Language Models for Collaborative Filtering Neighbourhoods

Daniel Valcarce, Javier Parapar, and Álvaro Barreiro

Information Retrieval Lab
Computer Science Department
University of A Coruña, Spain
{daniel.valcarce,javierparapar,barreiro}@udc.es

Abstract. Language Models are state-of-the-art methods in Information Retrieval. Their sound statistical foundation and high effectiveness in several retrieval tasks are key to their current success. In this paper, we explore how to apply these models to deal with the task of computing user or item neighbourhoods in a collaborative filtering scenario. Our experiments showed that this approach is superior to other neighbourhood strategies and also very efficient. Our proposal, in conjunction with a simple neighbourhood-based recommender, showed a great performance compared to state-of-the-art methods (NNCosNgbr and PureSVD) while its computational complexity is low.

Keywords: Recommender systems, Language Models, collaborative filtering, neighbourhood

1 Introduction

Recommender systems aim to provide useful items of information to the users. These suggestions are tailored according to the users's tastes. Considering the increasing amount of information available nowadays, it is hard to manually filter what is interesting and what is not. Additionally, users are becoming more demanding—they do not conform with traditional browsing or searching activities, they want relevant information immediately. Therefore, recommender systems play a key role in satisfying the users' needs.

We can classify recommendation algorithms in three main categories: content-based systems, which exploit the metadata of the items to recommend similar ones; collaborative filtering, which uses information of what other users have done to suggest items; and hybrid techniques, which combine both content-based and collaborative filtering approaches [15]. In this paper, we focus on the collaborative filtering scenario. Collaborative techniques ignore the content of the items since they merely rely on the feedback from other users. They tend to perform better than content-based approaches if sufficient historical data is available. We can distinguish two main types of collaborative methods. On the one hand, model-based techniques learn a latent factor representation from the data after a training process [10]. On the other hand, neighbourhood-based methods (also called memory-based algorithms) use the similarities among past user-item interactions [6]. Neighbourhood-based recommenders, in turn, are classified in two categories: user-based and item-based approaches depending on which type of similarities are computed. User-based recommenders rely on user neighbourhoods (i.e., they recommend items that similar users like). By contrast, item-based algorithms compute similarities between items (i.e., two items are related if users rate them in a similar way).

Neighbourhood-based approaches are simpler than their model-based counterparts because they do not require a previous training step—still, we need to compute the neighbourhoods. Multiple approaches to generate neighbourhoods exist in the literature [6] because this phase is crucial in the recommendation process. The effectiveness of these type of recommenders depends largely on how we calculate the neighbourhoods. A popular approach consists in computing the k Nearest Neighbours according to a pairwise similarity metric such as Pearson's correlation coefficient, adjusted cosine or cosine similarity.

Traditionally, recommender systems were designed as rating predictors; however, it has been acknowledged that it is more interesting to model the recommendation problem as an item ranking task [1, 8]. Top-N recommendation is the term coined to name this new perspective [4]. For this task, the use of Information Retrieval techniques and models is attracting more and more attention [2, 13, 17, 20]. The reason is that these methods were specifically conceived for ranking documents according to an explicit query. However, they can also rank items using the user's profile as an implicit query.

Previous work has found that the cosine similarity yields the best results in terms of accuracy metrics in the neighbourhood computation process [4]. In fact, it surpasses Pearson's correlation coefficient which is, by far, the most used similarity metric in the recommender system literature [6]. Thinking about cosine similarity in terms of retrieval models, we can note that it is the basic distance measure used in the Vector Space Model [16]. Following this analogy between Information Retrieval and Recommender Systems, if the cosine similarity is a great metric for computing neighbourhoods, it sounds reasonable to apply more sophisticated representations and measures to this task used in other more effective retrieval models. Thus, in this paper we focus on modelling the finding of user and item neighbourhoods as a text retrieval task. In particular, we propose an adaptation of the Language Modelling retrieval functions as a method for computing neighbourhoods. Our proposal leverages the advantages of this successful retrieval technique for calculating collaborative filtering neighbourhoods. Our proposal—which can be used in a user or item-based approach—in conjunction with a simple neighbourhood algorithm surpasses state-of-the-art methods (NNCosNgbr and PureSVD [4]) in terms of accuracy and is also very efficient.

