A positively directed mutual information measure for collaborative filtering

Armelle Brun, Sylvain Castagnos, Anne Boyer

A positively directed mutual information measure for collaborative filtering

Armelle Brun[1] and Sylvain Castagnos[2] and Anne Boyer[1]

[1] LORIA - BP 239 - 54506 Vandœuvre lès Nancy, France
[2] EPFL - IC IIF Station 14 - 1015 Lausanne, Switzerland
{armelle.brun, anne.boyer}@loria.fr, sylvain.castagnos@epfl.ch

Abstract. As resource spaces become ever larger, the need for tools to help users find pertinent and reliable resources quickly and easily is more and more acute. Recommender systems are an efficient way to tackle the problem of information overload, as they enable to inference from users' past behavior to suggest resources the users have not seen yet. Collaborative filtering, that uses ratings of user on items, has become a very popular technique for recommender systems. One crucial step of item-based collaborative filtering is the computation of pair-wise similarity between items. In order to improve the accuracy of collaborative filtering, we propose a new approach for the computation of similarities, by using mutual information. In this work, only positive and negative evidences are considered for training, whereas classical approaches use ratings. During the prediction step, we do not use the classical K value (KNN neighbors) as the size of the neighborhood of each item, but a size value that depends on the considered item. The proposed method is evaluated on the well-known MovieLens dataset. Our experiments show an improvement in the accuracy of collaborative filtering.

Keywords Mutual Information, Neighbor selection, Item-based approach

1 Introduction

Internet makes available to online users an over-abundance of information, e-services and products. Users have to decide which information to consult, which service to use or which product to buy. Recommender systems [1] aim at finding, given a user, the adequate items from the huge number of available items. An item is either an information, or an on-line service or a product, and is the elementary unit we consider.

Collaborative Filtering (CF) is a popular recommendation technique, which uses similarities between users (user-based approaches) or items (item-based approaches) to predict the most pertinent items for a given user. The input of the system is a set of ratings of users on a set of items, and the goal is to predict how well a user will like an item that he/she has not rated given his/her past ratings and the ratings of other users [14, 17, 3, 15]. Ratings can be explicitely given by the users or implicitely derived from the analysis of the users' actions, in the available logs or by mining usage [6].

This paper focuses on the item-based approach, that associates to each item its set of nearest neighbors according to a similarity measure. The prediction of a user's rating on an item is based on all the items he/she has already rated, or the nearest neighbors (mentors) of this item.

Similarity between two items represents how these two items are similarly co-voted. We propose a new way to compute item-to-item similarities, relying on mutual information [16]. Mutual information (MI) is an information-theoric measure of the dependence of a variable on another one. Furthermore, we propose to compute item similarities when only positive or negative evidences are available. The advantage of this method is the reduction of the rating scale to only two states : "I like" or "I dislike". It makes easier the task of the users who do not have anymore to rate items but only to express a global opinion. Ratings that correspond to neutral opinions are considered a non-informative and are discarded.

In order to decide which items to use during the prediction phase, and therefore reduce time complexity while guarantying a high level of accuracy, we propose to use an original way to select neighborhood, that is an adjustable neighborhood depending on the item and its reliability, instead of the K nearest neighbors approach

as introduced in [14].

The rest of this paper is organized as follows : section 2 describes the item-based approach. Similarity measure based on mutual information is first described in section 3, then a new similarity measure based on mutual information is proposed. Section 4 details an original way to determine the optimal size of the neighborhood of a given item. The experimental dataset and protocol are described in section 5. Performance of the propositions presented in section 3 and 4 are assessed and analyzed in section 6. Section 7 draws the conclusion and section 8 presents future work.

2 ITEM-BASED COLLABORATIVE FILTERING

Item-based collaborative filtering algorithms explore the relationships between items, and then compute recommendations for the active user by finding items similar to the items he/she liked. The item-based approach is well suited to applications where the set of items is relatively static [5].

Let U be the set of N users, $a \in U$ the active user and I the set of M items with $t \in I$ the target item. r_{ui} corresponds to the rating of user $u \in U$ on item $i \in I$. Let $sim(i,j)$ be the similarity value between the two items i and j. p_{at} is the prediction of rating of the active user a on the target item t.

The procedure of predicting ratings using the similarity measure can be described as follows:

1. compute the similarity $sim(i,j)$ $\forall (i,j) \in I \times I$.
2. select the neighborhood of each item.
3. compute the prediction of rating p_{at} for the targeted item t and the active user a.

