Probabilistic Association Rules for Item-Based Recommender Systems

Sylvain Castagnos, Armelle Brun, Anne Boyer

Probabilistic Association Rules for Item-Based Recommender Systems

Sylvain CASTAGNOS [a,1], Armelle BRUN [a,2] and Anne BOYER [a,3]
[a] *LORIA University Nancy 2, Campus Scientifique B.P. 239, 54506 Vandoeuvre-lès-Nancy Cedex, France*

Abstract. Since the beginning of the 1990's, the Internet has constantly grown, proposing more and more services and sources of information. The challenge is no longer to provide users with data, but to improve the human/computer interactions in information systems by suggesting fair items at the right time. Modeling personal preferences enables recommender systems to identify relevant subsets of items. These systems often rely on filtering techniques based on symbolic or numerical approaches in a stochastic context. In this paper, we focus on item-based collaborative filtering (CF) techniques. We show that it may be difficult to guarantee a good accuracy for the high values of prediction when ratings are not enough shared out on the rating scale. Thus, we propose a new approach combining a classic CF algorithm with an item association model to get better predictions. We deal with this issue by exploiting probalistic skewnesses in triplets of items. We validate our model by using the MovieLens dataset and get a significant improvement as regards the High MAE measure.

Keywords. Recommender Systems, Probabilistic reasoning, Ranking

Introduction

The advent of recommender systems is a turning point in the history of the Web. In the old days, people had to cope with their seeking alone. Despite powerful search engines, this task was often very time-consuming and arduous. As the general audience is not trained to a good use of information technologies, expected results were sometimes unreachable. Intelligent Recommender Systems must overcome several difficulties in order to improve the human/computer interactions. One way to make the web browsing easier is to assist users in specifying appropriate keywords. `Google Suggest` relies on this principle. Other systems highlight popular tags to guide users through highly consulted items. This is of course a non-exhaustive list of researches led in this field.

This paper focuses on personalizing services based on collaborative filtering techniques (CF). Personalization is an efficient way to save time of users. The latter can instantaneously access to content fitted to their needs. We can mention systems relying on adaptive interfaces, social navigation, or content adjustment. Another solution consists

[1]PhD Student, Contact Author, E-mail: sylvain.castagnos@loria.fr
[2]E-mail: armelle.brun@loria.fr
[3]E-mail: anne.boyer@loria.fr

in providing each user with items that are likely to interest him/her. Contrary to the content adjustment, this approach does not require adapting resources to users. Each item is only proposed to the pertinent persons by using push-and-pull techniques. This is the purpose of CF techniques. CF algorithms exploit the knowledge of a similar population to predict future interests of a given user (called "active user") as regards his/her known preferences.

In practical terms, this kind of algorithms is broken down into 3 parts. Firstly, the system needs to collect data about all users under the form of explicit and/or implicit ratings. Secondly, this data is used to infer predictions, that is to say estimate the votes that the active user would have assigned on unrated items. Finally, the recommender system suggests to the active user items with the highest estimated values.

As the highest values of prediction are the only ones of interest, we propose a new model that focuses on prediction of high values, to improve accuracy. We show that the error on these values are significant with a usual item-based CF algorithm. Therefore, we propose to re-evaluate them by using reinforcement rules. The latter are automatically inferred by selecting triplets of items in the dataset according to their joint probabilities. The difficulty relies on the ability to estimate the quality of a prediction, to decide to apply these rules or not.

This paper is organized as follows: after a review of state-of-the-art of collaborative filtering approaches, we described an Item-Based Algorithm (CIBA) that we will use as a base for our model. The whole model combining CIBA with Reinforcement Rules is called "Reinforced Item-Based Algorithm" (RIBA). At last, we will discuss the advantages and drawbacks of RIBA according to the experiments we have made from the well-known MovieLens dataset.

1. Related Work

1.1. Collaborative Filtering Approaches

CF techniques amount to identifying the active user with a set of persons having the same tastes, based on his/her preferences and his/her past actions. This kind of algorithms considers that users who liked the same items have the same topics of interest. Thus, it is possible to predict the relevancy of data for the active user by taking advantage of experiences of a similar population. To supply the active user with information that is relevant to his/her concerns, the system first builds his/her profile by collecting his/her preferences. Preferences may be collected through different modeling processes, and are finally transformed under the form of numerical user profiles. These profiles are then aggregated in a user-item rating matrix, where each line corresponds to a user, and each column to an item.