2 Background

An extensive literature has studied several neighbourhood-based approaches because they are simple, interpretable and efficient [4–6, 9]. After calculating the neighbourhoods, the recommendation process consists in computing correlations between item or user neighbours. Item-based approaches are usually preferred [4–6] because the number of items is usually smaller than the users. This enables efficient computation of the neighbourhoods. Also, they have been shown to report better results in terms of accuracy than user-based approaches [5, 6]. Also, item-based recommendations are easy to justify with explanations such as "you would like item B because you liked item A". However, item-based methods may generate less serendipitous recommendations because they tend to recommender similar items to those rated by the user [6]. In contrast, user-based approaches recommend items that similar users enjoyed. Thus, it is possible to suggest items that strongly differ from the ones rated by the target user.

2.1 Non-Normalised Cosine Neighbourhood (NNCosNgbr)

Non-Normalised Cosine Neighbourhood (NNCosNgbr) is an effective item-based neighbourhood algorithm presented in [4]. For computing the k Nearest Neighbours, this method uses cosine similarity instead of Pearson's correlation coefficient because the former is computed over all the ratings whilst the latter relies only on the shared ratings. Moreover, they introduced a shrinking factor based on common ratings into the similarity metric [9]. This modification penalises the similarity between very sparse vectors [4]. Additionally, NNCosNgbr removes user and item biases according to the definition in [9]. The predicted score $\hat{r}_{u,i}$ for the user u and the item i is given by the following expression:

$$\hat{r}_{u,i} = b_{u,i} + \sum_{j \in J_i} s_{i,j} \left(r_{u,j} - b_{u,j} \right) \tag{1}$$

where $b_{u,i}$ denotes the bias for the user u and the item i (computed as in [9]); $s_{i,j}$, the cosine similarity between items i and j; J_i, the neighbourhood of the item i, and $r_{u,j}$, the rating that the user u gave to the item j. The major difference between this method and the standard neighbourhood approach [6] is the absence of the normalising denominator. Since we are not interested in predicting ratings, we do not worry about getting scores in a fixed range. On the contrary, this method fosters those items with high ratings by many neighbours [4, 5, 9].

2.2 Language Models (LM)

Language Models (LM) represent a successful framework within the Information Retrieval (IR) field. Ponte and Croft presented the first approach of using Language Models for the text retrieval task in 1998 [14]. Nowadays, the use of Language Models has become so popular in the field that they have been improved to address several IR tasks achieving state-of-the-art performance [22]. Compared to previous techniques, the main contributions of these models are their solid statistical foundation and their interpretability [14, 22].

The Language Modelling framework is a formal approach with a sound statistical foundation. It models the occurrences of words in the documents and queries as a random generative process—usually, using a multinomial distribution. Within this framework, we infer a language model for each document in the collection. To rank those documents according to a user's query, we estimate the posterior probability of each document d given the particular query q, $p(d|q)$:

$$p(d|q) = \frac{p(q|d)\,p(d)}{p(q)} \overset{\text{rank}}{=} p(q|d)\,p(d) \tag{2}$$

where $p(q|d)$ is the query likelihood and $p(d)$, the document prior. We can ignore the query prior $p(q)$ because it has no effect in the ranking for the same query. Usually, a uniform document prior is chosen and the query likelihood retrieval model is used. The most popular approach in IR to compute the query likelihood is to use a unigram model based on a multinomial distribution:

$$p(q|d) = \prod_{t \in q} p(t|d)^{c(t,q)} \tag{3}$$

where $c(t, q)$ denotes the count of term t in the query q. The conditional probability $p(t|d)$ is computed via the maximum likelihood estimate (MLE) of a multinomial distribution smoothed with a background model [23].

