International Journal of Scientific Research and Engineering Development-– Volume 7 Issue 3, May-June 2024
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

A Comparison of Collaborative Filtering Algorithms on Recommender Systems

Le Van Thinh*, Tran Thi Dang Tam**,
Truong Thanh Tam***, Le Nhu Toan****
*, **, ***, **** Faculty of Information Technology – Economics,
Mientrung industry and trade college, Viet Nam
Email:* thinhcdcn@gmail.com, ** tranthidangtam@tic.edu.vn , ***
center1984@gmai.com , ****lenhutoantic@gmail.com

..... ********** .....

Abstract:
The inherent weakness of the data on user ratings collected from web, such as the Data Sparsity Problem and new user, has limited the data analysis capability and prediction accuracy in recommender systems (RS). To alleviate this problem, the technique of collaborative filtering is successful in generating personalized recommendations, many researchers has resulted in numerous algorithms but no comparison of the different strategies has been made and Comparison of collaborative filtering algorithm does not exist yet. In this paper, we compare different techniques found in the literature, and we study the characteristics of each one, highlighting their principal strengths and weaknesses. Several experiments have been performed based on a detailed data analysis of MovieLens dataset. The results have revealed us to highlight the advantages and drawbacks of each approach, and to propose some default options that we think should be used when using a given approach or designing a new RS in future.

Keywords: — Collaborative filtering, Recommender systems, User-Based, Item-Based

I. INTRODUCTION

Today, With hundreds of millions of websites and more coming online daily, has become the greatest source of information. In this context, information retrieval systems are essential tools to guide users to the information they are seeking. Specifically, users demand personalized search systems, not just limited to retrieving the most relevant items, but also more adequate for their particular tastes or interests. So, Quality of Service (QoS) plays a crucial role in service- oriented systems. Many researchers propose that QoS should be a key factor in the success of building critical service-oriented applications.

Recommender systems (RS) are software tools and techniques providing suggestions for items to be use of a user [18], collaborative filtering (CF) is one of popular recommendation algorithms, this is the aim of recommender systems, help automatic predictions (filtering) about the interests of a user by collecting preferences from many other users (collaborating). They use information about users, user profiles, to predict the utility or relevance of a particular item, thus providing personalized recommendations. Recommendation systems are attracting a lot of attention and have proven to be useful in contexts such as e-commerce, and they surely have a promising future in many other domains, like Web search engines, digital TV program recommenders, etc. Until now, recommender systems have been used basically in two tasks:

- First recommender systems, They have been used to predict the utility of a given item to the user[22,19]. In this task, often known as annotation in context, the user first selects the item (or items) which he is interested. This is

usually done after performing a search, browsing an online catalog, etc. The recommender system then predicts the rating the user would give to that item

- Second recommender systems have been used, to recommend a list of items to the user. In this case, often called the find good items task, the system chooses the items that it considers the most relevant. Actually, recommender systems can also be used for other tasks, such as find all good items, recommended sequence, just browsing or find credible recommender[11], although these have not yet attracted much interest among researchers.

With The explosive growth of e-commerce and online environments has made the issue of information search and selection increasingly serious; users are overloaded by options to consider and they may not have the time or knowledge to personally evaluate these options. so, Recommender systems been successfully deployed in commercial environments to attempts to recommend information items (movies, TV program/show/episode, video on demand, music, books, news, images, web pages, scientific literature such as research papers etc.) or social elements (e.g. people, events or groups) that are likely to be of interest to the user. There are four types of filtering technique used in Recommender System: demographic, content, collaborative and hybrid. The most widely and popularly used technique is collaborative filtering, numerous collaborative filtering algorithms based on different ideas and concepts have been developed to address Recommender systems problem. They can be categorized into two types:

- Memory Based Collaborative Filtering: Memory-based CF uses user-to-user or item-to-item correlations based on users' rating behaviour to recommend or predict ratings for users on future items. Correlations can be measured by various distance metrics, such as Pearson correlation coefficient, cosine distance, and Euclidean distance, ect. Memory-based collaborative filtering uses the whole training set each time it computes a prediction, which makes it easy to incorporate new data but suffers slow performance on large data sets. Speedup can be achieved by recalculating correlations and other needed information and incrementally updating them as item-based Collaborative Filtering algorithms[2, 14,] and user- based Collaborative Filtering algorithms [15, 4]
- Model Based Collaborative Filtering: Unlike memory-based CF, model-based approach does not use the whole data set to compute a prediction. Instead, it builds a model of the data based on a training set and uses that model to predict future ratings. For example, clustering based CF method builds a model of the data set as clusters of users, and then uses the ratings of users within the cluster to predict as model-based method as the Singular Value Decomposition (SVD) [5] .

However, the users in the datasets rated more than one MoviesLens, the lack of prior ratings makes it fundamentally difficult to find enough number of similar users and make accurate predictions for an individual with collaborative filtering method. On the other hand, due to the sparse ratings matrix with huge number of null values, large amount of computer memory will be wasted to store the useless values. Many research works has shown a rising interest in incorporating trust into ecommender systems to solve this problem, mainly by quantifying trust into numerical values and build a web of trust (WOT) for each user, using trust inference and trust propagation. The effectiveness of trust has been proved for many times, that it can improve the prediction accuracy efficiently. But, works that compare these techniques are scarce, making it difficult to select the best algorithm (or algorithms) in a given situation. Thus, In this paper we mainly focuses and compare different techniques of collaborative filtering, identifying their main advantages and limitations. We focus on both annotation in context and find good items tasks. The evaluation was performed following the most common methodology and metrics found in the literature. Our experiments show that Comparison and looking for the best algorithm. that it is an ideal candidate for online systems or for systems involving many users and/or items.

