Esta es la versión de autor de la comunicación de congreso publicada en:
This is an author produced version of a paper published in:


DOI:  http://dx.doi.org/10.1145/2507157.2507191

Copyright:  © 2013 ACM

El acceso a la versión del editor puede requerir la suscripción del recurso
Access to the published version may require subscription
Probabilistic Collaborative Filtering with Negative Cross Entropy

Alejandro Bellogín*, Javier Parapar*, Pablo Castells*
*Information Access, Centrum Wiskunde & Informatica
*Information Retrieval Lab, Department of Computer Science, University of A Coruña
Information Retrieval Group, Department of Computer Science, Universidad Autónoma de Madrid
alejandro.bellogin@cwi.nl, javierparapar@udc.es, pablo.castells@uam.es

ABSTRACT

Relevance-Based Language Models are an effective IR approach which explicitly introduces the concept of relevance in the statistical Language Modelling framework of Information Retrieval. These models have shown to achieve state-of-the-art retrieval performance in the pseudo relevance feedback task. In this paper we propose a novel adaptation of this language modeling approach to rating-based Collaborative Filtering. In a memory-based approach, we apply the model to the formation of user neighbourhoods, and the generation of recommendations based on such neighbourhoods. We report experimental results where our method outperforms other standard memory-based algorithms in terms of ranking precision.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Information Search and Filtering

General Terms: Algorithms, Experimentation, Performance

Keywords: Recommender systems, Collaborative Filtering, Relevance Models

1. INTRODUCTION AND MOTIVATION

Personalised recommendation is a fertile research area with roots in the eighties, which started to attract wider attention in the mid-nineties when the first works on Collaborative Filtering came to light [6]. Three classical approaches to recommendation are distinguished: content-based recommendation (CB), based on the user's history and data (descriptions, content, features) of the items; collaborative filtering (CF), based on the history of similar users and items; and hybrid approaches, based on combining CB and CF recommendation. One of the earliest and most popular CF methods is the so-called k nearest-neighbours (kNN) approach, which is still being used today in many commercial systems.

A common formulation of a user-based kNN method is [6]:

\[ \hat{r}(u, i) = C \sum_{v \in N_k(u)} \text{sim}(u, v)r(v, i) \]  

where the known preference of a user \( u \) for a particular item \( i \) is given by a numeric rating \( r(u, i) \), and \( \hat{r}(u, i) \) denotes the system's estimate of this value for an item for which the degree of user interest is unknown to the system. To provide that estimation, the method takes into account the ratings \( r(v, i) \) provided by the \( k \) users \( v \) who are most similar to \( u \), commonly referred to as neighbours and denoted here as \( N_k(u) \). The function \( \text{sim}(u, v) \) measures the similarity between two users \( u \) and \( v \), and the constant \( C \) is a normalisation factor. Thus, the predicted rating of user \( u \) for item \( i \) is computed as a weighted average of \( u \)'s neighbours' ratings (deviations with respect to the user and neighbour average can also be considered [6]), where the weights are the similarities between \( u \) and her neighbours \( v \).

Relevance-Based Language Models [3] (or Relevance Models for short, RM), on the other hand, are among the best-performing ranking techniques in text retrieval. In this paper, we study how RM can be adapted to rating-based recommendation, and whether they may lead to enhancements in terms of ranking-based metrics such as precision and nDCG. We do so by establishing a mapping between the variables involved in RM and the ones in user-based CF. The mapping is non-trivial as, to begin with, RM is formulated in text IR on a triadic space (query, documents, words), whereas the CF space is typically dyadic (users and items). Based on a proposed mapping, we bring the adaptation to computable terms, giving rise to a workable and effective recommendation framework. The resulting approach comprises two separable subcomponents: a neighbour selection approach and a ranking function, which can be used on their own or together.

In the remaining of this paper, we will answer the following research questions: (RQ1) Are the Relevance-Based Language Models useful to identify neighbours in recommendation? (RQ2) Is there any advantage in using a complete probabilistic representation of the problem? In other terms, does it make sense to replace the CF weighted average by the RM cross entropy as used in IR, and does this approach work well in practice?

After presenting our proposed method in detail in the next section, we address the research questions experimentally on MovieLens data, testing the effectiveness of our overall approach, and comparing the separate effect of the two subcomponents. As we shall see, the empirical results validate the approach, showing performance improvements over state of the art memory-based alternatives.

