The Efficient Imputation Method for Neighborhood-based Collaborative Filtering

Yongli Ren, Gang Li, Jun Zhang and Wanlei Zhou
School of Information Technology
Deakin University, 221 Burwood Highway
Vic 3125, Australia
{yongli, gang.li, jun.zhang, wanlei}@deakin.edu.au

ABSTRACT

As each user tends to rate a small proportion of available items, the resulted *Data Sparsity* issue brings significant challenges to the research of recommender systems. This issue becomes even more severe for neighborhood-based collaborative filtering methods, as there are even lower numbers of ratings available in the neighborhood of the query item. In this paper, we aim to address the *Data Sparsity* issue in the context of the neighborhood-based collaborative filtering. Given the $(user, item)$ query, a set of key ratings are identified, and an auto-adaptive imputation method is proposed to fill the missing values in the set of key ratings. The proposed method can be used with any similarity metrics, such as the *Pearson Correlation Coefficient* and *Cosine*-based similarity, and it is theoretically guaranteed to outperform the neighborhood-based collaborative filtering approaches. Results from experiments prove that the proposed method could significantly improve the accuracy of recommendations for neighborhood-based Collaborative Filtering algorithms.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering*

General Terms

Algorithms

Keywords

Imputation, Collaborative Filtering, Recommender Systems

1. INTRODUCTION

Recommender System attempts to automatically provide personalized recommendations based on the historical records of users' activities, and it is commonly constructed on the basis of *Collaborative Filtering* (CF) methods because of their insensitivity to the detailed features of users or items. In general, there are two types of collaborative filtering methods: the *neighborhood*-based method and the *model*-based method [1,10]. Although, a lot of research attention has been attracted to the *model*-based method, a lesson learned from the well-known *Netflix Competition* tells that *neighborhood*-based methods and *model*-based methods can explore very different levels of data patterns in the data set, therefore none of them can always output optimal results solely based on its own [3]. In this paper, we aim to address the data sparsity issue in the context of *neighborhood*-based method with theoretical analysis.

Neighborhood-based CF methods are generally based on the k nearest neighbor rule. Namely, its performance is highly affected by the selected nearest neighbors. However, there has been serious problems for practitioners trying to use this kind of method in areas where data is sparse. Specifically, *Data Sparsity* refers to insufficient information on a user's rating history, and the corresponding $user \times item$ rating matrix will thus become extremely sparse. When the data is prevailing with missing values, two like-minded users may not show any similarity. To overcome this problem for recommendation purposes, two major categories of approaches exist: one is to utilize more suitable similarity metrics to identify a better set of neighborhood [1, 5, 19]. The other approach is to design better aggregation methods which integrate the item ratings given by all the neighbors [6, 13]. It might be argued that a third category of approaches based on data imputation [5,16,23,25,26,28], to which this work belongs, has begun to emerge over the past couple of years. Among others, data imputation mainly involves selecting a set of $(user, item)$ pairs whose values are missing or unavailable in the $user \times item$ rating matrix, and filling them with imputed values before predicting the ratings for the active user. However, despite evidence of improved performance in the predictive accuracy, the research of imputation-based CF methods is still in its infancy. Issues such as *does all missing data possess the same importance to the recommendation purpose*, *how to select the most informative missing data to impute*, and *how to trade off the imputation benefit and error*, still largely remain unexplored.

In this paper, we propose an *Auto-Adaptive Imputation* (AutAI) method which can automatically identify a *key set* of missing data, and adaptively imputate them according to each individual user's rating history. Through this novel method, we can maximize the information contained in the neighborhood of the active user while minimizing the im-

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

putation error brought in. Inspired by Cover's [8] research on the nearest neighbor rule, which showed that the nearest neighbor contains at least 50% of the information in the infinite training set, we argue that not all missing data possesses equivalent information for a particular prediction, and some missing data is more informative than others. Then, we propose an auto-adaptive imputation method to identify the critical *key set* of missing data for each active user adaptively so as to impute the most informative missing data. Therefore, the proposed imputation method can lead to better recommendations in terms of accuracy, and we provide theoretical analysis on the performance benefit for the proposed AutAI method. On this basis, we propose a collaborative filtering algorithm by using the AutAI method from both user and item aspects (AutAI-Fusion).

The major contributions of this paper are as follows:

- We propose a novel imputation method (AutAI) to identify a *key set* of missing data for each rating prediction to address the problem of data sparsity.
- For the first time, we analyse the imputation-based methods from a theoretical perspective in the area of collaborative filtering.
- Based on the AutAI method, we propose a novel efficient collaborative filtering algorithm (AutAI-Fusion) by simultaneously considering both user activity and item popularity.
- We conduct a large set of experiments to examine the performance of the AutAI method on various similarity metrics, and to compare the proposed AutAI-Fusion algorithm with 6 state-of-the-art collaborative filtering algorithms, including other representative imputation-based algorithms.

The rest of the paper is organized as follows. We present the background in Section 2, define the *Auto Adaptive Imputation* (AutAI) method in Section 3, together with theoretical analysis about imputation benefit and error. Section 4 presents the experiment results, followed by conclusion in Section 5.

2. BACKGROUND AND RELATED WORK

In this section, we define the notations used in the paper, and we briefly recap the neighborhood-based method and the model-based method.

2.1 Notation

Let us assume $\mathcal{U} = \{u_1, u_2, \cdots, u_m\}$ be a set of m users, $\mathcal{T} = \{t_1, t_2, \cdots, t_n\}$ be the set of n items. For m users and n items, the *user* $\times$ *item* rating matrix $\mathcal{R}$ is represented as a $m \times n$ matrix. The *user* $\times$ *item* matrix $\mathcal{R}$ can be decomposed into row vectors: $\mathcal{R} = [\mathbf{u_1}, \mathbf{u_2}, \cdots, \mathbf{u_m}]^T$ and $\mathbf{u_i} = [r_{i1}, r_{i2}, \cdots, r_{in}]$, where T denotes *transpose*. The row vector $\mathbf{u_i}$ corresponds to the user u_i's rating information, and r_{ij} denotes the rating that user u_i gave to item t_j. $\mathcal{N}_k(u_x)$ denotes the set of user u_x's k nearest neighbors. $T_{xy} = \{t_i \in T | r_{xi} \neq \oslash, r_{yi} \neq \oslash\}$ denotes the set of items co-rated by both u_x and u_y. u_a denotes the active user for whom the recommendation is processing. t_s denotes the active item on which the recommendation is processing. $\bar{u}_a$ and $\bar{t}_s$ denote the average rating of user u_a and the average rating on item t_s, respectively.

2.2 Neighborhood-based CF

One of the pioneering works in *neighborhood-based Collaborative Filtering* was proposed by Paul Resnick [19]. This system, *GroupLens* utilizes users' ratings as input, then predicts how much the active user like an un-rated item. In general, the neighborhood-based Collaborative Filtering method is based on k nearest neighbors (KNN) rules. Its input is usually the entire *user* $\times$ *item rating* matrix $\mathcal{R}$. Two stages are involved in this process: *Neighbor Selection* and *Rating Aggregation*.

In the stage of *Neighbor Selection*, similarity could be evaluated between any two rows or any two columns in the *user* $\times$ *item* rating matrix, corresponding to the user-based approaches [5], or the item-based approaches [22]. It is clear that similarity measurements is one of the foundational problems in *collaborative filtering* methods. When the *user* $\times$ *item* rating matrix is sparse, this problem would be even more difficult. Several methods have been propoesd to address this problem. One way is to design better similarity metrics that tolerant the sparsity issue, e.g. the two popular metrics, *Pearson Correlation Coefficient* [19] and *Cosine-based Similarity* [1, 5]. Another way is to tackle the sparsity issue directly through data imputation, such as *Default Voting* [5], *EMDP* [16] etc. We are going to discuss this part with more details in the following Section 2.4.

