

Jun Wang · Stephen Robertson · Arjen P. de Vries ·

Marcel J. T. Reinders

# Probabilistic Relevance Ranking for Collaborative Filtering

**Abstract** Collaborative filtering is concerned with making recommendations about items to users. Most formulations of the problem are specifically designed for predicting user ratings, assuming past data of explicit user ratings is available. However, in practice we may only have implicit evidence of user preference; and furthermore, a better view of the task is of generating a top-N list of items that the user is most likely to like. In this regard, we argue that collaborative filtering can be directly cast as a *relevance* ranking problem. We begin with the classic Probability Ranking Principle of information retrieval, proposing a probabilistic item ranking framework. In the framework, we derive two different ranking models, showing that despite their common origin, different factorizations reflect two distinctive ways to approach item ranking. For the model estimations, we limit our discussions to implicit user preference data, and adopt an approximation method introduced in the classic text retrieval model (i.e. the Okapi BM25 formula) to effectively decouple frequency counts and presence/absence counts

---

J. Wang

University College London, The United Kingdom, E-mail: j.wang@adastral.ucl.ac.uk

S. Robertson

Microsoft Research, Cambridge, The United Kingdom, E-mail: ser@microsoft.com

A. P. de Vries

CWI, The Netherlands, E-mail: arjen@acm.org

M.J.T. Reinders

Delft University of Technology, The Netherlands, E-mail: m.j.t.reinders@tudelft.nl

in the preference data. Furthermore, we extend the basic formula by proposing the Bayesian inference to estimate the probability of relevance (and non-relevance), which largely alleviates the data sparsity problem. Apart from a theoretical contribution, our experiments on real data sets demonstrate that the proposed methods perform significantly better than other strong baselines.

## 1 Introduction

Collaborative filtering aims at identifying interesting information items (e.g. movies, books, websites) for a set of users, given their user profiles. Different from its counterpart, content-based filtering [3], it utilizes other users' preferences to perform predictions, thus making direct analysis of content features unnecessary.

User profiles can be explicitly obtained by asking users to rate items that they know. However these explicit ratings are hard to gather in a real system [7]. It is highly desirable to infer user preferences from implicit observations of user interactions with a system. These implicit interest functions usually generate frequency-counted profiles, like “playback times of a music file”, or “visiting frequency of a web-site” etc.

So far, academic research into frequency-counted user profiles for collaborative filtering has been limited. A large body of research work for collaborative filtering by default focuses on rating-based user profiles [1,20]. Research started with memory-based approaches to collaborative filtering [14,28,35,37] and lately came with model-based approaches [15,20,17].

In spite of the fact that these rating-based collaborative filtering algorithms lay a solid foundation for collaborative filtering research, they are specifically designed for rating prediction, making them difficult to apply in many real situations where frequency-counted user profiling is demanded. Most importantly, the purpose of a recommender system is to suggest to a user items that he or she might be interested in. The user decision on whether accepting a suggestion (i.e. to review or listen to a suggested item) is a binary one. As already demonstrated in [10,21], directly using predicted ratings as ranking scores may not accurately model this common scenario.

This motivated us to conduct a formal study on probabilistic item ranking for collaborative filtering. We start with the Probability Ranking Principle of information retrieval [24] and introduce the concept

---

of “binary relevance” into collaborative filtering. We directly model how likely an item might be relevant to a given user (profile), and for the given user we aim at presenting a list of items in rank order of their predicted relevance. To achieve this, we first establish an item ranking framework by employing the log-odd ratio of relevance and then derive two ranking models from it, namely an *item-based relevance model* and *user-based relevance model*. We then draw an analogy between the classic text retrieval model [27] and our models, effectively decoupling the estimations of frequency counts and (non-)relevance counts from implicit user preference data. Because data sparsity makes the probability estimations less reliable, we finally extend the basic log-odd ratio of relevance by viewing the probabilities of relevance and non-relevance in the models as parameters and apply the Bayesian inference to enforce different prior knowledge and smoothing into the probability estimations. This proves to be effective in two real data sets.

The remainder of the paper is organized as follows. We first describe related work and establish the log-odd ratio of relevance ranking for collaborative filtering. The resulting two different ranking models are then derived and discussed. After that, we provide an empirical evaluation of the recommendation performance and the impact of the parameters of our two models, and finally conclude our work.

## 2 Related Work

### 2.1 Rating Prediction

In the memory-based approaches, all rating examples are stored *as-is* into memory (in contrast to learning an abstraction), forming a heuristic implementation of the “Word of Mouth” phenomenon. In the rating prediction phase, similar users or (and) items are sorted based on the memorized ratings. Relying on the ratings of these similar users or (and) items, a prediction of an item rating for a test user can be generated. Examples of memory-based collaborative filtering include user-based methods [5, 14, 23], item-based methods [10, 28] and unified methods [34, 35]. The advantage of the memory-based methods over their model-based alternatives is that less parameters have to be tuned; however, the data sparsity problem is not handled in a principled manner.

In the model-based approaches, training examples are used to generate an “abstraction” (model) that is able to predict the ratings for items that a test user has not rated before. In this regard, many

probabilistic models have been proposed. For example, to consider user correlation, [22] proposed a method called personality diagnosis (PD), treating each user as a separate cluster and assuming a Gaussian noise applied to all ratings. It computes the probability that a test user is of the same “personality type” as other users and, in turn, the probability of his or her rating to a test item can be predicted. On the other hand, to model item correlation, [5] utilizes a Bayesian Network model, in which the conditional probabilities between items are maintained. Some researchers have tried mixture models, explicitly assuming some hidden variables embedded in the rating data. Examples include the aspect models [15,17], the cluster model [5] and the latent factor model [6]. These methods require some assumptions about the underlying data structures and the resulting ‘compact’ models solve the data sparsity problem to a certain extent. However, the need to tune an often significant number of parameters has prevented these methods from practical usage. For instance, in the aspect models [15, 17], an EM iteration (called “fold-in”) is usually required to find both the hidden user clusters or/and hidden item clusters for any new user.

## 2.2 Item Ranking

Memory-based approaches are commonly used for rating prediction, but they can be easily extended for the purpose of item ranking. For instance, a ranking score for a target item can be calculated by a summation over its similarity towards other items that the target user liked (i.e. in the user preference list). Taking this item-based view, we formally have the following basic ranking score:

$$o_{u_k}(i_m) = \sum_{i_{m'} \in L_{u_k}} s_I(i_{m'}, i_m) \quad (1)$$

where  $u_k$  and  $i_m$  denote the target user and item respectively, and  $i_{m'} \in L_{u_k}$  denotes any item in the preference list of user  $u_k$ .  $s_I$  is the similarity measure between two items, and in practice cosine similarity and Pearson’s correlation are generally employed. To specifically target the item ranking problem, researchers in [10] proposed an alternative, TFxIDF-like similarity measure, which is shown as follows:

$$s_I(i_{m'}, i_m) = \frac{Freq(i_{m'}, i_m)}{Freq(i_{m'}) \times Freq(i_m)^\alpha} \quad (2)$$

where  $Freq$  denotes the frequency counts of an item  $Freq(i_{m'})$  or co-occurrence counts for two items  $Freq(i_{m'}, i_m)$ .  $\alpha$  is a free parameter, taking a value between 0 and 1. On the basis of empirical observations, they also introduced two normalization methods to further improve the ranking.

In [33], we proposed a language modelling approach for the item ranking problem in collaborative filtering. The idea is to view an item (or its presence in a user profile) as the output of a generative process associated with each user profile. Using a linear smoothing technique [39], we have the following ranking formula:

$$o_{u_k}(i_m) = \sum_{i_{m'} \in L_{u_k}} \ln \left( \lambda P(i_{m'} | i_m) + (1 - \lambda) P(i_{m'}) \right) + \ln P(i_m) \quad (3)$$

where the ranking score of a target item is essentially a combination of its popularity (expressed by the prior probability  $P(i_m)$ ) and its co-occurrence with the items in the preference list of the target user (expressed by the conditional probability  $P(i_{m'} | i_m)$ ).  $\lambda \in [0, 1]$  is used as a linear smoothing parameter to further smooth the conditional probability from a background model ( $P(i_{m'})$ ).

Nevertheless, our formulations in [33] only take the information about presence/absence of items into account when modelling implicit user preference data, completely ignoring other useful information such as frequency counts (i.e. the number of visiting/playing times). We shall see that the probabilistic relevance framework proposed in this paper effectively extends the language modelling approaches of collaborative filtering. It not only allows us to make use of frequency counts for modelling implicit user preferences but has room to model non-relevance in a formal way. They prove to be crucial to the accuracy of recommendation in our experiments.

### 3 A Probabilistic Relevance Ranking Framework

The task of information retrieval aims to rank documents on the basis of their relevance (usefulness) towards a given user need (query). The Probability Ranking Principle (PRP) of information retrieval [24] implies that ranking documents in descending order by their probability of relevance produces optimal performance under a “reasonable” assumption, i.e. the relevance of a document to a user information need is independent of other documents in the collection [32].

