Grey forecast model for accurate recommendation in presence of data sparsity and correlation

Feng Xie[a,b,*], *Zhen Chen*[b,c], *Jiaxing Shang*[a], *Geoffrey C. Fox*[d]

[a] *Department of Automation, Tsinghua University*
[b] *Research Institute of Information Technology, Tsinghua University*
[c] *Tsinghua National Lab for Information Science and Technology*
Beijing, China 100084
[d] *School of Informatics and Computing, Indiana University, Indiana,USA 47408*

Abstract

Recently, recommender systems have attracted increased attention because of their ability to suggest appropriate choices to users based on intelligent prediction. As one of the most popular recommender system techniques, Collaborative Filtering (CF) achieves efficiency from the similarity measurement of users and items. However, existing similarity measurement methods have reduced accuracy due to problems such as data correlation and data sparsity. To overcome these problems, this paper introduces the Grey Forecast (GF) model for recommender systems. First, the Cosine Distance method is used to compute the similarities between items. Then, we rank the items, which have been rated by an active user, according to their similarities to the target item, which has not yet been rated by the active user; we use the ratings of the first k items to construct a GF model and obtain the required prediction. The advantages of the paper are three-fold: first, the proposed method introduces a new prediction model for CF, which, in turn, yields better performance of the model; second, it is able to alleviate the well-known sparsity problem as it requires less data in constructing the model; third, the model will become more effective when strong correlations exist among the data. Extensive experiments are conducted and the results are compared with several CF methods including item based, slope one, and matrix factorization by using two public data sets, namely, MovieLens and EachMovie.

*Corresponding author. Tel.: +86 15801438286
Email address: xiefeng10@gmail.com (Feng Xie)

The experimental results demonstrate that the proposed algorithm exhibits improvements of over 20% in terms of the mean absolute error (MAE) and root mean square error (RMSE) when compared with the item based method. Moreover, it achieves comparative, or sometimes even better, performance when compared to the matrix factorization methods in terms of accuracy and F-measure metrics, even with small k.

Keywords:
Recommender Systems, Collaborative Filtering, Grey Forecast Model, Data Sparsity, Data Correlation

1. Introduction

Recommender systems help users cope with the information overload experienced in a wide range of Web services and have been widely adopted in various applications, such as e-commerce (e.g., Amazon[1]), online video sharing (e.g., YouTube[2]), and online news aggregators (e.g., Digg[3]). Recommender systems have also been successfully developed for e-business and e-government applications [1], [2] and [3]. They can be used to present the most attractive and relevant items to the user based on the individual user's characteristics. As one of the most promising recommender techniques [4], collaborative filtering (CF) predicts the potential interests of an active user by considering the opinions of users with similar preferences. As compared to other recommender techniques (e.g., content based methods [5]), CF technologies have the capability to recommend unanticipated items to users, which are not similar to those they have seen before; this could work well in domains where the attribute content of items is difficult to parse. Generally, the representative CF technique, namely, the memory based CF technique [6], has been widely used in many commercial systems due to its simplistic algorithm and reasonably accurate recommendations. It obtains the user's ratings on different items by explicitly asking the user or by implicitly observing the user's interactions with the systems; these ratings are stored into a table known as the user-item rating matrix. Then, the memory based CF methods use similarity measurement methods to filter out the

[1] www.amazon.com
[2] www.youtube.com
[3] www.digg.com

users (or items) that are similar to the active user (or the target item) and calculate the prediction from the ratings of these neighbors. Memory based methods can be further classified into user based methods [7] or item based methods [8] depending on whether the process of defining neighbors follows the process of finding similar users or similar items.

Despite its widespread use, memory based CF techniques still suffer from several major problems, including the data sparsity problem [4] and [9], data correlation problem [10], and cold start problem [11] and [12]. The cold start problem can be regarded as a data sparsity problem. Hence, in this paper, we focus on the first two issues. In most recommender systems, each user rates only a small subset of the available items, and therefore, most of the entries in the rating matrix are empty. In such cases, determining similar users or items becomes a considerable challenge. Consequently, the similarity between two users or items cannot be calculated and the prediction accuracy becomes very low. Furthermore, the active users always tend to consume similar commodities, and the ratings for these items will be close, which indicates that there are strong correlations among the ratings. However, the existing similarity measurement methods, such as Cosine Distance and Pearson Correlation, suffer from such issues. Therefore, we cannot directly use similarities for rating prediction. To overcome these shortcomings, some researchers have developed algorithms that use models employing pure rating data to make predictions, such as clustering CF models [13] and [14], Bayesian belief nets (BNs) CF models [15] and [16], Markov decision process based (MDP-based) CF models [17], and latent semantic CF models [18]. However, some of these models are extremely complicated, require estimation of multiple parameters, and are sensitive to the statistical properties of data sets. In practice, many of these theoretical models have not been used in recommender systems due to the high costs involved.

In addition, dimensionality reduction techniques, such as singular value decomposition (SVD) [19], have been investigated to alleviate the data sparsity problem, where the unrepresentative users or items in the user-item rating matrix are removed to reduce the dimensionalities. However, useful information may be lost when certain users or items are discarded, and it is difficult to factor the matrix due to the high portion of missing values caused by its sparseness. Koren et al. [20] proposed a matrix factorization model, which is closely related to SVD. The model learns by only fitting the previously observed ratings. Its excellent performance enables it to be considered a state-of-the-art approach in rating prediction, but it also faces parameter

estimation problems and high time complexities. Luo et al. [21] and [22] improved the matrix factorization based method by including incremental computations and applying an adaptive learning rate.

In this paper, we present novel approaches that aim at overcoming data sparsity limitations and benefiting from the data correlations existing among the ratings rather than eliminating them altogether. In particular, the proposed algorithm calculates the similarities between the items using the simplest method, namely, the Cosine Distance measurement method. It is worth noting that we do not directly use the exact value of the similarities, but rather rank the items according to their similarities. Then, a grey forecast (GF) model is constructed for rating prediction. This model has been successfully adopted for forecasting in several fields, such as finance [23], integrated circuit industry [24], the market for air travel [25], and underground pressure for working surface [26]. We compare the performances of the proposed algorithm with several other CF methods, including item based methods, slope one, and the state-of-the-art matrix factorization based method. Extensive experiments were conducted on two public data sets, namely, MovieLens and EachMovie. The results provide empirical evidence that the GF model can indeed cope effectively with data sparsity and correlation problems.

The remainder of this paper is organized as follows. Section 2 provides a detailed description of conventional user based CF (UCF) methods, item based CF (ICF) methods, the definition of existing problems, and our contributions. Section 3 presents the proposed GF model based algorithm in detail. Section 4 describes the experimental study, including experimental data sets, evaluation metrics, methodology, analysis of results, followed by a final section on conclusions and future work.

2. Related Work

The CF technique is one of the most successful recommender techniques [27]: it can be classified into memory based CF techniques [7] and [8] such as similarity or neighborhood based CF algorithms, model based CF techniques such as clustering CF algorithms [13] and [14], and hybrid CF techniques such as personality diagnosis [28], hybrid fuzzy-based personalized recommender system [1], and hybrid semantic recommendation system [29]. As a representative memory based CF technique, the similarity based method represents one of the most successful approaches for recommendation. They have been extensively deployed into commercial systems and been compre-

hensively studied [4] and [30]. This class of algorithms can be further divided into user and item based methods. The former is based on the basic assumption that people who share similar past preferences tend to agree in their future preferences. Hence, for the target user, the potential interest for an object is predicted according to the ratings from the users who are similar to the target user. As opposed to the user based method, an item based method recommends the items that are similar to what the active user has consumed before. In a typical memory based CF scenario, there is a set of n users $U = \{u_1, u_2, \ldots, u_n\}$,a set of m items $I = \{i_1, i_2, \ldots, i_m\}$,and the $n \times m$ user-item rating matrix. The ratings can either be explicit indications, such as an integer number from 1 to 5 (The integer number represents the rating a user gives to the item. Usually, number 1 means that the user does not like the item, while number 5 indicates the user is very satisfied with the item.), or implicit indications, such as purchases or click-throughs [31]. For example, implicit user behaviors (Table 1a) can be converted into a user-item rating matrix R (Table 1b). When the k-th user has purchased the l-th item, $R(k,l)$ for the k-th row and the l-th column of the matrix is assigned to rating 1. If the k-th user has not purchased the l-th item yet, a *null* value is assigned to$R(k,l)$. Therefore, the recommendation problem is reduced to predicting the *null* entries (*Lily* is the active user for whom we want to make recommendations for in Table 1b). Generally, the procedure for this type of CF method consists of two steps: similarity measurement and rating prediction.

