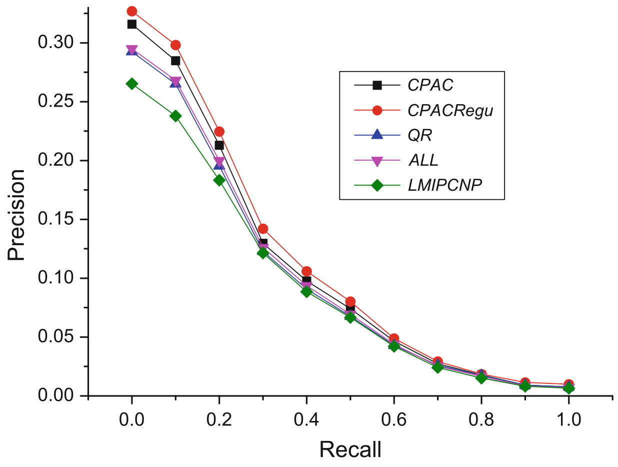


Fig. 2 Precision-Recall curves for the top performing systems

Table 2 A summary of six data fusion methods

CombMNZ	$SUM(RSV_i)$*(# of nonzero (RSV_i))
CombSUM	$SUM(RSV_i)$
CombRSV	$SUM(RSV_i/MAX_{RSV})$
CombRSVNorm	$SUM[(RSV_i - MIN_{RSV})/(MAX_{RSV} - MIN_{RSV})]$
CORI	As described in Callan et al. (1995)
LAMBDAMERGE	As described in Sheldon et al. (2011)

unsupervised data fusion algorithms (entire query set). *CPAC* and *CPACRegu* outperformed the top scoring fusion algorithm *CombRSVNorm* with statistically significant results. This finding confirms our suspicions. Our CF-based technique produces results that we cannot replicate with simple data fusion.

6.2 Baseline systems

In this section, we evaluate the performance of the baseline systems used in our experiment. Overall, the best performing baseline was *ALL* (i.e., the entire pre-processed

Table 3 Recall of collaborative patent search and various baselines

	R@10	R@50	R@100
ALL	0.1192	0.2374	0.2868
BM25	0.0990	0.1810	0.2267
LM	0.1128	0.2078	0.2608
LMIPC	0.1122	0.2169	0.2694
TF	0.0852	0.1676	0.2091
TFIDF	0.1122	0.2112	0.2647
UFT	0.1133	0.2114	0.2670
LMIPCNP	0.1153	0.2166	0.2694
QR	0.1209	0.2426	0.2926
CPAC	0.1292	0.2492	0.2978
CPACRegu	0.1383	0.2623	0.3014

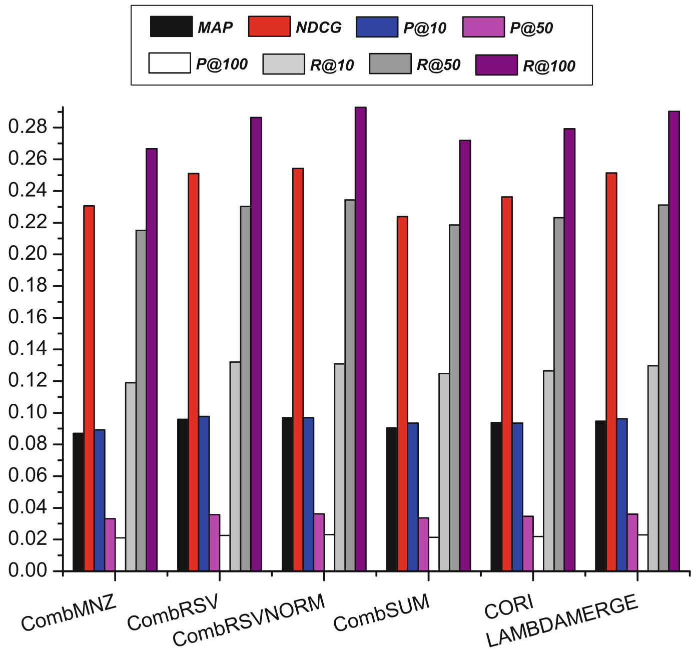


Fig. 3 Comparison of supervised and unsupervised data fusion methods (using a subset of CLEF-IP 2011 query set)

Description field). This finding is consistent with work published at CLEF 2010 and CLEF 2011. *UFT* also performed well, probably because it closely resembles *ALL*. Filtering out the unit frequency terms from the *Description* field leaves most of the original terms intact. A similar effect was observed in the *QR* run.

