Collaborative Filtering: Fallacies and Insights in Measuring Similarity. *

Panagiotis Symeonidis, Alexandros Nanopoulos, Apostolos N. Papadopoulos, and Yannis Manolopoulos

Aristotle University of Thessaloniki, Dept. of Informatics, 56224, Greece
{symeon, alex, apostol, manolopo}@delab.csd.auth.gr

Abstract. Nearest-neighbor collaborative filtering (CF) algorithms are gaining widespread acceptance in recommender systems and e-commerce applications. These algorithms provide recommendations for products, based on suggestions of users with similar preferences. One of the most crucial factors in the effectiveness of nearest-neighbor CF algorithms is the similarity measure that is used. The most popular measures are the Pearson correlation and cosine similarity. In this paper, we identify existing fallacies in the calculation of these measures. We propose a novel approach, which addresses the problem and substantially improves the accuracy of CF results. Moreover, we propose an evaluation procedure that produces reliable conclusions about the performance of nearest-neighbor CF algorithms. Through the proposed evaluation procedure, our experimental results identify the problems of existing approaches (which could not be revealed with existing evaluation procedures) and illustrate the superiority of the proposed approach.

1 Introduction

Information Filtering has become a necessary technology to attack the "information overload" problem. In our everyday experience, while searching on a topic (e.g., products, movies, etc.), we often rely on suggestions from others, more experienced on it. In the Web, however, the plethora of available suggestions renders it difficult to detect the trustworthy ones. The solution is to shift from individual to collective suggestions. Collaborative Filtering (CF) applies information retrieval and data mining techniques to provide recommendations based on suggestions of users with similar preferences. CF is a very popular method in recommender systems and e-commerce applications. Two types of CF algorithms have been proposed: (a) *nearest-neighbor* (a.k.a. memory-based) algorithms, which rely on finding the most similar ones among the past users, and (b) *model-based* algorithms, which develop a model about user ratings. Research results and practical experience have reported that nearest-neighbor algorithms present excellent performance in terms of accuracy, for multi-value rating data [7].

Nearest-neighbors CF algorithms are influenced by several factors. The similarity measure for finding nearest-neighbors, is among the most crucial ones.

* This work is conducted while the first two authors where scholars of the Greek State Scholarship Foundation

Related research has mainly used as similarity measures the Pearson correlation and the cosine similarity.[1] One issue that impacts the accuracy of CF is the *sparsity* of past users' ratings, which emanates from the fact that each user usually rates only a very small percentage of the total items (maybe less than 0.1%). The measuring of similarity is affected by sparsity, especially due to the choice that has been followed in related work to compute the similarity between two users only with respect to the items that have been rated by *both* of them. This leads to the finding of spurious neighbors and to inability of providing correct recommendations. The reason is twofold: (a) similarity is implausibly computed based on an inadequate number of items, and (b) by ignoring the items rated by only *one* of the two users, we do not consider how much their preferences may differ. Nevertheless, the procedures used so far for the assessment of nearest-neighbor CF algorithms, are not able to identify the inefficiencies caused by the aforementioned choice.

In this paper, we first provide a thorough analysis of the factors involved in the computation of similarity measures and in the evaluation of the nearest-neighbor CF algorithms. During our examination we identify choices that have been incorrectly adopted in related work. Next, we propose a new approach, which addresses the problem and substantially improves the accuracy of CF results. Our contributions are summarized as follows:

- The revealing of existing fallacies in popular similarity measures.
- A novel method for similarity computation and an evaluation procedure that produces reliable conclusions about the performance of nearest-neighbor CF algorithms.
- Experimental results which take into account many factors and demonstrate the superiority of the proposed method (more than 40% improvements in terms of precision).

The rest of this paper is organized as follows. Section 2 summarizes the related work, whereas Section 3 contains the analysis of the examined CF factors. The proposed approach is described in Section 4. Experimental results are given in Section 5. Finally, Section 6 concludes this paper.

