# Language Models of Collaborative Filtering

Jun Wang

Department of Computer Science, University College London
Malet Place, London, WC1E 6BT, UK
jun_wang@acm.org

**Abstract.** Collaborative filtering is a major technique to make personalized recommendations about information items (movies, books, webpages etc) to individual users. In the literature, a common research objective is to predict unknown ratings of items for a user, on the condition that the user has explicitly rated a certain amount of items. Nevertheless, in many practical situations, we may only have *implicit* evidence of user preferences, such as "playback times of a music file" or "visiting frequency of a web-site". Most importantly, a more practical view of the recommendation task is to directly generate a top-$N$ ranked list of items that the user is most likely to like.
In this paper, we take these two concerns into account. Item ranking in recommender systems is considered as a task highly related to document ranking in text retrieval. Firstly, two practical item scoring functions are derived by adopting the generative language modelling approach of text retrieval. Secondly, to address the uncertainty associated with the score estimation, we introduce a *risk-averse* model that penalizes the less reliable scores. Our experiments on real data sets demonstrate that significant performance gains have been achieved.

## 1 Introduction

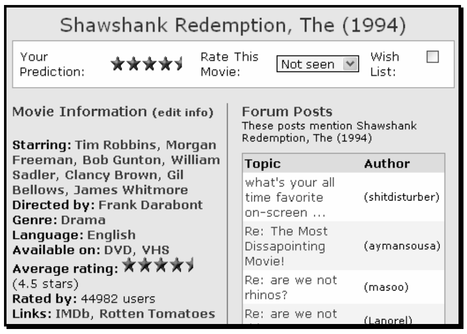
The Digital Revolution on information storage and transmission increases the amount of information that we deal with in our daily lives. Although we enjoy the entertainment and convenience brought to us by such a variety of sources, the volume of information is increasing far more quickly than our ability to digest it. For instance, the Internet has become the most significant media source and is growing at an exponential speed. But the user ability of obtaining useful information in the Internet grows slowly. Tools that support for the effective retrieval of relevant information are still primitive – most information retrieval systems heavily rely on textual queries of users to identify their information needs [13]. Queries constructed by keywords only are, however, not powerful enough to express the needs of a particular user both semantically and contextually. To see this, consider the following common search scenario: "find movies showing this weekend in nearby cinemas that I most likely to like." Such a user information need requires the retrieval system at least to be able to capture user interest ("most likely to like"). Unfortunately, most existing retrieval models and search technologies are incapable of achieving such a realistic retrieval goal, because they only focus on building the *correspondence* between textual queries and documents and lack mechanisms to model individual users who issue queries. Hence, it is essential to accurately model various user information needs beyond queries. With the recent advances in Human-Computer Interfacing and sensor technologies that make use of cameras, motion detectors, voice captures, GPSs etc., we have witnessed a research transition from information (document) centric computing into user centric computing; consider for example user profiling, that attempts to broadly understand users' various interests, intentions etc. on the basis of the recorded human-computer interactions.

Also, large amounts of information exist in a dynamic form. To process streams of incoming data, we need an information system that can play a more active role during the information seeking process. Therefore, information filtering systems arise [2]. In contrast to most retrieval systems that passively wait for user queries to respond accordingly, they aim to actively filter out, refine and systematically represent the relevant information and intuitively ignore superfluous computations on redundant data.

Combination of these two demands has created increasing interests in building a recommender system that can steer users towards their personal interests and actively filter relevant information items on the users' behalf. As one of the dominant techniques, collaborative filtering has appeared in the domain of Information Retrieval (IR) and Human-Computer Interaction (HCI) [8]. They attempt to filter information items such as books, CDs, DVDs, movies, TV programs, and electronics, based on a history of the user's likes and dislikes. Examples include the Amazon's book recommendation engine (amazon.com) and the Netflix DVD recommendation engine (netflix.com). We believe recommender systems will eventually support companies to realize a shift from offering mass products and services to offering customized goods and services that efficiently satisfy desires and needs of individual users.