This matrix is used by CF algorithms to compute predictions, that is to say the estimations of votes for items which have not been rated by the active user. There are several ways to classify CF algorithms. In this paper, we refer to [6] who has identified, among existing techniques, two major classes of algorithms: user-based and item-based algorithms.

User-based CF can be divided into roughly three main phases: neighborhood formation, pairwise prediction, and prediction aggregation [2]. The neighborhood formation

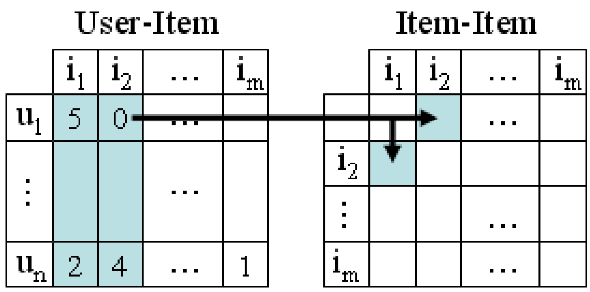

Figure 1. Computing the similarity between two items from a user-item matrix.

consists in building virtual communities of interests by computing correlation coefficient between users' profiles (i.e. between rows of the rating matrix). Then, the active user is associated to the nearest community according to the correlation measure. Members of this community are the most appropriate users to consider since they have common interests with the active user. The closer users are to the active user, the more their preferences are taken into account: this is the pairwise prediction phase. At last, the prediction aggregation consists in computing the weighted mean of the community's ratings in order to provide an estimated vote for each unrated item.

Item-based CF is based on the observation that the consultation of a given item often leads to the consultation of another one [8]. To translate this idea, the system builds a model that computes the relationships between items. Most of time, the model is generated by transforming the user-item matrix in an item-item matrix (cf. figure 1). This conversion requires the computation of similarities between items (i.e. columns of the user-item rating matrix). The active user's predictions are then computed by taking into account his/her known ratings, and the similarities between the rated items and the unrated ones.

User-based and Item-based approaches present both advantages and drawbacks. In [4], we argued that the choice of the method mainly relies on the context. If the recommender systems provides a highly-evolutive catalogue of items (a platform where the set of items can change radically over the time) and long-term users, it will be wise to favour a user-based algorithm. In the case where the set of items is stable, item-based algorithms provide high-quality results and deal with scalability and sparsity problems.

In this paper, we propose a model that can be plugged on an item-based collaborative filtering algorithm in order to refine some predictions.

1.2. Notations

To help the readers, we introduce the following notations:

- $U = \{u_1, u_2, \ldots, u_n\}$ is the set of the *n* users;
- $u_a \in U$ refers to the active user and u_j to any user in the dataset;
- $R = \{i_1, i_2, \ldots, i_m\}$ is the set of the *m* items;
- U_k refers to the set of users who have rated the item i_k;
- R_a is the list of items rated by u_a;
- $M : U \times R \rightarrow \mathbb{N}$ is the user-item rating matrix;
- $v(j, k)$ is the vote of the user u_j on the item i_k;

- v_{min} and v_{max} are respectively the minimum and maximum values on the rating scale;
- v_l the minimum value to reach in order to consider that a user likes an item;
- v_d the maximum value to consider that a user dislikes an item;
- $\bar{i_k}$ is the average of all users' ratings on i_k;
- $S : R \times R \rightarrow \mathbb{R}$ is the item-item similarity matrix;
- $s(k,t)$ the similarity measure between i_k and i_t;
- $p(a,k)$ is the prediction of u_a for item i_k;
- $pr(a,k)$ is the prediction of u_a for item i_k with reinforcement rules.

1.3. Classical Item-Based Algorithm

In this subsection, we present the Classical Item-Based Algorithm (CIBA) used as a base for our model.