2 Related work

In 1992, the Tapestry system [3] introduced Collaborative Filtering (CF). In 1994, the GroupLens system [12] implemented a CF algorithm based on common users preferences. Nowadays, it is known as user-based CF algorithm, because it employs users' similarities for the formation of the neighborhood of nearest users. Since then, many improvements of user-based algorithm have been suggested, e.g., [5].

In 2001, another CF algorithm was proposed. It is based on the items' similarities for a neighborhood generation [14, 8]. Now, it is denoted as item-based or item-item CF algorithm, because it employs items' similarities for the formation of the neighborhood of nearest users.

[1] Cosine similarity is related to Pearson correlation which represents the angular separation between two normalized data vectors measured from the mean, while the cosine similarity measures the angular separation of two data vectors measured from zero.

Most recent work followed the two aforementioned directions (i.e., user-based and item-based). Herlocker et al. [6] weight similarities by the number of common ratings between users/items, when it is less than some threshold parameter γ. Deshpande and Karypis [2] apply item-based CF algorithm combined with conditional-based probability similarity and Cosine Similarity measures. Xue et al. [15] suggest a hybrid integration of aforementioned algorithms (nearest neighbor CF algorithms) with model-based CF algorithms. Finally, recent extensions of CF include issues like streaming data [1] or privacy preserving [11]. In the following section we elaborate further on related work, through the analysis of the factors we examine.

3 Examined factors

In this section, we provide details for the examined factors that are involved in measuring similarity and evaluating CF results.

Similarity measure: Related work [6, 9, 10, 14] has mainly used Pearson correlation and cosine similarity. In particular, user-based (UB) CF algorithms use the Pearson correlation (Equation 1)[2], which measures the similarity between two users, u and v. Item-based (IB) CF algorithms use a variation of adjusted cosine-similarity (Equation 2)[3], which measures the similarity between two items, i and j, and has been proved more accurate [9, 14], as it normalizes bias from subjective ratings.

$$\text{sim}(u,v) = \frac{\sum_{\forall i \in S} (r_{u,i} - \overline{r}_u)(r_{v,i} - \overline{r}_v)}{\sqrt{\sum_{\forall i \in S} (r_{u,i} - \overline{r}_u)^2} \sqrt{\sum_{\forall i \in S} (r_{v,i} - \overline{r}_v)^2}}, \quad S = I_u \cap I_v. \tag{1}$$

$$\text{sim}(i,j) = \frac{\sum_{\forall u \in T} (r_{u,i} - \overline{r}_u)(r_{u,j} - \overline{r}_u)}{\sqrt{\sum_{\forall u \in U_i} (r_{u,i} - \overline{r}_u)^2} \sqrt{\sum_{\forall u \in U_j} (r_{u,j} - \overline{r}_u)^2}}, \quad T = U_i \cap U_j. \tag{2}$$

Herlocker et al. [6] proposed a variation of the previous measures, which henceforth is denoted as Weighted Similarity (WS). If *sim* is a similarity measure (e.g., Pearson or cosine), then WS is equal to $\frac{\max(c,\gamma)}{\gamma} \cdot sim$, where c is the number of co-rated items and γ is a threshold value used by WS.

Equation 1 takes into account only the set of items, S, that are *co-rated* by both users. This, however, ignores the items rated by only one of the two users. The number of the latter items denotes how much their preferences differ. Especially for the case of sparse data, by ignoring these items we discard significant information. Analogous reasoning applies for Equation 2, which considers (in the numerator) only the set of users, T, that co-rated both the examined pair of items, and for WS, which is based on Equations 1 or 2. To address the problem, in Section 4 we examine alternative definitions for S and T.

Neighborhood size: The number, k, of nearest neighbors, used for the neighborhood formation, directly affects accuracy. Related work [5, 13] utilizes a k in

[2] $r_{u,i}$ is the rating of u on item i. I_u is the set of items rated by u. $\overline{r}_u$, $\overline{r}_v$ are the mean ratings of u and v over their co-rated items.

