HOKKAIDO UNIVERSITY

Title	Adaptive Fusion Method for User-based and Item-based Collaborative Filtering
Author(s)	Yamashita, Akihiro; Kawamura, Hidenori; Suzuki, Keiji
Citation	Advances in Complex Systems, 14(2), 133-149 https://doi.org/10.1142/S0219525911003001
Issue Date	2011-04
Doc URL	https://hdl.handle.net/2115/45461
Rights	Electronic version of an article published as Advances in Complex Systems, 14(2), 2011, 133-149, 10.1142/S0219525911003001. © copyright World Scientific Publishing Company. http://www.worldscinet.com/acs/acs.shtml
Type	journal article
File Information	ACS14-2_133-149.pdf

Hokkaido University Collection of Scholarly and Academic Papers : HUSCAP

Adaptive Fusion Method for User-based and Item-based Collaborative Filtering

Akihiro Yamashita
Graduate School of Information Science and Technology, Hokkaido University
yama@complex.eng.hokudai.ac.jp

Hidenori Kawamura
Graduate School of Information Science and Technology, Hokkaido University
kawamura@complex.eng.hokudai.ac.jp

Keiji Suzuki
Graduate School of Information Science and Technology, Hokkaido University
suzuki@complex.eng.hokudai.ac.jp

ABSTRACT

In many E-commerce sites, recommender systems, which provide personalized recommendation from among a large number of items, are recently introduced. Collaborative filtering is one of the most successful algorithms which provide recommendations using ratings of users on items. There are two approaches such as user-based and item-based collaborative filtering. Additionally a unifying method for user-based and item-based collaborative filtering was proposed to improve the recommendation accuracy. The unifying approach uses a constant value as a weight parameter to unify both algorithms. However, because the optimal weight for unifying is actually different by the situation, the algorithm should estimate an appropriate weight dynamically, and should use it. In this research, first, we investigated the relationship between recommendation accuracy and the weight parameter. The results show the optimal weight is different depending on the situation. Second, we propose an approach for estimation of the appropriate weight value based on collected ratings. Then, we discussed the effectiveness of the proposed approach based on both multi-agent simulation and MovieLens dataset. The results show that the proposed approach can estimate the weight value within an error rate of 0.5% for the optimal weight.

1. INTRODUCTION

Recommender system predicts users' preference to provide personalized recommendation [6]. Effective recommendation improves usefulness of searching items and provides information of new items on the web. On the other hand, the recommender system also gives advertising effects for the users. Therefore, the system benefits both users and operators. Several websites especially e-commerce sites are already applied such as Amazon.com[1], MovieFinder.com[2], Ebay[3] effectively. Additionally, other applications with recommender system were introduced by Schafer, et al. [8].

User-based and item-based collaborative filtering are the most successful and adopted algorithms to provide recommendations in many systems. However, if the recommender system has few ratings at the beginning of operation for example, the system cannot compute the rating predictions. The problem is commonly known as cold start problem [9].

Therefore, Wang et. al. proposed unifying method for these algorithms by weighted sum with a constant weight parameter, which is represented as λ, in such a way that the system recommends more efficiently when the system has few ratings [11].

For the unifying method, one of the most important challenge is how determine the λ parameter in adapting to the situation. Wang et. al. propose the unifying method by weight parameter λ which is empirically determined as a constant value at all times. Generally, recommendation accuracy of collaborative filtering is affected by the system condition such as input frequency of ratings by users and distribution of users' preference. Thus, user-based collaborative filtering may be relatively more effective than item-based collaborative filtering, or we may have the opposite kind of situation. Consequently, optimal weight parameter λ seems to change according to the situations. Notice that, the system should estimate optimal λ not to adapt previously collected ratings, but to maximize next recommendation accuracy.

Therefore, we propose adaptive unifying method for user-based and item-based collaborative filtering by estimating appropriate weight parameter dynamically. Actually, the weight is determined by simulated recommendation using previously collected ratings. In this paper, we also describe the result of evaluation experiment for proposed adaptive unifying method with both multi-agent simulations based on rating-recommendation interaction model and MovieLens dataset.

2. COLLABORATIVE FILTERING

2.1 User-based Collaborative Filtering

The baseline of user-based collaborative filtering algorithm

[1] http://www.amazon.com/
[2] http://www.moviefinder.com/
[3] http://www.half.ebay.com/

Akihiro Yamashita, Hidenori Kawamura and Keiji Suzuki: Adaptive Fusion Method for User-based and Item-based Collaborative Filtering. In Frank Schweitzer, Akira Namatame, Hideyuki Nakashima, and Satoshi Kurihara (eds.): Proc. of 5th Int. Workshop on Emergent Intelligence on Networked Agents (WEIN 2010) at AAMAS 2010; May, 10, 2010, Toronto, Canada.

was proposed by GroupLens project in 1994, and it is used in some of the earliest collaborative filtering systems [5]. The algorithm remains a popular baseline algorithm of recommender systems today, since it is easy to implement and demonstrates high accuracy.

The algorithm is referred to as "user-based collaborative filtering", since the core of the algorithm is the computation of similarity weight between each pair of users. Figure 1 illustrates this process; here the matrix rows represent users and the columns represent items. Note that user-user similarity is computed based on ratings on items which were rated by both of the pair of users.