In this paper, we would like to emphasize the following two crucial observations:

1. Although collaborative filtering exists in various forms in practice, its purposes can be generally regarded as "item ranking" and "rating prediction". They are illustrated in Fig 1. The rating prediction (see Fig 1 (a) and (b)) aims at predicting an unknown rating of an item for the user, with the requirement that the user has to explicitly rate a certain amount of items. This type of recommendation has been widely conceived and well studied in the research literature, since the pioneering work on the MovieLens systems (http://movielens.umn.edu); from the early work on filtering netnews [15] and the movie recommender systems ([9]) to the latest Netflix competition (http://www.netflixprize.com/), most approaches accept by default that the rating prediction is the underlying task for recommender systems. However, in many practical systems such as Amazon (http://amazon.com) and Last.Fm (http://last.fm), it is sometimes more favorable to formulate collaborative filtering as an item ranking problem, because we often face the situation where our ultimate task is to generate the top-N list of the end user's most favorite items (see Fig 1 (c) and (d)).


Shawshank Redemption, The (1994)

Your Prediction: ★★★★½ Rate This Movie: Not seen Wish List: ☐

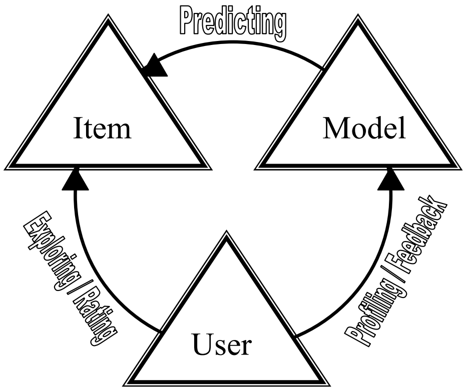
Movie Information (edit info)

**Starring:** Tim Robbins, Morgan Freeman, Bob Gunton, William Sadler, Clancy Brown, Gil Bellows, James Whitmore
**Directed by:** Frank Darabont
**Genre:** Drama
**Language:** English
**Available on:** DVD, VHS
**Average rating:** ★★★★½
(4.5 stars)
**Rated by:** 44982 users
**Links:** IMDb, Rotten Tomatoes

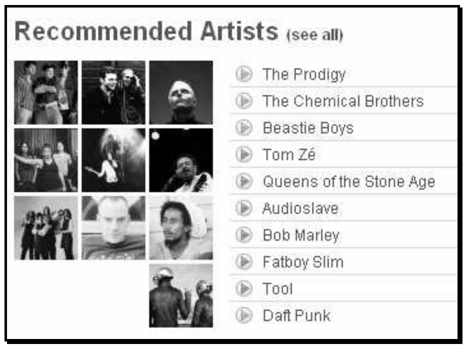
Forum Posts
These posts mention Shawshank Redemption, The (1994)

| Topic | Author |
|---|---|
| what's your all time favorite on-screen ... | (shitdisturber) |
| Re: The Most Dissapointing Movie! | (aymansousa) |
| Re: are we not rhinos? | (masoo) |
| Re: are we not | (Laporel) |

(a) A snapshot of the MovieLens System

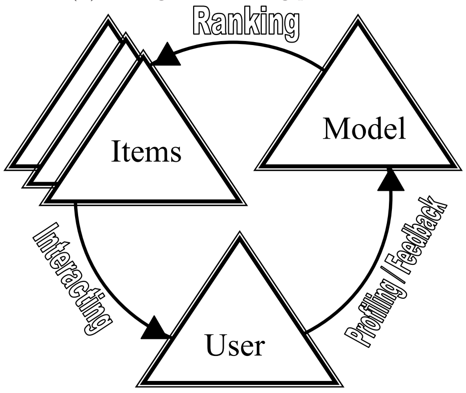


(b) A diagram of rating prediction.


Recommended Artists (see all)

- The Prodigy
- The Chemical Brothers
- Beastie Boys
- Tom Zé
- Queens of the Stone Age
- Audioslave
- Bob Marley
- Fatboy Slim
- Tool
- Daft Punk

(c) A snapshot of the Last.FM system



(d) A diagram of item ranking.

**Fig. 1.** The Two Forms of Recommendation.

2. 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 [5]. 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, 9, 10, 16, 21].

This motivated us to conduct a formal study on probabilistic item ranking for collaborative filtering. The remainder of the paper is organized as follows. We first describe related work, and then establish the generative language model for collaborative filtering. After that, we extend the model by considering the uncertainty of the estimation. Finally, we provide an empirical evaluation of the recommendation performance, and conclude our work.

## 2 Related Work

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 [3, 9, 15], item-based methods [7, 16] and unified methods [19]. 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, [14] 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, [3] 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 [10, 12], the cluster model [3] and the latent factor model [4]. 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 [10, 12], 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.