[3] U_i is the set of users that rated i. Means $\overline{r}_u$, $\overline{r}_v$ are taken over all ratings of u and v.

the range of values between 10 and 100. The optimum k depends on the data characteristics. Therefore, CF algorithms should be evaluated against varying k. Moreover, an issue that has not been precisely clarified in related work, is whether we include in the neighborhood a user or item with negative similarity. In order to improve accuracy, we suggest keeping only the positive similarities for the neighborhood formation, even if less than the specified number k of neighbors remain. This approach is also followed in several works [10].

Positive rating threshold: Recommendation for a test user is performed by generating the top-N list of items that appear most frequently in his formed neighborhood (this method is denoted as Most-Frequent item-recommendation). Nevertheless, it is evident that recommendations should be "positive". Recommending an item that will be rated with, e.g., 1 in 1-5 scale should not contribute to the increase of accuracy. We use a rating-threshold, P_τ, to recommended items whose rating is not less than this value. If we do not use a P_τ value, then the results become misleading.

Amount of sparsity: In many real cases, users rate only a very small percentage of items, thus rating data become sparse. Due to lack of sufficient information, sparsity leads to inaccurate recommendations. For this reason, several recent works concentrate only on sparse data [6, 8, 14] (e.g., Movielens). However, there exist dense rating data sets (e.g., Jester [4]). To provide complete conclusions, we have to examine both cases.

Evaluation Metrics: Several metrics have been used for the evaluation of CF algorithms, for instance the Mean Absolute Error (MAE) or the Receiving Operating Characteristic (ROC) curve [6, 7]. MAE represents the absolute differences between the real and the predicted values and is an extensively used metric. From our experimental study (Section 5) we understood that MAE is able to characterize the accuracy of prediction, but is not indicative for the accuracy of recommendation. Since in real-world recommender systems the experience of users mainly depends on the accuracy of recommendation, MAE may not be the preferred measure. For this reason we focus on widely accepted metrics from information retrieval. For a test user that receives a top-N recommendation list, let R denote the number of *relevant recommended items* (the items of the top-N list that are rated higher than P_τ by the test user). We define the following:

- *Precision* is the ratio of R to N.
- *Recall* is the ratio of R to the total number of relevant items for the test user (all items rated higher than P_τ by him).

Notice that with the previous definitions, when an item in the top-N list is not rated at all by the test user, we consider it as *irrelevant* and it counts negatively to precision (as we divide by N). In the following we also use $\mathrm{F}_1 = 2 \cdot \mathrm{recall} \cdot \mathrm{precision}/(\mathrm{recall} + \mathrm{precision})$. F_1 is used because it combines both the previous metrics.

Setting a baseline method: Existing experimental evaluations lack the comparison against a baseline algorithm. A baseline algorithm has to be simple and to indicate what can be attained with as little effort as possible. Through a baseline, we can see the actual improvement due to existing CF algorithms.

4 Proposed methodology

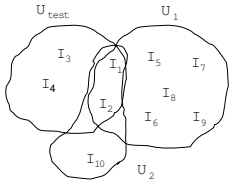
Next, we describe in more detail our proposed method. We first examine the factor of the similarity measure. Next, we elaborate on the issue of how to assign ratings to non-rated items, which is required by the proposed similarity measure. Finally, we describe the development of a baseline algorithm.

4.1 The UNION similarity measures

According to the definition of sets S and T given in Equations 1 and 2, only the items that are co-rated by both users are considered. For instance, Figure 1a depicts the ratings of two users, U_1 and U_2, over five items (dash denotes an unrated item). When only co-rated items are considered, then the similarity between U_1 and U_2 will be computed based on the ratings for I_1 and I_3.

	I_1	I_2	I_3	I_4	I_5
U_1	3	-	5	4	-
U_2	4	2	4	-	-

(a)



(b)

Fig. 1. Example of: (a) the ratings of two users over five items, (b) a test user compared against two past users.