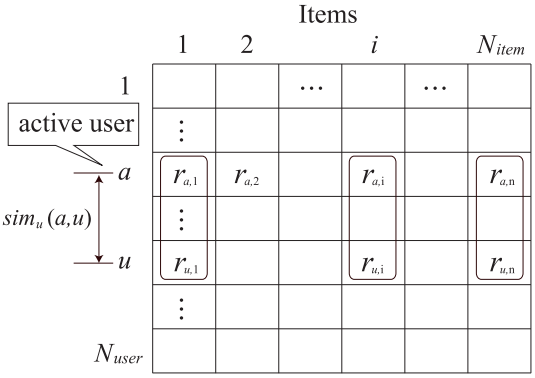


Figure 1: Similarity computation process in user-based collaborative filtering

Therefore, the first step of recommendation in the algorithm is to weight all users with respect to similarity with the active user. Several expressions are proposed as the similarity weight such as Pearson's correlation coefficient , cosine distance [2] and Tanimoto coefficient [10]. Selection of similarity weight expression is one of the most important design factor of collaborative filtering, affecting recommendation accuracy. In this paper, Pearson's correlation coefficient is adopted to conventional approach, since several previous studies conclude the Pearson's correlation coefficient perform better than the others [1]. The similarity is formulated as follows:

$$\mathrm{sim_u}(a,u) = \frac{\sum_{i \in I_a \cap I_u} (r_{a,i} - \overline{r}_a)(r_{u,i} - \overline{r}_u)}{\sqrt{\sum_{i \in I_a \cap I_u} (r_{a,i} - \overline{r}_a)^2} \sqrt{\sum_{i \in I_a \cap I_u} (r_{u,i} - \overline{r}_u)^2}}, \quad (1)$$

I_u is the set of items rated by user u while $r_{u,i}$ is the rating of user u on item i. The rating averages $\overline{r}_u$, $\overline{r}_a$ are taken over the common items rated by both users.

Having assigned similarity weights to users, in the next step, the system determines which other users' ratings will be used in the computation of a prediction for the active user. Formally, the neighbors $S(a) \in U_i$ of active user a are selected based on the similarity weights, where U_i is the set of all users who have rated item i. Selection of neighbors is also important design factor of collaborative filtering. Herlocker, et. al.[3] and Ma, et. al.[4] determined the effect of neighborhoods selection in recommendation accuracy. Typically, $S_u(a)$ is consisted of the top of k users most similar to active user a, i.e. k-Nearlest Neighbor method. Or weight threshold, which sets a minimum correlation weight that a neighbor must have in order to be accepted into the neighbors, are used to consist $S_u(a)$. In this paper, $S_u(a)$ is consisted by top of k_{user} similar users and weight threshold is ξ_{user}, i.e. $S_u(a) = \{u | \mathrm{sim_u}(a,u) \geq \xi_{\mathrm{user}}, \mathrm{rank}(\mathrm{sim_u}(a,u)) \leq k_{\mathrm{user}}\}$.

The ratings of these user neighbors are combined into a rating prediction $\widehat{r}_{a,i}$ as follows:

$$\widehat{r}_{a,i} = \overline{r}_a + \frac{\sum_{u \in U_i \cap S(a)} \mathrm{sim_u}(a,u)(r_{u,i} - \overline{r}_u)}{\sum_{u \in U_i \cap S(a)} \left| \mathrm{sim_u}(a,u) \right|} \quad (2)$$

If $U_i \cap S(a) = \phi$, rating prediction $\widehat{r}_{a,i}$ cannot be calculated. Figure 2 illustrates the rating prediction process based on user similarity. Here, the matrix rows represent users and the matrix columns represent items.

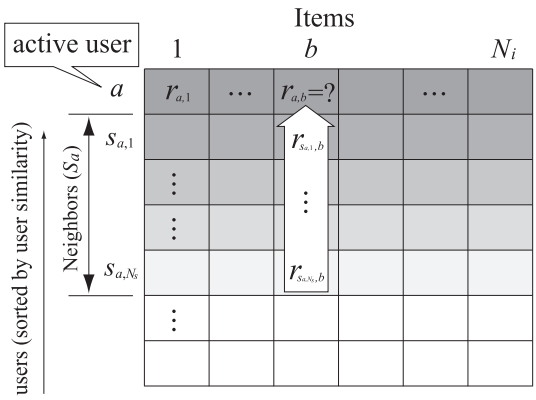


Figure 2: rating prediction based on user-based collaborative filtering

Finally, recommender system recommends the items which has high rating prediction $\widehat{r}_{a,i}$ for active user a.

2.2 Item-based Collaborative Filtering

The user-based collaborative filtering calculates rating prediction based on similarity weight between each pair of users. An alternate approach is to find items rated by the active user that are similar to the item being predicted. Sarwar et al.[7] proposed several different algorithms that used similarities between items, rather than users, to compute rating predictions.