We found that *TF* produced the worst performance on the CLEF-IP test collections. This result conflicts with previous work showing positive results on the USPTO corpus (Xue and Croft 2009). These results are possibly due to the citation practices common to that corpus. As expected, the effect of pseudo-relevance feedback on the top performing baselines was negative (see Figs. 5, 6). The use of IPC information improved overall performance (i.e., *LMIPC* scored better than *LM*). Consistent with Becks et al. (2011),

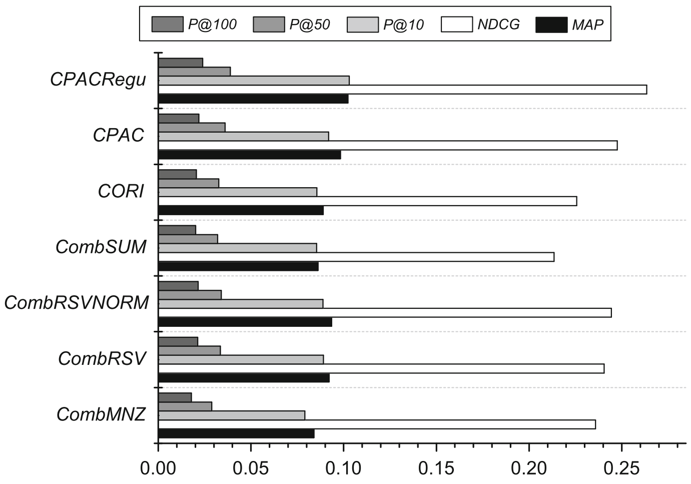


Fig. 4 Precision of collaborative patent search and unsupervised data fusion algorithms (entire query set)

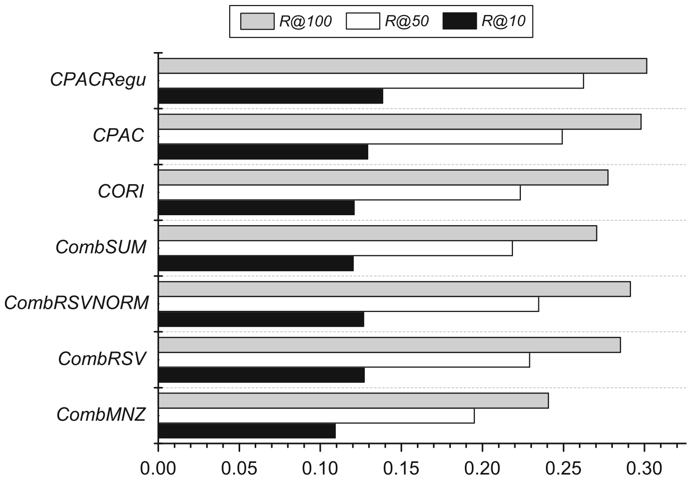


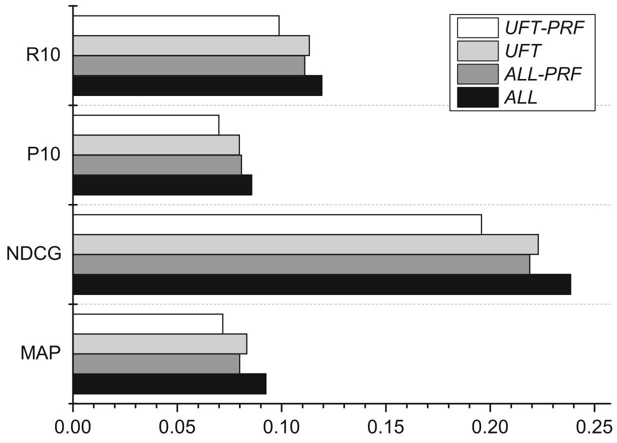
Fig. 5 Recall of collaborative patent search and unsupervised data fusion algorithms (entire query set)

adding phrases (*LMIPCNP*) led to a modest (i.e., not statistically significant) improvement in retrieval effectiveness.