By the same token, our task for collaborative filtering is to find items that are relevant (useful) to a given user interest (implicitly indicated by a user profile). The PRP applies directly when we view a

user profile as a query to rank items accordingly. Hereto, we introduce the concept of “relevancy” into collaborative filtering. By analogy with the relevance models in text retrieval [19,26,31], the top- $N$  recommendation items can be then generated by ranking items in order of their probability of relevance to a user profile or the underlying user interest.

To estimate the probability of relevance between an item and a user (profile), let us first define a sample space of relevance:  $\Phi_R$  and let  $R$  be a random variable over the relevance space  $\Phi_R$ .  $R$  is either ‘relevant’  $r$  or ‘non-relevant’  $\bar{r}$ . Secondly, let  $U$  be a discrete random variable over the sample space of *user id*’s:  $\Phi_U = \{u_1, \dots, u_K\}$  and let  $I$  be a random variable over the sample space of *item id*’s:  $\Phi_I = \{i_1, \dots, i_M\}$ , where  $K$  is the number of users and  $M$  the number of items in the collection. In other words,  $U$  refers to the user identifiers and  $I$  refers to the item identifiers.

We then denote  $P$  as a probability function on the joint sample space  $\Phi_U \times \Phi_I \times \Phi_R$ . The PRP now states that we can solve the ranking problem by estimating the probability of relevance  $P(R = r|U, I)$  and non-relevance  $P(R = \bar{r}|U, I)$ . The relevance ranking of items in the collection  $\Phi_I$  for a given user  $U = u_k$  can be formulated as the log odds of the relevance:

$$o_{u_k}(i_m) = \ln \frac{P(r|u_k, i_m)}{P(\bar{r}|u_k, i_m)} \quad (4)$$

For simplicity, the propositions  $R = r$ ,  $R = \bar{r}$ ,  $U = u_k$  and  $I = i_m$  are denoted as  $r$ ,  $\bar{r}$ ,  $u_k$ , and  $i_m$ , respectively.

### 3.1 Item-Based Relevance Model

Two different models can be derived if we apply the Bayes’ rule differently. This section introduces the item-based relevance model, leaving the derivations of the user-based relevance model in Section 3.2.

By factorizing  $P(\bullet|u_k, i_m)$  with  $P(u_k|i_m, \bullet)P(\bullet|i_m)/P(u_k|i_m)$ , the following log-odds ratio can be obtained from Eq. 4:

$$o_{u_k}(i_m) = \ln \frac{P(u_k|i_m, r)}{P(u_k|i_m, \bar{r})} + \ln \frac{P(r|i_m)}{P(\bar{r}|i_m)} \quad (5)$$

Notice that, in the ranking model shown in Eq. 5, the target user is defined in the user *id* space. For a given new user, we do not have any observations about his or her relevancy towards an unknown item. This makes the probability estimations unsolvable. In this regard, we need to build a feature

representation of a new user by his or her user profile so as to relate the user to other users that have been observed from the whole collection.

This paper considers implicit user profiling: user profiles are obtained by implicitly observing user behavior, for example, the web sites visited, the music files played etc., and a user is represented by his or her preferences towards all the items. More formally, we treat a user (profile) as a vector over the entire item space, which is denoted as a bold letter  $\mathbf{l} := (l^1, \dots, l^{m'}, \dots, l^M)$ , where  $l^{m'}$  denotes an item frequency count, e.g., number of times a user played or visited item  $i_{m'}$ . Note that we deliberately used the item index  $m'$  for the items in the user profile, as opposed to the target item index  $m$ . For each user  $u_k$ , the user profile vector is instantiated (denoted as  $\mathbf{l}_k$ ) by assigning *this* user's item frequency counts to it:  $l^{m'} = c_k^{m'}$ , where  $c_k^{m'} \in \{0, 1, 2, \dots\}$  denotes number of times the user  $u_k$  played or visited item  $i_{m'}$ . Changing the user presentation from Eq. 5, we have the following:

$$o_{u_k}(i_m) = \ln \frac{P(\mathbf{l}_k | i_m, r)}{P(\mathbf{l}_k | i_m, \bar{r})} + \ln \frac{P(r | i_m)}{P(\bar{r} | i_m)} = \sum_{\forall m'} \ln \frac{P(l^{m'} = c_k^{m'} | i_m, r)}{P(l^{m'} = c_k^{m'} | i_m, \bar{r})} + \ln \frac{P(r | i_m)}{P(\bar{r} | i_m)} \quad (6)$$

where we have assumed frequency counts of items in the target user profile are conditionally independent, given relevance or non-relevance<sup>1</sup>. Although this conditional independent assumption does not hold in many real situations, it has been empirically shown to be a competitive approach (e.g., in text classification [11]). It is worthwhile noticing that we only ignore the item dependency in the profile of the *target* user, while for all *other* users, we do consider their dependence. In fact, how to utilise the correlations between items is crucial to the item-based approach.

For the sake of computational convenience, we intend to focus on the items ( $i_{m'}$ , where  $m' \in \{1, M\}$ ) that are present in the target user profile ( $c_k^{m'} > 0$ ). By splitting items in the user profile into two groups, i.e. presence and absence, we have:

$$\begin{aligned} & o_{u_k}(i_m) \\ &= \sum_{\forall m': c_k^{m'} > 0} \ln \frac{P(l^{m'} = c_k^{m'} | i_m, r)}{P(l^{m'} = c_k^{m'} | i_m, \bar{r})} + \sum_{\forall m': c_k^{m'} = 0} \ln \frac{P(l^{m'} = 0 | i_m, r)}{P(l^{m'} = 0 | i_m, \bar{r})} + \ln \frac{P(r | i_m)}{P(\bar{r} | i_m)} \end{aligned} \quad (7)$$

Both subtracting

$$\sum_{\forall m': c_k^{m'} > 0} \ln \frac{P(l^{m'} = 0 | i_m, r)}{P(l^{m'} = 0 | i_m, \bar{r})}, \quad (8)$$

---

<sup>1</sup> The underlying model assumption might be weaker and more plausible by adopting Cooper's linked dependence assumptions instead of conditional independence [8].

to the first term and adding it from the second (where  $\ln x - \ln y = \ln \frac{x}{y}$ ) gives

$$\begin{aligned} o_{u_k}(i_m) &= \left( \sum_{\forall m': c_k^{m'} > 0} \ln \frac{P(l^{m'} = c_k^{m'} | i_m, r) P(l^{m'} = 0 | i_m, \bar{r})}{P(l^{m'} = c_k^{m'} | i_m, \bar{r}) P(l^{m'} = 0 | i_m, r)} \right) + \left( \sum_{\forall m'} \ln \frac{P(l^{m'} = 0 | i_m, r)}{P(l^{m'} = 0 | i_m, \bar{r})} \right) + \ln \frac{P(r | i_m)}{P(\bar{r} | i_m)} \end{aligned} \quad (9)$$

where the first term only deals with those items that are present in the user profile.  $P(l^{m'} = c_k^{m'} | i_m, r)$  is the probability that item  $i_{m'}$  occurs  $c_k^{m'}$  times in a profile of a user who likes item  $i_m$  (i.e. item  $i_m$  is relevant to this user). In other words, it means, given the evidence that a user who likes item  $i_m$ , what is the probability that this user plays item  $i_{m'}$   $c_k^{m'}$  times.

In summary, we have the following ranking formula:

$$o_{u_k}(i_m) = W_{u_k, i_m} + X_{i_m} + Y_{i_m} \quad (10)$$

where

$$W_{u_k, i_m} = \left( \sum_{\forall m': c_k^{m'} > 0} \ln \frac{P(l^{m'} = c_k^{m'} | i_m, r) P(l^{m'} = 0 | i_m, \bar{r})}{P(l^{m'} = c_k^{m'} | i_m, \bar{r}) P(l^{m'} = 0 | i_m, r)} \right) \quad (11)$$

$$X_{i_m} = \sum_{\forall m'} \ln \frac{P(l^{m'} = 0 | i_m, r)}{P(l^{m'} = 0 | i_m, \bar{r})} \quad (12)$$

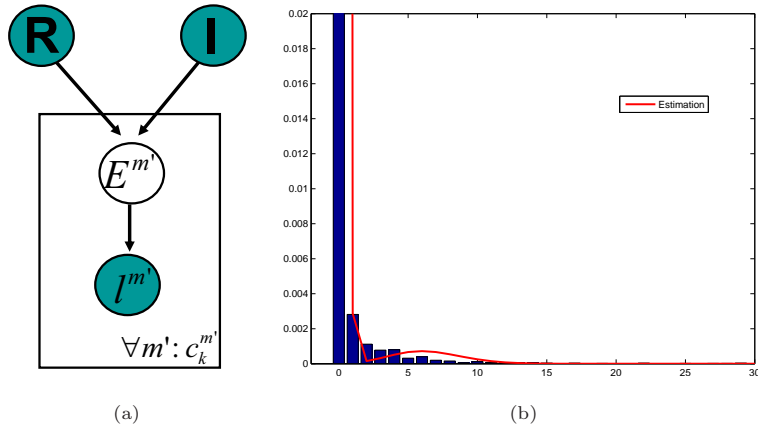
$$Y_{i_m} = \ln \frac{P(r | i_m)}{P(\bar{r} | i_m)} \quad (13)$$

From the final ranking score, we observe that the relevance ranking of a target item in the item-based model is a combination between the evidence that is dependent on the target user profile ( $W_{u_k, i_m}$ ) and that of the target item itself ( $X_{i_m} + Y_{i_m}$ ). However, we shall see in Section 3.2 that, due to the asymmetry between users and items, the final ranking of the user-based model (Eq. 27) only requires the “user profile”-dependent evidence.