One critical step in the item-based collaborative filtering algorithm is to compute the similarity weight between each pair of items and then to select the most similar items. Figure 3 illustrates this process; here the matrix rows represent users and the columns represent items. Like user-based collaborative filtering, several expression are proposed as the similarity weight such as Pearson's correlation coefficient and cosine distance [7]. In this paper, Pearson's correlation coefficient is adopted to compute item similarity weights formulated as follows:

$$\mathrm{sim_i}(b,i) = \frac{\sum_{u \in U_b \cap U_i} (r_{u,b} - \overline{r}_b)(r_{u,i} - \overline{r}_i)}{\sqrt{\sum_{u \in U_b \cap U_i} (r_{u,b} - \overline{r}_b)^2} \sqrt{\sum_{u \in U_b \cap U_i} (r_{u,i} - \overline{r}_i)^2}}, \quad (3)$$

U_b is the set of users who rated item b and U_i is the set of users who rated item i. The rating averages $\overline{r}_i$, $\overline{r}_b$ are taken over the common users who rated both items.

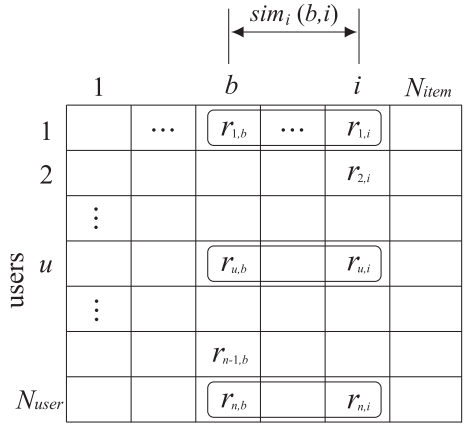


Figure 3: similarity computation in item-based collaborative filtering

As with the user-based collaborative filtering, the next step is determine which other items' ratings will be used in the computation of a prediction for the active user. Formally, the item neighborhoods $S_i(b) \in I_a$ of active user a are selected based on the item similarity weights, where I_a is the set of all items which are rated by active user a. In this paper, $S_i(b)$ is consisted by top of k_{item} similar items and correlation weight threshold is ξ_{item}, i.e. $S_i(b) = \{i|\mathrm{sim_i}(b,i) \geq \xi_{item}, rank(\mathrm{sim_i}(b,i)) \leq k_{item}\}$.

The ratings of these item neighbors are combined into a rating prediction $\widehat{r}_{a,i}$ based on item-based collaborative filtering as follows:

$$\widehat{r}_{a,b} = \frac{\sum_{i \in I_a \cap S_i(b)} \mathrm{sim_i}(b,i)(r_{a,i})}{\sum_{i \in I_a \cap S_i(b)} |\mathrm{sim_i}(b,i)|} \quad (4)$$

If $I_a \cap S_i(b) = \phi$, rating prediction $\widehat{r}_{a,b}$ cannot be calculated. Figure 4 illustrates the prediction process based on the item similarity. Here the matrix rows represent users and the matrix columns represent items.

Finally, recommender system recommends the items which has high rating prediction $\widehat{r}_{a,b}$ for active user a.

2.3 Fusion Method for User-based and Item-based Collaborative Filtering

User-based collaborative filtering calculates linear combination of other user's ratings to predict target rating. Similarly, Item-based collaobrative filtering calculates linear combination of other item's rating to predict target rating. Therefore, linear combination between rating prediction based on User-based and Item-based collaborative filtering is also rating prediction. Wang et.al. propose relative importance value λ to unifying User-based and Item-based collaborative filtering by mathematical form, and it could improve recommendation accuracy [11]. Actually, the rating prediction of the unifying method is computed by weighted sum of rating predictions from user-based and item-based collaborative filtering as follows:

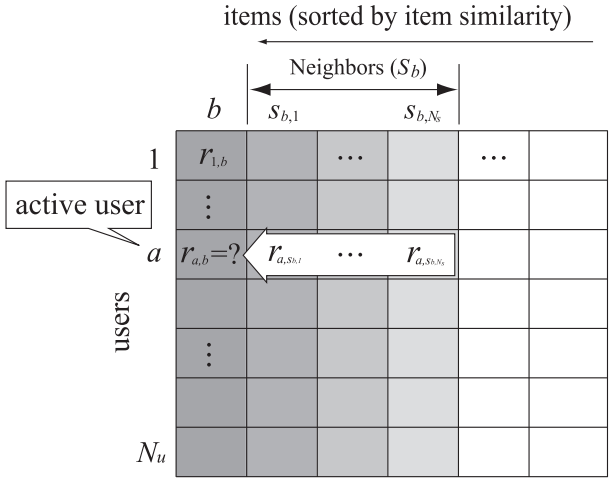


Figure 4: rating prediction based on item-based collaborative filtering

$$\widehat{r}_{a,b} = \lambda f_u(a,b) + (1-\lambda) f_i(a,b) \quad (5)$$

Figure 5 illustrates the unifying process. In this paper, we focus that how the λ, which is the weight parameter of the unifying process, was computed.