3 WSR and LM for Neighbours

In this section, we explain how we designed our neighbourhood algorithm WSR within the Language Modelling framework. First, we propose our recommendation algorithm and, next, we explain how we compute neighbours.

3.1 Neighbourhood-based Recommender for Ranking: WSR

Our recommendation algorithm stems from NNCosNgbr method and each modification described in this section was evaluated in Sec. 4.2. First, we kept the biases (see Eqs. 4 and 5), instead of removing them as in Eq. 1. Removing biases is very important in rating prediction recommenders because it allows to estimate ratings more accurately [6, 9], however it is useless on the top-N recommendation because we are concerned about rankings. Moreover, this process adds an extra parameter to tune [9]. Next, we focused on the similarity metric. In [4], the authors introduced a shrinking factor into the cosine metric to promote those similarities that are based on many shared ratings. This shrinkage procedure has shown good results in previous studies based on error metrics [6,9] at the expense of putting an additional parameter into the model. However, we found that its inclusion is detrimental in our scenario. This is reasonable because the main advantage of cosine similarity over other metrics such as Pearson's correlation coefficient is that it considers non-rated values as zeroes. In this way, cosine already takes into account the amount of co-occurrence between vectors of ratings which makes unnecessary the use of a shrinkage technique.

In conclusion, the final formula of the recommendation algorithm is a weighted sum of the ratings of the neighbours, which we coined as WSR (Weighted Sum Recommender). Eqs. 4 and 5 are the user and item-based versions, respectively.

$$\hat{r}_{u,i} = \sum_{v \in V_u} s_{u,v}\, r_{v,i} \qquad (4) \qquad\qquad \hat{r}_{u,i} = \sum_{j \in J_i} s_{i,j}\, r_{u,j} \qquad (5)$$

where s is the cosine similarity between the user or item vectors. V_u is the neighbourhood of user u, as J_i is the neighbourhood of item i.

3.2 Neighbourhoods using Language Models

Preliminary tests showed that our algorithm (Eqs. 4 and 5) performs very well compared to more sophisticated ones using plain cosine similarity and evaluating ranking quality. In fact, techniques such as biases removal or similarity shrinkage worsened the performance and introduced additional parameters in the model. Major differences in terms of ranking accuracy metrics occur when varying the neighbourhood computation method. In particular, our experiments showed that cosine similarity is a great metric for computing k Nearest Neighbours (k-NN). This process is analogous to the document ranking procedure in the Vector Space Model [16] if the target user plays the role of the query and the rest of the users are the documents in the collection. The outcome of this model will be a list of neighbours ordered by decreasing cosine similarity with respect to the user. Thus, choosing the k nearest neighbours is the same as taking the top k results using the user as the query.

Language Models (LM) are a successful retrieval method [14, 22] that deals with data sparsity [23], enables to introduce a priori information and performs document length normalisation [11]. Recommendation algorithms could benefit from LM because: user feedback is sparse, we may have a priori information and the profile sizes vary. We adapted LM framework to the task of finding neighbourhoods in a user or item-based manner. If we choose the former, we can model the generation of ratings by users as a random process given by a probability distribution (as Language Models do with the occurrences of terms). In this way, we can see documents and queries as users and terms as items. Thus, the retrieval procedure results in finding the nearest neighbours of the target user (i.e., the query). Analogously, we can flip to the item-based approach. In this case, the query plays the role of the target item while the rest of items play the role of the documents. In this way, a retrieval returns the most similar items.

The user-based analogy between the IR and recommendation tasks has already been stated. The consideration of a multinomial distribution of ratings has been used in [2, 13] under the Relevance-Based Language Modelling framework for computing recommendations. In our Language Modelling adaptation for calculating neighbourhoods, we can estimate the probability of a candidate neighbour v given a user u as follows:

$$p(v|u) \overset{\text{rank}}{=} p(v)\, p(u|v) = p(v) \prod_{i \in \mathcal{I}_u} p(i|v)^{r_{u,i}} \qquad (6)$$

where $\mathcal{I}_u$ are the items rated by user u. Here we only present the user-based approach for the sake of space: the item-based counterpart is derived analogously.