6.3 CF algorithms

We studied the performance of the different CF algorithms. The results are shown in Fig. 7. The model-based CF algorithms (SVD and Weighted *SlopeOne*) produced marginally better results than the memory-based alternatives (User-based and Item-based). Memory-based CF algorithms tend to perform poorly when the rating matrix is sparse. Model-based algorithms are generally less sensitive. All of the CF algorithms produced equivalent results. This supports our assertion that collaborative patent search is less sensitive to the problem of matrix sparsity.

Fig. 6 The effect of PRF on the *top*-performing baselines



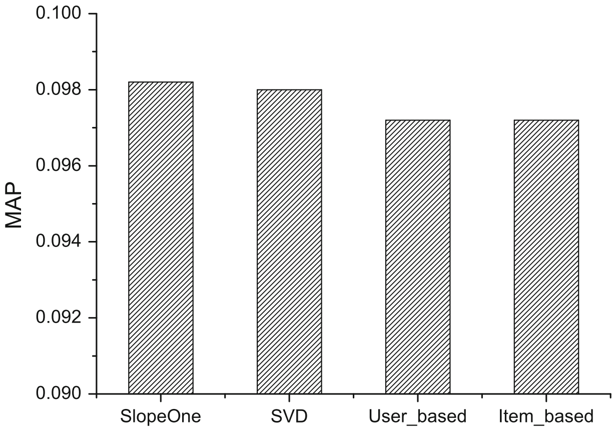


Fig. 7 The effect of changing the CF algorithm

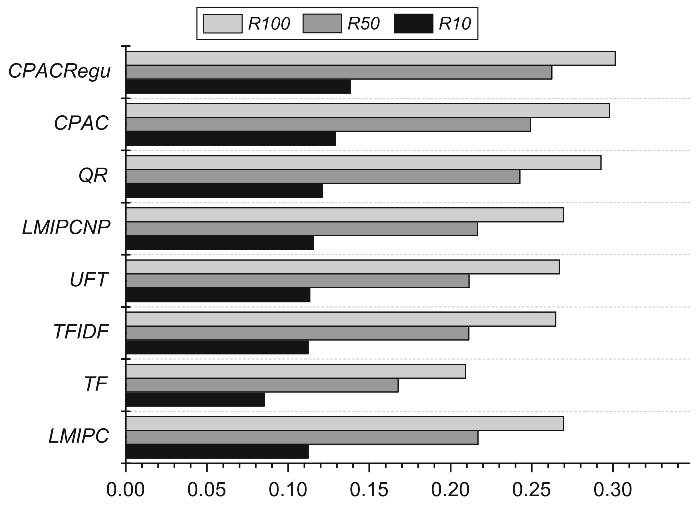


Fig. 8 Recall of collaborative patent search and various baselines

6.4 Per-query analysis

We performed a per-query analysis comparing the results produced by *ALL* and *CPACRegu*. We found that 61 % of all queries (824 out of 1,351) benefited from our refined, multiple query representations technique. Only 18.7 % of queries (252 out of 1351) were better off with a single query representations of the *Description* field.

6.5 Recall

In addition to the precision-based measurements described above, we evaluated our algorithms using recall-based metrics. Given the context, this is quite fitting. Patent prior-art search is a recall-oriented task wherein the primary focus is to retrieve relevant documents at early ranks. We found that *CPAC* and *CPACRegu* achieved better recall than the baseline systems. These improvements were quite stable across all evaluation metrics (see Fig. 8; Table 2). The high recall performance of our technique has an intuitive explanation. Relevant documents are being crowd-sourced (i.e., 'found' by other query representations).