In the stage of *Rating Aggregation*, for any item t_s, all the ratings on t_s by users in the $N_k(u_a)$, will be aggregated into the predicted rating value $\hat{r}_{as}$ by user u_a [1]. The *weighted majority prediction* algorithm predicts the rating on new items based on a weighted sum of the users ratings on those items [11]. Hence the determination of a suitable set of weights becomes an essential problem here. While most conventional work relies on the similarity between users or items to determine the weight, *Data Sparsity* therefore affects this stage too.

2.3 Model-based CF

In contrast to the *Neighborhood-based CF* methods, the *Model-based collaborative filtering* algorithms attempt to construct a model from the given rating matrix, and utilize the model to predict ratings on new items. A wide range of machine learning techniques have been adopted: *supervised learning*, *unsupervised learning* and *matrix decomposition* etc. For supervised learning techniques, Su and Khoshgoftaar applied the *Belief Nets* to treat rating prediction as a multi-class classification problem [24]. For unsupervised learning techniques, Lemire and Maclachlan proposed the *slope one* algorithm, based on the *popularity differential* principle between items, which means user rating behaviour is influenced by both the user's rating style (e.g. the user's average rating) and items popularity (e.g. the average rating for an item) [14]. For matrix decomposition techniques, Billsus and Pazzani proposed to use the *Singular Value Decomposition* (SVD) to exploit the '*latent structure*' in user ratings [4]. This method could utilize information from users whose ratings are not correlated with the active user. Since then, many other SVD-based methods have been proposed, such as the *SVD++* method which combines the user's explicit and implicit feedbacks [12].

2.4 Data Sparsity and Imputation

Data Sparsity is one of the most challenging issues in recommendation techniques. Since users tend to rate only a

small fraction of items in a system, the *user* × *item* rating matrix is usually very sparse, with a density of around 1% [22]. Furthermore, *Data Sparsity* may make the *neighborhood*-based CF methods incapable of finding neighbors and therefore fail to make accurate recommendations [17,22]. To overcome this problem, a number of methods have been proposed. One category of methods is to design some similarity measurement which can tolerate the sparsity in the recommendation field [9]. Another category is to design better aggregation methods which integrate the item ratings given by all neighbors [6,13]. The third category of these methods is to fill in the missing data by *imputation*, e.g. the *default voting* [5], the *smoothing method* [28], and the missing data prediction [16]. In this paper, we focus on the methods by using imputation.

Default Voting is a straight forward imputation-based method [5] which assumes default values for those missing ratings, such as exploiting average ratings by a small group of users as default ratings for other items in order to increase the size of the *co-rated item set* [7]. Xue et. al. proposed to use some machine learning methods to smooth all missing data in the *user* × *item* rating matrix [28]. Results from their experiment show that more accurate recommendations are obtained if imputing properly. Recently, Ma et. al. took the imputation confidence into consideration, and only filled in the missing data when it was confident to impute [16]. Specifically, they set a threshold to identify highly confident users, and only make a prediction about the missing data to users that have highly confident neighbors. This idea works well as it prevents poor imputation. However, their proposed EMDP algorithm treats all the missing data equally. Other previous imputation-related works focus on investigating various imputation techniques [23, 25, 26], e.g. the mean imputation, the linear regression imputation, the multiple imputation [21], and the predictive mean matching imputation [15].

Little work investigates *how to impute and which missing data should be imputed?* This question is interesting and of importance, since imputed data will bring in some imputation error as well as alleviating the sparsity issue. In this paper, we will investigate these questions, and propose an auto-adaptive imputation method that is presented in the following section.

3. AUTO-ADAPTIVE IMPUTATION BASED COLLABORATIVE FILTERING

In this section, we propose a new imputation method to effectively improve the performance of neighborhood-based collaborative filtering. Moreover, a theoretical analysis on the performance benefit of the proposed method is also provided. To our best knowledge, this work is the first one that analyses the imputation methods from a theoretical perspective in the area of collaborative filtering. Finally, by applying the proposed AutAI method, we, furthermore, propose a collaborative filtering framework by using AutAI form both user and item perspectives (AutAI-Fusion).

3.1 Formulation of Neighborhood-based CF

We first formulate the *neighborhood*-based CF using the probability theory [20]. The *neighborhood*-based CF provides recommendations by estimating how much the active user may like un-rated items, which is known as the *rating prediction* task. Given two variables, the user consuming history u and the available ratings r, the rating prediction task can be formulated as $\mu(u) = E(r|u)$, which is the expectation of dependent variable r given the independent variable u. For the recommendation purpose, it is interesting to estimate the value r_{as} on an un-rated item t_s for a singular independent variable value u_a. The estimator for u_a is then:

$$\hat{\mu}(u_a) = E(r_{as}|u_a). \tag{1}$$

From the perspective of probability theory, the observation values sampled at u_a can be used to estimate $\hat{\mu}(u_a)$.

However, there is no observation values at u_a in the context of collaborative filtering. To tackle this problem, certain similar users of u_a are selected and used to estimate $\hat{\mu}(u_a)$. This is the assumption of collaborative filtering: *similar users may have similar ratings on items*. Specifically, to estimate $\hat{\mu}(u_a)$, we can first select several *neighbors* of u_a, then treat the ratings of these neighbors as *samples* at u_a. Finally, we process the above estimation method as usual. This is the well-known k nearest neighbor estimator [10] (KNN):

$$\hat{\mu}(u_a) = E(r_{as}|u_a) = \frac{1}{k} \sum_{u_x \in \mathcal{N}_k(u_a)} r_{xs}, \tag{2}$$

where k is the number of selected neighbors, u_x is one of u_a's neighbors, $\mathcal{N}_k(u_a)$ is the set of k nearest neighbors, and r_{xs} is the rating of neighbor u_x on t_s. To further reduce the estimation bias, the KNN estimation usually apply the weighted average [2]:

$$\hat{\mu}(u_a) = E(r_{as}|u_a) = \frac{1}{\eta} \sum_{u_x \in \mathcal{N}_k(u_a)} w_{ax} r_{xs}, \tag{3}$$

where w_{ax} is the weight of neighbor u_x to u_a, and $\eta = \sum_{u_x \in \mathcal{N}_k(u_a)} w_{ax}$ is the normalizing factor. The weights reflect how close the neighbors are to u_a.

For neighborhood-based CF, the sparsity issue can lead to the critical problem of unreliable similarity since the similarity between two users is actually calculated using a very small set of co-rated items. The user relationship measured by the unreliable similarity cannot capture the overall relationship between two users [17]. Moreover, the similarities of an active user u_a to two users (u_x and u_y) are computed on two different sets of co-rated items, which results in incomparable similarities. Consequently, the performance of neighborhood-based CF is seriously affected by inaccurate similarities due to the sparsity issue.

3.2 A Novel Auto-adaptive Imputation Method

In this section, we propose a novel *Auto-Adaptive Imputation* (AutAI) method. To make a rating prediction on the active item t_s for the active user u_a, we argue that not all missing data in the *user* × *item* rating matrix possesses equivalent information for this rating prediction, and conversely there is a *key set* of missing data that are most informative for this particular prediction. Accordingly, this *key set* should be different for every prediction, even for the same user.