2.1 Computing Similarity Between Items

One crucial step in an item-based algorithm is to compute the similarity between items. The basic idea in similarity computation between two different items i and j, is to determine the set of users who rated both items and then to apply a similarity computation technique. An overview of the most usual similarity measures can be found in [18, 4], we only present here the most popular: the cosine and the Pearson coefficient measures.

Cosine Measure Item-based approach considers items as vectors in user space whose components are the users ratings. The cosine similarity between items i and j is computed as the cosine of the angle between $\overrightarrow{i}$ and $\overrightarrow{j}$ as follows:

$$sim(i,j) = cos(\overrightarrow{i}, \overrightarrow{j}) = \frac{\overrightarrow{i} \cdot \overrightarrow{j}}{||\overrightarrow{i}|| \ \ ||\overrightarrow{j}||} \tag{1}$$

The drawback of cosine measure is that the difference in rating scale between users is not taken into account [15].

Pearson Correlation Coefficient The Pearson correlation coefficient evaluates the deviation from the mean rating of items.

$$sim(i,j) = PCC(i,j) = \frac{\sum_k (r_{ki} - \overline{r_i})(r_{kj} - \overline{r_j})}{\sqrt{\sum_k (r_{ki} - \overline{r_i})^2 \sum_k (r_{kj} - \overline{r_j})^2}} \tag{2}$$

Where $\overline{r_i}$ is the mean rating of item i and k sums over users that have both rated item i and j.

The advantage of Pearson coefficient is that both negative and positive correlation values are obtained. It has been shown in [3, 4] that the Pearson Correlation Coefficient (PCC) leads to a higher accuracy than cosine measure. An additional measure can also be used, the adjusted cosine [15] that corresponds to the cosine of items deviation from the user mean rating. Adjusted cosine is similar to Pearson and offsets the cosine drawback by subtracting the corresponding user average for each covoted pair.

2.2 Selecting the Neighborhood of Items

To compute p_{at}, similarities between the target item t and all other items are used. However, to speed up the prediction computation, only a subset of these items is used. Usually, most similar items to t are used (called nearest neighbors). A value of K is fixed *a priori* and K items neighbors are considered [8, 15, 4] to compute predictions.

2.3 Predicting

In item-based collaborative filtering algorithms, the prediction p_{at} is computed on the items (and the corresponding ratings) that the user a has already voted. These items are weighted according to their correlations with item t. An overview of the prediction methods is presented in [4].

The weighted sum, that is often used in item-based collaborative filtering, evaluates p_{at} by computing the weighted sum of the ratings given by user a on the neighbor items of t.

$$p_{at} = \frac{\sum_i sim(i,t) * r_{ai}}{\sum_i |sim(i,t)|} \tag{3}$$

where i sums over all items in the nearest neighbors of item t, that have been rated by the active user a.

The following prediction method is classically used in the user-based algorithms [21], less often in item-based approaches [19]. The computation of p_{at} considers the sum of the target item mean rating and the weighted sum of deviations from the mean rating of items that have been rated by user a.

$$p_{at} = \overline{r_t} + \frac{\sum_i sim(i,t) * (r_{ai} - \overline{r_i})}{\sum_i |sim(i,t)|} \tag{4}$$

where $\overline{r_i}$ represents the mean rating of item i. This method will be used in this article to compute predictions, since it gets better results.

2.4 Assessing Performance

Performance of a recommender system can be evaluated in several ways, we are interested here in accuracy and coverage rate [13].

The accuracy of a recommender system can be evaluated by using various metrics [10]. One of the most widely used is the Mean Absolute Error (MAE). MAE measures the deviation of the recommendations from their true user-specified values. The lower the MAE is, the more accurate are the predictions of the recommender system. Given a test corpus T, composed of triplets (u, i, r) where u is a user, i an item and r the rating u gives on i, MAE is computed as follows:

$$MAE = \frac{1}{|T|} \sum_{(u,i,r) \in T} |p_{ui} - r| \tag{5}$$

Accuracy of a recommender can also be evaluated by using the HMAE (High MAE) measure [2]. HMAE represents the MAE computed only on the highest rating values. Two HMAE values can be considered. The first one, we call the TestHMAE, computes the accuracy on the high ratings of the test corpus. The TestHMAE represents the accuracy on the items the users like, it is related to the recall measure. The second measure, we call the RecHMAE, computes the accuracy on the items the recommender predicts with high ratings values. The RecHMAE is related to the precision measure. Both RecHMAE and TestHMAE are important to compute the accuracy of a recommender. Indeed, the items the recommender proposes to the user (items he rated highly) must be items the user actually likes, and conversely, the items a user likes have to be proposed by the recommender.