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:

$$o_u(i) \equiv \sum_{i' \in L_u} s_I(i', i) \tag{1}$$

where $u$ and $i$ denote the target user and item respectively, and $i' \in L_u$ denotes any item in the preference list of user $u$. $S_I$ 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 [7] proposed an alternative, TFxIDF-like similarity measure, which is shown as follows:

$$s_I(i', i) = \frac{Freq(i', i)}{Freq(i') \times Freq(i)^{\alpha}} \tag{2}$$

where $Freq$ denotes the frequency counts of an item $Freq(i')$ or co-occurrence counts for two items $Freq(i', i)$. $\alpha$ 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 our previous work, we have introduced the concept of relevance into collaborative filtering [17]. Items can be then ranked by estimating the probability of the relevance between users (preferences) and items [18, 20]. In this paper, we take another angle, considering a generative process between items and users.

## 3 A Statistic Language Model for Collaborative Filtering

Collaborative filtering aims at finding information items that a user is most likely to like, given his or her preference. To achieve this, we could formally measure how probable an item (denoted as $i$) is to be suggested to a given user (denoted as $u$): $p(i|u)$, and then rank items accordingly:

$$\begin{aligned} o_u(i) \equiv p(i|u) =& \frac{p(u|i)p(i)}{p(u)} \\ &\propto_i \log p(u|i) + \log p(i) - \log p(u) \\ &\propto_i \log p(u|i) + \log p(i) \end{aligned} \tag{3}$$

where $\log p(u)$ can be removed since it is independent of the target item $i$. The item ranking has two parts: its likelihood towards the user preference $p(u|i)$ and its popularity $p(i)$. The probability $p(i)$ can be easily estimated by counting the frequency from the collection.

To estimate the likelihood $p(u|i)$, we follow the argument of the language model of information retrieval. In the language modelling approach of information retrieval [6], one needs to assess how probable a query $q$ would be generated from a document *language model* $\theta_d$, and then rank each of the documents $d$ in the collection on the basis of the generative probability $p(q|\theta_d)$. Similarly, in collaborative filtering, we first choose an *optimal* generative model $\theta_i$ for each candidate item $i$; it captures the underlying distribution of users (or user preferences) who liked the item. Probability $p(u|\theta_i)$ is then used to estimate how probable a user preference (as a query) is to be generated by that model. Replacing it into Eq. (3) gives

$$p(i|u) \propto_i \log p(u|\theta_i) + \log p(i) \tag{4}$$

By doing this, we relate the language modelling of text retrieval and the collaborative filtering modelling at a probabilistic level. Yet, at the feature representation level they are quite apart from each other, as their input data and purposes are completely different. Consequently, applying the text retrieval model to collaborative filtering is not trivial. The difficulty lies in the fact that in text retrieval both queries and documents are represented by texts, which provide an important information channel to link queries (user needs) and documents. Due to the lack of relevance observations, the language models in text retrieval shift their focus from directly estimating the correspondence (relevance) between user needs (queries) and documents to estimating word statistics in the documents and/or queries and then building up the link through these statistics. Conversely, in collaborative filtering, in most cases, we do not have such extra information to relate user preferences and information items. Instead, recorded in the system are only user preferences, which are thought of as indirect observations of the relevance between a user interest and an information item. Thus, the central question in modelling collaborative filtering is how to relate users and items through this usually very sparse user-item matrix, where its elements record the frequency counts, like "playback times of a music file", or "visiting frequency of a web-site" etc.

The estimation of the likelihood $p(u|\theta_i)$ depends on the representation of the user preference. From the data stored in the user-item matrix, if we use a set of items $i' \in L_u$ to present user $u$, and assume that each item $i'$ in the user preference $L_u$ is independently generated, we have

$$p(i|u) \propto_i \sum_{i' \in L_u} \log p(i'|\theta_i) + \log p(i) \tag{5}$$

In text retrieval, the interpretation of the likelihood function $p(t|\theta_d)$ is relatively straightforward as both queries and documents are represented by the same set of features, i.e., words. Subsequently, $\theta_d$ is estimated conveniently by looking at the words occur in document $d$. By contrast, the interpretation of the likelihood $p(i'|\theta_i)$ in Eq. (5), which links the target item $i$ to another item $i'$ in the target user's preference, is slightly different; it measures how probable an item $i$ would be generated from a user preference where an item $i$ occurs. It is estimated by considering the following two steps: 1) aggregating the user preferences in which item $i$ occurs, and 2) from them, calculating how frequent item $i'$ is also present - the Maximum Likelihood Estimate (MLE) would be $\hat{\theta}_i = p(i'|\hat{\theta}_i) \equiv \frac{c(i',i)}{c(i)}$, where $c(i', i)$ denotes the number of user preferences where both items $i$ and $i'$ occur, and $c(i)$ denotes the number of user preferences where item $i$ occurs. The hat on $\hat{\theta}_i$ indicates that it is an estimated value.