### 3.1.1 Probability Estimation

Let us look at the weighting function  $W_{u_k, i_m}$  (Eq. 11) first. Item occurrences within user profiles (either  $P(l^{m'} = c_k^{m'} | i_m, r)$  or  $P(l^{m'} = c_k^{m'} | i_m, \bar{r})$ ) can be modeled by a Poisson distribution. Yet, an item occurring in a user profile does not necessarily mean that this user likes this item: randomness is another explanation, particularly when the item occurs few times only. Thus, a better model would





**Fig. 1** A Poisson mixture model for modelling the item occurrences in user profiles. (a) A graphical model of the Poisson mixtures. (b) An estimation of the Poisson mixtures for the Last.FM data set in the relevance case ( $\lambda_0 = 0.0028$ ,  $\lambda_1 = 6.4691$  and  $p = 0.0046$ ).

be a mixture of two Poisson models, i.e. a linear combination between a Poisson model coping with items that are “truly” liked by the user and a Poisson model dealing with some background noise. To achieve this, we introduce a hidden random variable  $E^{m'} \in \{e, \bar{e}\}$  for each of the items in the user profile, describing whether the presence of the item in a user profile is due to the fact that the user truly liked it ( $E^{m'} = e$ ), or because the user accidentally selected it ( $E^{m'} = \bar{e}$ ). A graphical model describing the probabilistic relationships among the random variables is illustrated in Fig. 1 (a). More formally, for the relevance case, we have

$$\begin{aligned}
 P(I^{m'} = c_k^{m'} | i_m, r) &= P(I^{m'} = c_k^{m'} | e)P(e | i_m, r) + P(I^{m'} = c_k^{m'} | \bar{e})P(\bar{e} | i_m, r) \\
 &= \frac{\lambda_1^{(c_k^{m'})} \exp(-\lambda_1)}{(c_k^{m'})!} p + \frac{\lambda_0^{(c_k^{m'})} \exp(-\lambda_0)}{(c_k^{m'})!} (1 - p)
 \end{aligned} \tag{14}$$

where  $\lambda_1$  and  $\lambda_0$  are the two Poisson means, which can be regarded as the expected item frequency counts in the two different cases ( $e$  and  $\bar{e}$ ) respectively.  $p \equiv P(e | i_m, r)$  denotes the probability that the user indeed likes item  $i'_m$ , given the condition that he or she liked another item  $i_m$ . A straight-forward method to obtain the parameters of the Poisson mixtures is to apply the Expectation-Maximization (EM) algorithm [9]. To illustrate this, Fig. 1 (b) plots the histogram of the item frequency distribution in the Last.FM data set as well as its estimated Poisson mixtures by applying the EM algorithm.

The same can be applied to the non-relevance case. Incorporating the Poisson mixtures for the both cases into Eq. 11 gives

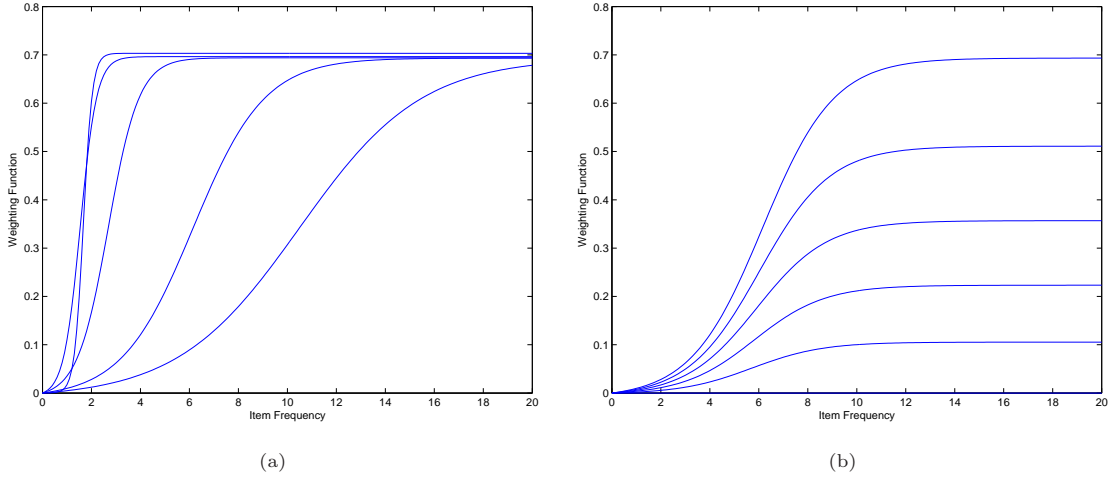
$$\begin{aligned}
W_{u_k, i_m} &= \sum_{\forall m': c_k^{m'} > 0} W_{i'_m, i_m} \\
&= \sum_{\forall m': c_k^{m'} > 0} \ln \frac{(\lambda_1^{(c_k^{m'})} \exp(-\lambda_1)p + \lambda_0^{(c_k^{m'})} \exp(-\lambda_0)(1-p)) (\exp(-\lambda_1)q + \exp(-\lambda_0)(1-q))}{(\lambda_1^{(c_k^{m'})} \exp(-\lambda_1)q + \lambda_0^{(c_k^{m'})} \exp(-\lambda_0)(1-q)) (\exp(-\lambda_1)p + \exp(-\lambda_0)(1-p))} \quad (15) \\
&= \sum_{\forall m': c_k^{m'} > 0} \ln \frac{(p + (\lambda_0/\lambda_1)^{(c_k^{m'})} \exp(\lambda_1 - \lambda_0)(1-p)) (\exp(\lambda_0 - \lambda_1)q + (1-q))}{(q + (\lambda_0/\lambda_1)^{(c_k^{m'})} \exp(\lambda_1 - \lambda_0)(1-q)) (\exp(\lambda_0 - \lambda_1)p + (1-p))}
\end{aligned}$$

where, similarly,  $q \equiv P(e|i_m, \bar{r})$  denotes the probability of the true preference of an item in the non-relevance case, while  $W_{i'_m, i_m}$  denotes the ranking score obtained from the target item and the item in the user profile.

For each of the item pairs  $(i'_m, i_m)$ , we need to estimate four parameters ( $p$ ,  $q$ ,  $\lambda_0$  and  $\lambda_1$ ), making the model difficult to apply in practice. Furthermore, it should be emphasised that the component distributions estimated by the EM algorithm may not necessarily correspond to the two reasons that we mentioned for the presence of an item in a user profile, even if the estimated mixture distribution may fit the data well.

In this regard, this paper takes an alternative approach, approximating the ranking function by a much simpler function. In text retrieval, a similar two-Poisson model has been proposed for modeling within-document term frequencies [13]. To make it applicable also, [27] introduced an approximation method, resulting in the widely-used BM25 weighting function for query terms. Following the same way of thinking, we can see that the weighting function for each of the items in the target user profile  $W_{i'_m, i_m}$  (Eq. 15) has the following characteristics: 1) Function  $W_{i'_m, i_m}$  increases monotonically with respect to the item frequency count  $c_k^{m'}$ , and 2) it reaches its upper-bound, governed by  $\log(p(1-q)/q(1-p))$ , when  $c_k^{m'}$  becomes infinity  $\infty$  [29, 30]. Roughly speaking, as demonstrated in Fig. 2, the parameters  $\lambda_0$  and  $\lambda_1$  can adjust the rate of the increase (see Fig. 2(a)), while the parameters  $p$  and  $q$  mainly control the upper bound (see Fig. 2(b)).