2.1. Similarity measurement

The critical step in memory based CF algorithms is the similarity computation between users or items [32], [33], [34] and [35]. In UCF methods, the similarity $s(u_x, u_y)$, between the users u_x, and u_y is determined by comparing the items that both of them have rated. In ICF methods, the similarity $s(i_x, i_y)$ between the items i_x, and i_y is determined by the users who have rated both the items. There are various methods to compute the similarity between two users or items. The two most popular methods are Cosine Distance [5] and [36] and Pearson Correlation [5] and [36]. To define them, let I be the set of all items rated by both the users u_x, and u_y, and let U be the set of all users who have rated the items i_x, and i_y. For example, in Table 1b, the co-rated items of *Alice* (u_x) and *Lily* (u_y) are Bread and Milk (item set I); therefore, these two items' ratings given by individual user form a d-dimensional vector, where d is equal to the size of set I. In this case, d is

Table 1: An example of a user-item rating matrix

(a)

User	Purchase	Not Purchase
Alice	Milk, Bread, Cake	Beer
Lily	Milk, Bread	Cake, Beer
Lucy	Milk, Cake	Bread, Beer
Bob	Bread, Beer	Milk, Cake

(b)

	Bread	Beer	Cake	Milk
Alice	1		1	1
Lily	1		?	1
Lucy			1	1
Bob	1	1		

equal to two. Analogously, items Cake (i_x) and Milk (i_y) are rated by both *Alice* and *Lucy* (user set U) whose ratings on each item form a d-dimensional vector, where d is equal to the size of set U. d is equal to two in this case.

2.1.1. Cosine distance

For the Cosine Distance approach, the cosine of the angle between the two vectors represents the similarity between them. For UCF, the similarity between two users with Cosine Distance method can be calculated as follows:

$$s(u_x, u_y) = \frac{\sum_{i \in I} r_{u_x,i} \cdot r_{u_y,i}}{\sqrt{\sum_{i \in I} r_{u_x,i}^2} \sqrt{\sum_{i \in I} r_{u_y,i}^2}} \tag{1}$$

where $r_{u_x,i}$ and $r_{u_y,i}$ are the ratings of the users u_x and u_y on item i. I has the same definition in Section 2.1. Analogously, for ICF, the Cosine Distance between two items can be expressed as follows:

$$s(i_x, i_y) = \frac{\sum_{u \in U} r_{u,i_x} \cdot r_{u,i_y}}{\sqrt{\sum_{u \in U} r_{u,i_x}^2} \sqrt{\sum_{u \in U} r_{u,i_y}^2}} \tag{2}$$

where r_{u,i_x} and r_{u,i_y} are the ratings of the user u for items i_x and i_y. U is defined in Section 2.1.

2.1.2. Pearson correlation

We should note that, during the computation of similarity, it is necessary to eliminate the rating correlations (e.g., the average rating of the user) to improve the significance of similarity. The Pearson Correlation is one such method, which can be used to improve the accuracy of similarity computation to a certain extent. For UCF, the Pearson Correlation between two users can be expressed as

$$s(u_x, u_y) = \frac{\sum_{i \in I}(r_{u_x,i} - \overline{r}_{u_x})(r_{u_y,i} - \overline{r}_{u_y})}{\sqrt{\sum_{i \in I}(r_{u_x,i} - \overline{r}_{u_x})^2}\sqrt{\sum_{i \in I}(r_{u_y,i} - \overline{r}_{u_y})^2}} \tag{3}$$

where the terms $r_{u_x,i}$ and $r_{u_y,i}$ mean the same as in Eq. (1) and $\overline{r}_{u_x}$ and $\overline{r}_{u_y}$ are the average ratings of the users u_x and u_y, respectively. Similarly, for ICF, the Pearson Correlation between two items can be formulated as

$$s(i_x, i_y) = \frac{\sum_{u \in U}(r_{u,i_x} - \overline{r}_{i_x})(r_{u,i_y} - \overline{r}_{i_y})}{\sqrt{\sum_{u \in U}(r_{u,i_x} - \overline{r}_{i_x})^2}\sqrt{\sum_{u \in U}(r_{u,i_y} - \overline{r}_{i_y})^2}} \tag{4}$$

where the terms r_{u,i_x} and r_{u,i_y} mean the same as in Eq. (2) and $\overline{r}_{i_x}$ and $\overline{r}_{i_y}$ are the average ratings of all the users for items i_x and i_y, respectively.

2.2. Rating prediction

The rating prediction stage aims to predict the value that the active user will assign to the target item. The k Nearest Neighbors (KNN) method [37] is usually used for prediction by weighting the sum of the ratings that similar users give to the target item or the ratings of the active user on similar items depending on whether UCF or ICF is used.

2.2.1. UCF

The UCF algorithm is based on the basic assumption that people who share similar past preferences will be interested in similar items. The algorithm uses the following steps: first, the similarities between the users are computed using similarity measurement methods introduced in Section 2.1; then, the prediction for the active user is determined by taking the weighted average of all the ratings of the similar users for a certain item [37] according to the formula in Eq. (5); finally, the items with the highest predicted ratings will be recommended to the user.

$$p_{u_x,i} = \overline{r}_{u_x} + \frac{\sum_{u_y \in U(u_x)} s(u_x, u_y)(r_{u_y,i} - \overline{r}_{u_y})}{\sum_{u_y \in U(u_x)} |s(u_x, u_y)|} \tag{5}$$

where $U(u_x)$ denotes the set of users similar to the user u_x, and $p_{u_x,i}$ is the prediction for the user u_x on item i.

2.2.2. ICF

The ICF algorithm recommends items to users that are similar to the items that they have already consumed. Similarly, after calculating the similarities between the items, the unknown rating of user u on item i_x can be represented as an aggregation of user u on similar items:

$$p_{u,i_x} = \overline{r}_{i_x} + \frac{\sum_{i_y \in I(i_x)} s(i_x, i_y)(r_{u,i_y} - \overline{r}_{i_y})}{\sum_{i_y \in I(i_x)} |s(i_x, i_y)|} \tag{6}$$

where $I(i_x)$ denotes the set of similar items of item i_x. Further, p_{u,i_x} denotes the prediction of user u on item i_x.

2.3. Problem analysis

After using the co-rated entries as a vector to represent the object, the Cosine Distance method measures the similarity between two users or items by computing the cosine of the angle between them. Pearson Correlation takes the rating correlation into consideration to eliminate the influence of average ratings. Obviously, Pearson Correlation can be considered a variation of Cosine Distance. Taking UCF as an example, we select the items that both users have rated earlier and then use these ratings of each user for these items to construct a d-dimensional vector, namely, $(r_{u,i_1}, r_{u,i_2}, \ldots, r_{u,i_d})$, where d is the number of co-rated items. If we subtract each element by the average rating of user u, the vector will be converted to $(r_{u,i_1} - \overline{r}_u, r_{u,i_2} - \overline{r}_u, \ldots, r_{u,i_d} - \overline{r}_u)$. In this case, the Pearson Correlation is equivalent to Cosine Distance. With Pearson Correlation, the accuracy of similarity computation can be improved to a certain extent. However, it still suffers from many issues.

- **Data sparsity.** It is difficult to determine co-rated entries when the data is sparse. For instance, *Bob* and *Lucy* have not consumed the same items before (Table 1). Therefore, the similarity between such users cannot be computed by using the existing methods elaborated in Section 2.1. Furthermore, it might not be possible to obtain the similarities between the users or items in the same dimensionality. For example, *Alice* and *Lucy* both rated milk and cake (Table 1): the similarity between them is computed in a 2-dimensional space; however, *Bob* and *Lily* have only one co-rated entry, namely, bread (Table 1);

therefore, the similarity between them is computed in a 1-dimensional space. Therefore, the results seem biased.

- **Data correlation.** In this paper, data correlation corresponds to the general features hidden in the data because of the similar attributes among the users or items. For example, people who like *Tom Cruise* tend to give similar rating to movies the "Mission: Impossible III" and "Mission: Impossible IV". People of the same age will have similar preferences; therefore, their ratings for the same item will also be close. These correlations among the ratings yield a nonorthogonal vector space since the elements in different dimensions are not independent. Although the Pearson Correlation eliminates the influence of average ratings, such rating correlations still exist. Therefore, the similarities computed with these similarity measurement methods are not accurate (see Appendix A).

Because of these issues, in practice, the similarity between two users or items computed using Cosine Distance or Pearson Correlation is not accurate. Consequently, if we take a weighted average of the ratings using the similarities to directly generate the prediction, we may not obtain a good result. To overcome these shortcomings, Xie et al. [38] utilized the statistical values of the ratings related to the object to form a vector for the similarity measurement, which improved the prediction accuracy. Moreover, similarity transitivity [39] was proven to be an effective method for sparse data sets, which can effectively balance the tradeoff between the quality and quantity of similarity. In this paper, we relate these problems as being those of data sparsity and data correlation, and we use the GF model for rating prediction.

2.4. Contributions

The GF process for prediction can be described as follows. The Cosine Distance method is used to measure the similarity between two items. Then, an $m \times m$ similarity matrix is generated, where m is the number of items. Although the similarity computation is not accurate, as discussed in Section 2.3, the value can represent the degree of similarity. Therefore, in our algorithm, we do not use the exact value of similarity; rather, we only rank the items according to the similarity. Then, to generate the prediction of the active user u on item i, the k most similar items that have been rated by the active user on item i are selected. Finally, we use these items as the input

to build a GF model and predict the rating of the active user u on item i. If the user u does not rate k items, a fixed value will be used to complete the k ratings. Empirically, the fixed value can be the median value of the rating scale. For example, when the rating scale is 1-5, the number 3 is selected as the fixed value. The proposed method provides the following three main contributions:

- **Overcoming data sparsity.** Although the data is sparse and few items are rated by each user, only a few ratings are needed to construct the GF model for our algorithm and the experimental results show that the prediction accuracy is still high even when k is equal to 5. Obviously, the proposed algorithm can efficiently address the data sparsity problem.