2. RELEVANCE MODELLING FOR RECOMMENDATION

Relevance-Based Language Models, as first proposed in [3], view the original user search query \( Q \) as a short sample of words obtained from an underlying relevance model \( R \). If one aims to add more words from \( R \) to the query then it is reasonable to choose those words with the highest estimated probability given a sample of observed words generated by the relevance model for the query. In [3] two different estimations for RM were originally presented, namely RM1 and RM2 from which, in our present approach, we adopt the former. In RM1 it is assumed that the query words \( q_i \in Q \) and the words \( w \) in the relevant documents are sampled identically and independently from a unigram distribution (i.i.d. sampling).
used in IR to incorporate the information learnt from the RM. We describe these two parts of our approach in the next two sections.

### 2.2 Neighbour Selection

The equivalent estimation to RM1 for neighbour selection according to our proposed mapping goes as follows. Modelled after RM1, the probability of a neighbour $v$ under the relevance model $R_u$ for a given user $u$ is estimated as:

$$p(v|R_u) = \sum_{i \in PRS(u)} p(i)p(v|i) \prod_{j \in I(u)} p(j|i) \quad (3)$$

where $p(i)$ is the probability of the item $i$ in the collection, $p(v|i)$ is the probability of the neighbour $v$ given the item $i$, and $p(j|i)$ is the conditional probability of item $i$ given another item $j$. As presented in Figure 1, $I(u)$ corresponds to the set of items rated by user $u$, and $PRS(u) \subset I(u)$ is the subset of items rated by $u$ above some specific threshold, i.e. items with a favorable rating value indicating the user likes them.

Besides folding a triadic space into a dyadic one, in our formulation the probabilistic RM framework is turned upside down to some extent. The ground probabilistic model upon which RM are formulated in text IR reflects a process in which words are sampled according to a certain (indirectly observed) distribution. While the relevance model steers the generation of words in text IR, in our approach it drives the sampling of users, or to be more specific, user profiles (i.e. a history of item ratings). In this perspective, $p(v|R_u)$ can be read as the probability to observe the ratings entered by user $v$ if her underlying tastes are defined by $R_u$.

### 2.3 Item Ranking

In document retrieval, once the terms with the highest estimated probability under the relevance model are selected for expansion, they are used to produce a second ranking by means of the negative cross entropy scoring function. In this paper we propose to use this very same method for ranking the item collection with respect to the preferences of the user $u$, that is, $\hat{r}(u,i) = H(p(\cdot|R_u); p(\cdot|i)) = \sum_{v \in C(u)} p(v|R_u) \log p(v|i)$. With this formulation, we rank the items according to the distance between the item and user probability models, so that the closest – more similar and (hypothetically) more relevant – items are ranked higher. Now, following the kNN CF strategy, we propose to restrict the sum over users $v$ to a subset $N_k(u) \subset C(u)$ of $k$ nearest neighbours with most similar tastes to user $u$. We propose the use of $p(v|R_u)$ as the similarity function, such that we select the neighbourhood for a given user $u$ as the set of $k$ users in the collection with the highest probability to share the user relevance model $R_u$. The resulting ranking function is:

$$\hat{r}(u,i) = H(p(\cdot|R_u); p(\cdot|i)|N_k(u)) = \sum_{v \in N_k(u)} p(v|R_u) \log p(v|i) \quad (4)$$

The result of this restriction is that we avoid the residual effect of a long tail of users with a low $p(v|R_u)$ (Eq. 3), whose contribution to the prediction of user tastes does not pay off the incurred computational cost. This conforms to the principle of neighbour selection in kNN CF, but also of keyword selection (a cutoff of most probable words given the relevance model) in query expansion by RM.

### 2.4 Parameter Estimation

Finally, some estimation details remain to be defined. We initially consider $p(i)$ and $p(u)$ as uniform priors, i.e. every neighbour $v \in N_k(u)$ has the same probability of being sampled, and same for every item in the collection. The conditional distribution $p(j|i)$ is computed by the maximum likelihood estimate $p_{ml}(j|i) = |U(j)\cap U(i)|/|U(i)|$, where $U(i)$ is the set of users who rated the item $i$. The probability of a user given an item is computed by Bayesian inversion $p(u|i) = p(i|u)p(u)/p(i)$, and the probability of an item