Language Modelling Smoothing. The probability of an item i given the user v, $p(i|v)$, is given by the MLE smoothed with the probability of an item in the collection. We explored three well-known smoothing methods (Absolute Discounting, Jelinek-Mercer and Dirichlet Priors [23]) that were recently applied to collaborative filtering [19] analysing their effects using Relevance-Based Language Models. For each method, the probability in the collection is computed as follows:

$$p(i|\mathcal{C}) = \frac{\sum_{v \in \mathcal{U}} \mathrm{r}_{v,i}}{\sum_{j \in \mathcal{I},\, v \in \mathcal{U}} \mathrm{r}_{v,j}} \tag{7}$$

Absolute Discounting (AD)

$$p_\delta(i|u) = \frac{\max(\mathrm{r}_{u,i} - \delta, 0) + \delta\, |\mathcal{I}_u|\, p(i|\mathcal{C})}{\sum_{j \in \mathcal{I}_u} \mathrm{r}_{u,j}} \tag{8}$$

Jelinek-Mercer (JM)

$$p_\lambda(i|u) = (1-\lambda)\, \frac{\mathrm{r}_{u,i}}{\sum_{j \in \mathcal{I}_u} \mathrm{r}_{u,j}} + \lambda\, p(i|\mathcal{C}) \tag{9}$$

Dirichlet Priors (DP)

$$p_\mu(i|u) = \frac{\mathrm{r}_{u,i} + \mu\, p(i|\mathcal{C})}{\mu + \sum_{j \in \mathcal{I}_u} \mathrm{r}_{u,j}} \tag{10}$$

We employ this Language Modelling approach only for computing neighbourhoods. Cosine similarity is still used in our WSR algorithm as the similarity in Eqs. 4 and 5. In this way, we generate recommendations independently of the neighbourhood strategy. In fact, we can use our proposal for computing neighbours in any neighbourhood-based algorithm.

4 Experiments

We ran our experiments on four collections: MovieLens 100k and 1M[1] film dataset, the R3-Yahoo! Webscope Music[2] dataset and the LibraryThing[3] book dataset. We present the details of these collections in Table 1.

We used the splits that MovieLens 100k and R3 Yahoo! provide for evaluation purposes. Since the MovieLens 1M and LibraryThing collections do not include predefined splits, we put 80% of the ratings of each user in the training subset and the rest in the test subset randomly.

[1] `http://grouplens.org/datasets/movielens/`
[2] `http://webscope.sandbox.yahoo.com`
[3] `http://www.macle.nl/tud/LT/`

Table 1: Datasets statistics

Dataset	**Users**	**Items**	**Ratings**	**Density**
MovieLens 100k	943	1,682	100,000	6.305%
MovieLens 1M	6040	3,706	1,000,209	4.468%
R3-Yahoo!	15,400	1,000	365,703	2.375%
LibraryThing	7,279	37,232	749,401	0.277%

4.1 Evaluation methodology

Instead of evaluating recommenders in the rating prediction task, we focused on the top-N recommendation task [4, 8]. We used precision-oriented metrics [1] but also other diversity and novelty measures [8]. We followed the *TestItems* approach to create the rankings [1]: for each user, we rank all the items that have a rating in the test set and have not been rated by the target user.

We evaluated all the metrics at a specified cut-off rank because we wanted to focus on how the recommenders behave in top positions of the rankings—users seldom consider more than the top suggestions. We used Normalised Discounted Cumulative Gain (nDCG) for quantifying the quality of the ranking using the ratings in the test set as graded relevance judgements. In particular, we employed the *standard nDCG* as formulated in [21]. We measured diversity using the complement of the Gini index [7] (the index is 0 when a single item is recommended for every user, 1 when all the items are equally recommended among the users). Finally, we computed the mean self-information (MSI) to measure the ability of the system to generate unexpected—unpopular—recommendations [24].