The proposed AutAI method can identify which missing data should be imputed automatically, with the imputed set determined adaptively according to a user's own rating history. Specifically, to make rating prediction on item t_s for user u_a, the imputed set is identified by two factors, the

users related to u_a and the items related to t_s. We denote the related users as $\mathcal{U}_a$, and denote the related items as $\mathcal{T}_s$:

$$\mathcal{U}_a = \{u_{a'}|r_{a's} \neq \oslash\} \quad (4)$$

$$\mathcal{T}_s = \{t_i|t_i \subset [T_{aa'_1} \cup \cdots \cup T_{aa'_i} \cup \cdots \cup T_{aa'_l}]\}, \quad (5)$$

where $T_{aa'_i}$ is the co-rated item set between the active user u_a and $u_{a'_i} \in \mathcal{U}_a$, and $l = |\mathcal{U}_a|$. For example, suppose that the rating histories for two users u_a and $u_{a'}$ are represented as:

$$\mathbf{u_a} = \begin{bmatrix} r_{a1}, & 0, & 0, & r_{a4}, & r_{a5}, & 0, & \cdots, & r_{an} \end{bmatrix}$$

$$\mathbf{u_{a'}} = \begin{bmatrix} r_{a'1}, & r_{a'2}, & 0, & 0, & r_{a'5}, & 0, & \cdots, & 0 \end{bmatrix},$$

where 0 indicates that the corresponding rating is missing. Then $T_{aa'} = [t_1, t_5, \cdots]$, since both user u_a and $u_{a'}$ have rated t_1, t_5 etc. Therefore, $\mathcal{T}_s$ is the union of $T_{aa'}$ over all $u_{a'} \in \mathcal{U}_a$. Furthermore, with respect to item t_s, we define the *key neighborhood* for the active user u_a as:

$$\mathcal{N}_{a,s} = \{r_{a'i}|u_{a'} \in \mathcal{U}_a, t_i \in \mathcal{T}_s\}, \quad (6)$$

where $r_{a'i}$ can either be an observing or missing rating. Normally, due to the sparsity of the *user* $\times$ *item* rating matrix, this selected *key neighborhood* is also sparse. We define all the missing data in this *key neighborhood* as the *key set* of missing data for the prediction $\hat{r}_{as}$. For each observing rating $r_{a'i}$ in $\mathcal{N}_{a,s}$, it plays a key role in the prediction for $\hat{r}_{as}$, as both user $u_{a'}$ and item t_i are highly related to $\hat{r}_{as}$. Even for the missing data in $\mathcal{N}_{a,s}$, they have equal importance for the prediction of $\hat{r}_{as}$. Please note that $\mathcal{N}_{a,s}$ is defined from the user's perspective, so we call this as the *user*-based *Auto-Adaptive Imputation* (*user*-based AutAI). After $\mathcal{N}_{a,s}$ is determined, for each missing data $r_{a'i}$ in $\mathcal{N}_{a,s}$, we use the following equation to impute its value:

$$\hat{r}_{a'i} = \bar{u}_{a'} + \frac{\sum_{u_x \in \mathcal{N}_k(u_{a'})} sim(u_{a'}, u_x) \times (r_{xi} - \bar{u}_x)}{\sum_{u_x \in \mathcal{N}_k(u_{a'})} sim(u_{a'}, u_x)}, \quad (7)$$

where $sim(u_{a'}, u_x)$ is the similarity between $u_{a'}$ and u_x as defined:

$$sim(u_{a'}, u_x) = \frac{\sum_{t_i \in T_{a'x}} (r_{a'i} - \bar{u}_{a'})(r_{xi} - \bar{u}_x)}{\sqrt{\sum_{t_i \in T_{a'x}} (r_{a'i} - \bar{u}_{a'})^2 \sum_{t_i \in T_{a'x}} (r_{xi} - \bar{u}_x)^2}}, \quad (8)$$

which is the *Pearson Correlation Coefficient* (PCC) between $u_{a'}$ and u_x [19]. Then, the imputed ratings in $\mathcal{N}_{a,s}$ can be put back to *user* $\times$ *item* rating matrix $\mathcal{R}$, and will be used for the rating prediction on item t_s for user u_a. It is clear that the missing data are imputed adaptively for user u_a on item t_s, and the *key set* of missing data is identified automatically from the user's perspective. The pseudocode of the *user*-based AutAI method is presented in Algorithm 1.

Similarly, AutAI can also work in an item-based manner. In this case, the *key neighborhood* for the active user u_a with respect to the active item t_s is defined as $\mathcal{N'}_{a,s} = \{r_{a'i}|u_x \in \mathcal{U'}_a, t_{s'} \in \mathcal{T'}_s\}$, where $\mathcal{T'}_s = \{t_{s'}|r_{as'} \neq \oslash\}$, $\mathcal{U'}_a = \{u_x|u_x \subset [U_{ss'_1} \cup \cdots \cup U_{ss'_i} \cup \cdots \cup U_{ss'_l}]\}$, $U_{ss'_i}$ denotes the set of users who rated both t_s and $t_{s'_i}$, and $l = |\mathcal{T'}_s|$. Consequently, the missing data in $\mathcal{N'}_{a,s}$ forms the *key set* of missing data in this item-based manner. Imputing this missing data is called the *item*-based *Auto-Adaptive Imputation* (*item*-based AutAI).

The imputed missing data in $\mathcal{N}_{a,s}$ contributes in two ways. Firstly, Cover [8]'s research on the nearest neighbor rule shows that the nearest neighbor contains at least 50% of the information in the infinite training set. In this sense, the missing data in $\mathcal{N}_{a,s}$ contains much more information than the missing data outside of $\mathcal{N}_{a,s}$. So, imputing this missing data will bring in much more benefit compared to imputing other missing data. Moreover, imputation in this way will also bring in less associated imputation error. We provide a theoretical analysis on this point in the following section. Secondly, AutAI makes a further similarity measurement between each neighbor to the active user in the same $\mathcal{T}_s$ space. Based on the common assumption of the *neighborhood*-based CF algorithms that *if two users rated n items similarly, they tend to rate similarly on other items* [5, 22], this auto-adaptive imputation method makes the similarity measurement between the active user and any other neighbor on the same $\mathcal{T}_s$ items, rather than on the co-rated items between two users. Formally, we denote the similarity between two users u_a and u_x on imputed data as $sim'(u_a, u_x)$, which can be measured by any similarity metric, for example, its PCC-based version is formulated as:

$$sim'(u_a, u_x) = \frac{\sum_{t_i \in \mathcal{T}_s} (r_{ai} - \bar{u}_a)(r_{xi} - \bar{u}_x)}{\sqrt{\sum_{t_i \in \mathcal{T}_s} (r_{ai} - \bar{u}_a)^2 \sum_{t_i \in \mathcal{T}_s} (r_{xi} - \bar{u}_x)^2}}, \quad (9)$$

where $\bar{u}_x$ is the average rating of user u_x.

On the other hand, the imputed missing data in $\mathcal{N'}_{a,s}$ also contributes in a similar way to its counterpart in $\mathcal{N}_{a,s}$, so we will not list them separately. The similarity between two items, after applying the item-based AutAI, can be calculated in a similar way, and its PCC-based version is defined:

$$sim'(t_s, t_i) = \frac{\sum_{u_{a'} \in \mathcal{U'}_a} (r_{a's} - \bar{t}_s)(r_{a'i} - \bar{t}_i)}{\sqrt{\sum_{u_{a'} \in \mathcal{U'}_a} (r_{a's} - \bar{t}_s)^2 \sum_{u_{a'} \in \mathcal{U'}_a} (r_{a'i} - \bar{t}_i)^2}}, \quad (10)$$

where $\bar{t}_s$ is the average rating of item t_s.