<table>
<thead>
<tr>
<th>RM for Retrieval</th>
<th>RM for Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>query $q$</td>
<td>target user $u$</td>
</tr>
<tr>
<td>query words $q_1, ..., q_n$</td>
<td>items rated by user $I(u)$</td>
</tr>
<tr>
<td>pseudo relevance set $PRS$</td>
<td>positively rated $PRS(u)$</td>
</tr>
<tr>
<td>candidate expansion terms</td>
<td>candidate user’s neighbours</td>
</tr>
</tbody>
</table>

Figure 1: Correspondence between the elements involved in Relevance-Based Language Models for document retrieval and its adaptation to item recommendation.
given a user is estimated by maximum likelihood smoothed with the probability in the collection (background collection model), using the Jelinek-Mercer smoothing [10]:

\[ p_A(i|u) = (1 - \lambda)p_{ml}(i|u) + \lambda p(i|C) \] (5)

where \( p_{ml}(i|u) \) and \( p(i|C) \) are estimated as:

\[ p_{ml}(i|u) = \frac{\sum_{u \in C_u} r(u, i)}{\sum_{u \in C_u} |V_u| r(u, j)} \quad p(i|C) = \frac{\sum_{v \in C_v} r(u, j)}{|C_u|} \] (6)

where \( C_u \) (\( C_f \)) is the set of users (items) in the collection.

3. EMPIRICAL EVALUATION

3.1 Evaluation Methodology

We test the proposed approach on two publicly available datasets\(^1\). The first one is MovieLens 100K, which contains 100,000 ratings on 1,682 items by 943 users. We perform a 5-fold cross validation using the data splits contained in the public distribution, which retain 80\% of the data for training, and the rest for testing. Using the terminology in [2], we report the results obtained following the TestItems evaluation approach described in [1]. In the TestItems methodology, a ranking is generated for each user by predicting a score for every item that has a rating in the test set. We can then compute any standard IR metric on the ranking, such as precision, nDCG or MRR. We report here the values for precision at 5 and 50, and nDCG at 5 and 10. We also report the user space coverage metric (cvg) as defined in [7], that is, the number of users for which the system is able to recommend at least one item.

Furthermore, for assessing the robustness of the methods across collections, we took the optimal parameter values for every method in terms of P@5 in the MovieLens 100K dataset, and tested the methods with those values on a second (and disjoint) public dataset, MovieLens 1M, containing 1,000,209 ratings by 6,040 users to 3,900 items. We do 5-fold cross validation again in this larger dataset – not with the aim of training the parameters but to enhance the randomisation of the data split, following standard methodology in the evaluation of recommender systems.

3.2 Baselines

We compare our methods against different well-known state-of-the-art recommenders. We take a user-based CF where ratings are predicted as in Eq. 1, with Pearson correlation as similarity measure (UB [6]). We also test our methods against the graph partitioning method [2] based on Normalised Cut (NC) with Pearson similarity (NC+P) which has demonstrated important improvements for neighbour selection. This method basically clusters the users in the collection by finding the optimal cut (NC) of the computed graph, where Pearson similarity is used to weight the edges between items. Then, it selects a neighbourhood \( NC_k(u) \) that outputs those users who belong to the same cluster as the target user \( u \) among the \( k \) clusters found by the algorithm; finally, the predicted rating is produced like in Eq. 1 with \( W_k(u) = NC_k(u) \).

Furthermore, we compare our approach to related work on recommendation algorithms that adapt IR models. Specifically, we compare our methods against the relevance model for log-based CF (UIR) proposed in [8], as formulated in the 16th page of that paper:

\[ \hat{r}(u, i) \sim \sum_{v \in U(i) \cap c(u, v) > 0} \log \left( 1 + \frac{(1 - \lambda)p_{ml}(v|u, r)}{\lambda p(v|r)} \right) + |U(i)| \log \lambda \] (7)

\(^1\)Both available at http://www.grouplens.org/node/73

Figure 2: Evolution of the performance of the compared methods in terms of P@5 when varying \( k \) on the MovieLens 100K collection

where \( c(u, v) \) is the number of items rated by both users \( u \) and \( v \) and the rest of probabilities are estimated as follows:

\[ p_{ml}(v|u, r) \propto \frac{c(v, u)}{c(u)} \quad p(v|r) \propto c(v) \]