The article is structured as follows. Section 2 briefly describes the state of the art on Collaborative

Filtering as algorithms and evaluated metrics approaches of recommender systems are also presented. Section 3 the experiments performed are presented and the results discussed comparing the behavior of various algorithms under different situations. Several aspects of each algorithm, such as the effect of the different parameters on the results or the variation in accuracy depending on the rating matrix sparsity, are analyzed. The differences between memory-based and model-based algorithms are highlighted. Moreover, the importance of a good fit between model and data is also considered. Finally, we present the conclusions of this study and discuss what direction to take in future works.

I. State of the art on Collaborative Filtering.

A. *Collaborative filtering approaches*

Many literature review have indicated that collaborative filtering is one of the most well known, successful and widely implemented techniques [1, 3, 20, 16, 13] on RS. In the most typical scenario, these techniques deal with a set of users $U=\{u_1,u_2,...,u_M\}$, and a set of items $I=\{i_1,i_2,...,i_N\}$. Each user $u_i \in U$, has an associated profile consisting of the subset of items he has rated, $I_u \subseteq I$, and the corresponding rating for each item. Similarly, the subset of users that have rated a certain item, $U_i \subseteq U$, is defined. The active user, the user for whom a prediction is being obtained, is denoted as u_a. The relationship between services users and items can be denoted by a user-item matrix M x N. The ratings usually correspond to integer numbers in a certain range, being R, the set of possible ratings. If user m has not rated the item n yet, $r_{m,n}=0$.

In collaborative filtering–based systems, the user profile is the set of ratings given to different items. These ratings can be captured explicitly, that is, by asking the user, or implicitly by observing his/her interaction with the system. Generally, the rating is represented as a unary value (showing only the relevant items), binary (allowing to distinguish between good and bad items) or, more commonly, as a numerical value on a finite scale. The user ratings are stored in a table known as the rating matrix. This table is processed in order to generate the recommendations. Depending on how the data of the rating matrix are processed.

The biggest advantage of CF over content-based approach is that it only relies on opinions on items described by users [13]. Instead content-based systems require more detailed descriptions of each item, so as to generate similarities between items. Two general classes of CF algorithms were examined in [11]: Memory-based algorithm and model-based algorithm. Model-based algorithm can be viewed as calculating the expected value of a vote from a probabilistic perspective, based on what we know about the user. Related methods include cluster models and Bayesian networks. As for the memory-based algorithm, we will describe it explicitly in section 2.2

However, CF approach still suffers from three fundamental challenges [19]: data sparsity, new user (or new item) and scalability. Data sparsity refers to the situation that users only rate a small portion of the available items, thus resulted in a sparse user-item matrix where we can hardly find co-rated items between users. In new user (or new item) problem, the lack of historical information occurs on new items or users consequently lead to a 'dumb' state in RS, that the system fails to consider users with an empty file or items no one has previously rated. Scalability entails a large amount of computation when there are millions of users and items, which is usually the case in reality. Several approaches have been adopted in previous work to cope with this challenge and received moderately good results, as Dimensionality reduction methods, such as SVD, Latent Semantic Indexing (LSI), reduce the dimensions of matrix by getting rid of unimportant users or items [21] and applied an associative retrieval framework and spreading activation algorithms to deal with the sparsity problem[6] or used trust inference to alleviate this problem[17].

B. *Collaborative filtering Algorithms*

Normally, the task in collaborative filtering can be of two forms [20] prediction and recommendation. Prediction is a numeric value expressing the predicted rating score on an item from a particular user (we will denote this user as the active user). Recommendation is to recommend a list of items the active user will like probably. This section presents the collaborative filtering algorithms found in the

literature that were evaluated and compared in this work. An illustration of dependencies in a simple recommender system at Fig 1 following.

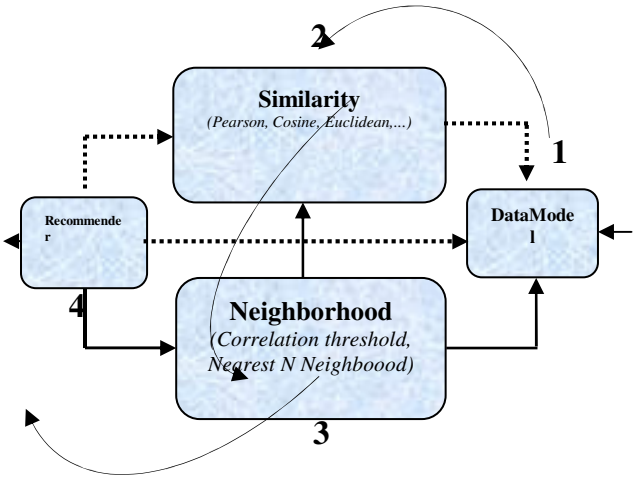


Fig 1: An illustration of dependencies in a simple recommender system, and the order in which components refresh their data structures

Hence, the user-based and items-based approach is considered a family of algorithms instead of a single algorithm. Each one combines different strategies for each step. In this work the following algorithms have been studied.

C. User-Based

User-Based algorithms[22,19] is suggested the first, different techniques to address each of these steps have been studied. Hence, the user-based approach is considered a set of algorithms instead of a single algorithm and different strategies. User-based algorithms, also known as user-based neighborhood approaches, are one of the most popular strategies of collaborative filtering. They follows a two-steps process [7]

(1) Calculate the similarity between the active user and the rest of the users.
(2) Select a subset of the users (neighborhood) according to their similarity with the active user and compute the prediction using the neighbor ratings.