Algorithm 1 Auto-Adaptive Imputation (AutAI)

Require: the user-item rating matrix $\mathcal{R}$. the active user u_a. the active item t_s.
Ensure: $\mathcal{R}'$: the imputed matrix $\mathcal{R}'$.

```
for each u_x ∈ U do
    for each u_y ∈ U & u_y ≠ u_x do
        calculate sim(u_x, u_y) according to Eq. 8;
    end for
end for
R' ← R;
U_a ← {u_a' | r_a's ≠ ∅};
T_s ← ∅;
for each u_a' ∈ U_a do
    T_aa' ← {t_i | r_ai ≠ ∅, r_a'i ≠ ∅};
    T_s ← T_s ∪ T_aa';
end for
N_a,s ← {r_a'i | u_a' ∈ U_a, t_i ∈ T_s};
for each r_a'i ∈ N_a,s do
    if r_a'i = ∅ then
        r̂_a'i ← ū_a' + (Σ_{u_x ∈ N_k(u_a')} sim(u_a', u_x) × (r_xi − r̄_x)) / (Σ_{u_x ∈ N_k(u_a')} sim(u_a', u_x));
        R'(u_a', t_i) ← r̂_a'i;
    end if
end for
return R';
```

3.3 Analysis on Auto-Adaptive Imputation

Why and how can the proposed AutAI method benefit the neighborhood-based CF? In this section, we attempt to answer this question from a theoretical perspective. Specifically, we study how the proposed AutAI method can help to select nearest neighbors more accurately. The analysis is conducted in the user-based AutAI, and it can be extended to the item-based AutAI in a straightforward manner.

Neighbor Selection determines the performance of the neighborhood based CF, which is based on the measured similarities among users [1, 2]. Let us consider a general case: suppose the active user u_a and two other users, u_x and u_y, have the following rating histories:

$$\begin{aligned}\mathbf{u_a} &= [r_{a1}, r_{a2}, \cdots, r_{ai}, \cdots, r_{al}]\\ \mathbf{u_x} &= [r_{x1}, \hat{r}_{x2}, \cdots, \hat{r}_{xi}, \cdots, \hat{r}_{xl}]\\ \mathbf{u_y} &= [r_{y1}, \hat{r}_{y2}, \cdots, r_{yi}, \cdots, \hat{r}_{yl}]\end{aligned}$$

where $\hat{r}_{xi}$ is the imputed rating on item t_i for u_x, and $l = |\mathcal{T}_s|$. According to Eq. 5, $\mathbf{u_a}$ contains the real ratings for u_a, but $\mathbf{u_x}$ contains both real ratings and imputed ratings. So, both $\mathbf{u_a}$ and $\mathbf{u_x}$ can be divided into two subsets: T_{ax} and $\mathcal{T}_s \setminus T_{ax}$, where T_{ax} denotes the co-rated item set between u_a and u_x, and $\mathcal{T}_s \setminus T_{ax}$ denotes the relative complement of T_{ax} in $\mathcal{T}_s$ [20]. Therefore, $\mathbf{u_a}$ and $\mathbf{u_x}$ can be represented as:

$$\begin{aligned}\mathbf{u_a} &= [\overbrace{r_{a(1)}, \cdots, r_{a(p)}}^{|T_{ax}|}, \overbrace{r_{a(p+1)}, \cdots, r_{a(l)}}^{|\mathcal{T}_s \setminus T_{ax}|}]\\ \mathbf{u_x} &= [\overbrace{r_{x(1)}, \cdots, r_{x(p)}}^{|T_{ax}|}, \overbrace{\hat{r}_{x(p+1)}, \cdots, \hat{r}_{x(l)}}^{|\mathcal{T}_s \setminus T_{ax}|}],\end{aligned}$$

where $p = |T_{ax}|$ and $l = |\mathcal{T}_s|$.

We measure the similarity between u_a and u_x by the distance in their item space. The distance on each item can be considered as an estimation for their similarity. The distance on item t_i is denoted as d_{t_i}. Then, the distance d_{ax} between u_a and u_x can be expressed by

$$d_{ax} = \begin{cases} d_{t_i}, & \text{if } r_{xi} \text{ is real rating,}\\ d_{t_i} + \varepsilon_i, & \text{if } r_{xi} \text{ is imputed,}\end{cases} \tag{11}$$

where $\varepsilon_i = r_{xi} - \hat{r}_{xi}$ denotes the imputation error on item t_i for user t_x, and $\hat{r}_{xi}$ denotes the imputed value.

In this study, for two users, we assume that the distance on any item is independent and identically distributed (i.i.d), and we adopt the normal distribution for theoretical analysis. Then, the Probability Density Function (PDF) [20, 29] of d_{t_i} is:

$$p(d_{t_i}) \sim \begin{cases}\mathcal{N}(\mu_1, \sigma_1^2), & \text{for } u_a \text{ and } u_x\\ \mathcal{N}(\mu_2, \sigma_2^2), & \text{for } u_a \text{ and } u_y\end{cases} \tag{12}$$

Similarly, we assume the imputation error ε on any item for any user is also independent and identically distributed (i.i.d), and its PDF is:

$$p(\varepsilon) \sim \mathcal{N}(\mu_\varepsilon, \sigma_\varepsilon^2), \tag{13}$$

Then, the distance d_{ax} between u_a and u_x is formulated as the expectation over all possible items, $d_{ax} = E(d_t)$. Practically, the expectation can be estimated by using the average of all observations, so,

$$d_{ax} = E(d_t) = \frac{1}{n}\sum_{i=1}^{n} d_{t_i}, \tag{14}$$

where n is the number of items taking into account when measuring d_{ax}. In the user-based AutAI method, this set of items is $\mathcal{T}_s$, and d_{ax}^{aai} denotes the distance d_{ax}. Considering $\mathcal{T}_s$ consists of two subsets, T_{ax} and $\mathcal{T}_s \setminus T_{ax}$, d_{ax}^{aai} is affected by items coming from both of them, and can be represented as:

$$\frac{1}{p}\sum_{i=1}^{p} d_{t_i} \text{ and } \frac{1}{q}\sum_{j=1}^{q} \hat{d}_{t_j},$$

where $p = |T_{ax}|$, $q = |\mathcal{T}_s \setminus T_{ax}|$, and $\hat{d}_{t_j} = d_{t_j} + \varepsilon_j$ represents the estimation from imputed values as defined in Eq. 11. Due to the existence of the imputation error ε, the similarity estimation coming from $\mathcal{T}_s \setminus T_{ax}$ has to take it into consideration. The *cumulative imputation error* for the estimation of d_{ax}^{aai} over $\mathcal{T}_s \setminus T_{ax}$ is:

$$\begin{aligned}\varepsilon_{aai} &= \frac{1}{q}\sum_{j=1}^{q}\hat{d}_{t_j} - \frac{1}{q}\sum_{j=1}^{q} d_{t_j}\\ &= \frac{1}{q}\left(\sum_{j=1}^{q}(d_{t_j} + \varepsilon_j) - \sum_{j=1}^{q} d_{t_j}\right)\\ &= \frac{1}{q}\sum_{j=1}^{q}\varepsilon_j. \end{aligned} \tag{15}$$