When implementing an item-based CF algorithm, the designer has to choose a pairwise similarity metric, and a prediction formula. There are a lot of metrics available to compute similarity between items, such as the Cosine vector [6], the Adjusted Cosine measure [8], the Pearson correlation coefficient [7], the Constrained Pearson coefficient [9] and the Mean Squared [9]. We decide to use the Pearson correlation coefficient, as litterature shows it works better [3].

$$s(k,t) = \frac{\sum_{u_j \in U_k \cap U_t} (v(u_j,i_k) - \bar{i_k})(v(u_j,i_t) - \bar{i_t})}{\sqrt{\sum_{u_j} (v(u_j,i_k) - \bar{i_k})^2} \sqrt{\sum_{u_j} (v(u_j,i_t) - \bar{i_t})^2}} \tag{1}$$

This similarity measure provides values $s(k,t)$ in the interval $[-1;1]$. A negative similarity means that the two items are inversely correlated. A positive similarity means they are correlated. A similarity equals to zero means they are independant.

With regards to the prediction formula, we can use the equation of the item-item algorithm in [6], the equation of the user-based algorithm in [3] and some variants. The weighted sum of the deviation from the mean is usually used in a user-based framework, we decide to adapt it to a item-based context. This method is given in Formula (2).

$$p(a,k) \quad = \quad max\Big(v_{min}, min\Big(\frac{\sum_{i_t \in R_a} s(k,t) \times (v(a,t) - \bar{i_t})}{\sum_{i_t \in R_a} |s(k,t)|} \; + \; \bar{i_k}, v_{max}\Big)\Big) \tag{2}$$

This formula leads to the highest accuracy.

2. Reinforced Item-Based Algorithm

Our model, called "Reinforced Item-Based Algorithm" (RIBA), is a combination of a Classic Item-Based Algorithm (CIBA) and probabilistic reinforcement rules. This section is dedicated to the way to combine these two approaches.

In [10], the authors have used association rules in collaborative filtering. These association rules are quantitative in the sense that they include ratings. For example, a usual association rule looks like : "`Star Trek`" triggers "`Star Wars`". A quantitative association rule looks like "rating 4 for the movie `Star Trek` triggers rating 5 for the movie `Star Wars`". In [10], they have shown an improvement of the accuracy when using these quantitative rules to fulfill the correlation matrix before using classical FC.

In our work, we first compute predictions with Formula (2) before refining the resulting predictions with quantitative rules.

In standard CF algorithms, similarity measures compute the correlation between only two elements. We argue that, in some cases, a single item is not sufficient to explain the interest of a user for an other item. Then, the goal of this work is to study the impact of triplets on the prediction computation process. A triplet is an association rule where the premisse is made up of two terms. The conclusion is the reinforced item.

To illustrate this statement, we can consider three items i_k="`Cinderella`", i_t="`Scary Movie`", and i_w="`Shrek`". A user may have liked i_k which is a fairytale without appreciating i_w. At the same time, a user who enjoys the horror film parody i_t should probably rate lowly i_w. However, a filmgoer who likes both fairytales and parodies will take fun when watching `Shrek`.

Let introduce the following additional notations:

- I_k denotes the fact to like i_k, i.e. when $v_{j,k} \geq v_l$;
- $\overline{I_k}$ is the fact to dislike i_k, i.e. when $v_{j,k} \leq v_d$;
- $\ddot{I_k}$ when i_k has not been rated (by convention, the vote is equal to 0 in this case);
- $\breve{I_k}$ when i_k has been rated (the vote is between v_{min} and v_{max});
- $P(I_k,\ I_t\ , I_w)$ the probability to like the three items i_k, i_t, and i_w;
- $P(I_k,\ I_t\ \mid\ \ddot{I_w})$ the probability to like i_k and i_t for users who have not rated i_w;
- $N(I_k,\ I_t\ ,\ \ddot{I_w})$ the number of users who have liked i_k and i_t, and not rated i_w.

Then a rule $< I_k, I_t >\Rightarrow I_w$ means that I_k alone does not explain I_w, I_t alone does not explain I_w, but $< I_k, I_t >$ together explain I_w.