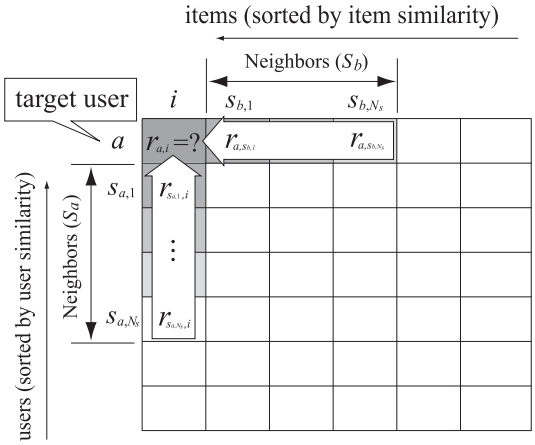


Figure 5: Unifying user-based and item-based collaborative filtering

3. ADAPTIVE FUSION METHOD BASED ON SIMULATED RECOMMENDATION PROCESS

Wang et. al. set λ as empirically determined a constant value at all times. However, the system cannot assign optimal λ as a constant value since the accuracy of next recommendation cannot be computed in advance. On the other hand, recommendation accuracy is affected by the situation such as input frequency of ratings by users, distribution of users' preference and the number of ratings. Therefore, recommendation accuracy increase when the system assigns

appropriate λ sequentially and adaptively by predicting an accuracy of next recommendation. Thus, the λ should be dynamically assigned based on previously collected ratings.

In this paper, we propose the computation method for the λ based on previously collected ratings. Particularly, the system extract one rating randomly from whole collected ratings. Then, the system calculates the rating prediction of the extracted ratings and calculates MAE (Mean Absolute Error) by the error of the prediction. Let us call this operation as simulated recommendation. The recommender system repeats the simulated recommendation where λ is assigned as several values. Then, the value which provide lowest MAE is adopted as λ. The flowchart of the proposed method is illustrated as figure 6.

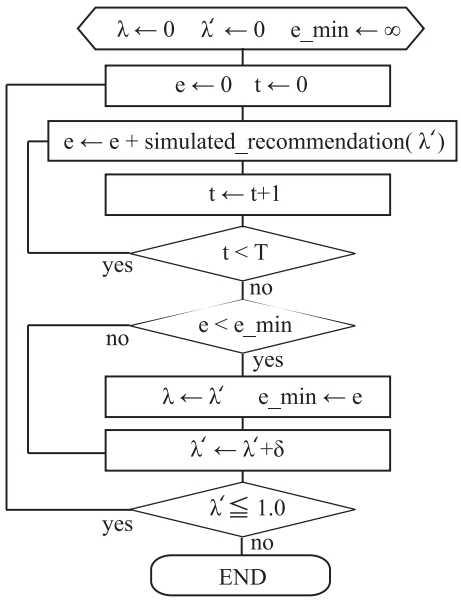


Figure 6: Computation procedure of λ based on simulated recommendation process

Notice that, there are several parameter such as the number of simulated recommendation represented as T and the step size of λ represented as δ. These parameter have to be assigned with consideration for trade-off between the adaptability and the time complexity.

4. EVALUATION MODEL BASED ON HUMAN-SYSTEM INTERACTION

The effectiveness of recommender systems improves as a result of interaction between the system and its users, i.e., the usage of collected ratings as user feedback. Therefore, certain real dataset is just one example which constructed based on the system-user interaction. In this paper, we use not only real dataset but also agent-based model to investigate the effectiveness of our proposed approach. In this section, the simulation model, which is named human-system interaction model, based on agent modeling is explained.

4.1 Model Components

The rating-recommendation interaction model has three components: agents (correspond to users), items and recommender system. This model can be formulated as follows: Let $U = \{u|u = 1, 2, \ldots, N_{user}\}$ be the set of all agents, let $I = \{i|i = 1, 2, \ldots, N_{item}\}$ be the set of all items. The preference of each agent u is represented by the vector $\mathbf{p_u} = (p_{u,1}, p_{u,2}, \ldots, p_{u,d})$, and the feature of each product i is represented by the vector $\mathbf{v_i} = (v_{i,1}, v_{i,2}, \ldots, v_{i,d})$, where d is the dimension of $\mathbf{p_u}$ and $\mathbf{v_i}$. The range of each elements in the preference vector $\mathbf{p_u}$ and feature vector $\mathbf{v_i}$ are $[-1, 1]$.

When an agent u is recommended an item i, the agent u computes rating $r_{u,i}$ according to the distance between the preference vector $\mathbf{p_u}$ and the feature vector $\mathbf{v_j}$. Then the agent u inputs rating $r_{u,i}$ to recommender system. The ratings can be represented as agent-item metrix in the recommender system, which is named rating matrix in this paper.

4.2 Distribution of the Preference Vector $\mathbf{p_u}$

In this model, each element of the vector $\mathbf{p_u}$ is set as random numbers according to the following distribution. Three types of the distribution were applied to investigate an effect of the recommendation by the difference of the distribution.

4.2.1 Uniform Distribution

Uniform distribution represents a situation without trend of the preference. $\mathbf{p_i}$ is set as random numbers according to the uniform distribution, and the range is $[-1, 1]$.

4.2.2 Multivariate Normal Distribution

Multivariate Normal distribution represents a situation where the trend exists. When the dimension of vector is d, random vector x from the d-dimensional multivariate normal distribution with mean vector μ and covariance matrix Σ is generated as follows: first, compute the cholesky decomposition of Σ, that is, find the unique lower triangular matrix L such that $LL^T = \Sigma$. then, Let $\mathbf{z} = (z_1, z_2, \ldots, z_d)$ be a vector whose components are d independent standard normal variates. x is computed by the product of L and $\mathbf{z}$ as follows:

$$\mathbf{x} = L\mathbf{z} + \mu \sim N(\mu, \Sigma), \tag{6}$$

In this paper, $\mu = \mathbf{0}$ and Σ is set as diagonal metrix as expression (7) for simplicity. Particularly, the trend is strong when the variance v in Σ is small in the multivariate normal distribution.

$$\Sigma_v = \begin{cases} \sigma_{ij} = v & \text{where } i = j \\ \sigma_{ij} = 0 & \text{otherwise} \end{cases}. \tag{7}$$

4.2.3 Two-peak Distribution

Two-peaks distribution is obtained by sum of two normal distributions: $N(\mu = -0.5, \sigma^2 = 0.2)$ and $N(\mu = 0.5, \sigma = 0.2)$. This distribution represents a situation that the users form several group by their preference. The number of groups is calculated by 2^d where d is the dimension of the vector $\mathbf{p_i}$, because random numbers are independently given to each element.

