Nantonac Collaborative Filtering: Recommendation Based on Order Responses

Toshihiro Kamishima
National Institute of Advanced Industrial Science and Technology (AIST)
AIST Tsukuba Central 2, Umezono 1-1-1, Tsukuba, Ibaraki, 305-8568 Japan
mail@kamishima.net (http://www.kamishima.net/)

ABSTRACT

A recommender system suggests the items expected to be preferred by the users. Recommender systems use collaborative filtering to recommend items by summarizing the preferences of people who have tendencies simliar to the user preference. Traditionally, the degree of preference is represented by a scale, for example, one that ranges from one to five. This type of measuring technique is called the semantic differential (SD) method. We adopted the ranking method, however, rather than the SD method, since the SD method is intrinsically not suited for representing individual preferences. In the ranking method, the preferences are represented by orders, which are sorted item sequences according to the users' preferences. We here propose some methods to recommend items based on these order responses, and carry out the comparison experiments of these methods.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Clustering, Information filtering*

Keywords

Order, Recommender System, Collaborative Filtering

1. INTRODUCTION

Recommender systems suggest the items expected to be preferred by the users [14]. These systems are playing an increasingly important role in business [16]. Collaborative filtering (*CF* for short) is one of the methods used to implement recommender systems. CF works according to this framework: First, the user inputs his/her preference patterns to the system. The system then searches its database for other people whose preferences are similar to those of the user. Next, to the user, the system recommends the items that those people prefer.

Though many methods of CF have been studied, almost all of these methods adopt the semantic differential (SD) method [12] to measure the users' preferences. In this method, the user's preferences are measured by a scale, the extremes of which are symbolized by antonymous adjectives and the divisions of which are cardinal numbers. An example of this type of scale is :

prefer `[ 5 4 3 2 1 ]` **not prefer**

A user represents the degree of his/her preference by choosing a number from 1 to 5. As will be explained in the next section, this measurement is intrinsically not fit for use to evaluate individual preferences.

We therefore advocate a CF framework which adopts the ranking method. In this method, users' preferences are represented by *orders*. An order is an item sequence which is sorted according to the user's preference. The system presents some items to a user, and receives its response of orders sorted according to the user's preferences. We will present several CF methods that can deal with these order responses. We will experimentally show how our framework is superior to the traditional one.

We would like to point out that the word *nantonac* is originates from a Japanese word, *nantonaku*, which means "unable to explain specifically, but I think such and such is the case." Since order responses allow a more vague and intuitive expression of users' preferences, we decided to call this type of filtering method *nantonac filtering*.

We show the framework of the traditional CF method and our new framework in Section 2. In Section 3, we propose some CF methods that can be used within this framework. The experimental results are presented in Section 4, and the summary in Section 5.

1.1 Merits of Using Order Responses

There are several reasons why the SD method is not appropriate for CF. Before explaining why, it is necessary to look at a notion in Stevens' *Scales of Measurement* [17]. Stevens classified scales into four levels: nominal, ordinal, interval, and ratio. Among these four levels, we discuss the ordinal and interval scales. It is possible to more or less determinate the values of the ordinal scales. For example, two values of the ordinal scales, 2 and 3, are given. An interpretation that "2 is smaller than 3" is appropriate, but that "3 is 1.5 times as much as 2" is inappropriate. On the other hand, the values of interval scales are allowed to determine the equality of the intervals. For example, three values of interval scales, 2, 3, and 4, are given. One proper determination is that "the difference between 2 and 4 is twice as much as that between 3 and 4."

Let's return to a discussion of the Semantic Differential

(SD) method [12]. By applying the SD method, we can get observations represented by ordinal scales. For example, a user rates the items A and B at 4 and 2, respectively. The proper interpretation is that the user prefers item A to B. It should not be interpreted that the user prefers A twice as much as B. However, due to the lack of an analysis technique, these scores are of necessity treated as interval scales, by introducing the following two unrealistic assumptions: *the divisions of scales are equivalent* and *all users share an understanding of scale extremes.* Why has the SD method been effectively used? The SD method should be intrinsically used for measuring the concepts of a group of users. For this purpose, even though intervals or extremes of scales are diverged, this divergence is lessened by averaging over users. However, in CF the system has to deal with individual preferences, and because of this, the divergence seriously influences the precision of preference prediction. If the users' responses deviate from the standard scale, the similarities to other people's responses are incorrectly identified, and the wrong items are then recommended.