Therefore, it is intuitively desirable to approximate these two characteristics separately. Following the discussion in [27], we choose the function  $c_k^{m'}/(k_3 + c_k^{m'})$  (where  $k_3$  is a free parameter), which increases from zero to an asymptotic maximum, to model the monotonic increase with respect to the



**Fig. 2** The relationship between weighting function  $W_{i'_m, i_m}$  and its four parameters  $\lambda_0$ ,  $\lambda_1$ ,  $p$  and  $q$ . We plot ranking score  $W_{i'_m, i_m}$  against various item frequency counts  $c_k^{m'}$  from 0 to 20. (a) We fix  $\lambda_0 = 0.02$ ,  $p = 0.02$  and  $q = 0.010$ , and vary  $\lambda_1 \in \{0.03, 0.04, 0.1, 0.4, 5\}$ . (b) We fix  $\lambda_0 = 0.02$ ,  $\lambda_1 = 0.04$  and  $p = 0.02$ , and vary  $q \in \{0.020, 0.018, 0.016, 0.014, 0.012, 0.010\}$ .

item frequency counts. Since the probabilities  $q$  and  $p$  cannot be directly estimated, a simple alternative is to use the probabilities of the presence of the item, i.e.  $P(l^{m'} > 0 | i_m, r)$  and  $P(l^{m'} > 0 | i_m, \bar{r})$  to approximate them respectively. In summary, we have the following ranking function:

$$W_{u_k, i_m} \approx \left( \sum_{\forall m': c_k^{m'} > 0} \frac{c_k^{m'}}{k_3 + c_k^{m'}} \ln \frac{P(l^{m'} > 0 | i_m, r) P(l^{m'} = 0 | i_m, \bar{r})}{P(l^{m'} > 0 | i_m, \bar{r}) P(l^{m'} = 0 | i_m, r)} \right) \quad (16)$$

where the free parameter  $k_3$  is equivalent to the normalization parameter of within-query frequencies in the BM25 formula [27] (also see Appendix A), if we treat a user profile as a query.  $P(l^{m'} > 0 | i_m, r)$  (or  $P(l^{m'} > 0 | i_m, \bar{r})$ ) is the probability that item  $m'$  occurs in a profile of a user who is relevant (or non-relevant) to item  $i_m$ . Eq. 16 essentially decouples frequency counts  $c_k^{m'}$  and presence (absence) probabilities (e.g.  $P(l^{m'} > 0 | i_m, r)$ ), thus largely simplifying the computation in practice.

Next, we consider the probability estimations of presence (absence) of items in user profiles. To handle data sparseness, different from the Robertson-Sparck Jones probabilistic retrieval (RSJ) model [26], we propose to use Bayesian inference [12] to estimate the presence (absence) probabilities. Since we have two events, either an item is present ( $l^{m'} > 0$ ) or absent ( $l^{m'} = 0$ ), we assume that the probability follow the *Bernoulli* distribution. That is, we define  $\theta_{m', m} \equiv P(l^{m'} > 0 | i_m, r)$ , where  $\theta_{m', m}$  is regarded as the parameter of a Bernoulli distribution. For simplicity, we treat the parameter as a

random variable and estimate its value by maximizing an *a posteriori* probability. Formally we have

$$\hat{\theta}_{m',m} = \arg \max_{\theta_{m',m}} p(\theta_{m',m} | r_{m',m}, R_m; \alpha_r, \beta_r) \quad (17)$$

where  $R_m$  denotes the number of user profiles that are relevant to an item  $i_m$ , and among these user profiles,  $r_{m',m}$  denotes the number of the user profiles where an item  $i_{m'}$  is present. This establishes a contingency table for each item pair (shown in Table 1). In addition, we choose the Beta distribution as the prior (since it is the conjugate prior for the Bernoulli distribution), which is denoted as  $Beta(\alpha_r, \beta_r)$ . Using the conjugate prior, the posterior probability after observing some data turns to the Beta distribution again with updated parameters.

$$p(\theta_{m',m} | r_{m',m}, R_m; \alpha_r, \beta_r) \propto \theta_{m',m}^{r_{m',m} + \alpha_r - 1} (1 - \theta_{m',m})^{R_m - r_{m',m} + \beta_r - 1} \quad (18)$$

Maximizing an a posteriori probability in Eq. 18 (i.e. taking the mode) gives the estimation of the parameter [12]

$$\hat{\theta}_{m',m} = \frac{r_{m',m} + \alpha_r - 1}{R_m + \alpha_r + \beta_r - 2} \quad (19)$$

Following the same reasoning, we obtain the probability of item occurrences in the non-relevance case.

$$P(l^{i_{m'}} > 0 | i_m, \bar{r}) \equiv \hat{\gamma}_i = \frac{n_{m'} - r_{m',m} + \alpha_{\bar{r}} - 1}{K - R_m + \alpha_{\bar{r}} + \beta_{\bar{r}} - 2} \quad (20)$$

where we used  $\hat{\gamma}_i$  to denote  $P(l^{i_{m'}} > 0 | i_m, \bar{r})$ .  $\alpha_{\bar{r}}$  and  $\beta_{\bar{r}}$  are again the parameters of the conjugate prior ( $Beta(\alpha_{\bar{r}}, \beta_{\bar{r}})$ ), while  $n_{m'}$  denotes the number of times that item  $i_{m'}$  is present in a user profile (See Table 1). Replacing Eq. 19 and Eq. 20 into Eq. 16, we have

$$\begin{aligned} W_{u_k, i_m} &\approx \sum_{\forall m': c_k^{m'}} \frac{c_k^{m'}}{k_3 + c_k^{m'}} \ln \frac{\hat{\theta}_i (1 - \hat{\gamma}_i)}{\hat{\gamma}_i (1 - \hat{\theta}_i)} \\ &= \sum_{\forall m': c_k^{m'}} \frac{c_k^{m'}}{k_3 + c_k^{m'}} \ln \frac{(r_{m',m} + \alpha_r - 1)((K - R_m) - (n_{m'} - r_{m',m}) + \beta_{\bar{r}} - 1)}{(n_{m'} - r_{m',m} + \alpha_{\bar{r}} - 1)(R_m - r_{m',m} + \beta_r - 1)} \end{aligned} \quad (21)$$

The four hyper-parameters  $(\alpha_r, \alpha_{\bar{r}}, \beta_r, \beta_{\bar{r}})$  can be treated as pseudo frequency counts. Varying choices for them leads to different estimators [38]. In the information retrieval domain [26,27], adding an extra 0.5 count for each probability estimation has been widely used to avoid zero probabilities. This choice corresponds to set tiny constant values  $\alpha_r = \alpha_{\bar{r}} = \beta_r = \beta_{\bar{r}} = 1.5$ . We shall see that in the experiments collaborative filtering needs relatively bigger pseudo counts for the non-relevance

**Table 1** Contingency table of relevance v.s. occurrence: Item Model

	Item $i_m$ is Relevant	Item $i_m$ is NOT Relevant	
Item $i_{m'}$ Contained in UP	$r_{m',m}$	$n_{m'} - r_{m',m}$	$n_{m'}$
Item $i_{m'}$ NOT Contained in UP	$R_m - r_{m',m}$	$(N - R_m) - (n_{m'} - r_{m',m})$	$K - n_{m'}$
	$R_m$	$K - R_m$	$K$

**Table 2** Contingency table of relevance v.s. occurrence: User Model

	User $u_k$ is Relevant	User $u_k$ is NOT Relevant	
User $u_{k'}$ Contained in UP	$r_{k',k}$	$n_{k'} - r_{k',k}$	$n_{k'}$
User $u_{k'}$ NOT Contained in UP	$R_k - r_{k',k}$	$(M - R_k) - (n_{k'} - r_{k',k})$	$M - n_{k'}$
	$R_k$	$M - R_k$	$M$

and/or absence estimation ( $\alpha_{\bar{r}}$ ,  $\beta_r$  and  $\beta_{\bar{r}}$ ). This can be explained because using absence to model non-relevance is noisy, so more smoothing is needed. If we define a free parameter  $v$  and set it to be equal to  $a_r - 1$ , we have the generalized *Laplace smoothing* estimator. Alternatively, the prior can be fit on a distribution of the given collection [39].

Applying the Bayesian inference similarly, we obtain  $X_{i_m}$  as follows:

$$\begin{aligned}
X_{i_m} &= \sum_{i_{m'}} \ln \frac{P(l^{i_{m'}} = 0 | i_m, r)}{P(l^{i_{m'}} = 0 | i_m, \bar{r})} \\
&= \sum_{i_{m'}} \ln \frac{(K - R_m + \alpha_r + \beta_r - 2)(R_m - r_{m',m} + \beta_r - 1)}{(R_m + \alpha_r + \beta_r - 2)(K - R_m - (n_{m'} - r_{m',m}) + \beta_{\bar{r}} - 1)}
\end{aligned} \tag{22}$$

For the last term, the popularity ranking  $Y_{i_m}$ , we have

$$Y_{i_m} = \ln \frac{P(r | i_m)}{P(\bar{r} | i_m)} = \ln \frac{R_m}{K - R_m} \tag{23}$$

Notice that in the initial stage, we do not have any relevance observation of item  $i_m$ . We may assume that if a user played the item frequently (say played more than  $t$  times), we treat this item being relevant to this user's interest. By doing this, we can also construct the contingency table to be able to estimate the probabilities.