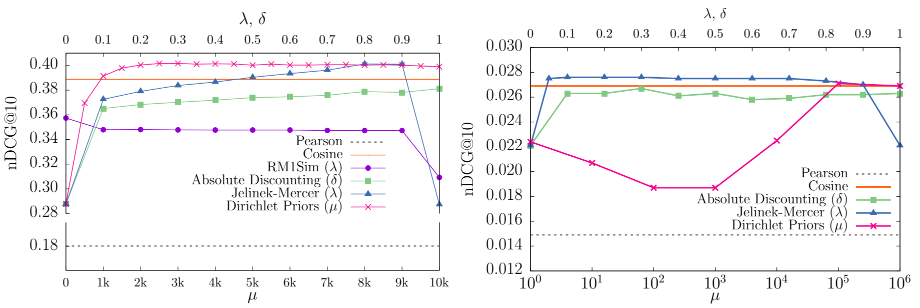
4.2 Testing WSR versus NNCosNgbr

In this section, we analyse the different options in WSR and NNCosNgr algorithms described in Sec. 3.1. Table 2 shows the best values of nDCG@10. We used cosine as the similarity metric in k-NN and tuned the number of nearest neighbours from $k = 50$ to 250 in steps of 50 neighbours. We chose a similarity shrinking factor of 100 as recommended in [4]. Biases were computed using L2 regularisation with a factor of 1 [9]. The first row corresponds to the NNCosNgbr algorithm [4]. The last two rows are the WSR method, our proposal for recommendation generation (Eqs. 5 and 4, respectively).

WSR variants performed the best and they significantly surpass NNCosNgbr (first row) in every dataset. The user-based approach reported the best figures on the dense film datasets while the item-based algorithm yielded the best results on the sparse songs and books collections. However, there are only statistically significant differences between these two methods in the LibraryThing dataset. This result agrees with the literature about neighbourhoods methods [4–6]: item-based approaches tend to work well on sparse datasets because they compute similarities among items which often contain more dense information than users.

Table 2: nDCG@10 best values on MovieLens 100k, Movielens 1M, R3-Yahoo! and LibraryThing datasets for different neighbourhood algorithms. The first column indicates whether a user-based (UB) or item-based (IB) approach is followed; the second, the pairwise similarity; the third, which treatment the user and item biases receive. Statistically significant differences (Wilcoxon two-sided test $p < 0.01$) with respect to the first, second, third and fourth method are superscripted with a, b, c and d, respectively. Values underlined are not statistically different from the best value.

Alg	**Type**	**Similarity**	**Bias**	**ML 100k**	**ML 1M**	**R3**	**LT**
NNCosNgr	IB	ShrunkCosine	Remove	0.1427	0.1042	0.0138	0.0550
NNCosNgr'	IB	ShrunkCosine	Keep	0.3704^{a}	0.3334^{a}	$\underline{0.0257}^{a}$	0.2217^{ad}
WSR	IB	Cosine	Keep	$\underline{0.3867}^{ab}$	$\underline{0.3382}^{ab}$	$\mathbf{0.0274}^{ab}$	$\mathbf{0.2539}^{abd}$
WSR	UB	Cosine	Keep	$\mathbf{0.3899}^{ab}$	$\mathbf{0.3430}^{ab}$	$\underline{0.0261}^{a}$	0.1906^{a}



(a) MovieLens 100k dataset, user-based approach, $k = 50$ neighbours

(b) R3-Yahoo! dataset, item-based approach, $k = 100$ neighbours

Fig. 1: Comparison in terms of nDCG@10 among different strategies (Pearson, cosine, RM1Sim and Language Models with AD, JM and DP) for computing neighbourhoods. Recommendations are computed using WSR algorithm.

4.3 Language Models for User Neighbourhoods

In this section, we compare our Language Modelling approach for computing neighbourhoods (see Eq. 6) against Pearson's correlation coefficient and cosine similarity. Recommendation are computed using WSR (Eqs. 4 and 5). Additionally, we implemented another baseline (RM1Sim [2]) for computing user neighbourhoods based on Relevance-Based Language Models. This method uses Jelinek-Mercer smoothing controlled by the interpolation parameter λ. Figure 1 presents the results of testing the different similarities (LM, cosine, Pearson and RM1Sim) experiments on the MovieLens 100k and R3-Yahoo! collections using user and item-based approaches, respectively.