4.3 Distribution of the Feature Vector $\mathbf{v_i}$

In this paper, various items are provided as the recommendation targets, for investigating effects of user preference on that recommendation performance. If there is bias in the distribution of feature vectors $\mathbf{v_i}$, it also effects the recommendation result. However, in these experiments, we focus on the effect of the distribution of preference vector. Thus, we eliminate the effect of the distribution of $\mathbf{v_i}$ on

the recommendation result by providing the feature vector $\mathbf{v_i}$ at random. More formally, the vector $\mathbf{v_i}$ is set as vectors according to the uniform distribution of range$[-1, 1]$.

4.4 Rating

In this model, let R be the set of all ratings by agents, and rating $r_{u,i} \in R$ represents the rating of agent $u \in U$ on item $i \in I$. $r_{u,i}$ is computed based on the distance of $\mathbf{p_u}$ and $\mathbf{v_i}$. In this model, every agent computes ratings based on a common rating function for simplicity, i.e. all agents rate for items without biasing to a low or high rating.

In this paper, five-grade rating was used in this model. Thus, the rating function has four thresholds. The thresholds were set where each grade occurs same frequency if the distribution of $\mathbf{p_u}$ is generated based on uniform distribution. Note that, if $\mathbf{p_u}$ is generated based on the other distribution, each grade occurs different frequency.

The parameter d, which is the dimension of vector $\mathbf{p_i}$ and vector $\mathbf{v_j}$, affects a computation of distance between $\mathbf{p_u}$ and $\mathbf{v_i}$. Then, the thresholds should be assigned each dimension d respectively, however, every experiments in this paper was performed where $d = 5$.

5. EXPERIMENT AND DISCUSSION BASED ON MULTI-AGENT BASED SIMULATION

5.1 Experimental Settings

In this section, several experiments are performed to evaluate the effectiveness of proposed unifying method based on rating-recommendation interaction model described in section 4. Generally, sequence of ratings and variance of rating frequency among users or items affect recommendation accuracy. First of all, selection process is defined to determine next active agent. Every agent was assigned its own score which is generated based on normal random number where average is 0.5, variance is σ^2 at the beginning of a simulation. Then, the system select next active agent by roulette-wheel selection based on the score. The selected agent (i.e. active agent) rates one item randomly. Notice that, the variance σ^2 was assigned as three different values as follows:

- $\sigma^2 = 0$ (Every agent is selected at same possibility)
- $\sigma^2 = 0.1$ (The possibility is assigned based on normal distribution)
- $\sigma^2 = \infty$ (The possibility is assigned based on uniform distribution)

In this section, the experiments were performed based on rating-recommendation interaction model. Table. 1 shows the settings for the experiments.

We use the metric of recommendation accuracy as MAE. Therefore, the optimal λ represented as λ_{opt} is defined as the value which provide the lowest MAE. In the first experiment described in section 5.2, an λ_{opt} was computed at each setting illustrated in table 1. Then, in the second experiment described in section 5.3, the λ which was obtained by the proposed method described in section 3 is compared with λ_{opt}, moreover, the availability and the performance are discussed.

5.2 Relationship between MAE and λ

First, the λ_{opt} at each experimental settings was computed. The total number of ratings is 25000 since the number of agent $N_u = 500$ and the number of items $N_i = 500$. In this experiment, the λ_{opt} was computed when the recommender system was obtained 5%, 10%, 15% ratings each. Figure 7 illustrates the experimental results at $\sigma^2 = 0$, $\sigma^2 = 0.1$ and $\sigma^2 = \infty$. In those figures, the horizontal axis denotes λ, and the vertical axis denotes MAE. Notice that, the MAE at $\lambda = 0$ represents the result based on item-based collaborative filtering only, meanwhile, the MAE at $\lambda = 1$ represents the results based on user-based collaborative filtering contrastively. Obviously, λ which provides the lowest MAE is λ_{opt}.

In all figures, MAE decreased according to increase the number of ratings. Moreover, the unifying method of user-based and item-based collaborative filtering can provide lower MAE than stand-alone algorithms. Let us focus about λ_{opt}. For example, λ_{opt} is less than 0.5 when $\mathbf{p_u}$ was generated based on normal distribution according to Figure 7(b)(e)(h). On the other hand, λ_{opt} is more than 0.5 when $\mathbf{p_u}$ was generated based on two-peak distribution according to Figure 7(c)(f)(i). Therefore, λ_{opt} is different depending on experimental settings.

5.3 Experimental Validation and Discussion

In this section, the λ provided based on the proposed method described in section 3 is compared with the λ_{opt} shown in section 5.2. Table 2 denotes averages and variances of λ provided based on the proposed method in 20 trials.