We also include as a baseline the unified user-based model (URM) presented in [9], which also introduces variations in the probability estimations. More specifically, we use the Eq. 40a from [9] which goes as follows:

\[ \hat{r}(u, i) = \sum_{v \in U(i) \cap c(u, v) > 0} e^{-\frac{1 - \cos (u, v)}{h_v^2}} \]

\[ \sum_{v \in U(i) \cap c(u, v) > 0} e^{-\frac{1 - \cos (u, v)}{h_v^2}} \] (8)

where \( \cos (u, v) \) is a cosine kernel based similarity measure between users \( u \) and \( v \) represented as vectors in an item space, where the missing ratings can be replaced by a constant value of 0, or by the average rating value. This approach requires a prior learning of the value \( h_v \) (the kernel bandwidth window parameter) by expectation-maximisation [9]. In order to provide a fair comparison, we use here the best value \( h_v^2 = 0.79 \) reported in [9], which was tuned on the very same collection.

3.3 Results and Discussion

We now assess the research questions RQ1 (are RM models useful to identify neighbours in recommendation?) and RQ2 (can we achieve better performance with a complete probabilistic formulation of the CF problem?), raised at the beginning of the paper, in light of the results, summarised in Figure 2 and Table 1. Figure 2 shows how sensitive the evaluated methods are to the neighbourhood size (or number of clusters for NC+P). In Table 1 we present the results for different evaluation metrics using the two datasets described previously. For our approaches, we first test a hybrid combination of our neighbour selection approach using Eq. 3 followed by the standard user-based CF formulation with Pearson similarity (Eq. 1); we refer to this method as RMUB. Additionally, we denote by RMCE the combination of the relevance model (Eq. 3) followed by the cross entropy ranking function (Eq. 4).

To address RQ1 we compare the performance of RMUB against that of the other baselines. We observe that the RMUB method clearly outperforms the UB, UIR and URM baselines for P@5, nDCG@5 and nDCG@10, in both MovieLens 100K (see Table 1a) and 1M (Table 1b). Its performance in terms of P@50, however, is similar to some of the baselines, showing that our method is able to rank higher than such baselines interesting items for the user,
at least until some reasonable cut-off, which in this case seems to be 50. Moreover, since this method takes two parameters \((k \text{ and } \lambda)\), we analyse now its performance sensitivity. Due to space constraints, we only explore in Figure 2 the neighbourhood size \(k\), but we include the performance for two values of \(\lambda\) – the optimal \((\lambda = 0.1)\) and neutral \((\lambda = 0.5)\) configurations – where a negligible difference is obtained. We can also notice in the same figure that the baseline NC+P obtains a much better performance than RMUB, consistently with results reported in [2]. Table 1 also shows that the coverage results for the NC+P baseline are better than for RMUB in their optimal settings. We further observed (we omit the detailed results here for the sake of space) that the coverage of NC+P decreases with larger \(k\)'s (as reported in [2]), whereas the coverage of RMUB increases, but at the expense of losing precision. All in all, our answer to RQ1 is that relevance models as a standalone method for neighbourhood selection are useful but not optimal.

To address RQ2 we focus on the RMCE approach. In this case our method consistently achieves statistically significant improvements against all the baselines for every reported metric, achieving a 100% improvement with respect to the best baseline (NC+P, which already demonstrated performance superior to a standard Matrix Factorisation baseline [2]). Furthermore, RMCE does not suffer from the coverage problem, although it is highly sensitive to the smoothing parameter, as Figure 2 shows. For this approach, the optimal parameter in MovieLens 100K is 0.9, that is, a configuration which relies heavily on the background collection model. This makes sense since, in such a setting, RMCE promotes popular recommendations which are known to perform very well in this dataset. Furthermore, as the same figure shows, the method outperforms the baselines even for a neutral setting of the smoothing parameter. Thus, we may conclude with a positive answer for RQ2, since the combination of RM-based neighbours and negative cross entropy as scoring function (RMCE) results in important improvements over the existing state-of-the-art CF methods.

Finally, it is interesting to observe how differently the RMUB and RMCE approaches perform, taking into account that the neighbours used for both methods are the same. Our hypothesis is that the classical user-based CF formulation (Eq. 1) has no formal justification to generate item rankings, mainly because it was proposed to predict ratings, not to rank items according to these predictions, in agreement with [4]. By using negative cross entropy as the retrieval function, the ranking shifts from guessing rating values to assessing relevance distribution distances, which proves to reward relevance over rating value accuracy, as we may notice in terms of relevance-oriented ranking metrics; according to the results, we can conclude that like in IR, negative cross entropy maintains its good properties in recommendation tasks.