Like text retrieval, due to the sparsity of the data, only considering the co-occurrence statistics is unreliable. One can smooth the estimate from the collection statistics; using the linear smoothing method [22], we have the following ranking formula:

$$\begin{aligned} o_u(i) &\equiv \sum_{i' \in L_u} \ln \Big(\lambda P(i'|i) + (1-\lambda) P(i')\Big) + \ln P(i) \\ &\equiv \sum_{i' \in L_u} \ln \Big(\lambda \frac{c(i',i)}{c(i)} + (1-\lambda) \frac{c(i')}{\sum_{i'} c(i')}\Big) + \ln \frac{c(i)}{\sum_{i'} c(i')} \end{aligned} \tag{6}$$

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

Alternatively, one can apply the Bayes-smoothing technique [22] to smooth the estimation. More formally, we have:

$$\hat{\theta}_i = p(i'|\hat{\theta}_i) \equiv \frac{c(i', i) + \mu \cdot p(i')}{c(i) + \mu} \tag{7}$$

where $\mu$ is the smoothing parameter and $p(i') \equiv \frac{c(i')}{\sum_{i'} c(i')}$. Replacing Eq. (7) into Eq. (5) results in the following Bayes-smoothing-based ranking formula:

$$o_u(i) \equiv \sum_{i' \in L_u} \ln(\frac{c(i', i) + \mu \cdot \frac{c(i')}{\sum_{i'} c(i')}}{c(i) + \mu}) + \ln \frac{c(i)}{\sum_i c(i)} \tag{8}$$

In summary, we have derived two ranking formulae in Eq. (6) and Eq. (8), respectively, by following the school of thinking in the language modelling approaches of text retrieval. The two scoring functions are item-based as the scoring relies on the co-occurrence statistics between items. It is worth noticing that a parallel user-based method can similarly be derived by considering the co-occurrence between users.

## 4 Risk-aware Ranking for Collaborative Filtering

As described, the classic language modelling approaches, thus including Eq. (6) and Eq. (8), consider the model parameters as unknown fixed constants, and apply point estimate such as the MLE, the linear-smoothing, or the Bayes-smoothing technique. The main drawback of this approach is that exact measures of the uncertainty associated with the estimation are not handled in a principled manner. As a result, unreliably-estimated items may be ranked highly in the ranked list, reducing the retrieval performance of the top-$N$ returned items.

To model the uncertainty of the estimate, we follow the Bayesian viewpoint, considering parameter $\theta_i$ itself has a probability distribution associated with it. We propose to use variance $Var(\theta_i)$ to summarize the uncertainty, inspired by the risk-aware language models introduced in [23]. A large variance indicates that the estimate is unreliable and its rank score should be penalized accordingly. Based on this, we have the following formula:

$$\hat{\theta}_i \equiv Mean(\theta_i) - \frac{b}{2} Var(\theta_i) \tag{9}$$

where $Mean(\theta_i)$ is the mean of $\theta_i$ while $Var(\theta_i)$ denotes its variance. $b > 1$, and it is a parameter that adjusts the risk preference and can be tuned from data.

**Table 1.** 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 (P-value $< 0.05$) over `SuggestLib` are marked as *.

| | **Top-**1 | **Top-**3 | **Top-**10 |
|---|---|---|---|
| **LM-LS** | 0.572 | 0.507 | 0.416 |
| **LM-BS** | **0.585***| **0.535*** | **0.456*** |
| **SuggestLib** | 0.547 | 0.509 | 0.421 |

(a) User Profile Length 5

| | **Top-**1 | **Top-**3 | **Top-**10 |
|---|---|---|---|
| **LM-LS** | 0.673 | 0.617 | **0.517*** |
| **LM-BS** | **0.674*** | **0.620*** | **0.517*** |
| **SuggestLib** | 0.664 | 0.604 | 0.503 |

(b) User Profile Length 10

| | **Top-**1 | **Top-**3 | **Top-**10 |
|---|---|---|---|
| **LM-LS** | 0.669 | 0.645 | 0.555 |
| **LM-BS** | **0.761*** | **0.684*** | **0.568*** |
| **SuggestLib** | 0.736 | 0.665 | 0.553 |