- **Benefiting from data correlation.** The stronger the data correlation, the more accurate is the GF model. In the experiments, when the user's average rating or overall average rating is eliminated, the GF model performs considerably worse. In other words, the proposed algorithm can effectively benefit from the data correlations rather than eliminating them.

- **Obtaining accurate predictions.** We test our algorithm on two public data sets, namely, MovieLens[4] and EachMovie[5] . The experimental results when compared with traditional ICF (using Cosine Distance for similarity measurement) reveal that our algorithm yields better performance with respect to the metrics of Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). In particular, with regard to the MovieLens data set, the accuracy has been improved by over 20% in terms of the MAE. Moreover, it achieves comparative or even better performance with regard to accuracy and F-measure when compared to the state-of-the-art matrix factorization based method.

3. Proposed algorithm

Memory based CF algorithms aggregate ratings of similar users for a target item or ratings of the active user for similar items to generate prediction.

[4]www.grouplens.org/

[5]www.kumpf.org/eachtoeach/eachmovie.html

Consequently, the prediction accuracy depends mainly on the similarity computation. However, when the data is sparse and exhibits strong correlations, the existing similarity measurement methods cannot obtain accurate similarities between the users or items. In other words, the similarities are not very accurate. Hence, we cannot use the similarities to directly obtain reliable predictions. In this paper, the GF model is used for rating prediction. It involves two steps: rating preprocessing and rating prediction.

3.1. Rating preprocessing

Since the similarities between the items computed by using existing similarity measurement methods have significance, we use them to preprocess the ratings. First, for simplicity, the Cosine Distance method is utilized to compute the similarity between two items. Then, an $m \times m$ similarity matrix is generated, where m is the number of items. If we want to predict the unrated entry of the user u on item i in the rating matrix, the k most similar items to the item i that have been rated by the user u are selected. Note that when the user u does not rate k items, the fixed value with the lowest similarity will be used to complete the k ratings. Finally, the k ratings are sorted according to their incremental similarities to the item i to produce a rating sequence. In the next step, the proposed algorithm inputs the rating sequence to the GF model and forecasts the rating that the user u will give to item i. For instance, a section of a rating matrix with ratings in the 1-5 scale is shown in Table 2. We want to predict the rating of user u_3 on item i_1. According to Cosine Distance, the similarities between item i_1 with the other items are shown in Table 3. The items rated by user u_3 can be sorted as their similarities with item i_1 increase: i_5, i_7, i_3, i_9, i_4. If we set $k = 3$, the last three items (namely i_3, i_9, and i_4) will be selected, since they have been rated by the user u_3 and have higher similarities to item i_1. Then, the rating sequence is (4, 3, 5). Furthermore, if we set $k = 7$, since the number of items rated by the user u_3 is less than 7, all the ratings of the items rated by the user u_3 will be selected and the median value (number 3) will be used to complete seven ratings with the lowest similarity. Then, the rating sequence is (3, 3, 5, 4, 4, 3, 5), where the first two numbers are replaced with the number 3 in the rating sequence. Note that when two or more items have the same similarity to the target item, the order of their ratings is random. For example, item i_5 is sorted in front of item i_7 although they have the same similarity to item i_1.

The rating sequence has several special attributes:

Table 2: Fragment of a rating matrix

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}
u_1	4	4		5		5		4	4	5
u_2	3	4	2			4		3		4
u_3	?		4	5	5		4		3	
u_4	1		3	2				3	4	

Table 3: Similarities between item i_1 with other nine items

	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}
i_1	0.989	0.789	0.991	0	0.999	0	0.942	0.857	0.999

- The similarities among the items are very high, since they are the k most similar items to the target item. Hence, these ratings are highly correlated. Intuitively, the user tends to allocate closer ratings to similar items.
- The ratings of the sequence are incrementally sorted by the items' similarities to the target item. Consequently, the ratings with higher similarities will contribute more to the final prediction, which makes the GF model more effective.

Due to these valid attributes, the rating prediction stage can be more effective.

3.2. Rating prediction

Grey theory was originally developed by Deng in 1982 [40]. It mainly focuses on model uncertainty and information insufficiency when analyzing and understanding systems via research on conditional analysis, prediction, and decision making. A recommender system can be considered as a grey system; further, with our algorithm, the GF model is used to yield the rating prediction. The GF model utilizes accumulated generation operations to build differential equations, which benefit from the data correlations. Meanwhile, it has another significant characteristic wherein it requires less data so it can overcome the data sparsity problem. The rating sequence generated in the rating preprocessing stage is the only input required for model

construction and subsequent forecasting. These are the reasons why we have selected the GF model for rating prediction: the GM(1,1) method is adopted in this paper. GM(1,1) indicates a single variable first-order GF model. The general procedure followed in a GF model is derived as follows [41]:

Step 1: Assume the original rating sequence to be $r_u^{(0)}$:

$$r_u^{(0)} = \{r_u^{(0)}(l)\}, l = 1, 2, \ldots, k \tag{7}$$

where $r_u^{(0)}(l)$ represents the original rating of the user u for the l-th value of the rating sequence. Further, k is the number of neighbors or the length of the rating sequence, and it must be equal to or greater than 4.

Step 2: A new sequence $r_u^{(1)}$ is produced by the Accumulated Generating Operation (AGO). The GF model is based on the generating sequence rather than the original one:

$$r_u^{(1)} = \{r_u^{(1)}(l)\}, l = 1, 2, \ldots, k \tag{8}$$

where $r_u^{(1)}(l) = \sum_{j=1}^{l} r_u^{(0)}(j)$, l = 1, 2,. . . , k.

This step is vital, since the randomness of the data is somehow smoothed and it further enhances the tendency of the new sequence due to the correlation between the values of the original sequence. For example, $r_u^{(0)} = \{3, 4, 3, 4, 5\}$ is a user's original rating sequence. Obviously, the sequence does not have a clear regularity. If AGO is applied to this sequence, $r_u^{(1)} = \{3, 7, 10, 14, 19\}$ is obtained which has a clear growing tendency.

Step 3: Based on the property, that the relation between the grey derivative and the background grey number is approximate linear regression, of smooth discrete function, a grey differential model called GM(1,1) can be defined as follows [41]:

$$d_u^{(1)}(l) + az_u^{(1)}(l) = b, l = 2, 3, \ldots, k \tag{9}$$

where a, b are the coefficients, especially in the terms of Grey System theory, a is the grey development coefficient and b is the grey input. They are estimated in Step 5. $d_u^{(1)}(l) = r_u^{(1)}(l)$ - $r_u^{(1)}(l-1) = r_u^{(0)}(l)$ is the grey derivative, therefore, the grey differential model is always denoted as $r_u^{(0)}(l) + az_u^{(1)}(l) = b$. $z_u^{(1)}(l)$ is the grey background number, which is the weighted sum of the values of the consecutive neighbors of the sequence $r_u^{(1)}$. More specially, $z_u^{(1)}(l) = \alpha r_u^{(1)}(l-1) + (1 - \alpha) r_u^{(1)}(l)$. Further, $\alpha(0<\alpha<1)$ is the weight. Here,

α is selected so as to yield the smallest prediction error rate. When α<0.5, the values in the sequence with higher rankings will make more contribution to the differential equation, and therefore, the final result. In fact, extensive experiments with different values of α have found that when α<0.5, the GF model based method performed well. This convinces us to incrementally sort the ratings with items' similarities to the target item. The more similar the items are to the target item, the more important are their ratings to the final prediction. Therefore, in our experiments, we set α= 1/3.

Step 4: If the discrete space is mapped into the continuous one (discrete variable l to the continuous variable t), the grey differential model can be whitened as:

$$dr_u^{(1)}(t)/dt + ar_u^{(1)}(t) = b \tag{10}$$

where $r_u^{(1)}$ is converted to the continuous function, $r_u^{(1)} = r_u^{(1)}(t)$. The grey derivative $d_u^{(1)}(l)$ is mapped to $dr_u^{(1)}(t)/dt$, and $z_u^{(1)}(l)$ to $r_u^{(1)}(t)$, since $z_u^{(1)}(t)$ $= r_u^{(1)}(t) = r_u^{(1)}(t-1)$ in the continuous space.

Step 5: From Step 4, the solution $\hat{r}_u^{(1)}(t)$ is:

$$\hat{r}_u^{(1)}(t) = (r_u^{(0)}(1) - b/a)e^{-a(t-1)} + b/a \tag{11}$$

where a, b have the same definitions in Step 3. Let $v = (a, b)^T$,

$$B = \begin{bmatrix} -z_u^{(1)}(2) & 1 \\ -z_u^{(1)}(3) & 1 \\ \vdots & \vdots \\ -z_u^{(1)}(k) & 1 \end{bmatrix}, Y = \begin{bmatrix} d_u^{(0)}(2) \\ d_u^{(0)}(3) \\ \vdots \\ d_u^{(0)}(k) \end{bmatrix} = \begin{bmatrix} r_u^{(0)}(2) \\ r_u^{(0)}(3) \\ \vdots \\ r_u^{(0)}(k) \end{bmatrix},$$

then, the GM(1,1) defined in Eq. (9) is equivalent to $Y = Bv$. By minimzing $J(\hat{v}) = (Y - B\hat{v})^T(Y - B\hat{v})$, the least squares estimation of parameters are: $\hat{v} = (a, b)^T = (B^TB)^{-1}B^TY$.