Coverage rate is used in addition to MAE and HMAE measures. Coverage computes the percentage of items for which a recommender can provide a prediction. In the case of items that have not been sufficiently co-voted, no reliable similarity values are computed, thus no prediction can be performed. In this article, coverage is studied, as some prediction methods may not reach a 100% coverage rate.

3 MUTUAL INFORMATION-BASED SIMILARITY MEASURE

To improve the accuracy of a recommender system, we propose in this article to use mutual information as a basis to compute the similarity measure between two items. Mutual information has already been introduced in user-based collaborative filtering algorithms. To our knowledge, it has not been much used to compute dependency between items [21].

In [2, 20, 22], the similarity between two users is computed on co-voted items with the Pearson Correlation Coefficient. A specific weight is then assigned to each co-voted item, this coefficient is defined with the mutual information measure. Mutual information can also be used during the prediction phase [22]. It allows to rank the relevance of the active user to the target item, thus performing feature selection.

In information theory, mutual information represents a measure of statistical dependence between two variables X and Y with associated probability distributions $p(x)$ and $p(y)$. In our context, x and y are rating values.

Following Shannon [16], the mutual information (MI) is defined as:

$$MI(X;Y) = \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{6}$$

where x (respectively y) sums over values (ratings) of X (respectively Y).

In this work, we use the normalised mutual information (NMI) [7], thus weights are in the range $[0;1]$. NMI is defined as follows:

$$NMI(X;Y) = \frac{2 \cdot MI(X;Y)}{H(X) + H(Y)} \tag{7}$$

where $H(X)$ is the entropy of X and is computed as:

$$H(X) = -\sum_x p(x) \log p(x) \tag{8}$$

From now on, we will refer to normalized mutual information as mutual information.

A large mutual information value between the votes of two items reflects a high level of dependence. However, this dependence can be of two kinds. On the one hand, it can be positive: the two items are mainly rated with similar values. On the other hand, the dependence can be negative: the two items are opposed items, they are rated with opposite values - when one item is liked by a given user, the second one is usually disliked. Thus mutual information does not give any information about the nature of this dependence: a high value may represent a positive or a negative dependence.

In the framework of recommender systems, mutual information is used to compute similarity between items, then to predict the rating of the target item. The nature of the dependence is thus a crucial information, that can be used to improve the prediction capability of the recommender system.

3.1 Using binary ratings

We propose, as mentioned in [9], to group the range of ratings into two sets: positive opinions and negative opinions. For example, on the most commonly used MovieLens dataset, ratings range from 1 to 5, where 1 represents the lowest possible rating and 5 represents the highest.

We propose to merge ratings 1 and 2, representing the evidence “do not like” and merge ratings 4 and 5, to represent the evidence “like”. We consider that rating 3, that represents a neutral opinion, is a non-informative

rating. It is the reason why we propose to discard it from the computation of the mutual information on the training dataset, as it can be considered as noise, thus decreasing performance.

In the case of binary opinions, Equation (7) becomes:

$$\begin{aligned} NMI(i;j) = \quad & \tfrac{2}{H(i)+H(j)} \cdot \\ & (\ p(i_l, j_l) \log \tfrac{p(i_l,j_l)}{p(i_l)p(j_l)} \quad (A) \\ & + p(i_l, j_d) \log \tfrac{p(i_l,j_d)}{p(i_l)p(j_d)} \quad (B) \\ & + p(i_d, j_l) \log \tfrac{p(i_d,j_l)}{p(i_d)p(j_l)} \quad (C) \\ & + p(i_d, j_d) \log \tfrac{p(i_d,j_d)}{p(i_d)p(j_d)} \ ) \ (D) \end{aligned} \tag{9}$$

where i_l represents a positive assessment (liked the item) and i_d a negative assessment (disliked the item). $p(i_l, j_l)$ represents the probability, for a given user, of liking both items i and j. This probability is computed as the normalized counts. $p(i_l)$ represents the probability of liking item i. This measure will be referred as the Global Mutual Information (GMI).

3.2 Positive Mutual Information Measure

As presented previously, we are interested in using a similarity measure that integrates the nature of the dependence between two items. In this way, we split the mutual information into two sub-measures. The first one reflects the "positive" dependence, *i.e.* how much items have similar ratings. This measure contains the terms (A) and (D). Indeed, a high value of term (A) represents that most users who liked i also liked j and a high value of term (D) represents that users who did do not like item i did not like item j neither. The second sub-measure represents "negative" dependence, *i.e.* how much items have opposite ratings. In this article, we are interested in the first measure, that is defined as:

$$\begin{aligned} PMI(i,j) = \quad & \tfrac{2}{H(i)+H(j)} \cdot \\ & (\ p(i_l, j_l) \log \tfrac{p(i_l,j_l)}{p(i_l)p(j_l)} \\ & + p(i_d, j_d) \log \tfrac{p(i_d,j_d)}{p(i_d)p(j_d)}) \end{aligned} \tag{10}$$

This measure will be refered to as the Positive Mutual Information (PMI) as it represents to what extent two items are similarly rated. This measure can have both positive and negative values.