The experiments showed that DP and JM smoothings yielded the best results. They outperform our baselines (cosine, Pearson and RM1Sim) and also AD. The behaviour of DP varies between collections and the scale of the optimal μ is very different (4500 on MovieLens 100k, 10000 on R3-Yahoo!). Accuracy figures of JM increased with a high amount of smoothing; however, we can observe a significant drop at $\lambda = 1$ which was expected because the estimate degenerates to the background model. On the other hand, AD is not competitive (cosine baseline is better). This also happens in Information Retrieval where DP and JM are the favourite methods [23].

In text retrieval, DP is preferred over JM, especially for short queries [23]. Nonetheless, since we are dealing with long profiles (the users' profiles contain multiple ratings), JM worked better than DP. However, in some cases, DP outperformed JM. The cause may be document length normalisation. Previous studies have found that DP applies a different amount of smoothing depending on the document length while JM smooths all documents in a length-independent manner [11,23]. This property is also very important for finding neighbourhoods because users and items have very diverse profile sizes. Thus, it would be interesting to consider this fact to leverage power users or popular items.

Tables 3, 4 and 5 present the nDCG@10, Gini@10 and MSI@10 values of two state-of-the-art algorithms [4] for top-N recommendation (NNCosNgbr and PureSVD) and our method using cosine and Language Models with DP and JM smoothings. The algorithms were tuned in steps of 50 neighbours/latent factors towards maximum nDCG@10. We tested the user-based approach on the MovieLens datasets and the item-based on the remaining collections.

Our methods significantly surpassed the baselines on all the datasets in terms of nDCG@10. In three out of four datasets, there were not significant differences between DP and JM. We also obtained good diversity and novelty figures. Still, the matrix factorisation technique, PureSVD, provided better results in terms of Gini and MSI in three out of four collections; however, the more sparse the dataset is, the less advantage PureSVD presented over our methods for these metrics. On the other hand, although NNCosNgbr also showed good results in these metrics, the accuracy figures were too low to be an effective alternative.

Finally, another benefit of our neighbourhood method is that inverted indexes enable the efficient computation of Language Models. These data structures are used in Information Retrieval to perform queries on web-scale scenarios. Thus, we can leverage search engines such as Indri or Terrier (which implement Language Models on inverted indexes) to compute neighbourhoods efficiently. In contrast, PureSVD needs to calculate a global matrix factorisation.

5 Related Work

The Language Modelling framework for collaborative filtering recommendation is recently attracting attention in the IR community. Bellogín et al. devised a model that uses any text retrieval method for generating recommendations in a user-based or item-based manner. They tested several techniques including Language Models using Dirichlet Priors and Jelinek-Mercer smoothings [3]. Although the framework is efficient and flexible, the accuracy figures are not outstanding.

Table 3: nDCG@10 best values on the four datasets. Statistically significant improvements (Wilcoxon two-sided test $p < 0.01$) w.r.t. the first, second, third, fourth and fifth method are superscripted with a, b, c, d and e, respectively. Values underlined are not statistically different from the best value. The number of neighbours/latent factors used in each case is indicated in the right side.

Algorithm	**ML 100k**		**ML 1M**		**R3-Yahoo!**		**LibraryThing**	
NNCosNgbr	0.1427	300	0.1042	50	0.0138	50	0.0550	50
PureSVD	0.3595^{a}	50	0.3499^{ac}	50	0.0198^{a}	50	0.2245^{a}	450
Cosine-WSR	0.3899^{ab}	50	0.3430^{a}	50	$\underline{0.0274}^{ab}$	150	0.2476^{ab}	100
LM-DP-WSR	$\mathbf{0.4017}^{abc}$	50	0.3585^{abc}	100	$\underline{0.0271}^{ab}$	100	0.2464^{ab}	50
LM-JM-WSR	$\underline{0.4013}^{abc}$	50	$\mathbf{0.3622}^{abcd}$	100	$\mathbf{0.0276}^{ab}$	100	$\mathbf{0.2537}^{abcd}$	50

Table 4: Gini@10 values on the same settings as Table 3.