In step (1) computation of similarity between users. There are several similarity algorithms that have been presented , when the values of these vectors are associated with a user's model then the similarity is called user-based similarity, whereas when they are associated with an item's model then it is called item-based similarity. The similarity measure can be effectively used to balance the ratings significance in a prediction algorithm and therefore to improve accuracy

In this work the following similarity algorithms have been studied.

- Euclidean Distance Similarity:

Euclidean Distance between two points is given by Minkowski distance metric. It is a metric on Euclidean space which can be considered as a generalization of both the Euclidean distance and the Manhattan distance. The formula of Euclidean distance is as following. [9].

$$d(x,y) = \sqrt{(\sum_{i=1}^{n}(x_i - y_i)^2)} \quad (1)$$

where n is the number of dimensions. It measures the numerial difference for each corresponding attributes of point x and point y. Then it combines the square of differencies in each dimension into an overal distance.

- Pearson Correlation degree

This is one of the first techniques proposed [19]. The pearson correlation of two variables is defined as their covariance divided by the product of their standard deviations. The pearson correlation has some problems: It doesn't take the preference overlap of two users into account, If two user overlap only on one item, the correlation can't be computed. Pearson correlation is defined by the following equation x and y represents two data objects.

$$p(x,y) = \frac{\sum x_i y_i - n\overline{xy}}{(n-1)s_x s_y} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}} \quad (2)$$

The similarity is not the best choice, it is not the worst choice, just because it is easy to understand, often raised in the earlier study. Pearson linear correlation coefficient must be assumed that the data pairs obtained from the normal distribution, and the data, at least in the context of logic must be equally spaced data. Pearson correlation calculated an extension to add an enumerated type (Weighting) parameters to make the number of overlapping calculated similarity factor.

- *Cosine Similarity*

Cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle between them, it can be derived by using the Euclidean dot product formula. (3)

$$x * y = \|x\|\|y\|\text{CoSine}(x,y)$$

Given two vectors of attributes, x and y, the cosine similarity, cosine(x,y), is represented using a dot product and magnitude as (4)

$$CoSine(x,y) = \frac{x * y}{\|x\|^2 \, x \, \|y\|^2} = \frac{\sum x_i y_i}{\sqrt{\sum x_i^2}\sqrt{\sum y_i^2}}$$

- *Spearman rank correlation coefficient*

Spearman Rank Correlation measures the correlation between two sequences of values. The two sequences are ranked separately and the differences in rank are calculated at each position. The distance between sequences x and y is computed using the following formula: (5)

$$P(x,y) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}$$

Where x_i and y_i are the i^{th} values of sequences x and y respectively. The range of Spearman Correlation is from -1 to 1. Spearman Correlation can detect certain linear and non-linear correlations.

- *Log-likelihood similarity*

One of the most fundamental concepts of modern statistics is that of likelihood. In each of the discrete random variables we have considered thus far, the distribution depends on one or more parameters that are, in most statistical applications. calculated the following formula

$$d(x,y) = 2 * \left(\left(x * \ln\left(x/E1\right) \right) + \left(y * \left(\ln\left(y/E2\right) \right) \right) \right) \quad (6)$$

Where E1 and E2 using the following formula:

$$E1 = N1 * (x+y) / (N1+N2)$$

$$E2 = N2 * (x+y) / (N1+N2) \quad (7)$$

The value 'N1' corresponds to the number of words in corpus one, and 'N2' corresponds to the number of words in corpus two (N values).

- *Tanimoto Coefficient*

The Tanimoto coefficient between two points, a and b is calculated as:

$$T(x,y) = \frac{x * y}{\|x\|^2 \, x \, \|y\|^2 - x * y} = \frac{\sum x_i y_i}{\sqrt{\sum x_i^2}\sqrt{\sum y_i^2} - \sum x_i y_i} \quad (8)$$

The Tanimoto similarity is only applicable for a binary variable, and for binary variables the Tanimoto coefficient ranges from 0 to +1 (where +1 is the highest similarity).

In setup (2) Neighborhood selection. In this work, Two alternatives have been studied.

- Correlation threshold [22]. This consists of selecting only those users whose similarity with the active user surpasses a given threshold.

- The N-nearest neighbors [19] is one of the simplest and oldest methods for pattern classification. Its performance crucially depends on the distance metrics used to identify nearest neighbors. Equation (10) is a simple expression of how k-nearest can be combined with CF algorithm. If user i belongs to the neighborhood of user a. i.e., $i \in N_a$, we simply set w(a,i)=1, otherwise w(a,i)=0. (9)

$$w(a,i) = \begin{cases} 1 & \text{if } i \in N_a \\ 0 & \text{else} \end{cases}$$

The two steps above can be normalized into several equations

- *Weighted by correlation*. The contribution of each neighbor is weighted by his/her similarity with the active user [22]. The more a user is similar to the active user, the more accurate his/her rating is supposed to be as a prediction.
- *Z-score normalization*. Before weighting the user rating according to similarity, it is normalized. Normalization assumes that the ratings of each user belong to different distributions: there are users that only give bad ratings to extremely bad items, others that save the good ratings for selected items, others that only rate good items, etc. Therefore, the mean, as well as the standard deviation, of the users are taken into account to normalize their contribution. The details of this normalization, proposed in [10], are shown in Equation (1). we use the classical Z-score normalization as the basis of our algorithm,

$$p_{ai} = \overline{v_a} + \delta_a \frac{\sum_{u \in Neigh_a} \left[\left(\frac{v_{ui} - \overline{v_u}}{\delta_u} \right) s(a,u) \right]}{\sum_{u \in Neigh_a} s(a,u)} \quad (10)$$

D. Item-Based

Item-based algorithms are similar to the user-based but, instead of looking for neighbors among users, they look for similar items. Just like user-based algorithms, different strategies can be used as similarity measure. After calculating the similarity between the different items, a subset with the N best neighbors, is selected. To compute the prediction, we add up the ratings the active user has given to such neighbors weighted by its similarity with the item to predict, calculated the following formula. (11)

$$p_{aj} = \frac{\sum_i (s(j,i) v_{ui})}{\sum_i |s(j,i)|}$$

One of the advantages of this algorithm over the user-based is that the similarity between items tends to be more static than the similarity between users, so the neighborhood can be computed offline.