Another drawback of the SD method is the so-called *Central Tendency* effects [12], which are well-known in experimental psychology literature. This phenomenon describes a tendency to use only the near neutral portion of the rating scale. Such rater effects also confuse the identification of preference similarity.

1.2 Related Works

There have been many studies published on CF [1, 9, 10, 11, 13]. Breese et al.'s work [2] is an empirical comparison survey in this research area. However, all of these methods employed the SD method to measure users' preferences.

Recently, there has been active research on the processing of orders. Cohen et al. [3] and Joachims [4] proposed a method to sort attributed items associated with pairwise precedence information. Kamishima and Akaho [5] and Kazawa et al. [7] studied the learning problem from ordered item sets. Sai et al. [15] proposed association rules between order variables.

2. COLLABORATIVE FILTERING

We will now describe the traditional CF framework, and compare it with our new one.

2.1 Traditional Collaborative Filtering

We will first describe the framework of CF developed as part of the Riedl et al.'s GroupLens project [13]. CF is a task to predict the preferences of a particular user (an active user) based on the preference data collected on other users (a user database). Formally, the task is defined as follows: Let s_{ij} be the score of item j by the user i. The score represents the preference of the user, and takes one of the values from, for example, $\{1,2,3,4,5\}$. $X_i = \{x^1,\dots,x^{|X_i|}\}$ denotes a set of items the user i rated and $S_i{=}\{s_{ij}|x^j{\in}X_i\}$. $X^*{=}\{x^1,\dots,x^{|X^*|}\}$ denotes the set of all items. Note that $|X|$ is the size of a set X. The user database, $D_S = \{S_1,\dots,S_{|D_S|}\}$, is a set of all S_i. We call the users in the database sample users. Let S_a be the set of scores rated by the active user, and X_a be the set of items the active user has already rated. Given the S_a and the D_S, the CF task is estimating the items that the active user is expected to rate high. Such items are then recommended to the active user.

The estimation method of the GroupLens works as follows: First, the similarity between the active user and the sample user i in D_S is measured by the Pearson correlation:

$$r_{ai}{=}\frac{\sum_{j\in I(X_{ai})}\bigl(s_{aj}-\bar{s}_a\bigr)\bigl(s_{ij}-\bar{s}_i\bigr)}{\sqrt{\sum_{j\in I(X_{ai})}\bigl(s_{aj}-\bar{s}_a\bigr)^2}\sqrt{\sum_{j\in I(X_{ai})}\bigl(s_{ij}-\bar{s}_i\bigr)^2}},$$

where $X_{ai}{=}X_a{\cap}X_i$ and $I(X){=}\{j|x^j{\in}X\}$. $\bar{s}_i$ is the mean score of user i for the items in X_{ai}.

The expected score of the item j for the active user is

$$\hat{s}_{aj}{=}\tilde{s}_a + \frac{\sum_{i\in\tilde{X}_j} r_{ai}\bigl(s_{ij}-\bar{s}_i\bigr)}{\sum_{i\in\tilde{X}_j}|r_{ai}|}, \tag{1}$$

where $\tilde{s}_a{=}|S_a|^{-1}\sum_{s_{aj}\in S_a} s_{aj}$ and $\tilde{X}_j{=}\{i|S_i{\in}D_S \text{ s.t. } s_{ij}{\in}S_i\}$. The system recommends the items for which the expected score $\hat{s}_{aj}$ is high.

This algorithm is simple, but very effective. In survey [2], it performed the best for the data set in which the items were rated on a 6-level scale. A Bayesian network performs better than this algorithm if the rating scale is binary (prefer or not). Because a multi-level scale was used in our experiment, we chose this algorithm for comparison.