For instance, the average of λ provided based on the proposed method increases from 0.455 (when the system was obtained 5% of ratings) to 0.700 (when the system was obtained 15% of ratings) in settings that $\sigma^2 = 0$ and $\mathbf{p_u}$ was generated based on two-peak distribution. By compare the results with Figure 7(c), the proposed method can provide λ close to λ_{opt}.

In contrast, the average of λ provided based on the proposed method decreases from 0.470 (when the system was obtained 5% of ratings) to 0.275 (when the system was obtained 15% of ratings) in settings that $\sigma^2 = 0.1$ $\mathbf{p_u}$ was generated based on multivariate normal distribution. Similarly, λ_{opt} illustrated in Figure 7(e) decreases according to the number of ratings increase. Likewise, the proposed method provides λ close to λ_{opt} in any experimental settings.

On the other hand, a variance of λ is about more than 0.1 in some settings. Actually, minimization of MAE is important instead of estimation of λ. Hence, error rate represented as ϵ between the MAE by λ provided based on the proposed method and the MAE by λ_{opt} was calcurated as expression (8). Table 3 denotes averages and a variances of the error rate ϵ.

$$\epsilon = \frac{\text{MAE of the proposed method} - \text{MAE of optimal } \lambda}{\text{MAE of optimal } \lambda} \quad (8)$$

In most settings, the error rate ϵ was less than 1%, however, the ϵ is relatively large when $\mathbf{p_u}$ is generated based on two-peak distribution. The cause is assumed that the number of simulated recommendation represented as T was insufficient. For this reason, the number of simulated recommendation should be assigned depending on the situation.

6. EXPERIMENT AND DISCUSSION BASED ON MOVIELENS DATASET

Table 1: Settings for experiments based on multi-agent based simulation

the number of agents N_u	500
the number of items N_i	500
The distribution of $\mathbf{p_u}$	uniform distribution
	multivariate normal distribution
	two-peak distribution
the distribution of $\mathbf{v_i}$	uniform distribution
selection probability of next active agent	$\sigma^2 = 0$ (same probability)
	$\sigma^2 = 0.1$(based on normal distribution)
	$\sigma^2 = \infty$(based on uniform distribution)
the number of simulated recommendations T	50
δ (the step size of λ)	0.1