When we set $t = l$, $\hat{r}_u^{(1)}(l)$ is the estimation of the l-th value of the AGO data.

Step 6: Inverse Accumulated Generation Operation (IAGO). Because the GF model is formulated using the data of AGO rather than the original data, we should use the IAGO to convert the AGO data to an actual rating prediction:

$$\hat{r}_u^{(0)}(l) = \hat{r}_u^{(1)}(l) - \hat{r}_u^{(1)}(l-1) = (r_u^{(0)}(1) - b/a)e^{-a(l-1)}(1 - e^a) \tag{12}$$

When we set $l = k + 1$, the rating prediction $p_{u,i}$ of the user u on item i can be represented by $\hat{r}_u^{(0)}(k+1) = (r_u^{(0)}(1) - b/a)e^{-ak}(1 - e^a)$.

Obviously, during the estimation of the parameters a and b in Step 5, a matrix inverse operation is needed. Hence, we cannot always forecast the ratings using the GF model. In these cases, the average of the k ratings is used as the rating prediction of the active user for the target item.

4. Experimental results

In this section, we present the results of the experimental evaluation of our novel algorithm. We describe the data sets used; the experimental methodology and performance improvement are compared with several CF methods, particularly the state-of-the-art matrix factorization based method.

4.1. Data sets

We conducted extensive experiments on two standard data sets: MovieLens [42] and EachMovie [43]. Both these data sets are publicly available data sets regarding movie ratings. MovieLens rating sets are collected by GroupLens research from the MovieLens Web site (http://movielens.umn.edu). Three different sizes of data sets are available. In this paper, the MovieLens 1M data set was used, which consists of 1 million ratings (in the scale of 1-5 stars) from 6,040 users on 3,952 movies. We also implemented the experiment for another data set, namely, EachMovie, which is collected by the DEC Systems Research Center. It consists of 2,811,983 numerical ratings from 74,424 users on 1,648 different movies (films and videos). Since the ratings are linearly mapped to the interval [0, 1], for convenience, we multiplied the ratings by 5 and deleted the records in which the ratings were zero. Finally, 2,464,792 ratings remained, which were in 1-5 rating scale. Table 4 summarizes the statistical properties of both these data sets. The sparsity level of the data set can be computed as follows [4]:

$$sparsity\ level = \frac{\#total\ entries - \#rating\ entries}{\#total\ entries} \tag{13}$$

Table 4: Statistical properties of MovieLens and EachMovie

	Movielens	EachMovie
Users	6,040	74,424
Items	3,952	1,648
Ratings	1,000,000	2,464,792
Ratings Per User	165	33
Ratings Per Item	253	1,495
Sparsity Level	95.81%	97.99%

4.2. Metrics and methodology

To evaluate the performance of a recommender algorithm, the data set needs to be partitioned into two sections: training set and testing set. The former is dedicated to the model's construction, while the latter is used for testing the model. Here, we set x as the training ratio, which is the proportion of the training set in the data set. For example, when x is equal to 80%, the training set comprises 80% of the data set, while the remaining 20% is the testing set. In this paper, two classes of metrics are used to evaluate the algorithmic performance: error metrics and classification metrics. Error metrics evaluate the error between the actual rating and the predicted value. MAE [44] and RMSE [45] are the most frequently used error metrics. Therefore, we use these two metrics to evaluate the accuracy of rating prediction. MAE and RMSE can be defined as

$$MAE = \frac{\sum_{(u,i)\in T} |r_{u,i} - p_{u,i}|}{|T|} \tag{14}$$

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in T} (r_{u,i} - p_{u,i})^2}{|T|}} \tag{15}$$

where T is the set of all the pairs $(u,\ i)$ in the testing set.

Generally, we are not interested in the precise prediction of ratings; rather, we are concerned about suggesting a short list of interesting items to the user [19] and [37].

Therefore, the information retrieval classification metrics are used to measure the recommendation accuracy more precisely. When using classification metrics, four different kinds of recommendations are distinguished. If the

algorithm recommends an interesting item to the user, we have a true positive (TP); however, if an uninteresting item is recommended, we have a false positive (FP). If the algorithm does not recommend an uninteresting item to the user, we have a true negative (TN); however, if an interesting item is missed, we have a false negative (FN). We set the number 3 as the threshold to determine whether the user is interested in the item or not. In particular, if the actual and predicted ratings are not less than 3, there is a TP; if the actual and predicted ratings are less than 3, there is a TN; if the actual rating is not less than 3 while the predicted rating is less than 3, there is a FN; and if the actual rating is less than 3 while the predicted rating is not less than 3, there is a FP.

The most popular classification metrics [4] and [19] are precision and recall:

$$precision = \frac{TP}{TP + FP} \tag{16}$$

$$recall = \frac{TP}{TP + FN} \tag{17}$$

Precision measures the percentage of interesting items recommended to the users with respect to the total number of recommended items, while recall measures the percentage of interesting items recommended to the users with respect to the total number of interesting items. Often, there is an inverse relationship between precision and recall. To better understand the recommendation quality, a combination between precision and recall is used, which is called F-measure [4] and [19]:

$$F\text{-}measure = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{18}$$

We compute the overall values of precision, recall, and F-measure.

To evaluate the performance of our proposed algorithm, we compare its performance with those of several other methods:

- **ICF** [8]**:** This is a well-known memory based CF approach, which calculates the similarity between two items using the Cosine Distance measurement. Due to its easy implementation and interpretability, it is one of the most popular recommender methods.

- **SlopeOne** [46]**:** This also belongs to a class of CF methods. Its prediction accuracy is relatively high, but this method requires high storage capacities.

- **SVD+:** This is a basic matrix factorization based method [20], which is closely related to SVD, and it uses the stochastic gradient descent for minimizing the regularized squared error on a set of known ratings.

- **SVD++:** This is another matrix factorization based method [20], which takes biases into account. These biases are the observed variations in the rating values induced by the effects associated with either users or items independent of any interactions. Examples of such effects are the overall average rating or the users' and items' average deviations. Therefore, SVD++ is an improved version of SVD+. It has been shown that this method yields accuracy superior than conventional memory based CF techniques. However, multiple parameters need to be estimated, which is time-consuming.

- **Cosine based GF:** The proposed method is based on the GF model and uses the Cosine Distance method to determine similarity between items.

4.3. Experimental results and analysis

4.3.1. Variations in training ratio

We increase the training ratio x from 10% to 90% for a variation of 10% in the MovieLens data set. The MAE and RMSE values for the ICF, cosine based GF, and correlation based GF methods are shown in Fig. 1 and Fig. 2; here, the correlation based GF method uses Pearson Correlation for item similarity measurement. We select the 100 nearest neighbors for the ICF method and 5 nearest neighbors for the cosine and correlation based GF methods having the following abbreviations in the figures: Item based CF (100), Cosine based GF (5), and Correlation based GF (5), respectively. In the subsequent experiments, the numbers given in the brackets have the same meaning.

Fig. 1 and Fig. 2 show that the cosine and correlation based GF methods perform much better than the ICF method with regard to error metrics, particularly when the training ratio is set at 80%. Moreover, although Pearson Correlation yields more accurate similarities than Cosine Distance, the results obtained from the correlation based GF method are slightly different

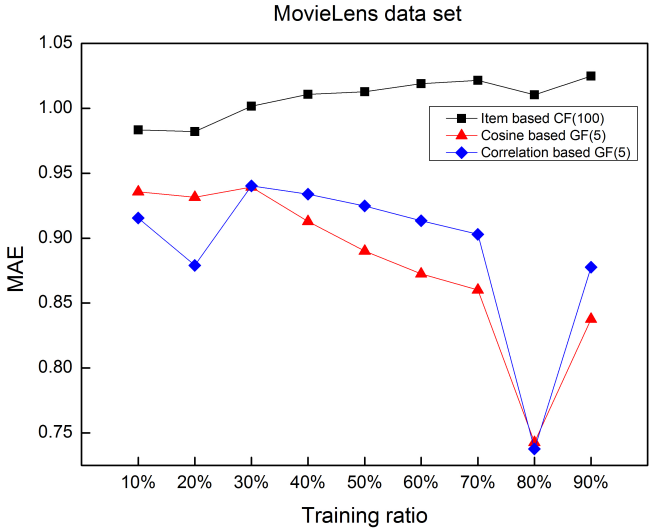

Figure 1: MAE values for different training ratios using MovieLens data set.