2.2 Nantonac Collaborative Filtering

We will next describe our new framework: *nantonac collaborative filtering*. This framework is the same as the traditional one described above, except for the representation of the users' preferences. Instead of using the set of scores acquired by the SD method, we adopted the order acquired by the ranking method. The system shows a set of items, X_i, to the user i, and the user sorts these items according to his/her preferences. The sorted sequences are denoted by $O_i{=}x^1{\succ}x^2{\succ}\cdots{\succ}x^{|X_i|}$. This indicates that, for example, the user i prefers item x^1 to item x^2. The rank, $r(O_i,x^j)$, is the cardinal number that indicates the position of the item x^j in the order O_i. For example, for the order $O_i{=}x^1{\succ}x^3{\succ}x^2$, $r(O_i,x^1){=}1$ and $r(O_i,x^2){=}3$. In our framework, the user database is a set of orders sorted by all the users, $D_S{=}\{O_1,\dots,O_{|D_S|}\}$. Let X_a be a set of items sorted by the active user, and O_a be the sorted order. Given the O_a and D_S, the task of nantonac CF is estimating the items that the active user is expected to prefer.

3. METHODS

We would like to propose some CF methods for the framework described in Section 2.2. In survey [2], the filtering methods are categorized: a *memory-based method* and a *model-based method*. To estimate the users' preferences, memory-based methods use an entire user database. In model-based methods, a user database is used for the learning model for prediction. In addition to these methods, we also propose a method which is a hybrid of these two methods. Below, we will show the modified version of the traditional methods, which has been adjusted to deal with orders.

3.1 Memory-Based Methods

In [2], Breese et al. described two basic methods for measuring similarities between two users by a correlation and a vector similarity, and several extensions of these methods. We will describe and present a modified version of these methods.

3.1.1 A Simple Correlation Method

In order to carry out CF, it is necessary to measure the similarities of preferences between the active and the sample users. In this method, we treat given ranks as scores of the GroupLens's method. That is to say, the similarities between the users are

$$R_{ai}=\frac{\sum_{x^j\in X_{ai}}\bigl(r(O_a,x^j)-\bar{r}_a\bigr)\bigl(r(O_i,x^j)-\bar{r}_i\bigr)}{\sqrt{\sum_{x^j\in X_{ai}}\bigl(r(O_a,x^j)-\bar{r}_a\bigr)^2}\sqrt{\sum_{x^j\in X_{ai}}\bigl(r(O_i,x^j)-\bar{r}_i\bigr)^2}}, \quad (2)$$

where $X_{ai}{=}X_a{\cap}X_i$ and $\bar{r}_i{=}|X_{ai}|^{-1}\sum_{x^j\in X_{ai}} r(O_i,x^j)$. Note that the items not contained in the other order are ignored, but the ranks are not renumbered. For example, from the order $O_a = x^1{\succ}x^2{\succ}x^3$, the item x^2 is eliminated, but the rank of x^3 remains $r(O_a,x^3)$=3. The system estimates preferences for the item j by the active user by the function:

$$\tilde{r}_a+\frac{\sum_{i\in\tilde{X}_j} R_{ai}\bigl(r(O_i,x^j)-\bar{r}_i\bigr)}{\sum_{i\in\tilde{X}_j}|R_{ai}|}, \quad (3)$$

where $\tilde{r}_a{=}|X_a|^{-1}\sum_{x^j\in X_a} r(O_a,x^j)$, $\tilde{X}_j{=}\{i|O_i{\in}D_S \text{ s.t. } x^j{\in}X_i\}$. The items are sorted in ascending order of these estimated preferences, and the highly ranked items are recommended.

3.1.2 Default Voting

In [2], three extensions are proposed. Though their experimental results show that the inverse user frequency extension is effective, we could not impose this idea onto our framework. Case amplification extension is not effective so much, and adopts the subjective parameter tuning; thus we didn't implement this. We imposed only the idea of default voting to our framework.