Consequently, d_{ax}^{aai} in AutAI is represented as the calculation based on real ratings plus the *cumulative imputation error* ε_{aai}:

$$\frac{1}{l}\sum_{i=1}^{l} d_{t_i} \text{ and } \varepsilon_{aai} = \frac{1}{q}\sum_{j=1}^{q}\varepsilon_j,$$

where $l = |\mathcal{T}_s|$. According to Eq. 13 and Eq. 15, ε_{aai} is also normal, and its PDF is:

$$p(\varepsilon_{aai}) \sim \mathcal{N}(\mu_\varepsilon, \frac{\sigma_\varepsilon^2}{q}), \tag{16}$$

where $q = |\mathcal{T}_s \setminus T_{ax}|$. Normally, due to $|\mathcal{T}_s \setminus T_{ax}| \gg 1$, $\frac{\sigma_\varepsilon^2}{q}$ is much smaller than σ_ε^2. For example, in *MovieLens* data set, $mean(|\mathcal{T}_s \setminus T_{ax}|) = 85.08$, which means the order of magnitude of $\frac{\sigma_\varepsilon^2}{q}$ is around two orders of magnitude less than σ_ε^2. Namely, ε_{aai} actually varies little around its mean value, and its variance can be ignored comparing with its initial distribution ε. Consequently, the measurement of d_{ax}^{aai} over $\mathcal{T}_s$ is defined:

$$d_{ax}^{aai} = E(d_t) = \frac{1}{l}\sum_{i=1}^{l} d_{t_i} + \mu_\varepsilon, \tag{17}$$

where $l = |\mathcal{T}_s|$. According to Eq. 12, Eq. 13 and Eq. 17, we obtain that:

$$p(d_{ax}^{aai}) \sim \mathcal{N}(\mu_1 + \mu_\varepsilon, \frac{\sigma_1^2}{l}). \tag{18}$$

Similarly, the PDF of the distance between u_a and u_y in AutAI, d_{ay}^{aai}, is:

$$p(d_{ay}^{aai}) \sim \mathcal{N}(\mu_2 + \mu_\varepsilon, \frac{\sigma_2^2}{l}). \tag{19}$$

We define the *distance divergence* ψ between the distance of two users u_x and u_y to the active user u_a as:

$$\psi = d_{ax} - d_{ay}, \tag{20}$$

which determines their order to the active user. It is well-known that the *confidence interval* (CI) can be used to indicate the reliability of an estimation [20], and it has been widely used in the research of neighborhood-based models [2]. Therefore, in this study, we apply CI to measure the reliability of the estimation of ψ. CI is a range of values that quantify the uncertainty of the estimation, and a narrow CI means high precision [20]. According to Eq. 18 and Eq. 19, the $100(1-\alpha)\%$ confidence interval for ψ in AutAI can be formulated as follows:

$$\begin{aligned} CI(\psi_{aai}) &= ((\mu_1+\mu_\varepsilon)-(\mu_2+\mu_\varepsilon)) \\ &\quad \pm z_{\alpha/2}\sqrt{\frac{\frac{\sigma_1^2}{l}}{l}+\frac{\frac{\sigma_2^2}{l}}{l}} \\ &= (\mu_1-\mu_2)\pm z_{\alpha/2}\sqrt{\frac{\sigma_1^2}{l^2}+\frac{\sigma_2^2}{l^2}}, \end{aligned} \quad (21)$$

where $z_{\alpha/2}$ is a standard normal variate which is exceeded with a probability of $\alpha/2$. We define $\sigma_{\psi_{aai}}$ as the standard error of ψ_{aai}:

$$\sigma_{\psi_{aai}} = \sqrt{\frac{\sigma_1^2}{l^2}+\frac{\sigma_2^2}{l^2}}. \quad (22)$$

Please note the width of $CI(\psi_{aai})$ is proportional to $\sigma_{\psi_{aai}}$.

Now, let us consider conventional neighborhood-based CF, the measurement of the distance between u_a and u_x, d_{ax}^{knn}, is estimated over T_{ax}, which is:

$$d_{ax}^{knn} = E(d_t) = \frac{1}{p}\sum_{i=1}^{p} d_{t_i}, \quad (23)$$

where $p = |T_{ax}|$. According to Eq. 12 and Eq. 23, we have

$$p(d_{ax}^{knn}) \sim \mathcal{N}(\mu_1, \frac{\sigma_1^2}{p_1}), \quad (24)$$

where $p_1 = |T_{ax}|$. Similarly, d_{ay}^{knn}, the distance between u_a and u_y in conventional neighborhood-based CF, also follows the normal distribution, and its PDF is:

$$p(d_{ay}^{knn}) \sim \mathcal{N}(\mu_2, \frac{\sigma_2^2}{p_2}), \quad (25)$$

where $p_2 = |T_{ay}|$. Therefore, according to Eq. 24 and Eq. 25, the $100(1-\alpha)\%$ confidence interval for ψ_{knn}, ψ in neighborhood based CF approaches, is:

$$\begin{aligned} CI(\psi_{knn}) &= (\mu_1-\mu_2)\pm z_{\alpha/2}\sqrt{\frac{\frac{\sigma_1^2}{p_1}}{p_1}+\frac{\frac{\sigma_2^2}{p_2}}{p_2}} \\ &= (\mu_1-\mu_2)\pm z_{\alpha/2}\sqrt{\frac{\sigma_1^2}{p_1^2}+\frac{\sigma_2^2}{p_2^2}}, \end{aligned} \quad (26)$$

where $z_{\alpha/2}$ is a standard normal variate which is exceeded with a probability of $\alpha/2$. $\sigma_{\psi_{knn}}$ is the standard error of the estimated ψ_{knn}:

$$\sigma_{\psi_{knn}} = \sqrt{\frac{\sigma_1^2}{p_1^2}+\frac{\sigma_2^2}{p_2^2}}, \quad (27)$$

where $p_1 = |T_{ax}|$ and $p_2 = |T_{ay}|$. And the width of $CI(\psi_{knn})$ is proportional to $\sigma_{\psi_{knn}}$.

According to Eq. 5, it is clear that $l = |\mathcal{T}_s| \geq |T_{ax}| = p_1$ and $l = |\mathcal{T}_s| \geq |T_{ay}| = p_2$. For example, in the benchmark dataset *MovieLens*, $mean(|\mathcal{T}_s|) = 103.47$, and $mean(|T_{ax}|) = 18.4$. Together with Eq. 22 and Eq. 27, we obtain:

$$\sigma_{\psi_{aai}} \leq \sigma_{\psi_{knn}} \quad (28)$$

with equality if and only if $|\mathcal{T}_s| = |T_{ax}|$ and $|\mathcal{T}_s| = |T_{ay}|$, which is not valid when facing the data sparsity issue in the field of collaborative filtering. Furthermore, according to Eq. 21 and Eq. 26, one can see that the $CI(\psi_{aai})$ is narrower than or equal to $CI(\psi_{knn})$, since $\sigma_{\psi_{aai}}$ is smaller than or equal to $\sigma_{\psi_{knn}}$. Based on the above theoretical analysis, we can conclude that the proposed AutAI method can effectively improve the performance of the neighborhood-based CF through more accurate nearest neighbor selections by using a sparse rating matrix in a novel way.

3.4 Rating Prediction

For the purpose of recommendation, we propose a new rating prediction algorithm, an Auto-Adaptive Imputation algorithm for both users and items (AutAI-Fusion), which can simultaneously take the user activity and the item popularity into account. Specifically, we apply the AutAI method for both users and items, and combine the predictions using a linear function.