In case of sparse data, we have a very small amount of provided ratings to compute the similarity measure. By additionally constraining S and T with co-rated items only, we reduce further the effective amount of used information. To avoid this, we consider alternative definitions for S and T, given in Equation 3:

$$S = I_u \cup I_v, \quad T = U_i \cup U_j \tag{3}$$

According to Equation 3, S includes items rated by at least one of the users. In the example of Figure 1a, except the ratings for I_1 and I_3, the ratings for I_2 and I_4 will be considered too (the issue of how to treat items rated by only one user, will be discussed in the following). Similar reasoning is followed for the set T, in the case of IB CF. By combining the definitions of S and T given in Equation 3 with the Pearson correlation and adjusted cosine similarity measures, we get two reformed measures: UNION Pearson correlation (for UB) and UNION adjusted cosine (for IB), respectively.[4] Notice that in case of UNION Pearson correlation, user mean ratings correspond to the average user ratings over all rated items. To further understand why should not base similarity only on co-rated items, consider the following example.

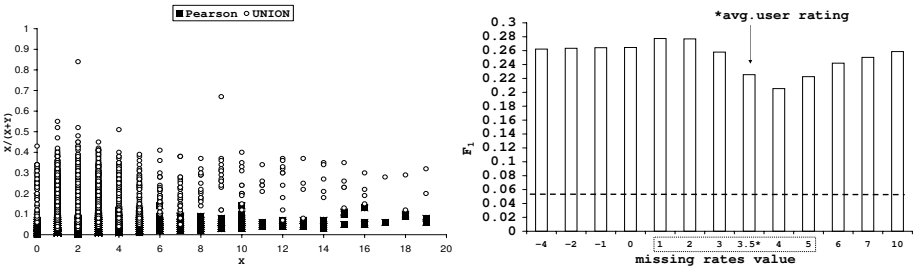
Example Figure 1b depicts the items rated positively (i.e., higher than P_τ) by a test user U_{test} and two users, U_1 and U_2, belonging in the training set. U_{test} and U_1 have co-rated items I_1 and I_2. Assume that U_{test} and U_1 rated I_1

[4] Henceforth, when it is clearly understood from the context whether we discuss about UB or IB, we use only the name UNION.

with 5 in the 1–5 scale, whereas I_2 have been rated by both of them with 4. Nevertheless, items I_3 – I_9 are rated only by U_{test} or U_1, and not by both. In this case, the Pearson measure of Equation 1, which is based on co-rated items only, results to the maximum possible similarity value (i.e., equal to 1) between U_{test} and U_1. However, this is based only on the 2 co-rated items and ignores the 7 items that are rated only by one of them. On the other hand, assume that U_2 rated I_1 and I_2 with 5 and 4, respectively. As previously, the Pearson measure of Equation 1 results to the maximum possible similarity between U_{test} and U_2, whereas U_{test} and U_2 differ in 3 items rated by only one of them. This example reflects the impotence of Equation 1 to capture the actual notion of similarity: despite the fact that U_{test} and U_1 differ at 7 items (which are rated by only one of them) and U_{test} and U_1 differ at 3, it assigns the same similarity value in both cases. □

In the previous example, if we designate U_1 as neighbor of U_{test}, we ignore two issues: (i) Items I_3 and I_4, will not be recommended to U_{test} by U_1, as U_1 has not rated them; this fact harms recall. (ii) Items I_5 – I_9 will be recommended by U_1, but as they have not been rated by U_{test}, this will harm precision. It follows that a desirable property from a similarity measure is to maximize the number, x, of items that are co-rated by the test user and each of his neighbors, relatively to the number, y, of items that are rated only by one of them (in the example of Figure 1b, for U_{test} and U_1, $x = 2$ and $y = 7$). In the best case, the ratio $x/(x + y)$ has value equal to 1 and in the worst 0.

To evaluate the previously described argument, we compared Pearson correlation against UNION UB by performing the following measurement. We used the MovieLens 100K data set and for each test user we computed its k nearest neighbors (k was set to 10) from the training set. Next, we measured x and y between each test user and each of his k neighbors. Figure 2a illustrates for each x, the resulting ratio $x/(x+y)$. Figure 2a clearly presents that Pearson measure results to significantly lower ratios than UNION UB. This explains why UNION UB compares favorably to Pearson correlation in terms of precision and recall, as will be presented experimentally in Section 5. (Due to lack of space we do not present the analogous comparison between adjusted cosine and UNION IB.)