(c) User Profile Length 15

Here the user preference data is assumed to follow Multinomial distribution, and the conjugate prior is Dirichlet distribution. Thus, the mean and variance are obtained as follows:

$$Mean(\theta_i) = \frac{c(i', i)}{c(i)}, Var(\theta_i) = \frac{c(i', i)\Big(c(i) - c(i', i)\Big)}{c(i)^2\Big(c(i) + 1)\Big)} \tag{10}$$

## 5 Experiments

### 5.1 Data Sets and Experiment Protocols

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, frequency-counted user profiles. This paper adopts two implicit user profile data sets.

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 {userID, artistID, 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

**Table 2.** 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 (P-value $< 0.05$) over `SuggestLib` are marked as *.

| | **Top-1** | **Top-3** | **Top-10** |
|---|---|---|---|
| **LM-LS** | **0.306*** | **0.253*** | **0.208*** |
| **LM-BS** | 0.253 | 0.227 | 0.173 |
| **SuggestLib** | 0.168 | 0.141 | 0.107 |

(a) User Profile Length 5

| | **Top-1** | **Top-3** | **Top-10** |
|---|---|---|---|
| **LM-LS** | **0.325*** | **0.256*** | **0.207*** |
| **LM-BS** | 0.248 | 0.226 | 0.175 |
| **SuggestLib** | 0.224 | 0.199 | 0.150 |

(b) User Profile Length 10

| | **Top-1** | **Top-3** | **Top-10** |
|---|---|---|---|
| **LM-LS** | **0.322*** | **0.261*** | **0.211*** |
| **LM-BS** | 0.256 | 0.231 | 0.177 |
| **SuggestLib** | 0.271 | 0.230 | 0.171 |

(c) User Profile Length 15

constructing a personalized "tag cloud" for the purpose of exploratory search . 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.

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 obtains 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. Our experiments here consider the *recommendation precision*, which measures the proportion of recommended items that are ground truth items.

## 5.2 Performance

**The Language Models** We choose a state-of-the-art item ranking algorithm [7] discussed in Section 2 as our strong baseline. We adopt their implementation, the top-N suggest recommendation library [1], which is denoted as `SuggestLib`. The proposed language modelling approach of collaborative filtering in Eq. (6) is denoted as `LM-LS` while its variant using the Bayes' smoothing given in Eq. (8) is denoted as `LM-BS`. The optimal parameters are tuned by applying cross-validation.

[1] http://glaros.dtc.umn.edu/gkhome/suggest/overview

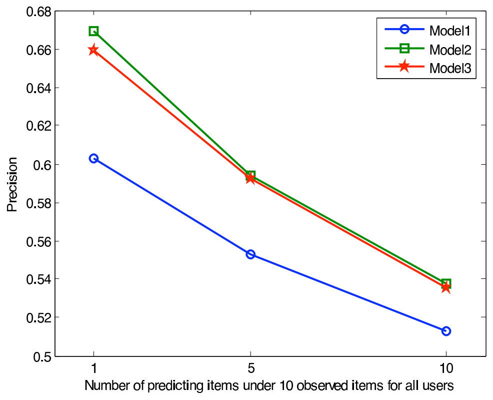


**Fig. 2.** Recommendation Precision in Last.FM data set. Model 1: the linear smoothing method; Model 2: risk-aware linear smoothing method; and Model 3: risk-aware Maximum Likelihood Estimate method.

The results are shown in Table 1 and 2. From the tables, we can see that our ranking methods, derived from the language models of text retrieval, performs consistently better than the heuristic ranking method, `SuggestLib`, over all the configurations. A Wilcoxon signed-rank test [11] is done to verify the significance. We believe that the effectiveness of our methods is due to the fact that the models naturally integrate frequency counts and probability estimation into the ranking formula. For the two smoothing methods, we have obtained a mixed result - in the Last.FM data, the Bayes smoothing method outperforms the linear smoothing, while in the `del.icio.us` data, the linear smoothing is better than the Bayes smoothing method.