## 4. CONCLUSIONS AND FUTURE WORK

We have presented a new approach to collaborative filtering in Recommender Systems. Our approach adapts the negative cross entropy ranking principle from the Relevance-Based Language Models in document retrieval to the item recommendation problem, combined with a neighbour selection step, drawing from the kNN CF principle. We tested our proposal first only for neighbourhood selection and then also for producing the recommendation, finding that the largest improvements are achieved when using the complete probabilistic model. Comparisons of our approach with other highly performing baselines shows consistent significant improvements for every evaluation metric. As future work, we plan to explore the behaviour of our proposal on larger datasets, and study how further the improvements can go with alternative estimation formulations such as different smoothing methods and RM models. We have also researched further alternatives in how the IR and recommendation variables are mapped [5], although more research is still needed on this point.

## 5. ACKNOWLEDGMENTS

This work was funded by Secretaría de Estado de Investigación, Desarrollo e Innovación from the Spanish Government under projects TIN2012-33867 and TIN2011-28538-C02.

## 6. REFERENCES


### Table 1: Summary of comparative effectiveness. Best values for each collection and metric are in bold. Statistical significant improvements w.r.t. UB, NC+P, UIR, URM, RMUB and RMCE are superscripted with a, b, c, d, e and f respectively. Trained parameter values are \(k = 50; k = 200; h^* = 0.79; \lambda = 0.1; k = 50\) and \(\lambda = 0.1\); and \(k = 700\) and \(\lambda = 0.9\) respectively.

<table>
<thead>
<tr>
<th></th>
<th>P@5</th>
<th>nDCG@5</th>
<th>nDCG@10</th>
<th>P@50</th>
<th>cvg</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MovieLens 100K</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UB</td>
<td>0.010* &amp;</td>
<td>0.311* &amp;</td>
<td>0.037* &amp;</td>
<td>0.034* &amp;</td>
<td>100%</td>
</tr>
<tr>
<td>NC+P</td>
<td>0.111* &amp;</td>
<td>0.097* &amp;</td>
<td>0.095* &amp;</td>
<td>0.058* &amp;</td>
<td>83%</td>
</tr>
<tr>
<td>UIR</td>
<td>0.004</td>
<td>0.002</td>
<td>0.002</td>
<td>0.002</td>
<td>100%</td>
</tr>
<tr>
<td>URM</td>
<td>0.005</td>
<td>0.003</td>
<td>0.018</td>
<td>0.054*</td>
<td>100%</td>
</tr>
<tr>
<td>RMUB</td>
<td>0.081* &amp;</td>
<td>0.064* &amp;</td>
<td>0.040* &amp;</td>
<td>0.050* &amp;</td>
<td></td>
</tr>
<tr>
<td>RMCE</td>
<td>0.124* &amp;</td>
<td>0.002* &amp;</td>
<td>0.000* &amp;</td>
<td>0.000* &amp;</td>
<td></td>
</tr>
<tr>
<td><strong>MovieLens 1M</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UB</td>
<td>0.013* &amp;</td>
<td>0.033* &amp;</td>
<td>0.030* &amp;</td>
<td>0.046*</td>
<td>100%</td>
</tr>
<tr>
<td>NC+P</td>
<td>0.037* &amp;</td>
<td>0.034* &amp;</td>
<td>0.036* &amp;</td>
<td>0.046* &amp;</td>
<td>99%</td>
</tr>
<tr>
<td>UIR</td>
<td>0.001</td>
<td>0.001</td>
<td>0.001</td>
<td>0.001</td>
<td>100%</td>
</tr>
<tr>
<td>URM</td>
<td>0.001</td>
<td>0.001</td>
<td>0.006</td>
<td>0.034* &amp;</td>
<td>100%</td>
</tr>
<tr>
<td>RMUB</td>
<td>0.075* &amp;</td>
<td>0.061* &amp;</td>
<td>0.054* &amp;</td>
<td>0.038* &amp;</td>
<td>100%</td>
</tr>
<tr>
<td>RMCE</td>
<td>0.187* &amp;</td>
<td>0.007* &amp;</td>
<td>0.008* &amp;</td>
<td>0.008* &amp;</td>
<td>100%</td>
</tr>
</tbody>
</table>