### 3.2 User-Based Relevance Model

Applying the Bayes' rule differently results in the following formula from Eq. 4:

$$o_{u_k}(i_m) = \ln \frac{P(i_m | u_k, r)}{P(i_m | u_k, \bar{r})} + \ln \frac{P(r | u_k)}{P(\bar{r} | u_k)} \tag{24}$$

**Table 3** Characteristics of the test data sets.

	Last.FM	Del.icio.us
Num. of Users	2408	1731
Num. of Items	1399	3370
Zero Occurrences in UP(%)	96.8%	96.7%

Similarly, using frequency counts over a set of users ( $l^1, \dots, l^{k'}, \dots, l^K$ ) to represent the target item  $i_m$ , we get

$$S_{u_k}(i_m) = \sum_{\forall k': c_{k'}^m > 0} \ln \frac{P(l^{k'} = c_{k'}^m | u_k, r) P(l^{k'} = 0 | u_k, \bar{r})}{P(l^{k'} = c_{k'}^m | u_k, \bar{r}) P(l^{k'} = 0 | u_k, r)} + \sum_{\forall k'} \ln \frac{P(l^{k'} = 0 | u_k, r)}{P(l^{k'} = 0 | u_k, \bar{r})} + \ln \frac{P(r | u_k)}{P(\bar{r} | u_k)} \quad (25)$$

where the last two terms in the formula are independent of target items, they can be discarded. Thus we have

$$S_{u_k}(i_m) \propto_{u_k} \sum_{\forall k': c_{k'}^m > 0} \ln \frac{P(l^{k'} = c_{k'}^m | u_k, r) P(l^{k'} = 0 | u_k, \bar{r})}{P(l^{k'} = c_{k'}^m | u_k, \bar{r}) P(l^{k'} = 0 | u_k, r)} \quad (26)$$

where  $\propto_{u_k}$  denotes same rank order with respect to  $u_k$ .

Following the same steps (the approximation to two-Poisson distribution and the MAP probability estimation) as discussed in the previous section gives

$$S_{u_k}(i_m) \propto_{u_k} \sum_{\forall k': c_{k'}^m > 0} \frac{c_{k'}^m}{\mathcal{K} + c_{k'}^m} \ln \frac{P(l^{k'} > 0 | u_k, r) P(l^{k'} = 0 | u_k, \bar{r})}{P(l^{k'} > 0 | u_k, \bar{r}) P(l^{k'} = 0 | u_k, r)} = \sum_{\forall k': c_{k'}^m > 0} \frac{c_{k'}^m}{\mathcal{K} + c_{k'}^m} \ln \frac{(r_{k',k} + \alpha_r - 1)(M - n_{k'} - R_k + r_{k',k} + \beta_{\bar{r}} - 1)}{(n_{k'} - r_{k',k} + \alpha_r \bar{r} - 1)(R_k - r_{k',k} + \beta_r - 1)} \quad (27)$$

where  $\mathcal{K} = k_1((1-b) + bL_m)$ .  $k_1$  is the normalization parameter of the frequency counts for the target item,  $L_m$  is the normalized item popularity (how many times the item  $i_m$  has been “used”) (i.e. the popularity of this item divided by the average popularity in the collection), and  $b \in [0, 1]$  denotes the mixture weight. Notice that if we treat an item as a document, the parameter  $k_1$  is equivalent to the normalization parameter of within-document frequencies in the BM25 formula (see Appendix A).

Table 2 shows the contingency table of user pairs.

---

### 3.3 Discussion

Previous studies on collaborative filtering, particularly memory-based approaches, make a distinction between user-based [5,14,23] and item-based approaches [10,28]. Our probabilistic relevance models were derived with an information retrieval view on collaborative filtering. They demonstrated that the user-based (relevance) and item-based (relevance) models are equivalent from a probabilistic point of view, since they have actually been derived from the same generative relevance model. The only difference corresponds to the choice of independence assumptions in the derivations, leading to the two different factorizations. But statistically they are inequivalent because the different factorizations lead to the different probability estimations; In the item-based relevance model, the item-to-item relevancy is estimated while in the user-based one, the user-to-user relevancy is required instead. We shall see shortly in our experiments that the probability estimation is one of the important factors influencing recommendation performance.

## 4 Experiments

### 4.1 Data Sets

The standard data sets used in the evaluation of collaborative filtering algorithms (i.e. MovieLens and Netflix) are rating-based, which are not suitable for testing our method using implicit user profiles. This paper adopts two implicit user profile data.

The first data set comes from a well known social music web site: `Last.FM`. It was collected from the play-lists of the users in the community by using a plug-in in the users' media players (for instance, Winamp, iTunes, XMMS etc). Plug-ins send the title (song name and artist name) of every song users play to the Last.FM server, which updates the user's musical profile with the new song. For our experiments, the triple  $\{\text{userID}, \text{artistID}, \text{Freq}\}$  is used.

The second data set was collected from one well-known collaborative tagging Web site, `del.icio.us`. Unlike other studies focusing on directly recommending contents (Web sites), here we intend to find relevance tags on the basis of user profiles as this is a crucial step in such systems. For instance, the tag suggestion is needed in helping users assigning tags to new contents, and it is also useful when

---

constructing a personalized “tag cloud” for the purpose of exploratory search [36]. The Web site has been crawled between May and October 2006. We collected a number of the most popular tags, found which users were using these tags, and then downloaded the whole profiles of these users. We extracted the triples {userID, tagID, Freq} from each of the user profiles. User IDs are randomly generated to keep the users anonymous. Table 3 summarizes the basic characteristics of the data sets<sup>2</sup>.

## 4.2 Experiment Protocols

For 5-fold cross-validation, we randomly divided this data set into a training set (80% of the users) and a test set (20% of the users). Results are obtained by averaging 5 different runs (sampling of training/test set). The training set was used to estimate the model. The test set was used for evaluating the accuracy of the recommendations on the new users, whose user profiles are not in the training set. For each test user, 5, 10, or 15 items of a test user were put into the user profile list. The remaining items were used to test the recommendations.

In information retrieval, the effectiveness of the document ranking is commonly measured by *precision* and *recall* [2]. Precision measures the proportion of retrieved documents that are indeed relevant to the user’s information need, while recall measures the fraction of all relevant documents that are successfully retrieved. In the case of collaborative filtering, we are, however, only interested in examining the accuracy of the top-N recommended items, while paying less attention to finding *all* the relevant items. Thus, our experiments here only consider the *recommendation precision*, which measures the proportion of recommended items that are ground truth items. Note that the items in the profiles of the test user represent only a fraction of the items that the user *truly* liked. Therefore, the measured precision *underestimates* the true precision.

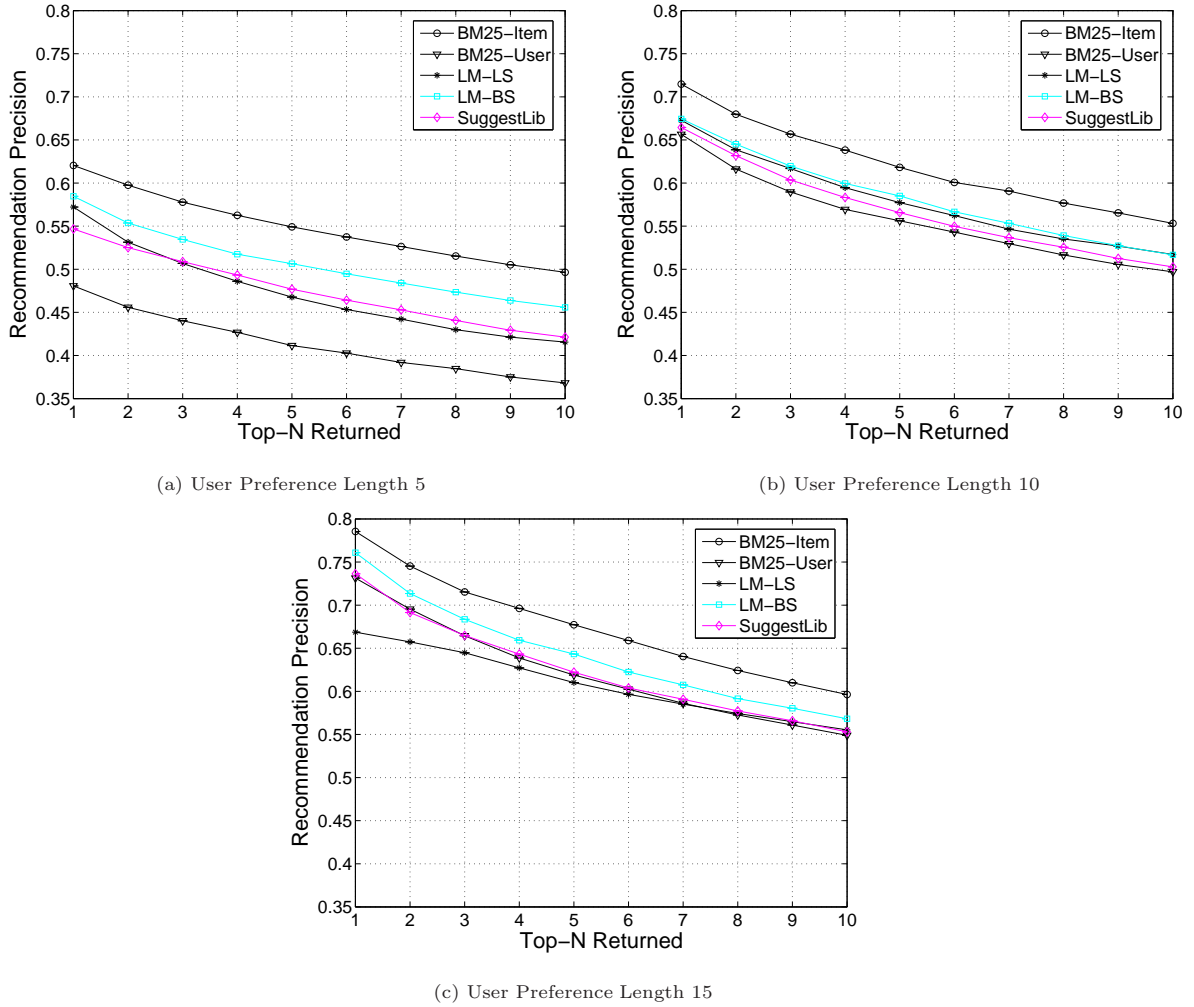
## 4.3 Performance

We choose the state-of-the-art item ranking algorithms that have been discussed in Section 2.2 as our baselines. For the method proposed in [10], we adopt their implementation, the top-N suggest

---

<sup>2</sup> The two data sets can be downloaded from <http://ict.ewi.tudelft.nl/~jun/CollaborativeFiltering.html>.