(a) (b)

Fig. 2. (a) Measuring the ratio $x/(x + y)$. (b) Impact of assigned value for unrated items.

4.2 Assigning a value to unrated items

To calculate the UNION Pearson correlation between two users U_1 and U_2, we have to assign a rating by U_1 to an item that is rated only by U_2 (e.g., I_2 in the example of Figure 1a) and vice-versa. The same requirement holds for the UNION adjusted cosine measure in the IB case. Notice that this problem cannot be effectively solved with existing techniques for filling missing values, because the sparsity of user-item matrices severely hinders this task (more than 99.9% missing values). For this reason we assign the same rating value to all the required cases. There could be several options for this value. For instance, in a 1–5 scale, we can assign the 0 value, to reflect that user is not interested at all to rate the item. Assuming that user u did not rate item i, another option is to assign the average value of the provided ratings on other items by u, or the average of the provided ratings on i by all users.

To examine the impact of the selected value, we measured F_1 versus the assigned value, which is depicted in Figure 2b for the MovieLens 100K data set (it uses 1–5 scale). The dashed line in Figure 2b corresponds to F_1 of the Pearson correlation (it is independent from the assigned value). As shown, values between the positive threshold (in this case P_τ was set to 3) and the maximum rating of the scale, result to reduced F_1 (notice that this range also includes the user average rating value). The reason is that these values impinge the ability to distinguish the assigned values from the actual positive ratings. However, when we assign values smaller than P_τ or *outside* the rating scale, F_1 is not affected. The reason is that with such assigned values we do not miss the ability to distinguish the assigned values from the actual positive ratings (as the latter are always within the provided scale). Thus, we conclude that UNION is not significantly affected by the selection for the assigned ratings, as all values outside the rating scale or below P_τ result to about the same F_1. Even more, the values between P_τ and the upper limit of the scale result to significantly higher F_1 than Pearson measure. Henceforth, we assume that the assigned value is equal to 0.

4.3 Baseline algorithm

Considering the factors described in Section 3 regarding the evaluation procedure, we detail a baseline algorithm. We propose the one that recommends the N items that are most frequently rated positively in the entire training set. This algorithm is denoted as BL. BL is very simple and, as will be shown in our experimental results, it is quite effective. For instance, our experiments with Movielens-100K data set have shown that, with BL, when we simply propose the $N = 20$ most positively rated movies (20 most popular movies), precision reaches 40%. Therefore, the most frequently rated items are very probable to be selected by the majority of the users. For the aforementioned reasons, BL is a tool to clearly evaluate the actual improvement of existing CF algorithms. We will see in our experiments that asymptotically, as k (neighborhood size) increases, the performance of Pearson correlation and adjusted cosine tend to become equal with that of BL. In the extreme case where k is equal to the number of all users in the training set, the result of the Most-Frequent item-recommendation procedure, for the generation of the top-N list, becomes equivalent to BL.

5 Performance study

In the sequel, we study the performance of the proposed approach against Pearson correlation and adjusted cosine. Both the UB and IB cases are examined. Regarding the parameters, the following default values are assumed: for the neighborhood size the default k value is 10, for the recommendation list the default N value is 20, and for the size of training set the default value is 75%. Regarding WS, the γ value was set to 5. Evaluation is performed with the precision and recall metrics (given as percentages). We also use F_1 metric and MAE.

We perform experiments with three real data sets that have been used as benchmarks in prior work. In particular, we examined two MovieLens data sets: (i) the first one with 100,000 ratings assigned by 943 users on 1,682 movies, and (ii) the second one with about 1 million ratings for 3,592 movies by 6,040 users. The range of ratings is between 1(bad)-5(excellent) of the numerical scale. Moreover, we ran our experiments on the Jester data set, which contains 4.1 million ratings of 100 jokes from 73,496 users. Due to lack of space, we present results only for the first MovieLens and the Jester data sets, because they correspond to a sparse and a dense data set, respectively. The performance of the former has been verified with the results of the 1M real data set. Finally, in all data sets, we normalized the rating scale in the range 1–5, whereas P_τ is set to 3.