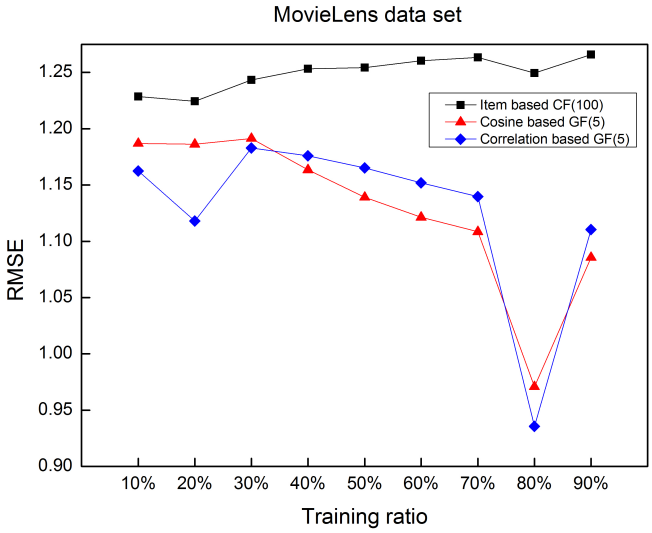

Figure 2: RMSE values for different training ratios using MovieLens data set.

than those obtained from the cosine based GF method. This indicates that the similarities derived from these two similarity measurements are valuable for effective item ranking. Therefore, GF model based methods can efficiently perform independent of the similarity accuracy. A similar conclusion is derived from the EachMovie data set; the experimental results are not shown here.

4.3.2. Influence of the number of neighbors

To determine the optimal number of neighbors for GF model based methods, the training ratio x is set at 80% and the number of neighbors k is adopted as 5, 10, 15, 30, 40, and 60. The cosine based GF method is compared with the ICF method in terms of the MAE and RMSE metrics using two data sets. The results are shown in Fig. 3, Fig. 4, Fig. 5, and Fig. 6.

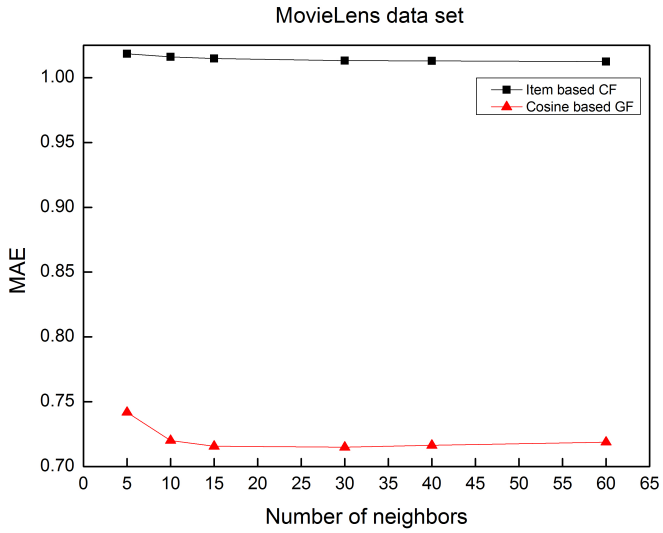

Figure 3: MAE values for different number of neighbors using MovieLens data set.

The results shown in these four figures reveal that the cosine based GF method outperforms the ICF method with regard to prediction error. Moreover, as the value of k increases, the performance of the cosine based GF method improves steadily. The optimal performance is achieved when k is approximately equal to 30. However, the prediction accuracy of the ICF method varies smoothly as k increases.

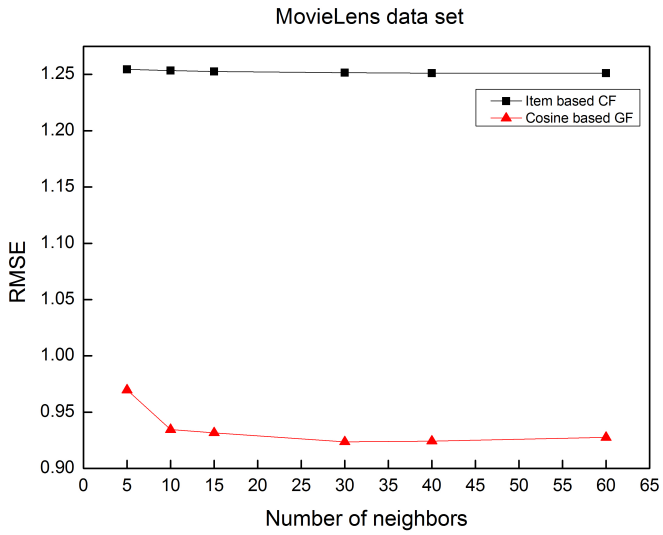

Figure 4: RMSE values for different number of neighbors using MovieLens data set.

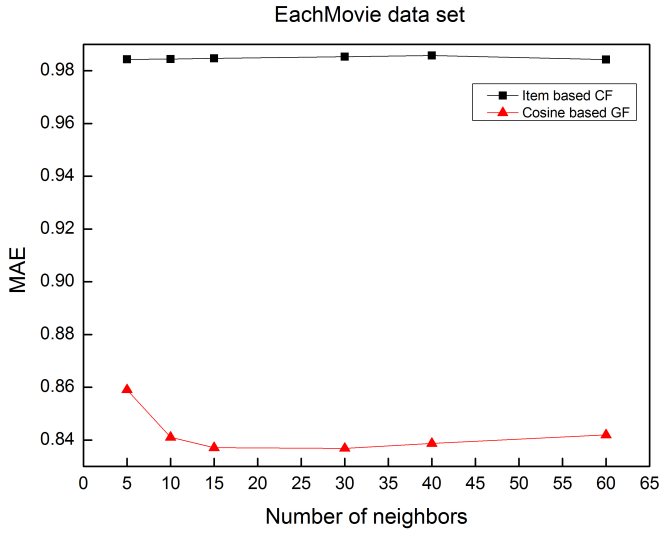

Figure 5: MAE values for different number of neighbors using EachMovie data set.

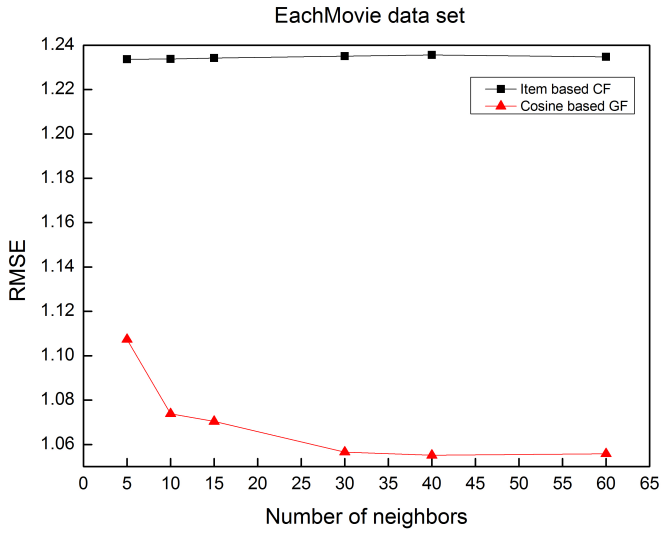

Figure 6: RMSE values for different number of neighbors using EachMovie data set.

4.3.3. Comparison with state-of-the-art methods in terms of error metrics

To effectively evaluate the performances of GF based methods, it is necessary to compare them with other CF methods, particularly the state-of-the-art methods. In the experiments, slope one and two variants of the matrix factorization based method are tested, except the ICF method. We set 15 as the number of neighbors for the cosine based GF method and 100 for the ICF method. These two methods use Cosine Distance for item similarity measurements. Moreover, the training ratios of the two data sets are set at 80%. The obtained MAE and RMSE values are shown in Fig. 7 and Fig. 8.

Fig. 7 and Fig. 8 show that the SVD++ method outperforms the other four methods in terms of MAE and RMSE, whereas SVD+ comes second. Since the EachMovie data set is sparser than the MovieLens data set, cosine based GF, SVD+, and SVD++ perform better when using the latter, while contradictory results are obtained when using it with the slope one and ICF methods. Although the cosine based GF yields poor performance when compared to matrix factorization based methods, an improvement of over 20% in terms of the MAE and RMSE values is observed when compared with the traditional ICF method.

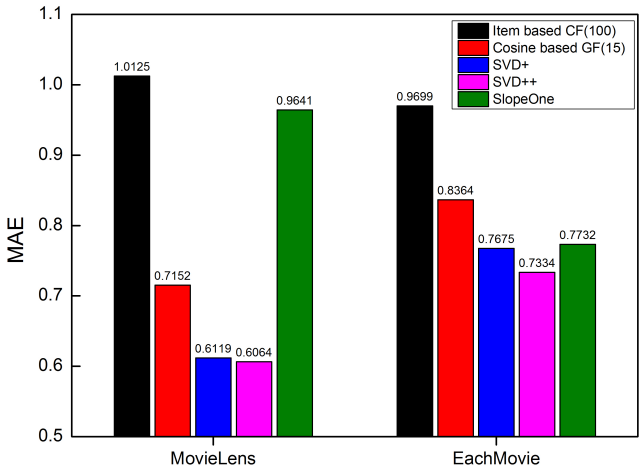

Figure 7: MAE value comparisons of five methods.