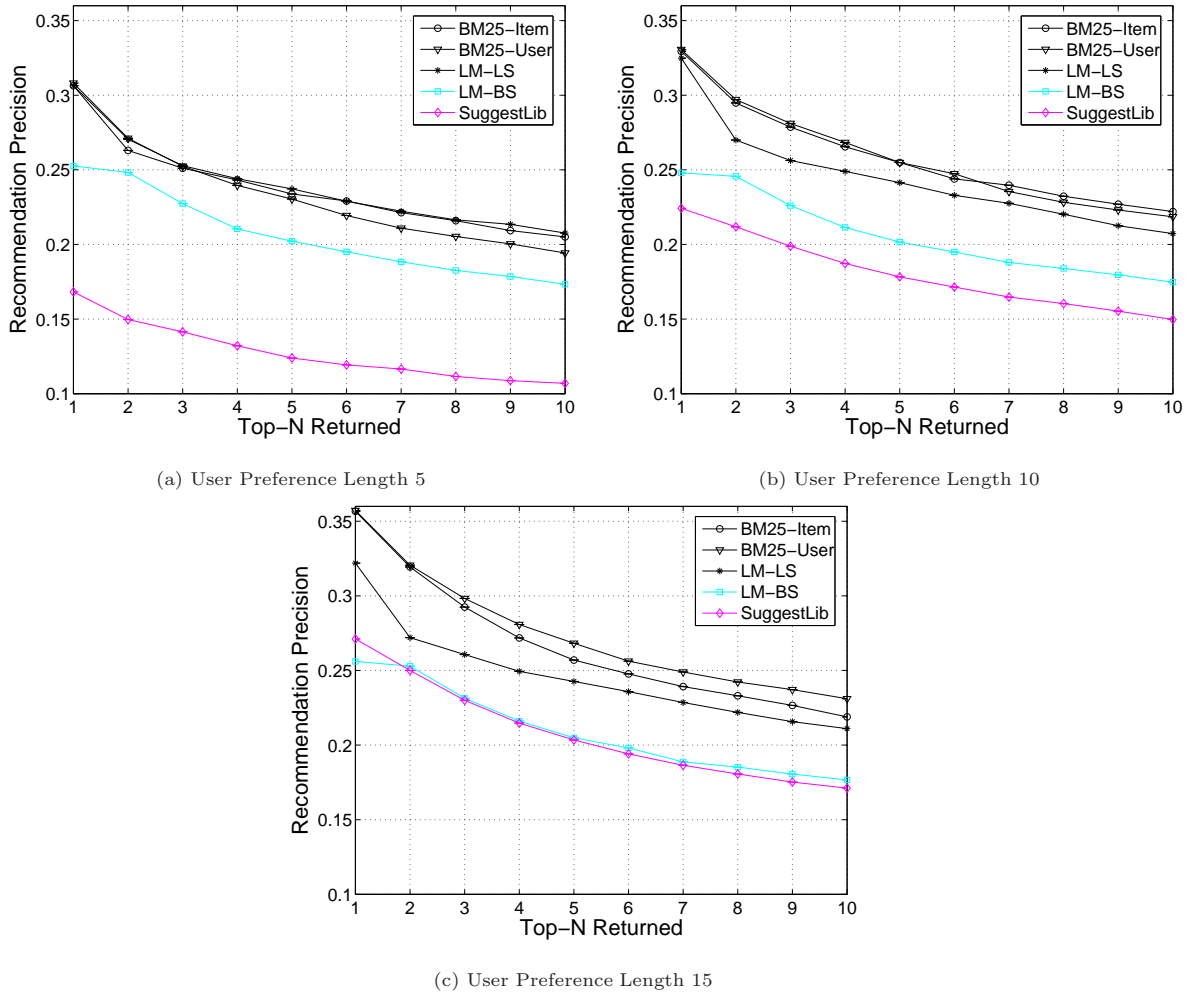


**Fig. 3** Precision of different methods in the Last.FM Data Set.

recommendation library<sup>3</sup>, which is denoted as `SuggestLib`. We also implement the language modelling approach of collaborative filtering in [33] and denote this approach as `LM-LS` while its variant using the Bayes' smoothing (i.e., a Dirichlet prior) is denoted as `LM-BS`. To make a comparison, the parameters of the algorithms are set to the optimal ones.

We set the parameters of our two models to the optimal ones and compare them with these strong baselines. The item-based relevance model is denoted as `BM25-Item` while the user-based relevance model is denoted as `BM25-User`. Results are shown in Fig. 3 and 4 over different returned items. Let us first compare the performance of the `BM25-Item` and `BM25-User` models. For the Last.FM data set (Fig. 3), the item-based relevance model consistently performs better than the user-based relevance

<sup>3</sup> <http://glaros.dtc.umn.edu/gkhome/suggest/overview>



**Fig. 4** Precision of different methods in the `Del.icio.us` Data Set.

model. This confirms a previous observation that item-to-item similarity (relevancy) in general is more robust than user-to-user similarity [28]. However, if we look at the `del.icio.us` data (Fig. 4), the performance gain from the item-based relevance model is not clear any more - we obtain a mixture result and the user-based one even outperforms the item-based one when the number of items in user preferences is set to 15 (see Fig. 4 (c)). We think this is because the characteristics of data set play an important role for the probability estimations in the models. In the `Last.FM` data set, the number of users is larger than the number of items (see Table 3). It basically means that we have more observations from the user side about the item-to-item relevancy while having less observations from the item side about user-to-user relevancy. Thus, in the `Last.FM` data set, the probability estimation for the item based relevance model is more reliable than that of the user-based relevance model. But

**Table 4** Comparison with the other approaches. Precision is reported in the `Last.FM` data set. The best results are in bold type. A Wilcoxon signed-rank test is conducted and the significant ones over the second best are marked as \*.

	Top-1	Top-3	Top-10
<b>BM25-Item</b>	<b>0.620*</b>	<b>0.578*</b>	<b>0.497*</b>
<b>LM-LS</b>	0.572	0.507	0.416
<b>LM-BS</b>	0.585	0.535	0.456
<b>SuggestLib</b>	0.547	0.509	0.421

(a) User Profile Length 5

	Top-1	Top-3	Top-10
<b>BM25-Item</b>	<b>0.715*</b>	<b>0.657*</b>	<b>0.553*</b>
<b>LM-LS</b>	0.673	0.617	0.517
<b>LM-BS</b>	0.674	0.620	0.517
<b>SuggestLib</b>	0.664	0.604	0.503

(b) User Profile Length 10

	Top-1	Top-3	Top-10
<b>BM25-Item</b>	<b>0.785*</b>	<b>0.715*</b>	<b>0.596*</b>
<b>LM-LS</b>	0.669	0.645	0.555
<b>LM-BS</b>	0.761	0.684	0.568
<b>SuggestLib</b>	0.736	0.665	0.553

(c) User Profile Length 15

in the `del.icio.us` data set (see Table 3), the number of items is larger than the number of users. Thus we have more observations about user-to-user relevancy from the item side, causing a significant improvement for the user-based relevance model.

Since the item-based relevance model in general outperforms the user-based relevance model, we next compare the item-based relevance model with other methods (shown in Table 4 and 5). From the tables, we can see that the item-based relevance model performs consistently better than the `SuggestLib` method over all the configurations. A Wilcoxon signed-rank test [16] is done to verify the significance. We also observe that in most of the configurations our item-based model significantly outperforms the language modelling approaches, both the linear smoothing and the Bayesian smoothing variants. We believe that the effectiveness of our model is due to the fact that the model naturally integrates frequency counts and probability estimation of non-relevance into the ranking formula, apart from other alternatives.

**Table 5** Comparison with the other approaches. Precision is reported in the `Del.icio.us` data set. The best results are in bold type. A Wilcoxon signed-rank test is conducted and the significant ones over the second best are marked as \*.