E. Slope One

The slope one algorithms are based on predictors of the form in the equation (12), therefore, simpler than those used in the regression-based algorithm[12]. (12)

$$f(x) = x + b$$

where b is defined as the mean difference between each item and the item to predict, computed among the users that have rated both items. The final prediction is calculated as in the equation:

$$p_{uj} = \overline{v_u} + \frac{1}{|R_j|} \sum_{j \in R_j} \sum_{x \in R_{ji}} \frac{v_{xi} - v_{xj}}{|S_{ji}|} \quad (13)$$

Where S_{ji} is the set of users that have rated both items j and i, $S_{ji} = U_j \cap U_i$, and R_j is the set of items rated by the user for which $|Sji|>0$. Finally, a third variant, bi-polar slope one, has also been studied. It divides the items into those that the user has rated positively and those that have been rated negatively, taking the user mean as threshold.

F. Singular Value Decomposition

Singular value decomposition(SVD) [1] takes a rectangular matrix of gene expression data (defined as A, where A is a n x m matrix) in which the n rows represents the genes, and the m columns represents the experimental conditions. The SVD theorem states:

$$A_{n \times m} = U_{n \times n} S_{n \times m} V^T_{m \times m}$$

Where $U^TU = I_{nxn}$ and $V^TV = I_{pxp}$ (i.e. U and V are orthogonal)

Where the columns of U are the left singular vectors (gene coefficient vectors); S (the same dimensions as A) has singular values and is diagonal (mode amplitudes); and V^T has rows that are the right singular vectors (expression level vectors). The SVD represents an expansion of the original data in a coordinate system where the covariance matrix is diagonal.

Calculating the SVD consists of finding the eigenvalues and eigenvectors of AA^T and A^TA. The eigenvectors of A^TA make up the columns of V, the eigenvectors of AA^T make up the columns of U. Also, the singular values in S are square roots of eigenvalues from AA^T or A^TA. The singular values are the diagonal entries of the S matrix and are arranged in descending order. The singular values are always real numbers. If the matrix A is a real matrix, then U and V are also real.

II. Evaluation Metrics

For the purpose of evaluating and comparing the performance of each algorithm, we follow the most commonly used approach of hiding a certain percentage of rating scores from the dataset then applying each algorithm in turn to predict the value of hidden ratings. In this case the accuracy of each algorithm can be evaluated from the difference between actual rating value and predicted value. We apply two metrics which are commonly used in the field of recommender systems to evaluate each algorithm: Prediction Accuracy and Precision /recall.

A. Prediction Accuracy

- *Mean Absolute Error (MAE)* is the average absolute deviation of the actual rating values to the predicted values as can be shown in Equation (duoi).

$$MAE = \frac{\sum_{i=1}^{P} |p_{i,j} - r_{i,j}|}{P} \quad (15)$$

(14)

Where $p_{i,j}$ is the predicted rating value that user i give to item j, $r_{i,j}$ is the actual rating value and P is the number of hidden ratings which are able to be predicted by the algorithm. Thus the lower the MAE, the more accurately the recommender algorithm predicts user ratings because the predicted values do not vary very far from true ratings. The statistical properties[11] have made this metric one of the most popular when evaluating recommender systems.

- *Root mean squared error(RMSE)*: Related to the previous metric, the root mean squared error, calculated using Equation (2), places greater emphasis on larger Errors.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{P}(p_{i,j} - r_{i,j})^2}{P}} \quad (16)$$

The principal reason for using this metric is that these errors can have the greatest impact on the user decision.

B. Precision and recall:

Precision[23] is defined as the ratio of relevant items to recommended items. Recall is the proportion of relevant items that have been recommended to the total number of relevant items. It is desirable for a system to have high precision and recall values. However, both metrics are inversely related, such that when precision is increased, recall usually diminishes, and vice versa.

Each item can be either relevant or irrelevant to the user. We get, therefore, the following matrix:

	Recommended	Not Recommended	Total
Relevant	RR	RN	R=RR+RN
Not Relevant	FP	NN	IR=FP+NN
Total	REC=RR+FP	NREC=RN+NN	N=R+IR=REC+REC

Precision is the fraction of all recommended items that are relevant.

$$\text{Precision} = \frac{RR}{RR + FP} = \frac{RR}{REC} \quad (17)$$

Recall is the fraction of all relevant items that were recommended.

$$\text{Recall} = \frac{RR}{RR + RN} = \frac{RR}{R} \quad (18)$$

F-measure. Recall and precision measure different facets of the accuracy of the recommender system. They can be combined in a single quantity, the F-measure:

$$F_1 = \frac{2.\text{Precision}.\text{Recall}}{\text{Precision}.\text{Recall}} \quad (19)$$

III. EXPERIMENTS

A. Datasets

We evaluated the algorithm on three datasets: MovieLens[8], EachMovie[] and LibimSeti[]. These are the most popular datasets used by researchers and developers in the field of collaborative filtering, datasets can be downloaded from the Internet and have been used in many works.