For comparison purposes, we also study the measure that contains (B) and (C) terms, that will be refered to as Negative Mutual Information measure (NMI).

We can notice that mutual information is highly influenced by the quantity of observed data. The smaller the size of observable data is, the less reliable the MI will be. That is why similarity between two items is computed only if a significant number of users have rated both items. In this paper, a threshold of 5 has been defined (see section 5.2).

Once similarities between items are evaluated, the ratings can be predicted. This prediction can be either computed by using all items or using a subset of items: its nearest neighbors.

4 SELECTION OF THE NEIGHBORHOOD FOR A GIVEN ITEM

In this section we address the problem of deciding which items to use during prediction and try to remove the unreliable ones to improve the quality and scalability of collaborative filtering.

The key idea is to highlight the fact that, weak dependent items (with low similarity value) are low informative for prediction. The use of such items may decrease the quality of the prediction. Furthermore, generating predictions over relevant items instead of operating over the entire dataset may decrease complexity time.

The most popular way to decide which items to use for prediction is to fix an *a priori* value K, that represents the number of items to use. The items are the K nearest neighbors of item t, *i.e.* items with the highest values [11, 2, 4]. The value of K is set experimentally and accuracy of collaborative filtering is influenced by the size of neighborhood.

The drawback of the K nearest neighbors method is that exactly K neighbors are used to compute ratings. Can the optimal predictions be computed by always using K neighbors ?

We propose here to improve this K nearest neighbor approach: the size of the neighborhood is now dependent on the considered item. Indeed some items may have few number of highly similar items, this small set may be sufficient to compute prediction. Conversely, some items may not have highly similar neighbors, and a higher number of neighbors may be used to compute predictions. Given an item, we propose to define its neighborhood as the set of similar items so that the value of their similarities sums up to a fixed value noted V. This approach has the advantage to determine the adequate size of the neighborhood for each item.

5 EXPERIMENTAL EVALUATION

5.1 The MovieLens Dataset

In our experiments we use the MovieLens dataset[3], containing 100K explicit ratings, for 1682 movies (items) and 943 users. Each rating varies in the range 1 to 5. Data sparsity on this corpus is 96%. This corpus is divided into five parts; to perform five-fold cross-validations, training is made up of 4 parts and testing is the last part.

As we consider that ratings with a value of 3 correspond to a neutral opinion, we propose, in a first step, to discard them from training corpus ; they represent 27% of the dataset. Obviously, all ratings in the test set are considered. The resulting data set is made up of 24% of negative opinions (merging ratings 1 and 2) and 76% of positive opinions (merging ratings 4 and 5).

5.2 Experimental Protocol

As previously mentioned, mutual information is unreliable when only few data are available. Therefore, we introduce a threshold - which represents the minimal amount of required data - in order to compute a reliable mutual information value. Otherwise, using a MI value when the threshold has not been reached may introduce a bias in the model. In that case, a 0 value is set in the similarity matrix, meaning that items are independent. A similar threshold is used for other methods.

RecHMAE and TestHMAE are evaluated on high ratings, we consider here that high ratings correspond to the "like" evidence, thus ratings 4 and 5.

6 RESULTS AND DISCUSSION

6.1 Comparison of Similarity Measures

In a first step, we implement various similarity measures: the cosine measure (Equation (1)) and the Pearson coefficient (Equation (2)) which are broadly used in the state of the art. In order to compare measures proposed in this article the GMI (Equation (8)) and PMI similarity measures (Equation (9)) are also implemented. For comparison purposes, NMI is also presented.

Recall that PMI is a sum of two terms:

- the first one, called the "like term" (LT), $\frac{-2}{H(i)+H(j)} p(i_l, j_l) \log \frac{p(i_l,j_l)}{p(i_l)p(j_l)}$,
- the second one, called the "dislike term" (DT), $\frac{-2}{H(i)+H(j)} p(i_d, j_d) \log \frac{p(i_d,j_d)}{p(i_d)p(j_d)}$.

[3] http://www.movielens.org

In many research areas, the "like term" is often used alone, as a simplified version of the mutual information. We study here the influence of each subterm on the MAE, in order to determine the pertinence exploiting both subterms.