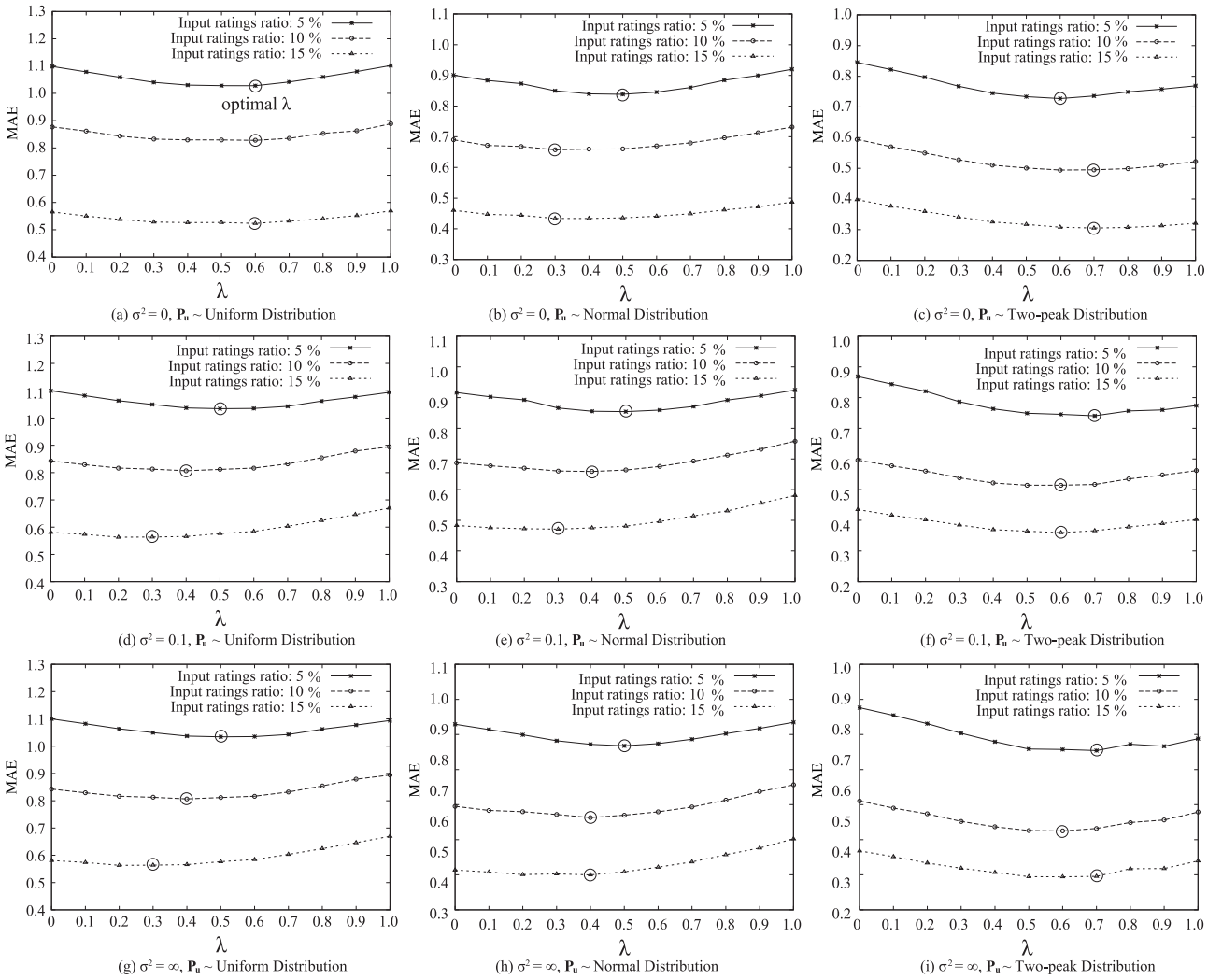


Figure 7: Relationship between MAE and λ at $\sigma^2 = \{0, 0.1, \infty\}$, $\mathbf{p_i}$ is generated based on uniform distribution (a)(d)(g), normal distribution (b)(e)(h) and two-peak distribution (c)(f)(i)

Table 2: Average and standard deviation of λ obtained using proposed method (20 trial runs)

	dist. of $\mathbf{p_u}$	5%		10%		15%	
		Ave.	SD	Ave.	SD	Ave.	SD
$\sigma^2 = 0$	uniform dist.	0.540	0.124	0.465	0.079	0.480	0.125
	multivariate normal dist.	0.500	0.077	0.445	0.116	0.425	0.134
	2-peak dist.	0.455	0.140	0.595	0.107	0.700	0.114
$\sigma^2 = 0.1$	uniform dist.	0.450	0.107	0.395	0.092	0.305	0.128
	multivariate normal dist.	0.470	0.119	0.350	0.107	0.275	0.089
	2-peak dist.	0.530	0.158	0.560	0.097	0.540	0.102
$\sigma^2 = \infty$	uniform dist.	0.550	0.107	0.445	0.107	0.365	0.101
	multivariate normal dist.	0.475	0.094	0.365	0.096	0.290	0.099
	2-peak dist.	0.595	0.102	0.575	0.062	0.560	0.102

Table 3: Average and standard deviation of error rate ϵ between the MAE by λ provided based on the proposed method and the MAE by λ_{opt}

	dist. of $\mathbf{p_u}$	5 %		10 %		15 %	
		Ave. [%]	SD	Ave. [%]	SD	Ave. [%]	SD
$\sigma^2 = 0$	uniform dist.	0.42335	1.12985	0.20293	0.13270	0.58782	0.46487
	multivariate normal dist.	0.30580	0.65538	0.71956	0.83600	0.69785	0.91000
	2-peak dist.	2.52720	3.35148	0.75057	1.48146	0.93756	1.46451
$\sigma^2 = 0.1$	uniform dist.	0.39421	0.69981	0.50005	0.35279	0.73569	1.07296
	multivariate normal dist.	0.61515	1.10575	0.54009	0.79992	0.38074	0.34992
	2-peak dist.	2.34030	2.94543	0.30855	0.49440	1.45296	0.88352
$\sigma^2 = \infty$	uniform dist.	0.54415	0.46722	0.62333	0.50215	0.40684	0.73066
	multivariate normal dist.	0.47115	0.63513	0.91722	0.87394	0.36984	0.47256
	2-peak dist.	0.45991	0.34179	0.24342	0.56416	0.70325	1.22513

The experiments and the results based on MovieLens dataset is described in this section. MovieLnes is recommender system of movies. The ratings, which is commonly used as a benchmark data for recommender system, are disclosed on the web [4].

The dataset contains 100,000 five-grade ratings provided by 943 users on 1682 items. Each user rated at least 20 items.

First, the ratings were divided into a training dataset R_{train} and a test dataset R_{test}. The recommender system computes the predictions of the ratings in R_{test} based on R_{train}. Then, MAE, which is an average of error between the prediction and the actual value of the ratings, were computed. In this paper, the experiments were performed in the settings that the number of R_{train} is the range from 5000 (5% of whole ratings) to 40000 (40 % of whole ratings).

6.1 Relationship between MAE and λ

First of all, λ_{opt} was calculated when the system obtained the R_{train} by computing MAE based on R_{test}. The experiments were performed where $\delta = 0.05$, which is the step size of λ, R_{test} and R_{train} were selected at random. Figure 8 illustrates the results (average of 60 trials).

In Figure 8, MAE = 1 constantly with any λ where $|R_{\text{train}}|=$ 5000. The MAE at $|R_{\text{train}}|=5000$ is similar to random recommendation, however, MAE is about 0.75 where $|R_{\text{train}}|=$ 40000. Additionally, λ_{opt} is shown close to 0.5, however, it is important to note that the λ_{opt} depends on the situation described in section 5.2.