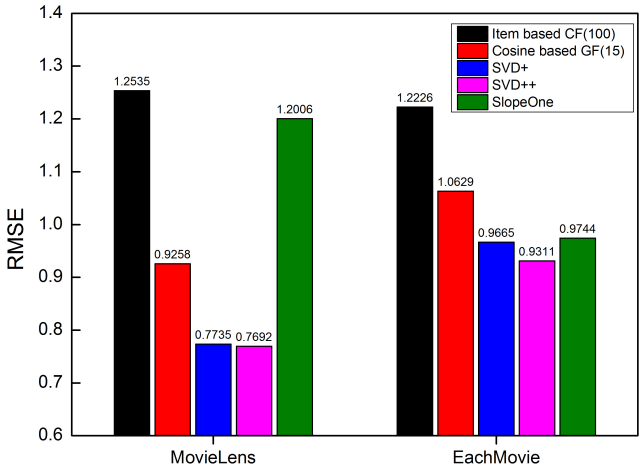

Figure 8: RMSE value comparisons of five methods.

4.3.4. Comparison with state-of-the-art methods in terms of classification metrics

In real-world recommender systems, we are interested in suggesting interesting items to users rather than accurately predicting ratings. Therefore, the precision, recall, and F-measure metrics are used to evaluate this capability. These metrics are defined in Section 4.2, and the settings of the evaluation methods are the same as the ones in Section 4.3.3. The experimental results with classification metrics are shown in Fig. 9 and Fig. 10.

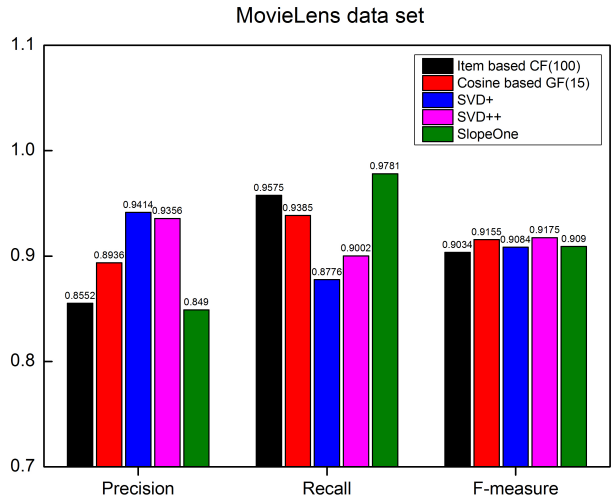

Figure 9: Classification metrics comparisons of five methods using MovieLens data set.

Fig. 9 and Fig. 10 show that both algorithmic precision and recall cannot be simultaneously high. For example, when using the MovieLens data set, SVD+ achieves the highest precision, but its recall is the lowest. Because of their accurate but conservative rating predictions, the prediction ratings of both the matrix factorization based methods are generally less than the actual ones. Therefore, the number of FPs is low, but the number of FNs is high. Consequently, they yield higher precision but lower recall. Moreover, with regard to the ICF method, if the rating of a user on an item in the testing set cannot be predicted, 3 is taken to be the default value for such a prediction. In other words, no matter if the actual ratings are more or less than 3, the predictions are set at 3. Therefore, the number of FPs

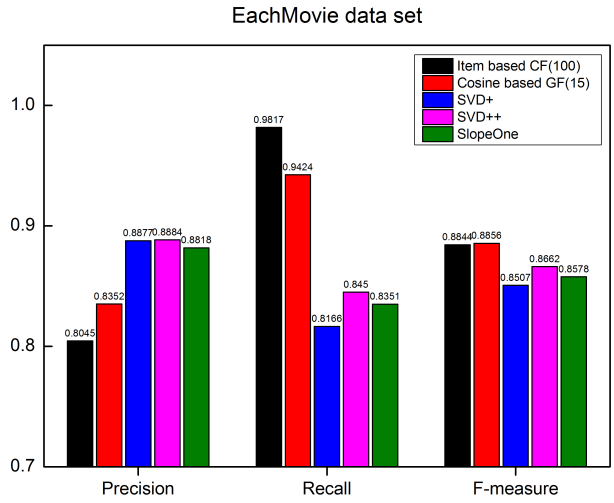

Figure 10: Classification metrics comparisons of five methods using EachMovie data set.

is high, but the number of FNs is low; this results in lower precision but higher recall. In fact, due to the sparsity of the data set, a large number of ratings cannot be predicted by the ICF method. Therefore, its precision, recall, and F-measure values seem insignificant. The cosine based GF method yields comparative performance in terms of both precision and recall. In particular, its F-measure value is considerably higher than the ones yielded by the matrix factorization based methods when using the EachMovie data set, while their F-measure values are similar when using the MovieLens data set. Overall, the GF model based methods can significantly outperform conventional ICF methods in terms of error metrics and achieve comparative or even better performance than the state-of-the-art methods in terms of classification metrics, which are more important in real-world systems.

4.3.5. Influence of correlations

Typically, some aggressive users tend to give higher ratings, while conservative users like to give lower ratings. This difference lies in the user average ratings. Intuitively, different systems may have different overall average ratings. We call these tendencies as correlations. To verify whether these correlations are useful for the GF model construction, we eliminate the user average rating and the overall average rating from the rating sequence

(see Section 3.1) before building the model. Consequently, two variants of the GF model based methods are generated, and they have the following abbreviations: cosine based GF-UA and cosine based GF-OA. A comparison of the results obtained using these two methods with the cosine based GF method in terms of the MAE and RMSE values are shown in Fig. 11, Fig. 12, Fig. 13, and Fig. 14.

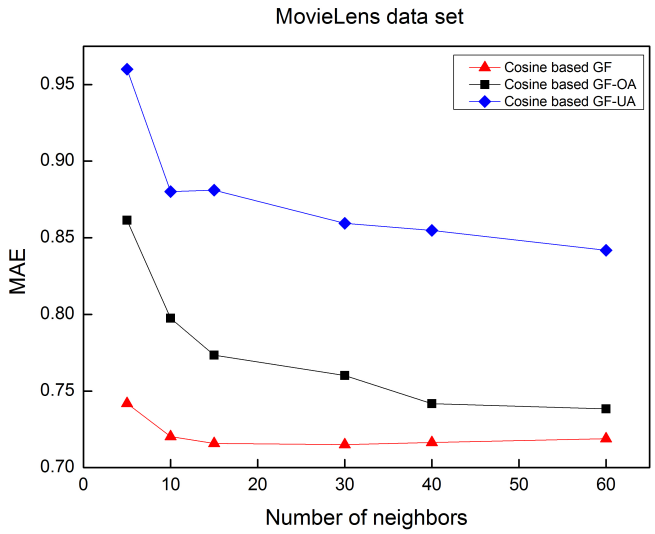

Figure 11: MAE value comparisons of three methods using MovieLens data set.

The experimental results show that the performances sharply deteriorate when the correlations are removed. Therefore, this suggests that the GF model based methods can benefit from data correlations. Moreover, it is also evident that the correlation of the user average rating is more significant than that of the overall average rating, since the performance decreases more drastically when the user average rating is eliminated. It can be supposed that the GF model based methods may be more effective when strong correlation exists among the data.

4.3.6. Effect of median value

As the median value is used to complete k ratings when the user does not rate enough items, it is necessary to compare with the traditional item based CF taking the same preprocessing. Item based CF++ is such a variant of

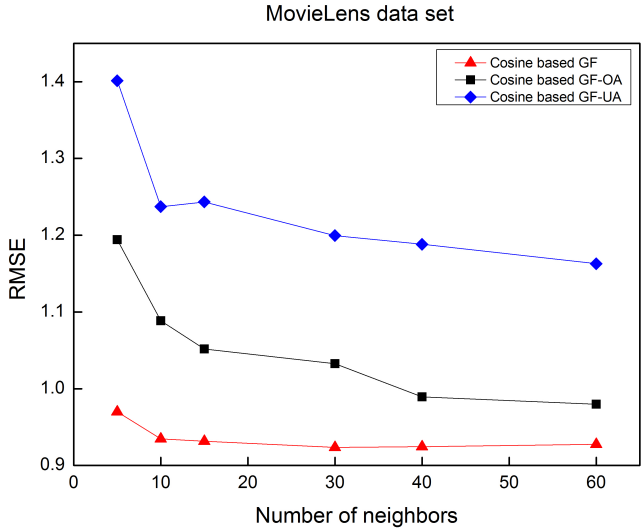

Figure 12: RMSE value comparisons of three methods using MovieLens data set.

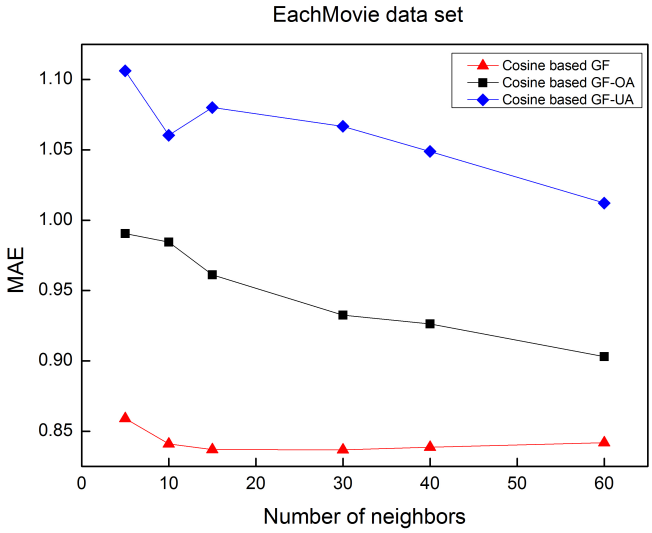

Figure 13: MAE value comparisons of three methods using EachMovie data set.