7 Conclusion and further work

In this paper, we have described a pseudo-collaborative approach to patent IR which combines results lists from multiple query representations. We have also proposed an iterative method for refining its performance. In a multi-stage evaluation using CLEF-IP data, our experimental system delivered statistically significant improvements over state-of-the-art baseline systems. In future work, we intend to explore the differences between IP test collections. We also plan to evaluate the use of citation information alongside more selective query generation techniques. Further scrutiny of data fusion algorithms, and their application to our technique, is another obvious extension.

Acknowledgments The work described in this paper was supported by the National Natural Science Foundation of China under Grant No. 61300129 and No. 61272063, the Excellent Youth Foundation of Hunan Scientific Committee No. 11JJ1011, and partially supported by the Hunan Provincial Natural Science Foundation of China No. 12JJ3059, the Science Foundation Ireland No. 07/CE/I1142 (Centre for Next Generation Localisation). The authors would like to thank the anonymous reviewers for their contribution to the paper.

Reference

Aggarwal, C. C., Wolf, J. L., Wu, K. L., & Yu, P. S. (1999). Horting hatches an egg: A new graph-theoretic approach to collaborative filtering. In *Proceedings of the fifth ACM SIGKDD international conference on knowledge discovery and data mining, KDD '99* (pp. 201–212). New York, NY, USA: ACM.

Becks, D., Eibl, M., Jürgens, J., Kürsten, J., Wilhelm, T., & Womser-Hacker, C. (2011). Does patent IR profit from linguistics or maximum query length? In *CLEF* (Notebook Papers/LABs/Workshops).

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 Transactions on the Web, 5*(1), 2:1–2:33

Callan, J. P., Lu, Z., & Croft, W. B. (1995). Searching distributed collections with inference networks. In *Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '95* (pp. 21–28). New York, NY, USA: ACM.

Cao, B., Shen, D., Wang, K., & Yang, Q. (2010). Clickthrough log analysis by collaborative ranking. In *Proceedings of the twenty-fourth AAAI conference on artificial intelligence, AAAI 2010*, Atlanta, Georgia, USA, July 11–15, 2010, pp. 224–229. AAAI Press.

Ganguly, D., Leveling, J., Magdy, W., & Jones, G. J. (2011). Patent query reduction using pseudo relevance feedback. In *Proceedings of the 20th ACM international conference on Information and knowledge management, CIKM '11* (pp. 1953–1956). New York, NY, USA: ACM.

Herlocker, J., Konstan, J. A., & Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval, 5*(4), 287–310

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

Huang, Z., Chen, H., & Zeng, D. (2004). Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems, 22*(1), 116–142.

Itoh, H., Mano, H., & Ogawa, Y. (2003). Term distillation in patent retrieval. In *Proceedings of the ACL-2003 workshop on patent corpus processing—volume 20, PATENT '03* (pp. 41–45). Association for Computational Linguistics, Stroudsburg, PA, USA.

Iwayama, M., Fujii, A., Kando, N., & Takano, A. (2003). Overview of patent retrieval task at NTCIR-3. In *Proceedings of the ACL-2003 workshop on patent corpus processing—volume 20, PATENT '03* (pp. 24–32). Association for Computational Linguistics, Stroudsburg, PA, USA.

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.

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.

Lopez, P., & Romary, L. (2010). Experiments with citation mining and key-term extraction for prior art search. In *CLEF* (Notebook Papers/LABs/Workshops).

Luo, H., Niu, C., Shen, R., & Ullrich, C. (2008). A collaborative filtering framework based on both local user similarity and global user similarity. *Machine Learning, 72*(3), 231–245

Magdy, W., & Jones, G. J. F. (2010). Applying the KISS principle for the CLEF-IP 2010 prior art candidate patent search task. In *CLEF* (Notebook Papers/LABs/Workshops).

Magdy, W., Lopez, P., & Jones, G. J. F. (2011). Simple versus sophisticated approaches for patent prior-art search. In *Proceedings of the 33rd European conference on advances in information retrieval, ECIR'11* (pp. 725–728). Berlin, Heidelberg: Springer.