To take users activity into consideration, we first perform a user-based AutAI imputation as described in Eq. 6, then make prediction as follows. The prediction for r_{as} in this user-based manner is calculated as:

$$\hat{r}_{as}^{u} = \bar{u}_a + \frac{\sum_{u_x\in\mathcal{N}_k(u_a)} sim'(u_a,u_x)\times(r_{xs}-\bar{u}_x)}{\sum_{u_x\in\mathcal{N}_k(u_a)} sim'(u_a,u_x)}, \quad (29)$$

where $\bar{u}_a$ is the average rating of u_a, and $sim'(u_a,u_x)$ is defined in Eq. 9.

Similarly, to take item popularity into consideration, we do an item-based AutAI imputation as described in Section 3.2, then the prediction for r_{as} in this item-based manner is calculated as:

$$\hat{r}_{as}^{i} = \bar{t}_s + \frac{\sum_{t_i\in\mathcal{N}_k(t_s)} sim'(t_s,t_i)\times(r_{ai}-\bar{t}_i)}{\sum_{t_i\in\mathcal{N}_k(t_s)} sim'(t_s,t_i)}, \quad (30)$$

where $\bar{t}_s$ is the average rating of t_s, and $sim'(t_s,t_i)$ is defined in Eq. 10.

After imputing and predicting from both user and item respectively, the final prediction for r_{as} in the proposed AutAI-Fusion algorithm is calculated as:

$$\hat{r}_{as} = \lambda\hat{r}_{as}^{u} + (1-\lambda)\hat{r}_{as}^{i}. \quad (31)$$

A similar fusion strategy is performed in [16, 27]. The parameter λ determines to what extent the final prediction is based on user-based AutAI or item-based AutAI imputation prediction. When $\lambda = 1$, the prediction is completely generated by taking user activity to perform an AutAI-based prediction. On the other hand, when $\lambda = 0$, the prediction is totally estimated by taking item popularity to perform an AutAI-based prediction. The value for λ can be determined by doing cross-validation, which we will discuss further in the experiment section.

4. EXPERIMENT

In this section, we conduct several experiments to answer the following questions:

1. How does the proposed AutAI method perform with various similarity metrics? For this point, we employ two well-known similarity metrics in the field of collaborative filtering, the *Pearson Correlation Coefficient* (PCC) [19] and the *Cosine*-based similarity (COS) [5]. To achieve a fair comparison, we perform this experiment in a user-based manner. Namely, we only do AutAI imputation from the perspective of users (the user-based AutAI), and do the prediction based on Eq. 29. Then we compare it with the user-based collaborative filtering algorithm.

2. How does the proposed AutAI-Fusion algorithm compare with other imputation-based CF algorithms? For this question, we compare the proposed AutAI-Fusion algorithm with other state-of-the-art *imputation*-based algorithms, including the well-known *EMDP* [16], the *SCBPCC* method [28], *Default Voting* (Default Voting) [5], and two traditional collaborative filtering algorithms, the user-based algorithm [19] and the item-based CF algorithm [22], and one model-based algorithm, the *Slope One* algorithm [14]. Please note that EMDP is an imputation-based algorithm by fusing the user-based CF algorithm and the item-based CF algorithm.

3. How does the parameter λ affect the performance of the proposed AutAI-Fusion algorithm? Since the proposed AutAI method take either user activity or item popularity into consideration, the parameter λ balance how much the prediction comes from the user-based AutAI and the item-based AutAI. We examine AutAI-Fusion's performance in accuracy by varying λ from 0 to 1.

4.1 Experimental Setup

The data set we experiment with is the popular benchmark data set *MovieLens* [1], which has been widely used in CF research [16, 18, 27, 28]. *MovieLens* includes around 1 million ratings collected from 6,040 users on 3,900 movies.

To evaluate the performance thoroughly, we extract a subset of 2000 users from *MovieLens* who rated at least 30 movies, and further we set up several different experimental configurations. Specifically, we split the selected subset into two sets, the Training set and the Test set. The size of the Training set varies from the first 500, 1000 and 1500 users, and are denoted as M_{500}, M_{1000} and M_{1500} respectively. The remaining 500 users are treated as the Test set. For each of the active user within the Test set, we alter the number of rated items provided from 10, 20 to 30, which are represented as $Given_{10}$, $Given_{20}$ and $Given_{30}$, respectively. This protocol is widely used in Collaborative Filtering research [16, 27, 28]. Furthermore, we also apply the *All-But-One* configuration, in which we randomly select one single rating for each user in the data set, then try to predict its value when observing all the other ratings the user has given.

4.2 Evaluation Metric

For consistency with other literatures [16, 27, 28], we apply the *Mean Absolute Error* (MAE) as the measurement metric, which is defined as:

$$MAE = \frac{\sum_{(a,s)\in X} |r_{as} - \hat{r}_{as}|}{|X|}, \tag{32}$$

where r_{as} is the true rating given by user u_a on item t_s, $\hat{r}_{as}$ is the predicted rating, X denotes the test data set, and $|X|$ represents its size. A smaller MAE value means better performance.

[1] http://www.grouplens.org/

Table 1: MAE comparison on different similarity measurement metrics on the *Given* Data set. (A smaller value means better performance)

Training Users		PCC	AutAI+PCC	COS	AutAI+COS
M_{500}	$Given_{10}$	0.7925	**0.7788**	0.7855	**0.7726**
	$Given_{20}$	0.7663	**0.7530**	0.7663	**0.7513**
	$Given_{30}$	0.7546	**0.7404**	0.7561	**0.7406**
M_{1000}	$Given_{10}$	0.7802	**0.7713**	0.7853	**0.7663**
	$Given_{20}$	0.7594	**0.7464**	0.7672	**0.7437**
	$Given_{30}$	0.7491	**0.7348**	0.7563	**0.7333**
M_{1500}	$Given_{10}$	0.7764	**0.7670**	0.7854	**0.7630**
	$Given_{20}$	0.7581	**0.7423**	0.7687	**0.7397**
	$Given_{30}$	0.7487	**0.7309**	0.7585	**0.7293**

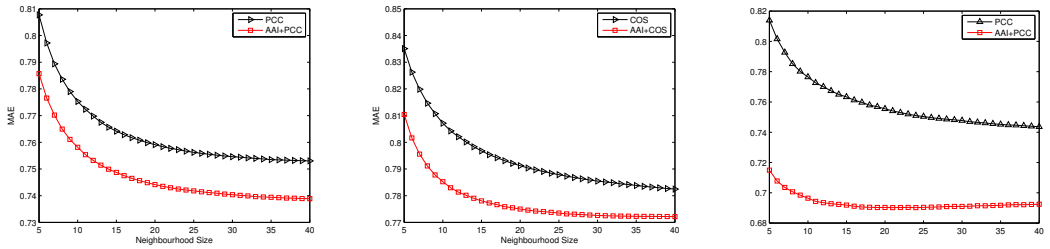
Table 2: MAE comparison on different similarity measurement metrics on the *All-But-One* Data set. (A smaller value means better performance)

Training Users	PCC	AutAI+PCC	COS	AutAI+COS
All-But-One	0.7477	**0.6910**	0.7652	**0.6932**

4.3 Performance with Different Similarity Metrics

In order to evaluate the performance of AutAI, we use the proposed AutAI method on two state-of-the-art similarity metrics in Collaborative Filtering [1, 10, 16, 27, 28], the PCC and the COS. We implement 4 algorithms in the user-based manner, namely the user-based PCC algorithm, user-based COS algorithm, and user-based AutAI with PCC and COS, respectively. Then, we compare their prediction performance, in which the number of neighbors is set to 30. We examine AutAI performance on all the experiment configurations, including *Given* and *All-But-One* data sets.