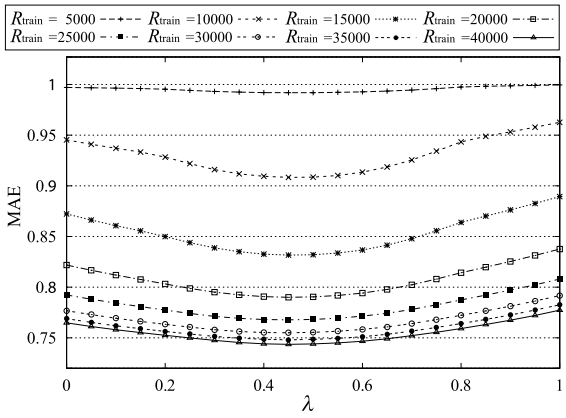


Figure 8: Relationship between MAE and λ obtained using Movie Lens Dataset

6.2 Experimental Validation and Discussion

The λ provided by the proposed method is validated by the expression (8) based on the results described in section 6.1. Figure 9 illustrates an error rates by λ in the settings that the number of R_{train} = { 5000, 10000, 15000, 20000, 25000, 30000, 35000, 40000} and the number of simulated recommendation $T = \{100, 500, 1000\}$.

[4] http://www.grouplens.org/node/73#attachments

In Figure 9, horizontal axis denotes T and vertical axis denotes error rate ϵ. The results show ϵ decrease regardless of the number of training dataset R_{train}. Note that the ϵ is low where $|R_{\text{train}}| = 5000$ since MAE is approximately constant value due to deficiency of training dataset accorting to Figure 8. ϵ is about less than 0.005 where the number of simulated recommendation $T = 1000$. Thus, in this experiment using real dataset, the proposed unigying method can compute λ with an error rate $\epsilon = 0.005$ if the system can run 1000 times simulated recommendation. For practical application, the number of simulated recommendation should be assigned depending on the situation or time complexity.

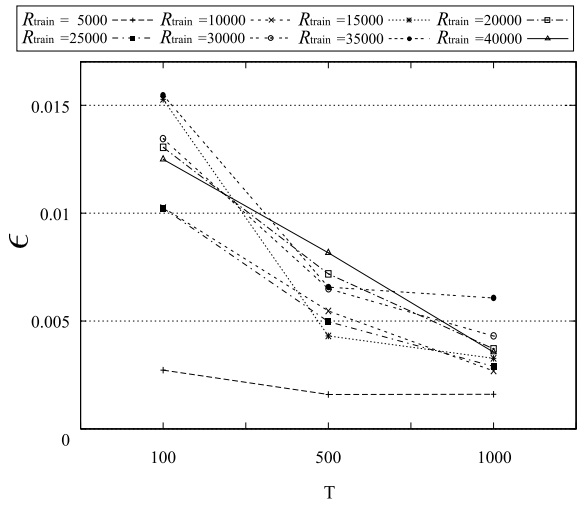


Figure 9: Experimental results based on agent-based simulation

Figure 9 shows that ϵ was about less than 0.005 when the number of simulated recommendation $T = 1000$. Thus, in this experiment, the proposed unigying method can compute λ with an error rate $\epsilon = 0.005$ if the system can run 1000 times simulated recommendation.

7. CONCLUSION

Wang et. al proposed unifying method for user-based and item-based collaborative filtering by computing weighted sum of two rating predictions with weight parameter λ. In this research, the experimental results show the optimal value of weight parameter λ appears to change under different circumstances, because recommendation accuracy is affected by system factors such as the input frequency of ratings by users and the distribution of user preferences. Then we proposed adaptive unifying method for user-based and item-based collaborative filtering by doing simulated recommendation process based on previously collected ratings. Moreover, the experimental validation was performed to evaluate the effectiveness based on both of agent-based simulation and MovieLens dataset. The result of evaluation experiment shows that the proposed method can estimate weight parameter λ with an error rate, which denotes the recommendation accuracy, about less than $\epsilon = 0.5\%$ to unify these collaborative filtering. Note that computational cost is needed when the system predicts the optimal rambda by simulated recommendation process. However, it not really matter in most cases, because the process can run in the background when the system not busy and it need not be real-time. Therefore, the recommender system should optimize weight parameter dynamically when it uses the unifying algorithm.

8. REFERENCES

[1] J. S. Breese et al. Empirical analysis of predictive algorithms for collaborative filtering. In Proc. of the 14th Conference on UAI'98, pages 43–52, 1998.

[2] E. Garcia. Cosine similarity and term weight tutorial. http://www.miislita.com/information-retrieval-tutorial/cosine-similarity-tutorial.html#Cosim, 2006.

[3] J. L. Herlocker et al. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Information Retrieval, 5(4):287–310, 2002.

[4] H. Ma et al. Effective missing data prediction for collaborative filtering. In Proc. of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 39–46, 2007.

[5] P. Resnick et al. Grouplens: An open architecture for collaborative filtering of netnews. In Proc. of the CSCW '94, ACM, pages 175–186, 1994.

[6] P. Resnick and H. Varian. Recommender systems. Comm. of the ACM, 40(3):56–58, 1997.

[7] B. Sarwar et al. Item-based collaborative filtering recommendation algorithms. In Proc. of WWW '01, pages 285–295, 2001.

[8] J. B. Schafer et al. E-commerce recommendation applications. Data Mining and Knowledge Discovery, 5(1-2):115–153, 2001.

[9] A. I. Schein et al. Methods and metrics for cold-start recommendations. In Proc. of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 253–260, 2002.

[10] T. Segaran. Programming Collective Intelligence: Building Smart Web 2.0 Applications. O'reilly, 2007.

[11] J. Wang et al. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In Proc. of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 501–508, 2006.