In the rest of this article, we will use the notation of the equation (3) for this rule.

$$< I_k, I_t >\Rightarrow I_w \tag{3}$$

Let notice that 3 items could lead up to 8 reinforcement rules, as shown in equation (4).

$$< I_k, I_t >\Rightarrow I_w \tag{4}$$

$$< I_k, \overline{I_t} >\Rightarrow I_w \tag{5}$$

$$< \overline{I_k}, I_t >\Rightarrow I_w \tag{6}$$

$$< \overline{I_k}, \overline{I_t} >\Rightarrow I_w \tag{7}$$

$$< I_k, I_t >\Rightarrow \overline{I_w} \tag{8}$$

$$< I_k, \overline{I_t} >\Rightarrow \overline{I_w} \tag{9}$$

$$< \overline{I_k}, I_t >\Rightarrow \overline{I_w} \tag{10}$$

$$< \overline{I_k}, \overline{I_t} >\Rightarrow \overline{I_w} \tag{11}$$

2.2. Determination of the reinforcement rules

A triplet $< i_k, i_t, i_w >$ is candidate to be a reinforcement rule $< I_k, I_t > \ \Rightarrow\ I_w$ if the similarities between each pair of its items are around the mean similarity. In that case, the resulting reinforcement rule could impact accurately I_w.

Thus a triplet is a candidate if the constraints of the equation (12) are satisfied.

$$0 < t_{min} \leq |s(k,t)| \leq t_{max} < 1 \tag{12}$$

$$0 < t_{min} \leq |s(k,w)| \leq t_{max} < 1 \tag{13}$$

$$0 < t_{min} \leq |s(t,w)| \leq t_{max} < 1 \tag{14}$$

where t_{min} and t_{max} respectively refer to the minimum and maximum similarity threshold that will be set experimentally.

For each reinforcement rule candidate, we compute the probability of the corresponding triplet. Thus for each triplet $< i_k, i_t, i_w >$, we compute the joint probabilities $P(I_k,\ I_t\ ,\ I_w)$, $P(I_k,\ I_w\ \mid\ \ddot{I}_t)$, and $P(I_t,\ I_w\ \mid\ \ddot{I}_k)$ (cf. equation 15).

$$P(I_k,\ I_t\ ,\ I_w) = \frac{N(I_k,\ I_t\ ,\ I_w)}{N(\breve{I}_k,\ \breve{I}_t\ ,\ \breve{I}_w)} \tag{15}$$

$$P(I_k,\ I_w\ \mid\ \ddot{I}_t) = \frac{N(I_k,\ \ddot{I}_t\ ,\ I_w)}{N(\breve{I}_k,\ \ddot{I}_t\ ,\ \breve{I}_w)} \tag{16}$$

If this probability is significantly higher than the probability of each pair of its items, than this triplet is selected as a reinforcement rule. The reinforcement rule of the Equation (3) is then generated when the conditions of the equation (17) are fulfilled.

$$P(I_k,\ I_t\ ,\ I_w) \gg P(I_k,\ I_w\ \mid\ \ddot{I}_t) \tag{17}$$

$$P(I_k,\ I_t\ ,\ I_w) \gg P(I_t,\ I_w\ \mid\ \ddot{I}_k) \tag{18}$$

2.3. Prediction Confidence Metric

The generated reinforcement rules allow to refine some predictions. However, some estimated votes are already accurate and do not need any refinement. We consequently have to introduce a prediction confidence metric, in order to know if it can be relevant to apply rules.

In order to define this metric, we start from the observation that the more similar the items of R_a are, the more accurate the prediction $p(a,k)$ is. In Figure 2, we prove this statement by comparing the accuracy difference between the two following cases:

- when taking into account all the items of R_a;
- when only items of R_a whose similarity is higher than a threshold settled are considered.

We obtain better results when using this threshold. However, this reduces significantly the coverage: the percentage of items for which the recommendation system can provide predictions is decreased (cf. figure 2).

Thus, we define the confidence metric $cm(a,k)$ as the average of the absolute values of similarities used to compute the prediction $p(a,k)$ (cf. equation 19).

$$cm(a,k) = \frac{\sum_{i_t \in R_a} |s(k,t)|}{\mid R_a \mid} \tag{19}$$

$\mid R_a \mid$ is the number of items i_t in R_a where $s(k,t) \neq 0$.