The results on the *Given* and *All-But-One* data sets are shown in Table 1 and Table 2, respectively. The results show that AutAI achieves significant improvements on both similarity metrics in all experiment configurations. This indicates that the proposed AutAI method works well and robustly in different sparsity situations. Specifically, on the *Given* data set, the number of given ratings for the active user is restricted to 10, 20 or 30, which suppresses the effectiveness of AutAI, as AutAI determines the *key set* of missing data to fill in according to the rated items by this user as shown in Eq. 6. However, AutAI can still significantly outperform the user-based CF on all the *Given* data sets. On the *All-But-One* data set, AutAI achieves even larger improvements on both PCC and COS metrics as shown in Table 2. For PCC, the MAE is reduced from 0.7477 to 0.6910 by 7.58%, and for COS, it is reduced from 0.7652 to 0.6932 by 9.41%. This is mainly because there is no limitation to users' rating history, so AutAI can fully work in this natural setting.



(a) PCC on $M_{500}Given_{30}$ (b) COS on $M_{500}Given_{10}$ (c) PCC on *All-But-One*

(d) PCC on $M_{1500}Given_{30}$ (e) COS on $M_{500}Given_{30}$ (f) COS on *All-But-One*

Figure 1: Sensitivity to the Size of Neighborhood

Table 3: Paired-t-test for MAE on the *Given* Dataset and the *All-but-one* Dataset

	Paired-t statistics		
Methods	df	t	*p*-value
AutAI+PCC vs. PCC	9	4.0137	0.0030
AutAI+COS vs. COS	9	4.8512	0.0009

Moreover, in order to examine AutAI's performance thoroughly, we vary the number of neighbors from 5 to 40 so as to examine its sensitivity to the neighborhood size. In this paper, we report results on $M_{500}Given_{30}$ and $M_{1500}Given_{30}$ for PCC, results on $M_{500}Given_{10}$ and $M_{500}Given_{30}$ for COS, and results on *All-But-One* data sets for both PCC and COS in Fig. 1. We can observe that the neighborhood size does affect the performance. The AutAI method outperforms its counterpart across all neighborhood sizes from 5 to 40 on all data sets. This indicates that the AutAI method identifies the neighborhood relationship more accurately as we demonstrate in Section 3.3, and therefore, achieves better performance across all neighborhood sizes.

The two-tailed, paired t-test with a 95% confidence level has been applied to evaluate the performance of AutAI on both similarity metrics. The results show that the difference of performance with and without AutAI is statistically significant. The detailed paired-t statistics are shown in Table 3.

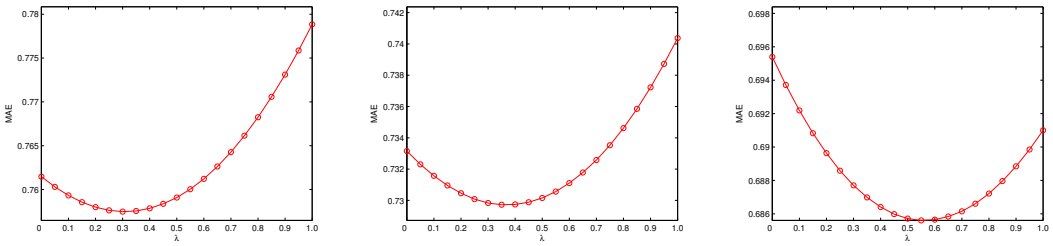
4.4 Comparison with Other Methods

In this section, we examine the proposed AutAI-Fusion algorithm by comparing it with other state-of-the-art *imputation*-based algorithms, including the *default voting* (Default Voting) method [5], the EMDP method [16], and the SCBPCC meth-od [28], and two traditional collaborative filtering algorithms, the *user-based* CF (UPCC) and the *item-based* CF (IPCC), and one model-based algorithm, the *Slope One* algorithm [14]. For fair comparison, we set λ and K with the values used in [28], but they may not lead to the best performance. Specifically, the parameters in SCBPCC are set as $\lambda = 0.35$, and the cluster number $K = 20$. All the parameters in EMDP are set as used in [16], namely $\lambda = 0.7, \gamma = 30, \delta = 25, \eta = \theta = 0.4$. Please note that EMDP is an imputation-based algorithm by fusing the user-based CF algorithm and the item-based CF algorithm. The parameter λ in our algorithm is set to 0.4.

Table 4: MAE comparison with other methods on the *Given* Data set. (A smaller value means better performance)

Training users	Methods	$Given_{10}$	$Given_{20}$	$Given_{30}$
M_{500}	AutAI-Fusion	**0.7579**	**0.7387**	**0.7298**
	EMDP	0.7724	0.7573	0.7421
	SCBPCC	0.7837	0.7633	0.7550
	Default Voting	0.7762	0.7594	0.7513
	UPCC	0.7925	0.7663	0.7546
	IPCC	0.7927	0.7613	0.7493
	Slope One	0.7704	0.7455	0.7341
M_{1000}	AutAI-Fusion	**0.7488**	**0.7298**	**0.7211**
	EMDP	0.7602	0.7440	0.7301
	SCBPCC	0.7784	0.7602	0.7499
	Default Voting	0.7776	0.7614	0.7535
	UPCC	0.7802	0.7594	0.7491
	IPCC	0.7673	0.7402	0.7293
	Slope One	0.7601	0.7374	0.7263
M_{1500}	AutAI-Fusion	**0.7445**	**0.7256**	**0.7172**
	EMDP	0.7548	0.7383	0.7264
	SCBPCC	0.7792	0.7587	0.7506
	Default Voting	0.7812	0.7659	0.7580
	UPCC	0.7764	0.7581	0.7487
	IPCC	0.7572	0.7315	0.7214
	Slope One	0.7581	0.7355	0.7246

The results on the *Given* data sets are shown in Table 4. Clearly, it is observed that AutAI-Fusion outperforms all of the other 6 algorithms on all configurations. Specifically,



(a) M_{500} $Given_{10}$ (b) M_{500} $Given_{30}$ (c) *All-But-One*

Figure 2: Impact of Parameter λ on MAE

Table 5: MAE comparison with other methods on the *All-But-One* Data set. (A smaller value means better performance)

Training Users	AutAI-Fusion	EMDP	SCBPCC	Default Voting	UPCC	IPCC	Slope One
All-But-One	**0.6864**	0.7244	0.7613	0.7954	0.7477	0.7056	0.7164

Table 6: Paired-t-test for MAE on the *Given* Dataset and the *All-but-one* Dataset

	Paired-t statistics		
Methods	df	t	p-value
AutAI-Fusion vs. EMDP	9	5.5460	0.0004
AutAI-Fusion vs. SCBPCC	9	7.2837	< 0.0001
AutAI-Fusion vs. Default Voting	9	4.5898	0.0013
AutAI-Fusion vs. UPCC	9	10.3010	< 0.0001
AutAI-Fusion vs. IPCC	9	5.3386	0.0005
AutAI-Fusion vs. Slope One	9	4.6457	0.0012

on the $M_{500}Given10$ data set, although all imputation-based methods, AutAI-Fusion, EMDP, SCBPCC and Default Voting, achieve a better performance than UPCC and IPCC, AutAI-Fusion achieves the largest improvement. This trend is obvious on all data sets as shown by the figures in bold in Table 4. This indicates that when the training set is small and the rating history of users is limited, AutAI-Fusion can still identify the most determinant missing values to fill in, in order to identify more appropriate relationships among neighbors. On the *All-But-One* data set as shown in Table 5, AutAI-Fusion obtains even better results than all compared algorithms by achieving a much smaller MAE 0.6864. This is mainly because there is no limitation to the rating history of users, and also because AutAI can work more efficiently in this natural configuration. Table 6 lists the paired-t test statistics (with a 95% confidence level) between AutAI-Fusion and these algorithms, and it is clear that the differences between the performance of AutAI-Fusion and that of other methods are statistically significant.