Note that, before evaluating the MAE or HMAE scores, all predicted ratings are rounded as the ratings given by the users are integer values.

We can remark that for all similarity values, the similarity matrix contains about 77% of null values. This rate is explained by the fact that the similarity value between two items is set to 0 if they have not been sufficiently co-voted. A 0 value can also mean that two items are independent. When a no minimum co-voted value is set, the similarity matrix contains only about 45% null values.

Table 1 presents MAE and the coverage rate (defined as the percentage of predicted items), on the five test corpora, using all neighbors, for all the similarity measures listed above.

Table 1. Comparing similarity measures

measure	**MAE**	**coverage**
Cosine	0.0.699	98.4 %
Pearson	0.693	98.5 %
GMI	0.696	98.5 %
PMI	**0.686**	**98.4 %**
NMI	0.724	99.3 %
LT	0.692	98.5 %
DT	0.696	94.8 %

We can first notice that all the methods have quite similar coverage rates. A 100% coverage rate is never achieved, this can be explained by the fact that a subset of pairs of items in the dataset does not reach the co-voted threshold (see Section 2.4). We can also notice that, although ratings with a value 3 have been discarded, the coverage rate is high.

The Pearson Coefficient results in a better MAE than the cosine measure (2%), as previously shown in [4].

The Global Mutual Information measure performs slightly worse than Pearson, but performs better than Cosine. When comparing measures using mutual information, we can notice that the Positive Mutual Information, that uses a subset of terms of the GMI, slightly improves its MAE off 1.4%. Moreover, NMI, that represents how opposed items are, reaches the worst MAE value among all similarity values. We can deduce that the two sub-terms (B) and (C) from the GMI, that are not part of the PMI, are not good terms to compute the similarity. Indeed, discarding these two terms does not increase the MAE, it does even decrease it.

In order to study the pertinence of using both terms in PMI, we study the MAE obtained for the LT or DT. Both terms lead to higher MAE (respectively 0.8% and 1.4% higher than PMI), which shows that the combination of both subterms reinforces the accuracy of the prediction.

We can notice that NT leads to a higher MAE value and lower coverage rate. This can be explained by the small percentage of disliked items within the dataset (24%), which lowers the quality of the model.

We are also interested in the study of TestHMAE and RecHMAE values for all similarity measures, presented in Table 2. The coverage rate for TestHMAE is constant as it represented the rate of items highly rated in the test corpus. Concerning RecMAE, all methods have comparable coverage rates: they all rate about 25.4 % of the item with a high value. When comparing Pearson and GMI, we can remark that they have opposite behaviors: Pearson has a better TestHMAE whereas GMI has a better RecHMAE. PMI, that leads to the best MAE, slightly improves TestHMAE and RecHMAE over all similarity measures.

6.2 K nearest neighbors

In this section, we study the variation of the MAE according to the size of the neighborhood used to compute the prediction. The neighbors used are the most correlated neighbors. A value of K is fixed *a priori* and a rating

Table 2. HMAE values for all similarity measures

measure	**Test HMAE**	**cov.**	**Rec HMAE**	**cov.**
Cosine	0.613	54.9 % 0.608	24.7 %	
Pearson	0.603	54.9%	0.606	25.6 %
GMI	0.608	54.9%	0.603	25.2 %
PMI	**0.596**	54.9 %	**0.598**	25.4 %
LT	0.608	54.9%	0.603	25.2 %
DT	0.601	54.9%	0.603	25.5 %

is computed according to K neighbors. We can notice that whatever the value of K is, the coverage does not vary.

Fig. 1. Evolution of the MAE value according to the number of neighbors used to compute prediction

We can notice that, as presented in [8, 15, 12], MAE does not highly improve when using only a subset of the nearest neighbors. However, the improvement of PMI and GMI is five times higher than the improvement of Pearson. We can also notice that the optimal value of neighbors for Pearson (about 70) is higher than the optimal value for PMI and GMI (about 30).

6.3 A dynamic size of the neighborhood

When using the K nearest neighbors presented in the previous section, exactly K items are used to compute prediction. We can suppose that some ratings may be computed by using less than K neighbors (for example when a few number of highly correlated neighbors exists). In some cases, more than K neighbors may be useful to accurately compute one rating (when only low correlated items are available). In this section we present an original way to choose neighborhood: it is now made up of the highest similar items such that the sum of the values of their similarities is over a value V. This approach has the advantage of selecting a neighborhood whose size depends on the target item. For example, when highly correlated items are available, a few numbers of neighbors is used, and low-correlared neighbors are not considered, avoiding a decrease of performance. As for K nearest neighbors, this way to select neighborhood leads to a constant coverage.