MovieLens is provided by GroupLens a research group at university of Minnesota, It 100000 ratings in five-point scale(1, 2, 3, 4, 5) by 943 users on 1682 items. Each user has rated at least 20 movies. The data is randomly ordered. This is a tab separated list of user id | item id | rating.

The second dataset, EachMovie is a database of movie ratings collected by Systems Research Center of Digital Equipment Corporation. The dataset contains 2811983 ratings given by 72916 users for 1628 movies. User ratings were recorded on a numeric sixpoint scale (0.0, 0.2, 0.4, 0.6, 0.8, 1.0).
The Third dataset, LibimSeti is provided by the Charles University. These files contain 17,359,346 anonymous ratings of 168,791 profiles made by 135,359 LibimSeTi users as dumped on April 4, 2006. Ratings are on a 1-10 scale where 10 is best (integer ratings only). Each user has rated at least 20 ratings were included.

B. Methodology

The experiments were performed by dividing the dataset into two groups, a training subset and an evaluation subset, The first set corresponds to data the algorithm already knows, that is, the data used to

train the algorithm. With such information, the algorithm computes the recommendation that will be later compared with the original data present in the evaluation subset.

For EachMovie and LibimSeti dataset we randomly selected 20% of the data. For MovieLens dataset we randomly selected 80% of the data.

We have selected the training subset to constructed from a percentage of the available ratings, randomly chosen. For our tests we used all the following percentages: 20%, 40%, 60% and 80%. The evaluation subset was composed by randomly selecting 10% of the dataset. Obviously, ratings that appear in the evaluation subset were never included in the training subset. With high percentages of ratings in the training set, we can evaluate the behavior of the algorithm under relatively high density conditions. In contrast, a small percentage allows to evaluate the algorithm under sparsity conditions, common in the initial phases, or in domains with a large number of users and/or items.

In the evaluation, the most popular metrics in the literature, presented in Section 2.B. we are denoted as follows

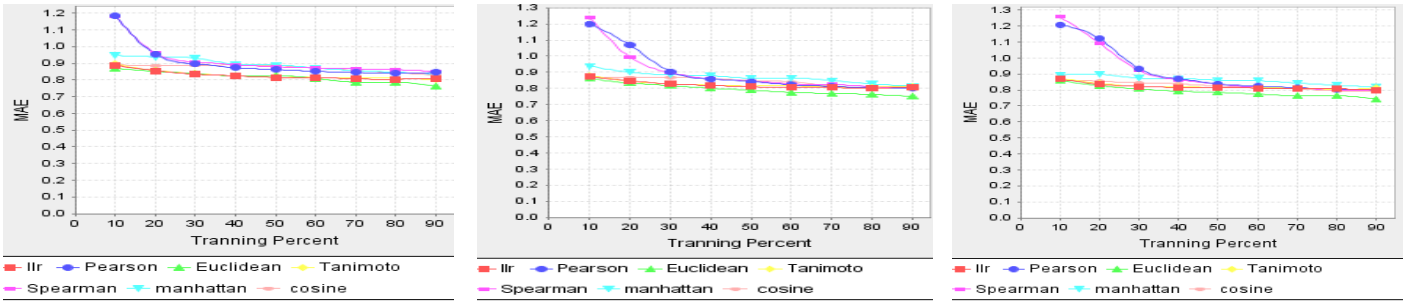
- Euclidean Distance Similarity (Euclidean)
- Pearson Correlation degree (Pearson)
- Cosine Similarity (Cosine)
- Spearman rank Correlation coefficient (Spearman)
- Mahattan distance (Manhattan)
- Log Likelihood Similarity (llr)

We used MAE and RMSE; Precision; Recall. Each test was repeated 3 times, so the results presented in the results section.

C. Results

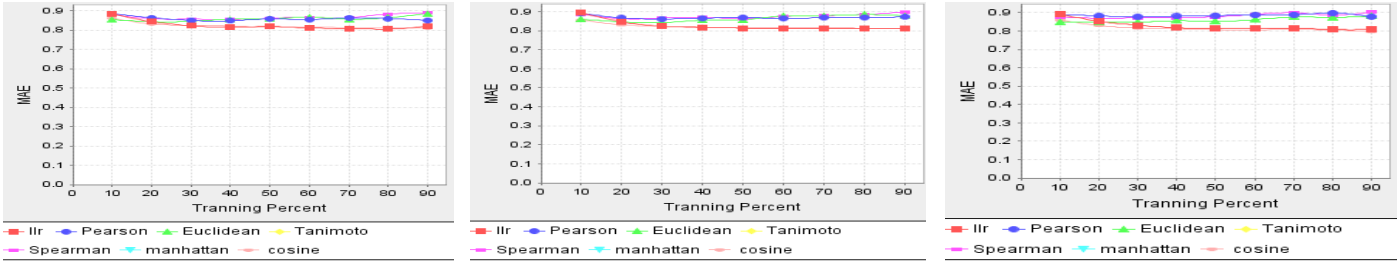
1) Experiment results 1

In this experiment, we evaluate the user-based algorithm using the method as described in Section 3. For the 1st evaluation method based on MAE and RMSE, we choose the neighbors users (Neighborhood selection) for the following two cases:



N=100 *N=200* *N=300*

Fig 2. User – Based N-nearest neighbor



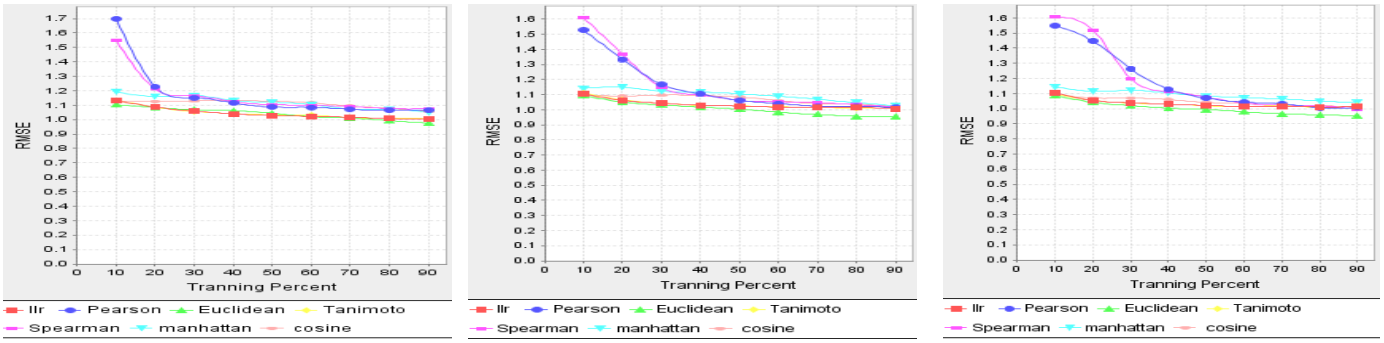
Threshold=0.7 *Threshold=0.8* *Threshold=0.9*

Fig 3. User – Based threshold

Case 1: We selected N-nearest neighbor equals to 100, 200 and 300 respectively. The results in Figure 2 and 4 show two methods MAE and RMSE, the Euclidean algorithm gives the best results for three values of N including training data density from 10% to 90%. While choosing the training data density is less than 30%, the Person and Spearman algorithm give not good results. The LLr, Cosine, Tanimoto and Manhattan algorithms give the same results with all the training data set, in which LLr has better results. Case 2: We selected threshold values

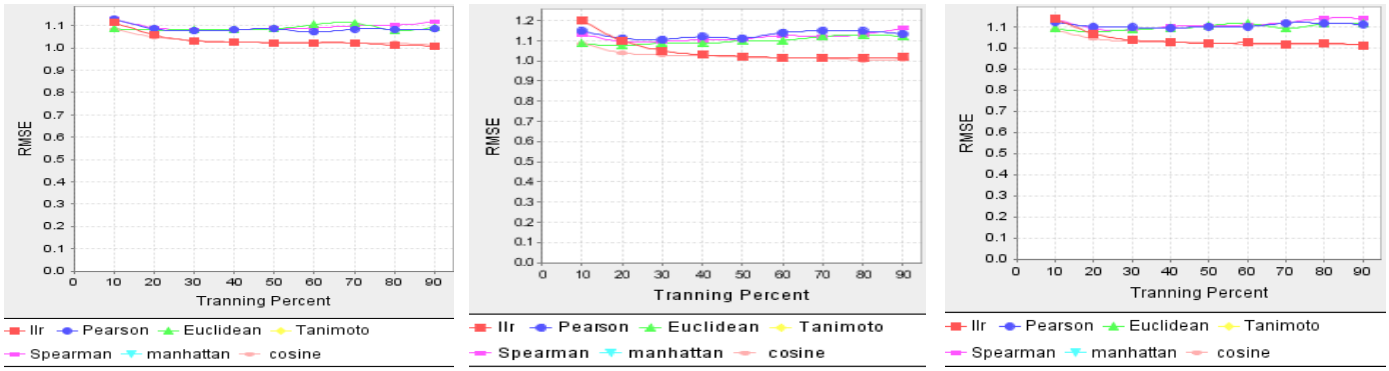
corresponding to 0.7, 0.8 and 0.9. The results in Figure 3 and 5 is similar. Euclidean and Cosine algorithm gives the best results when the training data set is less than 20%, Cosine and LLr algorithm give the best results when training set density from 20% to 90%. The other algorithms give the nearly unchanged results for each training set.

For the 2nd evaluation method are Precision, recall and F1. We chose Precision@50, recall@50, F1@50.



N=100 ***N=200*** *N=300*

Fig 4. User – Based N-nearest neighbor



Thres=0.7 **Thres=0.8** **Thres=0.9**

Fig 5. User – Based threshold

We make choose the Neighborhood selection for two above cases. Selected N-nearest neighbor equals 300, the results in Figure 6a shows the LLr algorithm has the best results, and case 2, in Figure 6b, when we selected threshold values corresponding to 0.9, the results show that LLr and cosine algorithm have the best results.

Therefore, the User-Based algorithm for both two evaluation methods. If choose N-nearest neighbor, the algorithm computes the similarity between users: Euclidean and LLr has the best results, even if choose the threshold, LLR and Cosine algorithm have the best results.

2) Experiment results 2

In this section, we evaluate the Item-Based algorithm with the evaluation method as Experiment 1 (MAE, RMSE, Precision, recall and F1) but in Item-Based not choose threshold and neighbors users N. The results in Figure 7 shows

Pearson algorithm has not good results for all the training data. As the other algorithm almost have the same result for each training set, in which Tanimoto and Manhattan algorithm have better results. Sothat, Item-Based algorithm, the best results are Tanimoto, manhattan.

3) Experiment results 3

In this experiment, we chose evaluation methods as two above experiments. But in this experiment, we turn choose the best algorithm in experiment 1

(User-Based) and experiment 2 (Item-Based) to compare with two other algorithms commonly used in collaborative filtering (SlopeOne and SVD algorithms).