Algorithm	**ML 100k**	**ML 1M**	**R3-Yahoo!**	**LibraryThing**
NNCosNgbr	0.0910	0.0896	0.0256	0.0058
PureSVD	0.1364	0.0668	0.1335	0.0367
Cosine-WSR	0.0549	0.0400	0.0902	0.1025
LM-DP-WSR	0.0659	0.0435	0.1557	0.1356
LM-JM-WSR	0.0627	0.0435	0.1034	0.1245

Table 5: MSI@10 values on the same settings as Table 3.

Algorithm	**ML 100k**	**ML 1M**	**R3-Yahoo!**	**LibraryThing**
NNCosNgbr	18.4113	19.5975	43.4348	56.5973
PureSVD	14.2997	14.8416	30.9107	37.9681
Cosine-WSR	11.0579	12.4816	21.1968	41.1462
LM-DP-WSR	11.5219	12.8040	25.9647	46.4197
LM-JM-WSR	11.3921	12.8417	21.7935	43.5986

Also, in the same line, literature about using Relevance-Based Language Models for collaborative filtering is growing. This model was designed for the pseudo-relevance feedback task but it has been adapted for finding user neighbourhoods [2] and computing recommendations directly [13,18,19]. As a neighbourhood technique, our experiments showed that its accuracy is worse than our proposal in addition to being more computationally expensive. Regarding their use as a recommender algorithm, Relevance-Based Language Models have proved to be a very effective recommendation technique [13]. Since this method is based on user neighbourhoods, it would be interesting to combine it with our Language Modelling proposal. We leave this possibility as future work.

6 Conclusions and Future Work

In this paper, we presented a novel approach to find user or item neighbourhoods based on the LM framework. This method, combined with an adaptation of a neighbourhood-based recommender (WSR algorithm), yields highly accurate recommendations which surpass the ones from two state-of-the-art top-N recommenders (PureSVD and NNCosNgr) in term of nDCG with good values of diversity and novelty. This method is also very efficient and scalable. On the one hand, we can take advantage from inverted indexes for neighbours computation—these structures were designed for dealing with web-scale datasets in IR. On the other hand, WSR is simpler than NNCosNgbr and PureSVD without requiring a previous training step for computing biases or the latent factor representation.

Our experiments revealed that Jelinek-Mercer is the best choice for smoothing the estimate of Language Models for neighbourhoods although Dirichlet Priors is also a great choice. This result is analogous to the Information Retrieval task where JM works better than DP for long queries. To overcome the problem that JM does not vary the amount of smoothing applied depending on the document length (in contrast to DP), Losada et al. proposed the use of a length-based document prior [11]. This prior is equivalent to Linear Prior proposed for the Relevance-Based Language Modelling of collaborative filtering recommendations [18]. Testing the applicability of this prior combined with JM smoothing would be an interesting avenue for further work.

We also envision to expand this study to other language modelling approaches. The method proposed in this paper is based on multinomial distributions. A future study that explores the applicability of different probability distributions (such as the multivariate Bernoulli [12]) may be worthwhile.

Acknowledgments. This work was supported by the *Ministerio de Economía y Competitividad* of the Goverment of Spain under grants TIN2012-33867 and TIN2015-64282-R. The first author also wants to acknowledge the support of *Ministerio de Educación, Cultura y Deporte* of the Government of Spain under the grant FPU014/01724.