The confidence measure is judged satisfying when it is greater or equal to the average of all strictly positive values in the matrix of confidence measures. Otherwise, it means that a refining process is pertinent.

2.4. Rating Refining Process

Each applicable rule associated to $p(a,k)$ is set to a weight $w(r,a,k)$. This weight is equal to 1 when the conclusion of the rule is I_k, and it $w(r,a,k) = -1$ if the conclusion of the rule is $\overline{I_k}$.

For each prediction $p(a,k)$, a rule is applicable if i_k corresponds to the item in the conclusion and if the premises are valid.

We call $AR_{a,k}$ the set of rules that can be applied for the prediction computation of $p(a,k)$.

We define a parameter "min_{rules}", which is a minimum threshold. If $|AR_{a,k}| \leq min_{rules}$ and $cm(a,k)$ is not satisfying, we refine the vote with the equation (20).

$$pr(a,k) = p(a,k) + \frac{coef * \sum_{r \in AR_{a,k}} w(r,a,k)}{\sum_{r \in AR_{a,k}} |w(r,a,k)|} \tag{20}$$

"`coef`" is the coefficient of refinement. The greater this coefficient is, the more important the refinement will be.

Table 1. Distribution of votes in the MovieLens dataset.

Rating Scale Datasets	1	2	3	4	5
U.data	6.11%	11.37%	**27.14%**	**34.17%**	21.20%
U1.base	5.90%	11.47%	**27.45%**	**34.25%**	20.93%
U2.base	6.06%	11.48%	**27.26%**	**34.12%**	21.07%
U3.base	6.15%	11.43%	**26.95%**	**34.05%**	21.40%
U4.base	6.24%	11.21%	**26.97%**	**34.26%**	21.31%

Table 2. Distribution of similarities in the item-item matrix.

Similarity Range	Number of values	Range	Number
[-1.0;-0.8)	104,724	(0.8;1.0]	187,852
[-0.8;-0.6)	21,246	(0.6;0.8]	53,383
[-0.6;-0.4)	29,424	(0.4;0.6]	73,505
[-0.4;-0.2)	42,507	(0.2;0.4]	95,740
[-0.2;-0.0)	67,357	[0.0;0.2]	737,983
Total number of pairwise similarities in matrix S: 1,413,721			

3. Results

3.1. Dataset

In order to evaluate the prediction accuracy of our model, we use the `MovieLens` dataset provided by GroupLens Research[4]. `MovieLens`[5] is a movie recommendation website. People have the opportunity to share their preferences by rating items with integer values from $v_{min} = 1$ to $v_{max} = 5$. The service uses these pieces of information to generate personalized recommendations. The dataset extracted from this platform has been widely used by researchers to evaluate collaborative filtering algorithms and constitutes a good support of validation. It is composed of 100.000 ratings of real users. Each of them has rated at least 20 items. We considered a matrix M of 943 users (U) and 1682 items (R). Thus, there are 93.7% of missing data. The distribution of votes is displayed in Table 1. The dataset has been divided 4 times into a training set (also called "base") and a test set. They respectively include 80% and 20% of all ratings.

U.data corresponds to the whole dataset. *U[1-4].base* are 4 generated training sets.

By using the Pearson correlation coefficient, we get the distribution of similarities in the item-item matrix (S), shown in Table 2.

3.2. Accuracy Metric

Training data is used to compute predictions and retrieve test data. It is then possible to measure the accuracy of a collaborative filtering algorithm by comparing these predictions with the real votes.