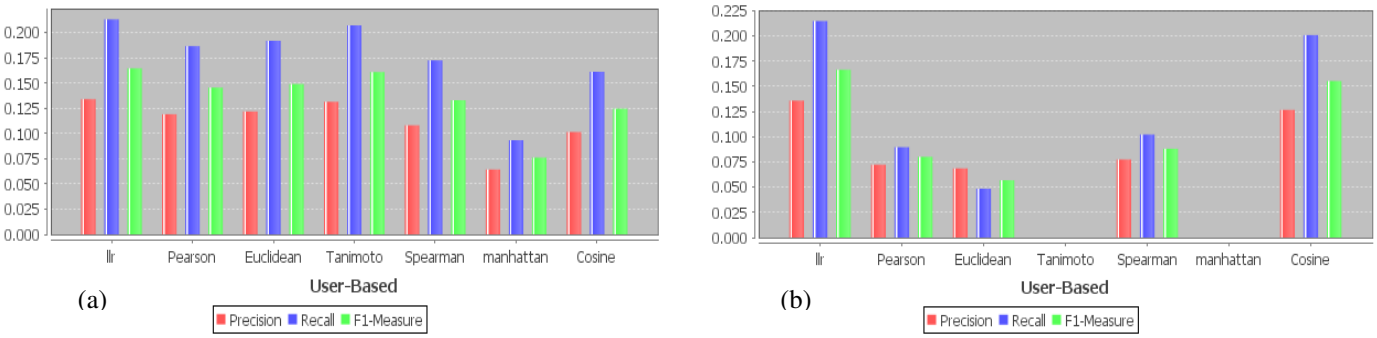


Fig 6. User – Based precision and recall (*fig a: N-nearest neighbor and fig b: threshold*)

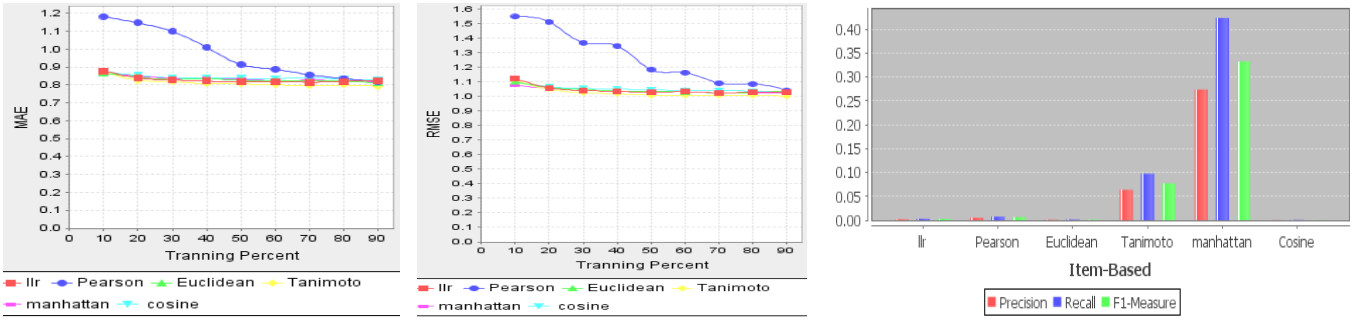


Fig 7. Item-based

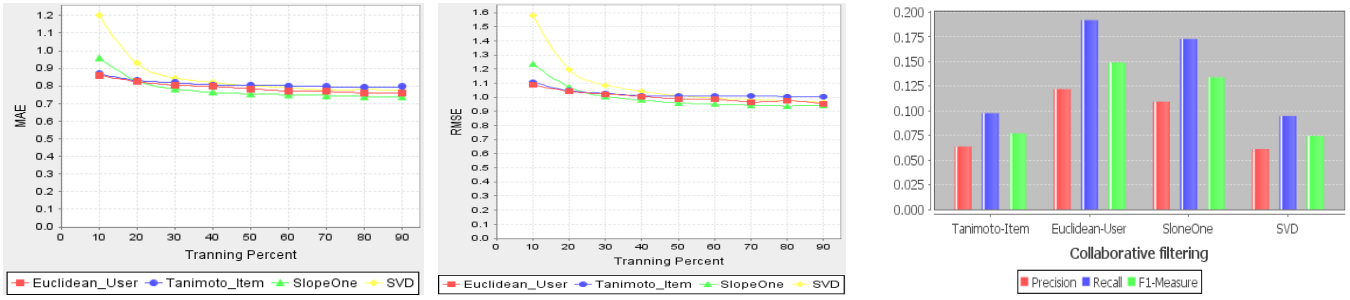


Fig 8. Collaborative filtering algorithms

In user-based algorithm, we choose the similarity calculates algorithm between two users is Eucliean, denoted Eucliean-User, and in Item-Based algorithm, we choose Tanimoto, denoted Tanimoto-Item. The results in Figure 8 shows the SVD algorithm with a training set of 10% -40% have not good result. SlopeOne algorithm gives the best results when the training data set from 20% to 90%, while Euclidea-user and Tanimoto-Item algorithm give relatively good results for all the training data set, but Euclidea-user algorithm gives better results .

In this article, we have performed an comparison of different collaborative filtering algorithms to observe the behavior of the algorithms under diverse situations, not only under the most favorable conditions and we are to explore an effective method which is able to make rating value predictions for users even when there exists sparsity problem on data sets and test the effect of trust, in line with the conventional CF algorithm. Similarly, the use of a clearly defined methodology will allow the comparison of other techniques in the future,

resolving one of the biggest problems in the evaluation of collaborative filtering algorithms.