References

1. Alejandro Bellogín, Pablo Castells, and Iván Cantador. Precision-Oriented Evaluation of Recommender Systems. In *RecSys '11*, page 333. ACM, 2011.
2. Alejandro Bellogín, Javier Parapar, and Pablo Castells. Probabilistic Collaborative Filtering with Negative Cross Entropy. In *RecSys '13*, pages 387–390. ACM, 2013.
3. Alejandro Bellogín, Jun Wang, and Pablo Castells. Bridging Memory-Based Collaborative Filtering and Text Retrieval. *Inf. Retr.*, 16(6):697–724, 2013.
4. Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of Recommender Algorithms on Top-N Recommendation Tasks. In *RecSys '10*, pages 39–46. ACM, 2010.
5. Mukund Deshpande and George Karypis. Item-based top-N Recommendation Algorithms. *ACM Trans. Inf. Syst.*, 22(1):143–177, 2004.

6. Christian Desrosiers and George Karypis. A Comprehensive Survey of Neighborhood-based Recommendation Methods. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, pages 107–144. Springer, 2011.
7. Daniel Fleder and Kartik Hosanagar. Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity. *Manage. Sci.*, 55(5):697–712, 2009.
8. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating Collaborative Filtering Recommender Systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.
9. Yehuda Koren. Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model. In *KDD 08*, pages 426–434. ACM, 2008.
10. Yehuda Koren and Robert Bell. Advances in Collaborative Filtering. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B Kantor, editors, *Recommender Systems Handbook*, pages 145–186. Springer, 2011.
11. David E. Losada and Leif Azzopardi. An Analysis on Document Length Retrieval Trends in Language Modeling Smoothing. *Inf. Retr.*, 11(2):109–138, 2008.
12. David E. Losada and Leif Azzopardi. Assessing Multivariate Bernoulli Models for Information Retrieval. *ACM Trans. Inf. Syst.*, 26(3):17:1–17:46, 2008.
13. Javier Parapar, Alejandro Bellogín, Pablo Castells, and Álvaro Barreiro. Relevance-Based Language Modelling for Recommender Systems. *Inf. Process. Manage.*, 49(4):966–980, 2013.
14. Jay M. Ponte and W. Bruce Croft. A Language Modeling Approach to Information Retrieval. In *SIGIR '98*, pages 275–281. ACM, 1998.
15. Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B Kantor. *Recommender Systems Handbook*. Springer, 2011.
16. G. Salton, A. Wong, and C. S. Yang. A Vector Space Model for Automatic Indexing. *Commun. ACM*, 18(11):613–620, 1975.
17. Daniel Valcarce. Exploring Statistical Language Models for Recommender Systems. In *RecSys '15*, pages 375–378. ACM, 2015.
18. Daniel Valcarce, Javier Parapar, and Álvaro Barreiro. A Study of Priors for Relevance-Based Language Modelling of Recommender Systems. In *RecSys '15*, pages 237–240. ACM, 2015.
19. Daniel Valcarce, Javier Parapar, and Álvaro Barreiro. A Study of Smoothing Methods for Relevance-Based Language Modelling of Recommender Systems. In *ECIR '15*, volume 9022, pages 346–351. Springer, 2015.
20. Jun Wang, Arjen P. de Vries, and Marcel J. T. Reinders. A User-Item Relevance Model for Log-based Collaborative Filtering. In *ECIR '06*, volume 3936, pages 37–48. Springer-Verlag, 2006.
21. Yining Wang, Liwei Wang, Yuanzhi Li, Di He, Wei Chen, and Tie-Yan Liu. A Theoretical Analysis of NDCG Ranking Measures. In *COLT '13*, pages 1–30. JMLR.org, 2013.
22. ChengXiang Zhai. *Statistical Language Models for Information Retrieval.* Synthesis lectures on human language technologies. Morgan & Claypool, 2009.
23. ChengXiang Zhai and John Lafferty. A Study of Smoothing Methods for Language Models Applied to Information Retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, 2004.
24. Tao Zhou, Zoltán Kuscsik, Jian-Guo Liu, Matúš Medo, Joseph Rushton Wakeling, and Yi-Cheng Zhang. Solving the Apparent Diversity-Accuracy Dilemma of Recommender Systems. *PNAS*, 107(10):4511–5, 2010.