	Top-1	Top-3	Top-10
<b>BM25-Item</b>	<b>0.306</b>	0.251	0.205
<b>LM-LS</b>	<b>0.306</b>	<b>0.253</b>	<b>0.208</b>
<b>LM-BS</b>	0.253	0.227	0.173
<b>SuggestLib</b>	0.168	0.141	0.107

(a) User Profile Length 5

	Top-1	Top-3	Top-10
<b>BM25-Item</b>	<b>0.329</b>	<b>0.279*</b>	<b>0.222*</b>
<b>LM-LS</b>	0.325	0.256	0.207
<b>LM-BS</b>	0.248	0.226	0.175
<b>SuggestLib</b>	0.224	0.199	0.150

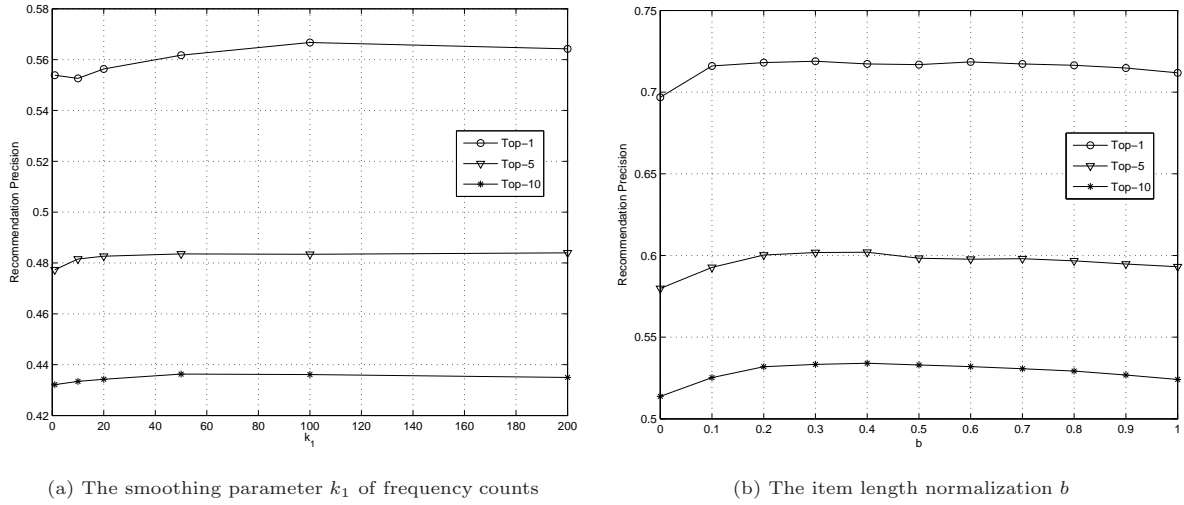
(b) User Profile Length 10

	Top-1	Top-3	Top-10
<b>BM25-Item</b>	<b>0.357*</b>	<b>0.292*</b>	<b>0.219*</b>
<b>LM-LS</b>	0.322	0.261	0.211
<b>LM-BS</b>	0.256	0.231	0.177
<b>SuggestLib</b>	0.271	0.230	0.171

(c) User Profile Length 15

#### 4.4 Parameter Estimation

This section tests the sensitivity of the parameters, using the `del.icio.us` data set. Recall that for both the item-based relevance model (shown in Eq. 10) and the user-based relevance model (shown in Eq. 27), we have frequency smoothing parameter  $k_1$  (and  $b$ ) or  $k_3$ , and co-occurrence smoothing parameters  $\alpha$  and  $\beta$ . We first test the sensitivity of the frequency smoothing parameters. Fig. 5 shows recommendation precision against the parameters  $k_1$  and  $b$  of the user-based relevance model while Fig. 6 shows recommendation precision varying the parameter  $k_3$  of the item relevance model. The optimal values in the figures demonstrate that both the frequency smoothing parameters ( $k_1$  and  $k_3$ ) and the length normalization parameter  $b$ , inspired by the BM25 formula, indeed improve the recommendation performance. We also observe that these parameters are relatively insensitive to different data sets and their different sparsity setups.

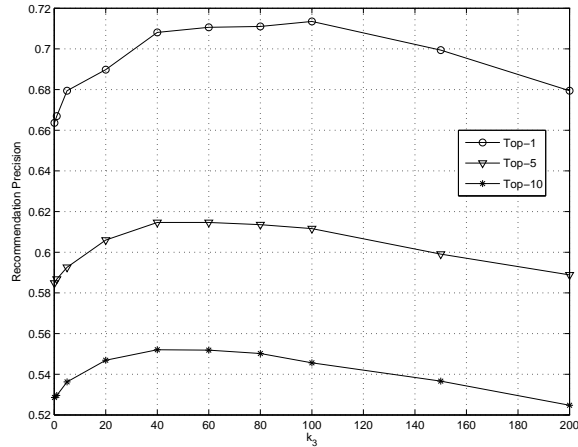


**Fig. 5** Parameters in the user-based relevance model.

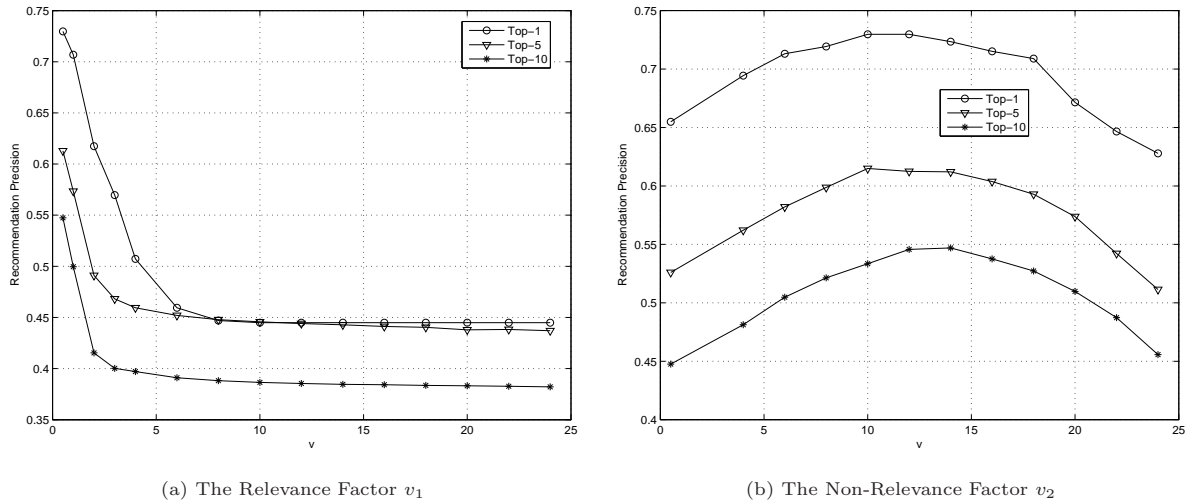
Next we fix the frequency smoothing parameters to the optimal ones and test the co-occurrence smoothing parameters for both models. Fig. 7 and Fig. 8 plot the smoothing parameters against the recommendation precision. More precisely, Fig. 7 (a) and Fig. 8 (a) plot the smoothing parameter for the relevance part  $v_1 = \alpha_r - 1$  while Fig. 7 (b) and Fig. 8 (b) plot that of the non-relevance or absence parts; all of them are set to be equal ( $v_2 = \alpha_{\bar{r}} - 1 = \beta_r - 1 = \beta_{\bar{r}} - 1$ ) in order to minimize the number of parameters while still retaining comparable performance. From the figures, we can see that the optimal smoothing parameters (pseudo counts) of the relevance part  $v_1$  are relatively small, compared to those of the non-relevance part. For the user-based relevance model, the pseudo counts of the non-relevance estimations are in the range of [10, 15] (Fig. 7(b)) while for the item-based relevance model, they are in the range of [50, 100] (Fig. 8(b)). It is due to the fact that the non-relevance estimation is not as reliable as the relevance estimation and thus more smoothing is required.

## 5 Conclusions

This paper proposed a probabilistic item ranking framework for collaborative filtering, which is inspired by the classic probabilistic relevance model of text retrieval and its variants [26, 27, 29, 30]. We have derived two different models in the relevance framework in order to generate top-N item recom-



**Fig. 6** The smoothing parameter of frequency counts  $k_3$  in the item-based relevance model.



**Fig. 7** The relevance/non-Relevance smoothing parameters in the user-based relevance model.

recommendations. We conclude from the experimental results that the proposed models are indeed effective, and significantly improve the performance of the top-N item recommendations.