Fig. 2. Evolution of the MAE value according to the value of the threshold neighborhood for Pearson

Figure 2 shows the evolution of the MAE for Pearson coefficient, according to the value V. We can notice that the lower MAE is obtained with a value $V = 25$. This MAE is similar to the one reached when using a fixed number of neighbors K (section 6.2). A value $V = 25$ corresponds to 56 neighbors on average. The optimal number of neighbors with the KNN method was 70, the number of neighbors to use is thus reduced by 20%.

Figure 3 represents the evolution of MAE for PMI and GMI according to the V value. We can remark that this new method slightly improves PMI and GMI. The value of the MAE obtained for GMI is 0.689 whereas the one obtained with KNN was over 0.691. For PMI, MAE obtained is also improved compared to KNN. Concerning the value V, the optimal value for PMI is 4.5, that corresponds to on average 24 neighbors (23 % less than when using KNN). The maximum number of neighbors used is 97.

Fig. 3. Evolution of the MAE value according to the value of the threshold neighborhood for GMI and PMI

To conclude, the use of the threshold V improves the K nearest neighbors while reducing the mean size of the neighborhood.

6.4 Influence of the Similarity Threshold

In this section, we are interested, as in the two previous sections, in a better way to select the neighbors to compute prediction. We study the influence of an item, according to its similarity with target item t. We introduce here a similarity threshold T that represents the lower bound of pertinence of a similarity value: items correlated with a value below T are not taken into account.

We consider that low dependent items are not useful for prediction, and may lead to even lower performance. The question is which value of T should be chosen for threshold ? In Table 3, MAE and RecHMAE are presented according to the threshold T, for several similarity measures.

Table 3. Influence of the similarity threshold T on MAE and coverage for Pearson, Cosine, GMI and PMI

Threshold	**Pearson**	**Pearson**	**Cosine**	**Cosine**	**GMI**	**GMI**	**PMI**	**PMI**
	MAE	**coverage**	**MAE**	**coverage**	**MAE**	**coverage**	**MAE**	**coverage**
0	0.693	98.5 %	0.699	98.4	0.696	98.5 %	0.686	98.4 %
0.1	0.693	98.3 %	0.696	98.5 %	0.705	93.4 %	0.685	97.6 %
0.2	0.693	98.2 %	0.686	97.1 %	0.737	46.1 %	0.700	90.2 %
0.3	**0.700**	**97.9** %	0.684	88.9 %	**0.610**	**8.4 %**	0.735	56.1 %
0.4	0.717	95.5 %	0.689	66.7 %	**0.402**	**1.9 %**	0.675	10.2 %
0.5	0.749	88.9 %	0.681	35.6 %	0.365	0.9 %	**0.423**	**1.4 %**
0.6	0.788	76.2 %	0.663	9.8 %	0.237	0.3 %	0.237	0.3 %
0.7	0.832	59.0 %	0.504	1.3 %	0.151	0.1 %	0.151	0.1 %
0.8	0.880	37.6 %	-	-	-	-	-	-
0.9	0.914	14.0 %	-	-	-	-	-	-
1.0	**0.872**	**0.6 %**	-	-	-	-	-	-

On the one hand, we can notice that the coverage rate decreases quickly with the threshold value: this is due to the fact that only few similarity values are above the similarity threshold when it grows up. For example, when the threshold is fixed at a value of 0.3 the coverage rate dramatically decreases by 85%. We can also notice that decrease in the coverage for Pearson and Cosine is slower than for GMI and PMI. For example, for the same threshold of 0.3, Pearson computes a prediction for 97.9% of the test set.

On the other hand, we can see that the MAE has a similar variation: accuracy is improved with the threshold value. For Cosine, MAE slowly decrease according to T. We can notice that with low values of T, the MAE of GMI and PMI increases, while the coverage decreases slightly. For example, for GMI with a threshold $T = 0.2$, MAE is increased by 6% compared to the use of all neighbors. This can be explained by the low number of neighbor items used to compute prediction: when using a threshold $T = 0.2$, the number of neighbors used is on average 30 times lower than when using a threshold $T = 0.0$. The difficulty is to find the right balance between the MAE and the coverage rate by tuning this threshold experimentally. However, Pearson MAE does not decrease when the threshold increases, it does even increase. When the threshold is fixed to 1.0, coverage is equal to 0.6% and MAE is 0.872: when only the items correlated with a value of 1.0 are used (the highest correlation value that can be reached), MAE increases by 20% compared to using all positive correlations. A similar behavior is observed when 5 ratings are used (without binary ratings). Cosine, GMI and PMI have expected behaviors: MAE decreases when the value of the threshold is high. For example when the threshold is

fixed to 0.5, the MAE of PMI is equal to 0.423. The most correlated the items used for prediction are, the most accurate the recommender is. Of course, coverage highly decreases according to the threshold. These results can be linked with Figure 1: when Pearson is used, a low improvement of the MAE is reached when using KNN, whereas a large improvement is reached for PMI and GMI.