**The Risk-aware Ranking** We continue our experiment with the risk-aware model given in Eq. (9). As we intend to investigate whether the added variance bit could improve the recommendation accuracy, the linear smoothing approach (denoted as Model 1) is now regarded as a baseline, Two different risk-aware models are evaluated: Model 2, using the linear smoothing to estimate the mean ($Mean(\theta_i)$), and Model 3, using the maximum likelihood to estimate the mean ($Mean(\theta_i)$). The results, under the three configurations top-1, top-10, and top-15, are shown in Fig. 2 and Table 3. We can see that Model 2 and Model 3 significantly outperform Model 1 in all configurations. Model 2 is slightly better than Model 3, implying that, the variance plays a more critical role than the smoothing from the collection. And, even without smoothing from the collection, the risk-model that considers the variance only provides a robust scoring function.

## 6 Conclusions

In this paper, we have presented a novel statistic model for item ranking in collaborative filtering. It is inspired by the widely-adopted language models of text

**Table 3.** Recommendation Precision in Last.FM. Model 1: the linear smoothing method; Model 2: risk-aware linear smoothing model ; and Model 3: risk-aware Maximum Likelihood Estimate model. A Wilcoxon signed-rank test is conducted and the significant ones (P-value $< 0.05$) over Model 1 the linear smoothing model are marked as *.

| Observed items | Method | Parameter | Top-10 | Top-5 | Top-1 |
|---|---|---|---|---|---|
| 5 | ① | λ=0.9949 | 0.4696 | 0.5081 | 0.5988 |
|  | ② | λ=0.989 b=22.9 | **0.4888*** | **0.5281*** | **0.6279*** |
|  | ③ | b =130.8 | 0.4786 | 0.5243 | 0.5925 |
| 10 | ① | λ=0.99718 | 0.5131 | 0.5530 | 0.6029 |
|  | ② | λ=0.9969 b=16 | **0.5378*** | **0.5938*** | **0.6694*** |
|  | ③ | b =170.8 | 0.5356 | 0.5925 | 0.6590 |
| 15 | ① | λ=0.998363 | 0.5632 | 0.6146 | 0.6757 |
|  | ② | λ=0.99712 b=17.01 | **0.5969*** | **0.6674*** | **0.7526*** |
|  | ③ | b =190.8 | 0.5784 | 0.6482 | 0.7339 |

retrieval. To consider the uncertainty of the parameter estimation and reflect it during the ranking, we then presented a risk-averse ranking model by considering the variance of the parameters. The experiments on two real data sets have shown that the significance of our approaches.

One of the assumptions in our model is that the items in users' profiles are independent of each other. This is unrealistic in practice. In the future, we intend to explore this dependence. It is also of great interest to study the method of combing content descriptions under the proposed framework.

## 7 Acknowledgement

The experiment of the risk-aware model was conducted by Mofei Han when he was working on his MSc thesis under the supervision of the author.

## 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. 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.

3. 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)*, 1998.
4. J. Canny. Collaborative filtering with privacy via factor analysis. In *SIGIR '02*, 2002.
5. M. Claypool, P. Le, M. Wased, and D. Brown. Implicit interest indicators. In *IUI '01*, 2001.
6. B. W. Croft and J. Lafferty. *Language Modeling for Information Retrieval.* Springer, 2003.
7. M. Deshpande and G. Karypis. Item-based top-N recommendation algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, 2004.
8. D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70, 1992.
9. J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR '99*, 1999.
10. T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Info. Syst.*, Vol 22(1):89–115, 2004.
11. D. Hull. Using statistical testing in the evaluation of retrieval experiments. In *SIGIR '93*, 1993.
12. R. Jin, L. Si, and C. Zhai. A study of mixture models for collaborative filtering. *Inf. Retr.*, 9(3):357–382, 2006.
13. C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval.* Cambridge University Press, 2008.
14. 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*, 2000.
15. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94*, 1994.
16. B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *WWW '01*, 2001.
17. J. Wang. *Relevance Models for Collaborative Filtering.* Delft University of Technology, ISBN 978-90-9022932-4, 2008.
http://web4.cs.ucl.ac.uk/staff/jun.wang/papers/phdthesis.pdf.
18. 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.
19. J. Wang, A. P. de Vries, and M. J. Reinders. Unified relevance models for rating prediction in collaborative filtering. *ACM Trans. on Information System (TOIS)*, 2008.
20. J. Wang, S. E. Roberston, A. P. de Vries, and M. J. T. Reinders. Probabilistic relevance models for collaborative filtering. *Journal of Information Retrieval*, 2008.
21. 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*, 2005.
22. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01*, 2001.
23. J. Zhu, J. Wang, M. Taylor, and I. Cox. Risky business: Modeling and exploiting uncertainty in information retrieval. In *SIGIR09*, 2009.