[4]http://www.grouplens.org/
[5]http://www.movielens.org/

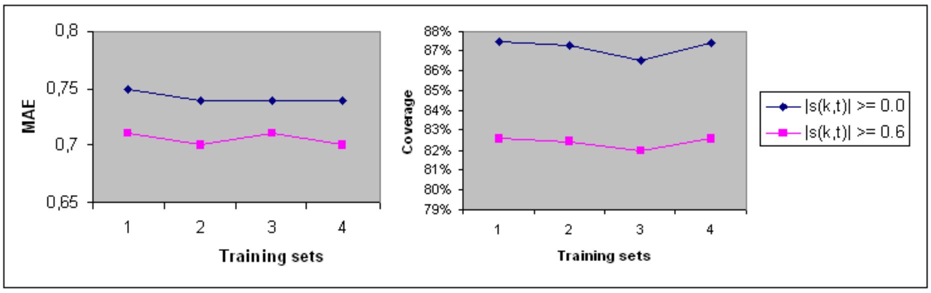

Figure 2. Significant accuracy improvement by only taking into account high pairwise similarities.

In this paper, we compare our algorithm with the classic item-based algorithm of subsection 1.3 by computing the *Mean Absolute Error* (`MAE`) and the High MAE.

`MAE` is a metric which shows the deviation between predictions and real user-specified values. For each rating-prediction pair $< p_i, q_i >$, we compute the absolute error between them. Then, we get the MAE measure by summing these absolute errors and dividing this sum by the number N of corresponding rating-prediction, as shown in formula 21.

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N} \tag{21}$$

A low MAE means obviously an accurate recommendation engine.

The High MAE measure is defined as the MAE obtained only on ratings with values 4 and 5 [1]. This metric evaluates the accuracy of the most important predictions, since a recommender system only suggests items for which the estimated values are the highest (top-N). An estimation error on low predictions will not penalize the active user. We preferred the High MAE to the Precision measure, since it allows us to evaluate our model qualitatively rather than quantitatively. Precision is defined as the ratio of relevant items selected to the number of suggested items [5].

In the following subsection, HMAE1 refers to the MAE on items high-rated by users ($ratings \geq 4$) in the test sets *U[1-4].test*. HMAE2 estimates the error on high-predicted items ($predictions \geq 4$). A better HMAE1 means that the refining process is relevant. A worse HMAE2 would show that too many predictions have been refined.

3.3. Experiments and Discussion

As explained in subsection 2.3, we first evaluated the accuracy of CIBA in the normal case, and in the case where we only take into account items of R_a whose similarity is greater than a threshold settled to 0.6 (cf. Figure 2). The threshold improves the MAE measure to the expense of the coverage which widely reduces. This experiment is independent from our model and only aims at justifying our choice for the confidence metric. The coverage is the same for RIBA and CIBA, since the difference is only a refinement of some predictions.

In order to validate our model RIBA, we generated reinforcement rules from the 4 training sets *U[1-4].base*. In our experiments, items are disliked when ratings are lower or equal to $v_d = 2$. Users like items when their ratings are greater or equal to $v_l = 4$.

Table 3. Accuracy measures of Classic Item-Based Algorithm (CIBA) and Reinforced Item-Based Algorithm (RIBA).

Datasets	U1.base		U2.base		U3.base		U4.base	
Metrics	CIBA	RIBA	CIBA	RIBA	CIBA	RIBA	CIBA	RIBA
MAE	0.75	0.75	0.74	0.74	0.74	0.74	0.74	0.74
HMAE1	**0.64**	**0.60**	**0.64**	**0.62**	**0.63**	**0.62**	**0.64**	**0.62**
HMAE2	0.64	0.65	0.62	0.62	0.65	0.65	0.64	0.64
Evaluation of predictions for which rules are applicable								
Number of predictions	2244		2481		2505		2316	
MAE	0.75	0.75	0.80	0.80	0.78	0.78	0.79	0.79
HMAE1	**0.64**	**0.60**	**0.84**	**0.77**	**0.84**	**0.77**	**0.84**	**0.78**
HMAE2	0.64	0.65	0.72	0.71	0.65	0.63	0.66	0.68

We considered that the difference between probabilities is significant when the minimum range is 0.3 (cf. equation 17). We configured the algorithm to compute triplets whose similarities are between $t_{min} = 0.4$ and $t_{max} = 0.6$, since the Table 2 shows there is a reasonable number of pairwise similarities within this interval. The value of `coef` has been chosen in order to provide small refinements. The goal is not to recompute the predictions, but to supply a minor correction on badly estimated votes. Thus, `coef` was equal to 0.1 in our tests. At last, we set the parameter min_{rules} to 20. We generated about 432,000 reinforcement rules. The results in term of accuracy are shown in Table 3.