In current settings, we fix a threshold when considering frequency counts as relevance observations. In the future, we may also consider graded relevance with respect to the number of times a user played an item. To do this, we may weight (sampling) the importance of the user profiles according to the number of times the user played/reviewed an item when we construct the contingency table. In current models, the hyperparameters are obtained by using cross-validation. In the future, it is worthwhile investigating the evidence approximation framework [4] by which the hyperparameters can

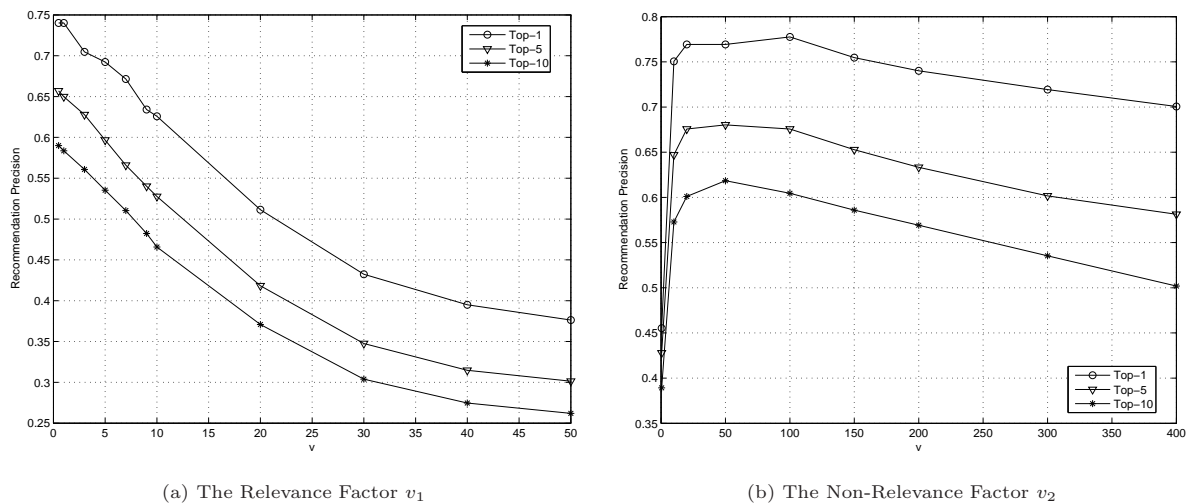


Fig. 8 The relevance/non-relevance smoothing parameters in the item-based relevance model.

be estimated from the whole collection; or we can take a full Bayesian approach that integrates over the hyperparameters and the model parameters by adopting variational methods [18].

It has been seen in this paper that relevance is a good concept to explain the correspondence between user interest and information items. We have setup a close relationship between the probabilistic models of text retrieval and these of collaborative filtering. It facilitates a flexible framework to tryout more of the techniques that have been used in text retrieval to the related problem of collaborative filtering. For instance, relevance observations can be easily incorporated in the framework once we have relevance feedback from users. An interesting observation is that, different from text retrieval, relevance feedback for a given user in collaborative filtering is not dependent of this user’s “query” (a user profile) only. It instead has a rather global impact, and affects the representation of the whole collection; Relevance feedback from one user could influence the ranking order of the other users. It is also worthwhile investigating query expansion by including more relevant items as query items or re-calculating (re-constructing) the contingency table according to the relevance information.

Finally, a combination of the two relevance models is of interest [34,35]. This has some analogies with the “unified model” idea in information retrieval [25]. However, there are also some differences: in information retrieval, based on explicit features of items and explicit queries, simple user relevance feedback relates to the current query only, and a unified model is *required* to achieve the global impact

which we have already identified in the present (non-unified) models for collaborative filtering. These subtle differences make the exploration of the unified model ideas particularly attractive.

## References

1. G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.
2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, May 1999.
3. N. J. Belkin and W. B. Croft. Information filtering and information retrieval: two sides of the same coin? *Commun. ACM*, 35(12):29–38, 1992.
4. C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
5. J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 43–52, San Francisco, CA, 1998. Morgan Kaufmann.
6. J. Canny. Collaborative filtering with privacy via factor analysis. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 238–245, New York, NY, 2002. ACM Press.
7. M. Claypool, P. Le, M. Wased, and D. Brown. Implicit interest indicators. In *IUI '01: Proceedings of the 6th international conference on Intelligent user interfaces*, pages 33–40, New York, NY, USA, 2001. ACM.
8. W. S. Cooper. Some inconsistencies and misidentified modeling assumptions in probabilistic information retrieval. *ACM Trans. Inf. Syst.*, 13(1):100–111, 1995.
9. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.
10. M. Deshpande and G. Karypis. Item-based top-N recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, 2004.
11. S. Eyheramendy, D. Lewis, and D. Madigan. On the naive bayes model for text categorization. In *Proc. of Artificial Intelligence and Statistics*, 2003.
12. A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall, 2003.
13. S. Harter. A probabilistic approach to automatic keyword indexing. *Journal of the American Society for Information Science*, 35:197–206 and 280–289, 1975.
14. J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237, New York, NY, 1999. ACM Press.



- 
15. T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Info. Syst.*, Vol 22(1):89–115, 2004.
  16. D. Hull. Using statistical testing in the evaluation of retrieval experiments. In *SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 329–338, New York, NY, 1993. ACM Press.
  17. R. Jin, L. Si, and C. Zhai. A study of mixture models for collaborative filtering. *Inf. Retr.*, 9(3):357–382, 2006.
  18. M. Jordan. *Learning in Graphical Models*. MIT Press, 1999.
  19. J. Lafferty and C. Zhai. Probabilistic relevance models based on document and query generation. *Language Modeling and Information Retrieval, Kluwer International Series on Information Retrieval*, V.13:1–10, 2003.
  20. B. Marlin. Collaborative filtering: a machine learning perspective. Master's thesis, Department of Computer Science, University of Toronto, 2004.
  21. M. R. McLaughlin and J. L. Herlocker. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 329–336, New York, NY, USA, 2004. ACM Press.
  22. D. M. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 473–480, San Francisco, CA, 2000. Morgan Kaufmann Publishers Inc.
  23. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, 1994. ACM Press.
  24. S. E. Robertson. The probability ranking principle in IR. *Readings in information retrieval*, pages 281–286, 1997.
  25. S. E. Robertson, M. E. Maron, and W. Cooper. Probability of relevance: a unification of two competing models for document retrieval. *Information Technology: Research and Development*, 1(1):1–21, 1982.
  26. S. E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–46, 1976.
  27. S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 232–241, New York, NY, 1994. Springer-Verlag New York, Inc.

- 
28. B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 285–295, New York, NY, 2001. ACM Press.
  29. K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments, part1. *Information Processing and Management*, V. 36(6):779–808, November 2000.
  30. K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments, part2. *Information Processing and Management*, V. 36(6):809–840, November 2000.
  31. M. J. Taylor, H. Zaragoza, and S. E. Robertson. Ranking classes: Finding similar authors. Technical report, Microsoft Research, Cambridge, 2003.
  32. C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, London, UK, 1979.
  33. J. Wang, A. P. de Vries, and M. J. Reinders. A user-item relevance model for log-based collaborative filtering. In *Proc. of ECIR06, London, UK*, pages 37–48, Berlin, Germany, 2006. Springer Berlin / Heidelberg.
  34. J. Wang, A. P. de Vries, and M. J. Reinders. Unified relevance models for rating prediction in collaborative filtering. *To appear in ACM Trans. on Information System (TOIS)*, 2008.
  35. J. Wang, A. P. de Vries, and M. J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 501–508, New York, NY, 2006. ACM Press.
  36. J. Wang, J. Yang, M. Clements, A. P. de Vries, and M. J. T. Reinders. Personalized collaborative tagging. *Technical Report, University College London*, 2007.
  37. G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, and Z. Chen. Scalable collaborative filtering using cluster-based smoothing. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 114–121, New York, NY, 2005. ACM Press.
  38. H. Zaragoza, D. Hiemstra, and M. Tipping. Bayesian extension to the language model for ad hoc information retrieval. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 4–9, New York, NY, USA, 2003. ACM Press.
  39. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 334–342, New York, NY, 2001. ACM Press.

---

## A The Okapi BM25 Document Ranking Score

To make the paper self-contained and facilitate the comparison between the proposed model and the BM25 model of text retrieval [27,30], here we summarise the Okapi BM25 document ranking formula. The commonly-used ranking function  $S_q(d)$  of a document  $d$  given a query  $q$  is expressed as follows:

$$S_q(d) = \sum_{\forall t: c_q^t > 0} \frac{(k_3 + 1)c_q^t (k_1 + 1)c_d^t}{k_3 + c_q^t \mathcal{K} + c_d^t} \log \frac{(r_t + 0.5)(N - n_t - R + r_t + 0.5)}{(n_t - r_t + 0.5)(R - r_t + 0.5)} \quad (28)$$

where

- $c_q^t$  denotes the within query frequency of a term  $t$  at query  $q$ , while  $c_d^t$  denotes the with document frequency of a term  $t$  at document  $d$ .
- $k_1$  and  $k_3$  are constants. The factors  $k_3 + 1$  and  $k_1 + 1$  are unnecessary here, but help scale the weights. For instance, the first component is 1 when  $c_q^t = 1$ .
- $\mathcal{K} \equiv k_1((1 - b) + bL_d)$ .  $L_d$  is the normalised document length (i.e. the length of this document  $d$  divided by the average length of documents in the collection).  $b \in [0, 1]$  is constant.
- $n_t$  is the number of documents in the collection indexed by this term  $t$ .
- $N$  is the total number of documents in the collection.
- $r_t$  is the number of relevant documents indexed by this term  $t$ .
- $R$  is the total number of relevant documents.

For detailed information about the model and its relationship with the Robertson-Sparck Jones probabilistic retrieval (RSJ) model [26], we refer to [27,30].