On the basis of data analysis, We have also highlighted the characteristics of the methodology and metrics used, pointing out the limitations of the offline evaluation to determine the quality of the recommendations, their utility for the user by many different ways evaluated in experimental (MAE, RMSE, precision, recall). We choose the neighbors users for the following two cases: N-nearest neighbor and threshold values, compute similarity between users or items with all llr, Euclidean, Pearson, cosine, manhattan, Tanimoto algorithms. we choose different density of training data from 10% to 90% and looked the best results . The results demonstrate the great influence of matrix density on the accuracy of the algorithms and how this influence depends on the type of algorithm.

We have Seen also observed how user-based or item-based algorithms offer good behavior, worsening. The different experiments performed let us relate this behavior with the ability of the different algorithms to interpret the available data and extract useful information. We have found that most of the difficulties algorithms have in extracting information are related to the limitations of the techniques based on capturing similarities between users and/or items. However, most current algorithms still use these kinds of techniques. Two possible explanations for these limitations are the great diversity in opinions and tastes and the difficulty in finding such similarities under sparsity conditions. It would be interesting for future experiments to include new algorithms and other datasets . The application of the tendencies-based algorithm to contexts such as information retrieval on the Web may also be of special interest. This technique seems appropriate for the large amounts of data found in such domain. Another possibility is the study of other alternatives to the traditional techniques based on the similarities between users or items. The good results obtained by the tendencies based algorithm in our experiments are a good indication of what direction to take in future works.

REFERENCES

[1] Alter O, Brown PO, Botstein D. (2000) Singular value decomposition for genome-wide expression data processing and modeling. *ProcNatl Acad Sci U S A*, 97, 10101-6

[2] B.M. Sarwar, G. Karypis, J.A. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International World Wide Web Conference, pages 285-295, 2001.

[3] Burke, Robin. "Hybrid recommender systems: Survey and experiments." User modeling and user-adapted interaction12.4 (2002): 331-370.

[4] Feng Ge, A User-Based Collaborative Filtering Recommendation Algorithm Based on Folksonomy Smoothing, International Conference, CSE 2011, Qingdao, China, July 9-10, 2011. Proceedings, Part II.

[5] G. Linden, B. Smith, and J. York, “Amazon.com recommendations: item-to-item collaborative filtering,” IEEE Internet Computing, vol. 7, no. 1, pp. 76–80, 2003.

[6] Huang, Zan, Hsinchun Chen, and Daniel Zeng. "Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering." ACM Transactions on Information Systems (TOIS)22.1 (2004): 116-142.

[7] http://en.wikipedia.org/wiki/Collaborative_filtering.

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

[9] http://en.wikipedia.org/wiki/Euclidean_distance.

[10] HERLOCKER,J.,KONSTAN,J.A.,ANDRIEDL, J. 2002. An empirical analysis of design choices in neighborhoodbased collaborative filtering algorithms.Inform. Retr. 5,4, 287–310.

[11] HERLOCKER,J.L.,KONSTAN,J.A.,TERVEEN,L.G.,ANDRIEDL, J. T. 2004. Evaluating collaborative filtering recommender systems.ACM Trans. Inform. Syst. 22,1, 5–53.

[12] LEMIRE,D.ANDMACLACHLAN, A. 2005. Slope one predictors for online rating-based collaborative filtering. In Proceedings of SIAM Data Mining (SDM’05).

[13] Massa, Paolo, and Paolo Avesani. "Trust-aware collaborative filtering for recommender systems." On the Move to

Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE(2004): 492-508.

[14] M.Deshpande and G. Karypis. Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst., 22(1):143-177, 2004.

[15] Zhi-Dan Zhao, Ming-sheng Shang User-Based Collaborative-Filtering Recommendation Algorithms on Hadoop. In proceeding of: Third International Conference on Knowledge Discovery and Data Mining, WKDD 2010, Phuket, Thailand, 9-10 January 2010.

[16] O'Doherty, Daire, Salim Jouili, and Peter Van Roy. "Trust-based recommendation: an empirical analysis." Submitted to: Proceedings of the Sixth ACM SIGKDD Workshop on Social Network Mining and Analysis SNA-KDD, Beijing, China, ACM. 2012

[17] Papagelis, Manos, Dimitris Plexousakis, and Themistoklis Kutsuras. "Alleviating the sparsity problem of collaborative filtering using trust inferences." Trust management(2005): 125-140.

[18] Ricci, Francesco, Lior Rokach, and Bracha Shapira. Recommender Systems Handbook(2011).

[19] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994): 'GroupLens: An Open Architecture for Collaborative Filtering of Netnews', in Proceedings of the Conference on CSCW,Chapel Hill, NC, 1994, pp. 175-186.

[20] Sarwar, Badrul, et al. "Item-based collaborative filtering recommendation algorithms." Proceedings of the 10th international conference on World Wide Web. ACM, 2001.

[21] Su, Xiaoyuan, and Taghi M. Khoshgoftaar. "A survey of collaborative filtering techniques." Advances in Artificial Intelligence2009 (2009).

[22] Shardanand, U., & Maes, P. (1995). Social information filtering: Algorithms for automating 'Word of Mouth'. Proceedings of CHI '95. ACM. xx+598, 210-17.

[23] SARWAR,B.,KARYPIS,G.,KONSTAN,J.,A NDRIEDL, J. 2000. Application of dimensionality reduction in recommender systems–a case study. InProceedings of the ACM WebKDD Workshop.