The MAE and HMAE2 values remain stable between CIBA and RIBA. The accuracy is globally the same. The interesting point is that our model increases the quality of high predictions. Among the 20% of votes in test sets that we tried to retrieve with RIBA, the ratings greater or equal to 4 are those that have to be suggested by recommender systems. Before applying our model, the error for these predictions was quite high (up to 0.85). This can be due to the great number of missing data and the bad distribution of ratings in the matrix M (cf. Table 1). The HMAE1 measure highlights a noteworthy improvement, particularly for predictions where we applied the reinforcement rules (up to a MAE decrease of 0.07).

We also tried to set the number min_{rules} to 0. In this case, the HMAE1 was quite the same, but the HMAE2 values were increasing. This means that we were refining too many predictions and confirms the interest of using this minimum threshold. The increase of HMAE2 when $min_{rules} = 0$ can be due to the fact that we generated much more positive rules (i.e. with a conclusion I_k) than negative rules (with a conclusion $\overline{I_k}$). It can be explained by the low number of disliked items ($ratings \leq v_d$).

Conclusion and Perspectives

In order to increase the quality of suggestions in recommender systems, we proposed a new approach combining an item-based collaborative filtering model with reinforcement rules. These rules are generated automatically by analyzing joint probabilities in triplets, and allow us to refine predictions of items where pair-wise similarities are not sufficient. The experiments show that this approach significatively improves the accuracy of high predictions.

Our model can easily be plugged on other item-based algorithms. We plan to test our work on different collaborative filtering techniques and with different similarity metrics. We also plan to find ways to reduce the computational weight of the algorithm in order to face the scalability problem.

References

[1] Linas Baltrunas and Francesco Ricci, Dynamic item weighting and selection for collaborative filtering, in Workshop PriCKL07, in conjunction with the 18th European Conference on Machine Learning (ECML) and the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), Warsaw, Poland, September 2007.

[2] Shlomo Berkovsky, Tsvi Kuflik, and Francesco Ricci, Enhancing privacy while preserving the accuracy of collaborative filtering, in Workshop on Recommender Systems, in conjunction with the European Conference on Artificial Intelligence (ECAI 2006), Riva del Garda, Italy, August 2006.

[3] Sylvain Castagnos and Anne Boyer, A client/server user-based collaborative filtering algorithm: Model and implementation, in 4th Prestigious Applications of Intelligent Systems special section (PAIS 2006), in conjunction with the European Conference on Artificial Intelligence (ECAI 2006), Riva del Garda, Italy, August 2006.

[4] Sylvain Castagnos and Anne Boyer, Modeling preferences in a distributed recommender system, in 11th International Conference on User Modeling (UM 2007), Corfu, Greece, June 2007.

[5] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst., **22(1)**, pp. 5-53, 2004.

[6] Bradley N. Miller, Joseph A. Konstan, and John Riedl, Pocketlens: Toward a personal recommender system, in ACM Transactions on Information Systems, volume **22**, pp. 437Ű476, July 2004.

[7] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl, Grouplens: An open architecture for collaborative filtering of netnews, in Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work, pp. 175-186, Chapel Hill, North Carolina, 1994. ACM.

[8] Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John Reidl, Item-based collaborative filtering recommendation algorithms, in World Wide Web, pp. 285Ű295, 2001.

[9] Upendra Shardanand and Patti Maes, Social information filtering: Algorithms for automating Şword of mouthŤ, in Proceedings of ACM CHIŠ95 Conference on Human Factors in Computing Systems, volume **1**, pp. 210-217, 1995.

[10] Xiaohua Sun, Fansheng Kong, and Hong Chen, Using Quantitative Association Rules in Collaborative Filtering, volume **3739**/2005 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2005.

[11] John S. Breese, David Heckerman and Carl Kadie, Empirical Analysis of Predictive Algorithms for Collaborative Filtering, in Proceedings of the fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98), pp. 43-52, San Francisco, CA, July 1998.