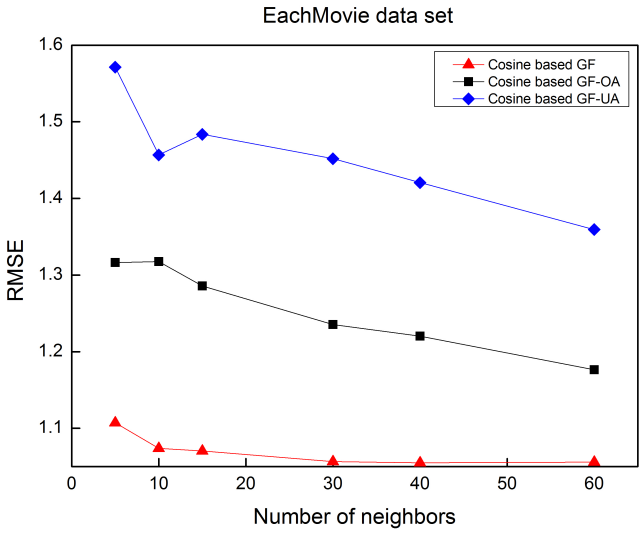

Figure 14: RMSE value comparisons of three methods using EachMovie data set.

item based CF method, which uses the median value as the default rating when the user have not given a rating to the item which is one of the k nearest neighbors of the target item. Therefore, there are always k ratings for the weighted sum in the prediction process. The performance comparison is shown in Table 5, where k is set to be 15 for both.

Table 5: Comparison with the improved item based CF

Metric	MovieLens		EachMovie	
	Item based CF++	Cosine based GF	Item based CF++	Cosine based GF
MAE	0.7956	**0.7152**	0.8796	**0.8364**
RMSE	0.9922	**0.9258**	1.0777	**1.0629**
Precision	**0.8986**	0.8936	**0.8451**	0.8352
Recall	0.8451	**0.9385**	0.8374	**0.9424**
F-measure	0.871	**0.9155**	0.8412	**0.8856**

The GF model based method consistently outperforms item based CF++ in terms of error metrics and classification metrics except precision. However,

their precisions are about the same. Because item based CF++ alleviates data sparsity using median value as the default, it achieves better performance than traditional item based CF. However, its predictions are around the median value, which results in mediocre performance in classification metrics. Actually, the ratings for most users (Ratings Per User in Table 4) are enough for GF model based methods when k equals to 15, therefore, only small part of testing pairs need to use the median value to complete the rating sequences. Furthermore, since the median values have the lowest similarity, they contribute less to the final predictions as described in Section 3.2. Consequently, these indicate that GF model based methods achieve outstanding performance independent of the median value.

4.3.7. Time complexity analysis

GF model based methods have the same time complexity with that of ICF method in constructing item similarity matrix, which is $O(m^2)$, where m is the number of items. In the procedure of prediction, k_{CF} multiplications are needed for ICF, where k_{CF} is the number of nearest neighbors. Intuitively, when the input rating sequences are the same, GF model based methods will produce the same prediction. Therefore, if we assume s (All ratings are integers from 1 to s) as the rating scale and k_{GF} as the number of neighbors, there are only $s^{K_{GF}}$ combinations for the rating sequence. In our experiment design, the $s^{K_{GF}}$ unique predictions are generated offline and stored in memory. Each rating sequence is mapped into a key which is used to get the prediction from memory. Then, the time complexity decreases to k_{GF} with binary search. As illustrated in Figs. 3-6, GF model based methods can achieve high performance even when k_{GF} is small, while k_{CF} is much bigger in ICF. For example, k_{CF} =100, k_{GF}=5, s=5, then, $k_{GF} \log s \ll K_{CF}$, while the storage consumption is less than 1M. Therefore, we can consume little storage space to achieve better time efficiency. The time consumption of model building in SVD is proportional to the size of training set which is much less than the one of item similarity matrix constructing when the data set is sparse. After the model is constructed, the time consumption of prediction is k_{SVD} for SVD, where k_{SVD} is the number of factors. In general, k_{SVD} has the same scale with k_{CF} and increases as the sparsity of data set increases. However, the model parameters need to be re-estimated when new ratings are injected. In practice, item similarity is stable which means we do not need to frequently update the similarity matrix. Therefore we can reduce the frequency of similarity calculation to make GF model based meth-

ods more efficient. Moreover, its prediction efficiency can be greatly improved by using little storage space to store the possible predictions beforehand.

5. Conclusions and future work

Since the existing similarity measurement methods, such as Cosine Distance and Pearson Correlation, cannot accurately compute the similarities between users or items when the data is sparse or when there are strong data correlations, UCF and ICF methods do not perform well when it comes to prediction accuracy. In this paper, we used the GF model for rating prediction in recommender systems and conducted a series of experiments on two public movie data sets, namely, MovieLens and EachMovie. The experimental results demonstrated that the GF model based methods can overcome the problem of data sparsity, benefit from data correlations, and outperform conventional memory based CF (ICF) methods. In particular, even when only 15 nearest neighbors are adopted, the GF model based method still reduces the prediction error by over 20% in terms of MAE and RMSE when compared to the ICF method with 100 nearest neighbors. Although the state-of-the-art methods, such as the matrix factorization based methods, perform better than the GF model based methods in terms of error metrics, the latter presents comparative, or sometimes even better, performance in terms of classification metrics—which are more valuable for algorithmic estimation in real-world systems as compared to error metrics.

Improving the accuracy of recommendations has been extensively investigated. In this paper, we adopt a mature forecasting model used in economics—called the GF model—to gain high-accuracy recommendations. This fosters a new era in prediction wherein advanced technologies in other fields can employ novel recommender algorithms, and the various problems in recommender systems, such as data sparsity and data correlation, can be overcome. As an effective rating prediction method, the GF model has room for improvement. In our future work, we consider the case when the user does not rate a sufficient number of k items, the average of the user's ratings on all the items is used instead of a fixed value to complete the k ratings. Moreover, we will also try to employ GF model based methods for larger data sets, and it is our target to implement the proposed method into recommender systems for real world applications. Actually, we are planning to apply it for video recommendation on iqiyi.com website.

6. Acknowledgements

The authors would like to thank Prof. Jun Li of Tsinghua University for his careful guidance about the paper structure and the paper written.

References

[1] Z. Zhang, H. Lin, K. Liu, D. Wu, G. Zhang, J. Lu, A hybrid fuzzy-based personalized recommender system for telecom products/services, Information Sciences 235 (2013) 117–129.

[2] Q. Shambour, J. Lu, A trust-semantic fusion-based recommendation approach for e-business applications, Decision Support Systems 54 (2012) 768–780.

[3] Q. Shambour, J. Lu, A hybrid trust-enhanced collaborative filtering recommendation approach for personalized government-to-business e-services, International Journal of Intelligent Systems 26 (2011) 814–843.

[4] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Analysis of recommendation algorithms for e-commerce, in: Proceedings of the 2nd ACM Conference on Electronic Commerce, 2000, pp. 158–167.

[5] G. Adomavicius, A. Tuzhilin, 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 (2005) 734–749.

[6] J.-M. Yang, K. F. Li, Recommendation based on rational inferences in collaborative filtering, Knowledge-Based Systems 22 (2009) 105–114.

[7] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, Grouplens: an open architecture for collaborative filtering of netnews, in: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, 1994, pp. 175–186.

[8] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web, 2001, pp. 285–295.

[9] H. Ma, T. C. Zhou, M. R. Lyu, I. King, Improving recommender systems by incorporating social contextual information, ACM Transaction On Information Systems 29 (2011) 9:1–9:23.

[10] F. Cacheda, V. Carneiro, D. Fernández, V. Formoso, Comparison of collaborative filtering algorithms: limitations of current techniques and proposals for scalable, ACM Transaction on Web 5 (2011) 2:1–2:33.

[11] N. N. Liu, X. Meng, C. Liu, Q. Yang, Wisdom of the better few: cold start recommendation via representative based rating elicitation, in: Proceedings of the Fifth ACM Conference on Recommender Systems, 2011, pp. 37–44.

[12] R. Hu, P. Pu, Enhancing collaborative filtering systems with personality information, in: Proceedings of the Fifth ACM Conference on Recommender Systems, 2011, pp. 197–204.

[13] L. H. Ungar, D. P. Foster, Clustering methods for collaborative filtering, in: Proceedings of the AAAI Workshop on Recommendation Systems, 1, 1998.

[14] S. H. S. Chee, J. Han, K. Wang, Rectree: an efficient collaborative filtering method, in: Proceedings of the 3rd International Conference on Data Warehousing and Knowledge Discovery, 2001, pp. 141–151.

[15] J. S. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in: Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence, 1998, pp. 43–52.

[16] X. Su, T. M. Khoshgoftaar, Collaborative filtering for multi-class data using belief nets algorithms, in: Proceedings of the International Conference on Tools with Artificial Intelligence, 2006, pp. 497–504.

[17] G. Shani, R. I. Brafman, D. Heckerman, An mdp-based recommender system, in: Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence, 2002, pp. 453–460.