We can also notice that when a threshold is fixed, GMI is more accurate than GMI, whereas PMI is more accurate when all items are used.

MAE does not get better when using a high value of T for Pearson, since a high Pearson value does not reflect that the two items have identical ratings: a correlation of almost 1 can be obtained even when items are not identically rated: this can for example be due to the influence of the deviation to the mean.

To highlight the bad influence of weakly dependent items in the computation of prediction, we studied the MAE of PMI on the 0.3% of the corpus that correspond to a 0.237 MAE value (threshold value $T = 0.6$). Computing the MAE on this corpus without any threshold (using all neighbors) leads to a MAE of 0.57. It shows that using a large number of weak dependent items actually increases MAE drastically. Similar results are obtained with GMI.

6.5 Combining several similarity measures

In the previous sections, we first presented a new method that selects the neighbors to use for prediction so that the sum of their similarities is equal to a fixed value V. An improvement of the MAE of PMI and GMI has been reached compared to KNN while having a constant coverage. We then studied MAE when only highly similar neighbors (with a similarity value higher than a fixed threshold T) are used for prediction. We noticed that this method reduced dramatically the coverage. However, MAE of PMI and GMI highly decreased according to this threshold.

In this section, we aim at taking advantage of both preceding methods, by combining them: when neighbors with similarity higher than a fixed threshold are available, only these neighbors are used (section 6.4), when no such neighbors are available, neighborhood with dynamic size is used (section 6.3). In Table 3, we have shown that GMI was the most accurate measure when threshold grew, whereas it was less accurate when all neighbors were used. That is why, we used GMI when neighbors with similarity higher than threshold T are available, and PMI is used in other cases.

Fig. 4. Evolution of the MAE value according to the value of the threshold neighborhood for GMI and PMI

Table 4 presents the evolution of the MAE according to T. We can remark that the evolution of the MAE according to the threshold T has the same aspect than the MAE for GMI used alone: it first increases, then decreases. In this combination, coverage is constant. The MAE obtained by using all neighbors with GMI used alone was 0.696 and 0.686 for PMI. The combination of both measures reaches a MAE of 0.677 with a threshold value of 0.4 that improves the reference MAE by 2.7%.

7 CONCLUSION

This paper aims at improving accuracy of item-based collaborative filtering approach and presents a new way to compute the similarity between items. This measure, called Positive Mutual Information (PMI), relies on the information theory measure of mutual information. It has the characteristics of using only a subpart of the classical mutual information measure. We compute its performance on the well-known MovieLens dataset and compare it with other well-known similarity measures such as the Cosine, Pearson correlation and the General Mutual Information. We also propose to consider only two assessment states instead of the usual rating scale. The users only have to express their opinions in terms of “I like it” or “I dislike it”. All the “no opinion” values

are discarded. Such ratings are easier to estimate from logs or usage, that is to say the implicit ratings that users should put on items if they had explicitely voted those items.

We show that the PMI increases the performance of both Cosine and Pearson correlation when used in identical contexts. It also increases performance of the General Mutual Information. This result is highly important as it shows that the computation of similarities between items can be restricted to those with similar behaviors, while decreasing time complexity.

We also propose an original way to compute the neighborhood of a given item. This neighborhood is made up of the most correlated items so that the sum of their similarities sums up to a fixed value. We showed that it decreases the MAE while using a smaller neighborhood size on average.

Finally, we showed that MAE can be increased by combining both GMI and PMI, leading to a MAE of 0.677 that corresponds to an improvement of 2.7%

Our major contributions are mainly the computation of similarity based on mutual information, the improvement of the accuracy when using only two opinions from the users ("I like" or "I dislike") and an item-dependent neighboorhood size. The basic hypothesis is that it is better for a system not to recommend any item than to suggest an item the user will dislike. Considering the HMAE, this goal has been achieved.

8 PERSPECTIVES

In a near future we plan to investigate the impact of the use of opposite items (terms (B) and (C) of Equation 7). We plan to replace the GMI equation by a combination of two sub-equations depending on the item characteristics. For example, if items have similar behaviors (as studied in this paper) the PMI will be used, otherwise NMI will be used.