4.5 Impact of Parameter

As we discussed in Section 3.4, we introduced a parameter λ to balance the prediction from the user-based AutAI and the item-based AutAI, to simultaneously consider both the activity of users and the popularity of items. We conducted several experiments to determine the impact of λ to the proposed AutAI-Fusion algorithm. Specifically, we vary the λ value from 0 to 1 with an increasing step of 0.05. We report the results on $M_{500}Given_{10}$, $M_{500}Given30$, and *All-But-One* data sets in Fig. 2.

When $\lambda = 0$, the prediction totally depends on the item-based AutAI imputation; when $\lambda = 1$, the prediction fully depends on the user-based AutAI imputation. Results on the *Given* data sets show that when there are few training users or few ratings of the active user, item-based AutAI can achieve significant improvement compared to user-based AutAI. For example, on data sets $M_{500}Given_{10}$ and $M_{500}Given30$ as shown in Fig. 2a and Fig. 2b, the performance of AutAI-Fusion with $\lambda = 0$ is better than its performance with $\lambda = 1$. This is because a limited user rating history suppresses the effectiveness of user-based AutAI. However, AutAI-Fusion achieves an even better performance with $\lambda = 0.4$. On the other hand, Fig. 2c shows that user-based AutAI and item-based AutAI achieve a similar performance on the *All-But-One* configuration, which indicates that in a natural situation there is no big difference between them. Yet, on all configurations, it is clear that better accuracy can be obtained by combining imputations from users and items.

5. CONCLUSION

In this paper, we treat each rating given by a user on an item as an observation for the predictions on this item to other users, which matches the theory of nearest neighbor rules [2, 8]. From this point of view, we define the notation of the *key set* of missing data for the rating prediction, and propose AutAI, a method to automatically and adaptively identify these key missing data for each prediction from both the user and the item perspective. Theoretical and empirical analysis show that imputation based on this *key set* of missing data leads to a more accurate identification of neighborhood relationships. Therefore, the AutAI-based algorithm can make more accurate predictions than neighborhood-based CF algorithms, including other imputation-based algorithms. To the best of our knowledge, no previous work has studied the theoretical analysis for imputation-based CF methods like this paper has. Therefore, the proposed AutAI method guarantees to work regardless of whichever similarity metric is applied. Moreover, we also propose an AutAI-Fusion algorithm based on the AutAI method to make predictions by taking user activity and item popularity into consideration. The experiment results show that: (1) the proposed AutAI method can identify neighbors

more accurately, and (2) the proposed AutAI-Fusion algorithm can achieve better predictions in terms of accuracy.

6. REFERENCES

[1] G. Adomavicius and 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(6):734–749, 2005.

[2] N. S. Altman. An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician*, 46(3):175, Aug. 1992.

[3] R. M. Bell and Y. Koren. Lessons from the Netflix Prize Challenge. *SIGKDD Explorations*, 9(2):75–79, 2007.

[4] D. Billsus and M. Pazzani. Learning collaborative information filters. In *Proceedings of the Fifteenth International Conference on Machine Learning*, volume 54, page 48, 1998.

[5] J. Breese, D. Heckerman, C. Kadie, and Others. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.

[6] L. M. D. Campos, J. M. Fernández-luna, J. F. Huete, and M. A. Rueda-morales. Measuring Predictive Capability in Collaborative Filtering. In *Proceedings of the 3rd ACM Conference on Recommender Systems*, pages 313–316, 2009.

[7] S. Chee, J. Han, and K. Wang. Rectree: An efficient collaborative filtering method. *Data Warehousing and Knowledge Discovery*, pages 141–151, 2001.

[8] T. Cover. Estimation by the nearest neighbor rule. *IEEE Transactions on Information Theory*, pages 50–55, 1968.

[9] C. Desrosiers and G. Karypis. A Novel Approach to Compute Similarities and Its Application to Item Recommendation. In *Proceeding of PRICAI 2010*, pages 39–51, 2010.

[10] C. Desrosiers and G. Karypis. *Chapter 4 A comprehensive Survey of Neighborhood-based Recommendation Methods*, chapter 4, pages 107–144. Springer US, Boston, MA, 2011.

[11] S. a. Goldman and M. K. Warmuth. Learning binary relations using weighted majority voting. *Machine Learning*, 20(3):245–271, Sept. 1995.

[12] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *SIGKDD 2008*, pages 426–434. ACM, 2008.

[13] M. Larson and A. Hanjalic. Exploiting User Similarity based on Rated-Item Pools for Improved User-based Collaborative Filtering. In *Proceedings of the 3rd ACM Conference on Recommender Systems*, pages 125–132, 2009.

[14] D. Lemire and A. Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *In SIAM Data Mining (SDM'05)*, volume 05, 2005.

[15] R. Little. Missing-data adjustments in large surveys. *Journal of Business & Economic Statistics*, 6(3):287–296, 1988.

[16] H. Ma, I. King, and M. R. Lyu. Effective missing data prediction for collaborative filtering. In *SIGIR 2007*, pages 39–46, New York, New York, USA, 2007. ACM Press.

[17] M. R. Mclaughlin and J. L. Herlocker. A Collaborative Filtering Algorithm and Evaluation Metric that Accurately Model the User Experience. *Proceeding of SIGIR 2004*, 2004.

[18] Y. Ren, G. Li, and W. Zhou. Learning Rating Patterns for Top-N Recommendations. In *Proceeding of The IEEE/ACM International Conference on Social Networks Analysis and Mining (ASONAM 2012)*, 2012.

[19] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens : An Open Architecture for Collaborative Filtering of Netnews. In *ACM Conference on Computer Supported Cooperative Work*, pages 175–186, 1994.

[20] G. K. B. Richard A. Johnsom. *Statistics: Principles and Methods*. John Wiley and Sons, 6 edition, 2009.

[21] D. Rubin. Multiple imputation for nonresponse in surveys. *New York, USA: John Willey & Sons*, 1987.

[22] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[23] X. Su and R. Greiner. A Mixture Imputation-Boosted Collaborative Filter. *Artificial Intelligence*, pages 312–317, 2008.

[24] X. Su and T. Khoshgoftaar. Collaborative Filtering for Multi-class Data Using Belief Nets Algorithms. In *2006 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)*, pages 497–504. Ieee, Nov. 2006.

[25] X. Su, T. Khoshgoftaar, and R. Greiner. Imputed Neighborhood Based Collaborative Filtering. In *Web intelligence 2008*, volume 1, pages 633–639. IEEE, 2008.

[26] X. Su, T. M. Khoshgoftaar, X. Zhu, and R. Greiner. Imputation-boosted collaborative filtering using machine learning classifiers. *Proceedings of the 2008 ACM symposium on Applied computing - SAC '08*, (2):949, 2008.

[27] J. Wang, A. P. de Vries, and M. J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR 2006*, pages 501–208, New York, New York, USA, 2006. ACM Press.

[28] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, and Z. Chen. Scalable collaborative filtering using cluster-based smoothing. In *SIGIR 2005*, pages 114–121, New York, New York, USA, 2005. ACM Press.

[29] J. Zhang, Y. Xiang, Y. Wang, W. Zhou, Y. Xiang, and Y. Guan. Network Traffic Classification Using Correlation information. *IEEE Transaction on Parallel and Distributed Systems*, 2012.