[18] T. Hofmann, Latent semantic models for collaborative filtering, ACM Transactions on Information Systems 22 (2004) 89–115.

[19] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Application of dimensionality reduction in recommender systems-a case study, in: Proceedings of the SIGKDD Workshop on Web Mining and Web Usage Analysis, 2000.

[20] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37.

[21] X. Luo, Y. Xia, Q. Zhu, Incremental collaborative filtering recommender based on regularized matrix factorization, Knowledge-Based Systems 27 (2012) 271–280.

[22] X. Luo, Y. Xia, Q. Zhu, Applying the learning rate adaptation to the matrix factorization based collaborative filtering, Knowledge-Based Systems 37 (2012) 154–164.

[23] L.-C. Hsu, C.-H. Wang, Grey forecasting the financial ratios, Journal of Grey System 14 (2002) 399–408. (In Chinese).

[24] L.-C. Hsu, Applying the grey prediction model to the global integrated circuit industry, Technological Forecasting and Social Change 70 (2003) 563–574.

[25] C.-I. Hsu, Y.-H. Wen, Improved grey prediction models for the trans-pacific air passenger market, Transportation planning and Technology 22 (1998) 87–107.

[26] D. Ma, Q. Zhang, Y. Peng, S. Liu, A particle swarm optimization based grey forecast model of underground pressure for working surface, Electronic Journal of Geotechnical Engineering 16 (2011) 811–830.

[27] X. Su, T. M. Khoshgoftaar, A survey of collaborative filtering techniques, Advances in artificial intelligence 2009 (2009) 4.

[28] D. Y. Pavlov, D. M. Pennock, A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains, in: Advances in Neural Information Processing Systems, volume 15, 2002, pp. 1441–1448.

[29] J. Lu, Q. Shambour, Y. Xu, Q. Lin, G. Zhang, Bizseeker: a hybrid semantic recommendation system for personalized government-to-business e-services, Internet Research 20 (2010) 342–365.

[30] G. Linden, B. Smith, J. York, Amazon.com recommendations: item-to-item collaborative filtering, IEEE Internet Computing 7 (2003) 76–80.

[31] B. N. Miller, J. A. Konstan, J. Riedl, Pocketlens: toward a personal recommender system, ACM Transactions on Information Systems (TOIS) 22 (2004) 437–476.

[32] J. Bobadilla, F. Ortega, A. Hernando, J. Alcal, Improving collaborative filtering recommender system results and performance using genetic algorithms, Knowledge-Based systems 24 (2011) 1310–1316.

[33] J. Bobadilla, F. Serradilla, J. Bernal, A new collaborative filtering metric that improves the behavior of recommender systems, Knowledge-Based Systems 23 (2010) 520–528.

[34] K. Choi, Y. Suh, A new similarity function for selecting neighbors for each target item in collaborative filtering, Knowledge-Based Systems 37 (2012) 146–153.

[35] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowledge-Based Systems 46 (2013) 109–132.

[36] J. L. Herlocker, J. A. Konstan, L. G. Terveen, J. T. Riedl, Evaluating collaborative filtering recommender systems, ACM Transactions on Information Systems 22 (2004) 5–53.

[37] J. L. Herlocker, J. A. Konstan, A. Borchers, J. Riedl, An algorithmic framework for performing collaborative filtering, in: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, 1999, pp. 230–237.

[38] F. Xie, M. Xu, Z. Chen, Rbra: A simple and efficient rating-based recommender algorithm to cope with sparsity in recommender systems, in: Proceedings of the 26th IEEE Conference on Advanced Information Networking and Applications Workshops, 2012, pp. 306–311.

[39] F. Xie, Z. Chen, H. Xu, X. Feng, Q. Hou, Tst: Threshold based similarity transitivity method in collaborative filtering with cloud computing, Tsinghua Science and Technology 18 (2013) 318–327.

[40] J.-L. Deng, Control problems of grey systems, Systems and Control Letters 1 (1982) 288–294.

[41] J.-L. Deng, Introduction to grey system theory, The Journal of grey system 1 (1989) 1–24.

[42] B. N. Miller, I. Albert, S. K. Lam, J. A. Konstan, J. Riedl, Movielens unplugged: experiences with an occasionally connected recommender system, in: Proceedings of the 8th international conference on Intelligent user interfaces, 2003, pp. 263–266.

[43] K. Yu, X. Xu, M. Ester, H.-P. Kriegel, Selecting relevant instances for efficient and accurate collaborative filtering, in: Proceedings of the 10th ACM Conference on Information and Knowledge Management, 2001, pp. 239–246.

[44] A. Karatzoglou, X. Amatriain, L. Baltrunas, N. Oliver, Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering, in: Proceedings of the fourth ACM conference on Recommender systems, 2010, pp. 79–86.

[45] M. Jamali, M. Ester, A matrix factorization technique with trust propagation for recommendation in social networks, in: Proceedings of the fourth ACM conference on Recommender systems, 2010, pp. 135–142.

[46] D. Lemire, A. Maclachlan, Slope one predictors for online rating-based collaborative filtering, Society for Industrial Mathematics 5 (2005) 471–475.

Appendix A.

Theorem 1. In a nonorthogonal coordinate space, the cosine of the angle between two vectors cannot be computed directly by using Eq. (1) or Eq. (2). An orthogonal transformation is needed.

Proof. We define (e_1,e_2,...,e_n) as the standard orthogonal basis of an n-dimensional vector space, i.e., $e_i = \left(\underbrace{0,\ldots,0}_{i-1},1,0,\ldots,0\right)^{\mathrm{T}}$

Assume (α_1, α_2,..., α_n)is an arbitrary basis of $\Re^n$;then, (α_1, α_2,..., α_n) = (e_1, e_2,..., e_n)A, where e_i, α_i are column vectors and A is the $R^{n\times n}$ transition matrix from(e_1, e_2,..., e_n) to (α_1, α_2,..., α_n). In terms of the

basis (α_1, α_2,..., α_n), suppose the coordinates of the vectors β_1,β_2 are $x = \left(x_1, x_2, \ldots, x_n\right)^{\mathrm{T}}$ and $y = \left(y_1, y_2, \ldots, y_n\right)^{\mathrm{T}}$, respectively. Equivalently, $\beta_1 = (\alpha_1, \alpha_2, \ldots, \alpha_n)x$ and $\beta_2 = (\alpha_1, \alpha_2, \ldots, \alpha_n)y$.The cosine of the angle between vectors β_1,β_2 can be calculated as $\cos<\beta_1,\beta_2>=\frac{\beta_1 \cdot \beta_2}{||\beta_1||||\beta_2||}$.

$$\beta_1 \cdot \beta_2 = {\beta_1}^T \beta_2 = x^T ({\alpha_1}^T, {\alpha_2}^T, \ldots, {\alpha_n}^T)^T (\alpha_1, \alpha_2, \ldots, \alpha_n) y = x^T F y \quad \text{(A.1)}$$

Similarly,$||\beta_1|\,|^2 = \beta_1 \cdot \beta_2 = x^T F x$,$||\beta_1|\,|^2 = \beta_1 \cdot \beta_2 = y^T F y$. Further, F denotes the measurement matrix in terms of the basis (α_1, α_2,..., α_n), namely,

$$\begin{pmatrix} (\alpha_1, \alpha_1) & (\alpha_1, \alpha_2) & \cdots & (\alpha_1, \alpha_n) \\ (\alpha_2, \alpha_1) & (\alpha_2, \alpha_2) & \cdots & (\alpha_2, \alpha_n) \\ \vdots & \vdots & \ddots & \vdots \\ (\alpha_n, \alpha_1) & (\alpha_n, \alpha_2) & \cdots & (\alpha_n, \alpha_n) \end{pmatrix} \quad \text{(A.2)}$$

Let E be the measurement matrix in terms of the basis (e_1,e_2,...,e_n); obviously, $E = I$, where I is the n-dimensional unit vector; then,

$$\begin{aligned} F &= (\alpha_1^T, \alpha_2^T, \ldots, \alpha_n^T)^T (\alpha_1, \alpha_2, \ldots, \alpha_n) \\ &= A^T (e_1^T, e_2^T, \ldots, e_n^T)^T (e_1, e_2, \ldots, e_n) A \\ &= A^T E A = A^T A \end{aligned} \quad \text{(A.3)}$$

Therefore,

$$\begin{aligned} \beta_1 \cdot \beta_2 &= x^T F y = x^T A^T A y = (Ax)^T A y \\ \|\beta_1\|^2 &= x^T F x = (Ax)^T A x \\ \|\beta_2\|^2 &= y^T F y = (Ay)^T A y \end{aligned} \quad \text{(A.4)}$$

Hence,$\cos< \beta_1,\beta_2 > = \frac{(AX)^T Ay}{(AX)^T Ax (Ay)^T Ay}$, which is $\frac{X^T y}{||x||||y||}$ only when $A = E$; therefore, the basis is a standard orthogonal basis. Therefore, in a nonorthogonal coordinate space, we cannot directly use Eq. (1) or Eq. (2) to compute the cosine of the angle between two vectors. As the Pearson Correlation is a variation of the Cosine Distance method for similarity measurement, the theorem is also applicable to Eq. (3) and Eq. (4).