5.1 Results for user-based CF

First, we examine the UB CF case and compare the existing Pearson similarity and WS measures against UNION. We also include the baseline (BL) algorithm. The results for precision and recall vs. k are displayed in Figure 3a and b, respectively.

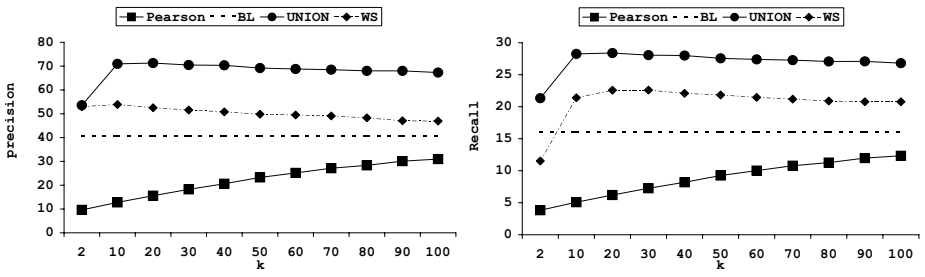
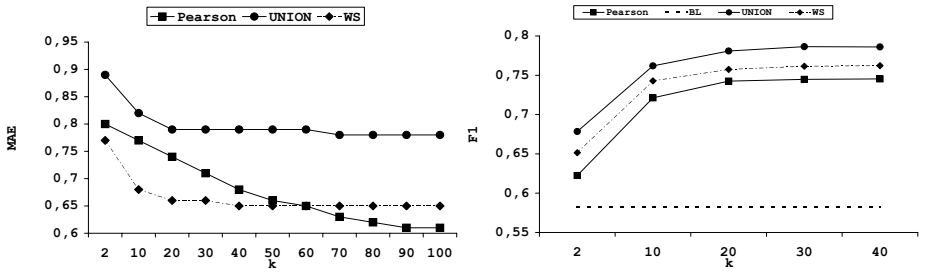


Fig. 3. Performance of user-based CF vs. k: (a) precision, (b) recall.

As shown, the existing Pearson measure, which is based on co-rated items, performs worst than BL. This result is surprising, as BL is very simple. WS improves Pearson, because the disadvantage of Pearson, due to co-rated items, is downsized by the weighting with the number of common items. UNION clearly outperforms all other measures for the reason that have been described in Section 4. Outside the examined k range (not displayed), Pearson stabilizes and never exceeds BL. As we already described, with increasing k, Pearson measure practically becomes equivalent to BL.

We now examine the MAE metric. Results are illustrated in Figure 4a (BL is only for recommendation, not prediction, thus omitted). As expected, Pearson yields the lowest MAE values, whereas WS is second best. This fact supports our explanation that MAE is indicative only for the evaluation of prediction and not of recommendation, as these measures did not attain the best performance in terms of precision and recall.



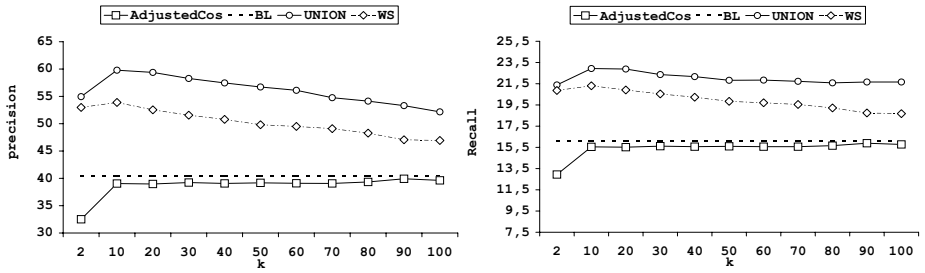
(a) (b)

Fig. 4. Performance of user-based CF vs. k: (a) MAE, (b) F_1 for dense data.

To consider the impact of density, we also examine the Jester data set. The results for the F_1 metric are depicted in Figure 4b. In this case, the relative differences are smaller than for the case of sparse data. The reason is that dense data have a sufficient amount of information, thus there is less need to exploit information in the way UNION does. Nevertheless, UNION still presents the best performance.