We also plan to associate a confidence value with each similarity measure. This confidence value may be related to the number of users involved in the computation of the similarity. This value will be used to weight the similarity measure during the prediction phase.

References

1. G. Adomavicius, '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, (june 2005).
2. Linas Baltrunas and Francesco Ricci, 'Dynamic item weighting and selection for collaborative filtering', in *Proceedings ECM-PKDD*, pp. 135–145, (2007).
3. J. S. Breese, D. Heckerman, and C. Kadie, 'Empirical analysis of predictive algorithms for collaborative filtering', in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, ed., San Francisco Morgan Kaufmann, pp. 43–52, (1998).
4. Laurent Candillier, Frank Meyer, and Marc Boullé, 'Comparing state-of-the-art collaborative filtering systems', in *Proceedings of 5th International Conference on Machine Learning and Data Mining in Pattern Recognition, MLMD'07*, pp. 548–562, (2007).
5. Sylvain Castagnos and Anne Boyer, 'Personalized communities in a distributed recommender system', in *Proceedings of the 29th European Conference on Information Retrieval (ECIR 2007)*, Rome, Italy, (April 2007).
6. P. Chan, 'A non-invasive learning approach to building web user profiles', in *5th International Conference on Knowledge Discovery and Data Mining - Workshop on Web Usage Analysis and User Profiling*, San Diego, USA, (august 1999).
7. A. Fred and A. Jain, 'Robust data clustering', in *In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pp. 128–133, (2003).
8. Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins, 'Eigenstate: A constant time collaborative filtering algorithm', Technical report, UCB Electronics Research Laboratory, (2000).
9. J. Herlocker, J. Konstan, L. Terveen, and J. Riedl, 'Evaluating collaborative filtering recommender systems', *ACM Transactions on Information Systems*, **22**(1), 5–53, (2004).
10. J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl, 'An algorithmic framework for performing collaborative filtering', in *Proceedings of the SIGIR conference*, pp. 230–237, (1999).
11. George Karypis, 'Evaluation of item-based top-n recommendation algorithms', in *Proceedings of CIKM*, pp. 247–254, (2001).

12. Jin Liu, QianPing Wang, Kun Fang, and Qinfeng Mi, 'An optimized collaborative filtering approach combining with item-based prediction', in *Proceedings of the 11th International Conference on Computer Supported Cooperative Work in Design*, pp. 157–161, (2007).
13. Manos Papagelis and Dimitris Plexousakis, 'Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents', *Engineering Applications of Artificial Intelligence*, **18**, 781–789, (2005).
14. P. Resnik, N. Iacovou, M. Suchak, P. Bergstrom, and J Riedl, 'Grouplens: An open architecture for collaborative filtering of netnews', in *Proceedings of the Conference on Computer Supported Collaborative Work*, ed., ACM Press New York, pp. 175–186, (1994).
15. B. M. Sarwar, G. Karypis, Konstan J. A., and J. Riedl, 'Item-based collaborative filtering recommendation algorithms', in *Proceedings of 10th International World Wide Web Conference (WWW10)*, (2001).
16. C.E. Shannon, 'A mathematical theory of communication', *Bell Sys. Tech. Journal*, **27**, 398–403, (1948).
17. U. Shardanand and P. Maes, 'Social information filtering: Algorithms for automating "word of mouth"', in *Proceedings of AC CHI'95 Conference on Human Factors in Computing Systems*, pp. 210–217, (1995).
18. Ellen Spertus, Mehran Sahami, and Orkut Buyukkokten, 'Evaluating similarity measures: A large-scale study in the orkut social network', in *Proceedings of the 11th International Conference on Knowledge Discovery and Data Mining*, pp. 678–684, (august 2005).
19. Fu Lee Wang, 'Improvements to collaborative filtering systems', in *Proceedings of the International Symposium on Computational and Information Science*, volume 3314, pp. 975–981, (2004).
20. K. Yu, X. XU, M. Ester, and H.P. Kriegel, 'Feature weighting and instance selection for collaborative filtering: An information-theoretic approach', *Knowledge and Information Systems*, **5**, 201–224, (2003).
21. Kai Yu, Xiaowei Xu, Martin Ester, and Hans-Peter Kriegel, 'Selecting relevant instances for efficient and accurate collaborative filtering', in *In Proceedings of the Conference on Information and Knowledge Management CIKM*, pp. 239–246, (2001).
22. W. Ziqiang and F. Boqin, 'Collaborative filtering based on mutual information', in *Proceedings of the APWeb Conference, LNCS 3007*, pp. 405–415, (2004).