Mahdabi, P., Andersson, L., Hanbury, A., & Crestani, F. (2011). Report on the CLEF-IP 2011 experiments: Exploring patent summarization. In *CLEF* (Notebook Papers/Labs/Workshop).

Mahdabi, P., Andersson, L., Keikha, M., & Crestani, F. (2012). Automatic refinement of patent queries using concept importance predictors. In *Proceedings of the 35th international ACM SIGIR conference on research and development in information retrieval, SIGIR '12* (pp. 505–514). New York, NY, USA: ACM.

Mahdabi, P., Keikha, M., Gerani, S., Landoni, M., & Crestani, F. (2011). Building queries for prior-art search. In *IRFC*, pp. 3–15.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. New York, NY, USA: Cambridge University Press.

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)*.

Piroi, F. (2010). Clef-ip 2010: Retrieval experiments in the intellectual property domain. In *CLEF* (Notebook Papers/LABs/Workshops).

Piroi, F., Lupu, M., Hanbury, A., & Zenz, V. (2011). CLEF-IP 2011: Retrieval in the intellectual property domain. In *CLEF* (Notebook Papers/Labs/Workshop).

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

Powell, A. L., French, J. C., Callan, J., Connell, M., & Viles, C. L. (2000). The impact of database selection on distributed searching. In *Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '00* (pp. 232–239). New York, NY, USA: ACM.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). 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. (1991). On term selection for query expansion. *Journal of Documentation, 46*(4), 359–364.

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.

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. *ACM Transaction on Asian Language Information Processing, 4*(2), 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.

Shaw, J. A., & Fox, E. A. (1994). Combination of multiple searches. In *Proceedings of TREC-3*, pp. 243–252.

Sheldon, D., Shokouhi, M., Szummer, M., & Craswell, N. (2011). LambdaMerge: merging the results of query reformulations. In *Proceedings of the fourth ACM International conference on Web search and data mining, WSDM '11* (pp. 795–804). New York, NY, USA: ACM.

Si, L., & Callan, J. (2005). Modeling search engine effectiveness for federated search. In *Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '05* (pp. 83–90). New York, NY, USA: ACM.

Tait, J. (Ed.). (2008). Proceedings of the 1st ACM workshop on patent information retrieval, PaIR 2008, Napa Valley, California, USA, October 30, 2008. ACM.

Tsai, M. F., Wang, Y. T., & Chen, H. H. (2008). A study of learning a merge model for multilingual information retrieval. In *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval—SIGIR '08* (pp. 195–202). New York, New York, USA: ACM Press.

Wang, F., Ma, S., Yang, L., & Li, T. (2006). Recommendation on item graphs. In *Proceedings of the sixth international conference on data mining, ICDM '06* (pp. 1119–1123). IEEE Computer Society, Washington, DC, USA.

Weimer, M., Karatzoglou, A., Le, Q., & Smola, A. (2007). CofiRank—maximum margin matrix factorization for collaborative ranking. In *Advances in neural information processing systems*, vol. $\sim$20, pp. 1593–1600.

Xue, X., & Croft, W. B. (2009). Transforming patents into prior-art queries. In *Proceedings of the 32nd international ACM SIGIR conference on research and development in information retrieval, SIGIR '09* (pp. 808–809). New York, NY, USA: ACM.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., & Scholkopf, B. (2004). Learning with local and global consistency. In *Advances in neural information processing systems*, vol. 16, pp. 321–328.

Zhou, D., Lawless, S., & Wade, V. (2012). Improving search via personalized query expansion using social media. *Information Retrieval, 15*(3–4), 218–242.

Zhou, D., Truran, M., Liu, J., & Zhang, S. (2013). Collaborative pseudo-relevance feedback. *Expert Systems with Applications, 40*(17), 6805–6812.

Zhu, X., Ghahramani, Z., & Lafferty, J. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *The twentieth international conference on machine learning*, August 21–24, 2003, Washington, DC USA, pp. 912–919.