5.2 Results for item-based CF

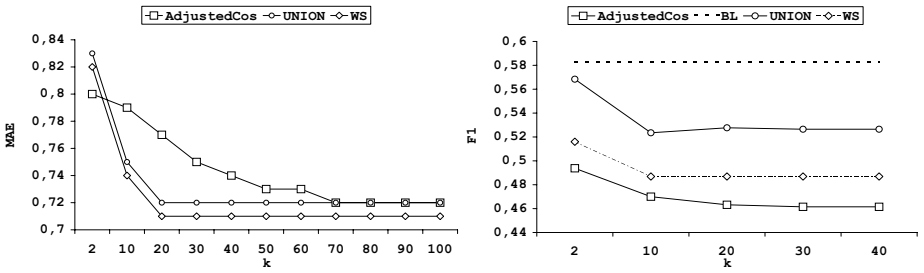
We perform similar measurements for the case of IB CF. Thus, we first examine the precision and recall for the adjusted cosine (considers co-rated items) against UNION. The results are depicted in Figure 5 and are analogous to those of the UB case. UNION clearly outperforms adjusted cosine and WS. Again, it is surprising to find that the adjusted cosine looses out by BL.



(a) (b)

Fig. 5. Performance of item-based CF vs. k: (a) precision, (b) recall.

Next, we compare adjusted cosine, UNION, and WS against MAE. The results are illustrated in Figure 6a. Differently to UB, all measures have similar MAE, and for larger k values they converge to the optimum MAE. Adjusted cosine does not present better MAE, because in its denominator it considers all items and not just the co-rated ones (see Equation 2). This improves its performance for the task of recommendation and worsens the performance of prediction.[5]



(a) (b)

Fig. 6. Performance of item-based CF vs. k: (a) MAE, (b) F_1 for dense data.

Regarding the examination of the dense data set (Jester), the results for the F_1 metric are illustrated in Figure 6b. Since IB CF has been designed to suit the needs of sparse data, we find out that for dense data all item-based algorithms are outperformed by BL. This is the case even for UNION, although it performs better than adjusted cosine. This result clarifies the need to examine CF algorithms for all the involved factors, in this case the amount of sparsity, in order to draw more complete conclusions.

5.3 Comparative results

In this section, we compare UNION for the UB and IB cases, as the corresponding UNION measures were shown to have the best performance in each case separately. The results for precision are depicted in Figure 7a, whereas those for recall are depicted in Figure 7b.

These results demonstrate that UB CF compares favorably against IB CF when UNION is used. The difference in precision is larger than 10%, whereas with respect to recall, it exceeds 5% (we refer to the optimum values resulting from the tuning of k). This conclusion contrasts the existing one, that IB is more preferable than UB, for the case of sparse data. The reason is that UB CF is more focused towards the preferences of the target user. In contrast, with IB CF, the recommended items may have been found similar by transactions of users with much different preferences than the ones of the target user. Thus, they may not directly reflect the preferences of the latter. However, this property could not be revealed with the existing similarity measures and evaluation procedures.

[5] We have examined a variation of adjusted cosine that uses only the co-rated items in the denominator. As expected, it resulted to worse precision and recall, but to better MAE.

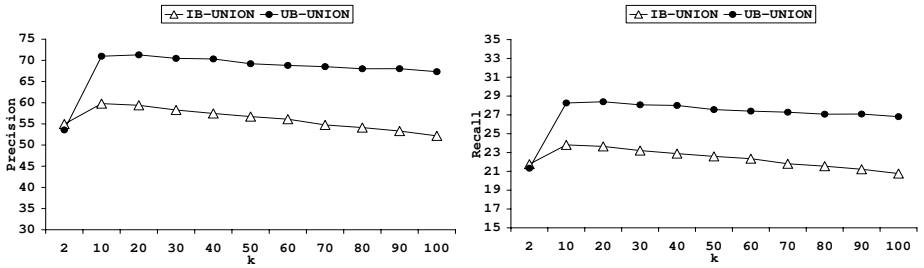


Fig. 7. Comparison between UB and IB: (a) precision, (b) recall.

The previous conclusion is in accordance with the one resulting from the comparison for the dense data set (Jester). Due to lack of space we do not present a graph for this case. However, from Figure 4b and Figure 6b it is easy to see that UB performs much better than IB when UNION is used, as the former is better than BL and the latter is worse.

6 Conclusions

In this paper, we performed a thorough study of neighborhood-based CF, which brought out several factors that have not been examined carefully in the past. We proposed a novel approach (UNION) for measuring similarity in nearest-neighbor CF applications. UNION successfully exploits more information in case of sparse data and considers how much the ratings of two users differ in order to provide accurate recommendations. We carried out extensive experimentation which reforms several existing beliefs and provides new insights. In particular, we highlight the following conclusions from our examination:

- In contrast to what is reported in majority of related work, MAE is not indicative for the accuracy of the recommendation process. It is, though, useful to characterize the quality of the similarity measure (as reflected in the process of prediction).
- Constraining similarity measures with co-rated items, weaknesses the measure. Though it is somewhat useful to consider the number of co-rated items (as WS does), the strict constraining inside the formulae for similarity measures is not suitable.
- The proposed approach, which does not use co-rated items only, substantially improves the performance of CF in terms of precision and recall, especially for sparse data. This conclusion was also explained through the measurement of ratio $x/(x+y)$ in Section 4. This measurement demonstrated the need to minimize the number, y, of items that are rated only by one of the users, a fact that is attained by UNION and not by existing similarity measures.
- Our results showed that, user-based compares favorably to item-based CF, and that item-based CF is not appropriate for dense data.

– Finally, the proposed baseline (BL) algorithm can better characterize the performance of existing CF algorithms. Its comparison against widely-accepted algorithms has produced surprising results.

We have to notice that item-based algorithms employ off-line computation, which is an advantage over user-based algorithms in terms of execution time. For this reason, in our future work we will consider the issue of scalability and compare the two approaches for this factor as well. Moreover, we will examine new algorithms for the generation of the top-N recommendation list.

References

1. J. Barajas and X. Li. Collaborative filtering on data streams. In *Proc. PKDD Conf.*, pages 429–436, 2005.
2. M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Trans. on Information Systems*, 22(1):143–177, 2004.
3. D. Goldberg, D. Nichols, M. Brian, and D. Terry. Using collaborative filtering to weave an information tapestry. *ACM Communications*, 35(12):61–70, 1992.
4. K. Goldberg, T. Roeder, T. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.
5. J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proc. ACM SIGIR Conf.*, pages 230–237, 1999.
6. J. Herlocker, J. Konstan, and J. Riedl. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval*, 5(4):287–310, 2002.
7. J. Herlocker, J. Konstan, L. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. on Information Systems*, 22(1):5–53, 2004.
8. G. Karypis. Evaluation of item-based top-n recommendation algorithms. In *Proc. ACM CIKM Conf.*, pages 247–254, 2001.
9. R. McLauglin and J. Herlocher. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In *Proc. ACM SIGIR Conf.*, pages 329–336, 2004.
10. B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Improving the effectiveness of collaborative filtering on anonymous web usage data. In *Proc. Workshop Intelligent Techniques for Web Personalization*, pages 53–60, 2001.
11. H. Polat and W. Du. Privacy-preserving collaborative filtering on vertically partitioned data. In *Proc. PKDD Conf.*, pages 651–658, 2005.
12. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering on netnews. In *Proc. Conf. Computer Supported Collaborative Work*, pages 175–186, 1994.
13. B. Sarwar, G. Karypis, J. Konstan, and R. J. Analysis of recommendation algorithms for e-commerce. In *Proc. ACM Electronic Commerce Conf.*, pages 158–167, 2000.
14. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proc. WWW Conf.*, pages 285–295, 2001.
15. G. Xue, C. Lin, and Q. e. a. Yang. Scalable collaborative filtering using cluster-based smoothing. In *Proc. ACM SIGIR Conf.*, pages 114 – 121, 2005.