The idea of default voting is designed for a situation in which relatively few items are evaluated. According to a decrease of $|X_i|$ relatively to $|X^*|$, the frequency of the event, $X_a{\cap}X_i{=}\emptyset$, increases. Since the R_{ai} is always 0 in a case in which no commonly evaluated items exist, the similarities between users can no longer be precisely measured. When deriving similarities between two orders O_a and O_i, if we assign *default ranks* to the items either of the user i or the active user evaluated, then the similarities can be calculated over $X_a \cup X_i$. Similar to [2], we give neutral preferences to these items. The items that are not in one order are inserted into the middle of the order. For example, the responses of the active user the user i are $O_a = x^1{\succ}x^3{\succ}x^6$ and $O_i = x^5{\succ}x^3{\succ}x^2{\succ}x^6$, respectively. The items evaluated only by the other user is inserted: $O'_a = x^1{\succ}x^3{\succ}x^2{\sim}x^5{\succ}x^6$ and $O'_i = x^5{\succ}x^3{\succ}x^1{\succ}x^2{\succ}x^6$ ($\sim$ denotes the tie in rank). To all tied items, we give the same rank, which is the mean of these ranks. For example, the items x^5 and x^2 which are at the 3rd and 4th positions in the order O'_a, are tied, thus The rank of these items are 3.5. The similarities between O'_a and O'_i are calculated by Equation (2). The expected preference is derived by Equation (3), except that $r(O'_i,x^j)$ is used instead of $r(O_i,x^j)$. The other procedures are the same as those described in Section 3.1.1.

3.2 Model-Based Methods

In [2], the two model-based method is provided: a cluster model and a Bayesian Network model. Since we do not know the Bayesian Network designed for dealing with orders, only the cluster model was examined.

```
Algorithm k-o'means(S, k, maxIter)
$S = \{O_1,\ldots,O_{|S|}\}$: a set of orders
$k$: the number of clusters
$maxIter$: the limit of iteration times
1) $S$ is randomly partitioned into a set of clusters
   $\pi = \{C_1,\ldots,C_k\}$, $\pi' := \pi$, $t := 0$.
2) $t := t+1$, if $t > maxIter$ goto step 6.
3) for each cluster $C_j \in \pi$, derive the order means $\bar{O}_j$
   by the procedure in Section 3.2.
4) for each order $O_i$ in $S$, assign it to the cluster:
   $\arg\min_{C_j} d(\bar{O}_j, O_i)$.
5) if $\pi = \pi'$ then goto step 6
        else $\pi' := \pi$, goto step 2.
6) output $\pi$.
```

Figure 1: The k-o'means algorithm

The k-o'means [6] is the algorithm for clustering orders. We here simply describe this algorithm in Figure 1. The k-o'means is the same as the well-known k-means algorithm, except for the notions of a dissimilarity and a mean. We define a dissimilarity based on the ρ between two orders as $d(O_i,O_a) = 1-\rho$. The ρ denotes a widely used similarity of orders, the *Spearman's Rank Correlation* [8]. For the two orders O_i and O_a consisting of the same item set (i.e., $X_i{=}X_a$), the ρ between the two orders is equal to the Equation (2). The ρ becomes 1 if the two orders are coincident, and -1 if one order is a reverse of the other order. If $X_i{\neq}X_a$, the items not contained in the other order are again eliminated, but the ranks are renumbered. For example, if the item x^3 is eliminated from the order $x^1{\succ}x^3{\succ}x^4{\succ}x^6$, the rank of the item x^6 changes from 4 to 3. Note that if no common items exist between the two orders, the $\rho = 0$ (i.e., no correlation).

As a mean, we used the following notion of the *order mean*, that is a representative of given orders. For a set of orders, C, the order mean is defined as

$$\bar{O}_C = \arg\max_{O_j}\sum_{O_i\in C}\rho_{ij}. \quad (4)$$

Unfortunately, the method to derive the optimal solution of Equation (4) could not be developed. Instead, the following method based on Thurstone's paired comparison, which gives a good performance empirically. First, the probability $\Pr[x^a{\succ}x^b]$ is estimated. From the order $O_i \in C$, all the item pairs, (x^a,x^b), are extracted such that x^a precedes x^b in the order. For example, from the order $O_i = x^3{\succ}x^1{\succ}x^2$, three item pairs, (x^3,x^1), (x^3,x^2), and (x^1,x^2), are extracted. Such pairs are extracted from all $|C|$ orders, and are collected into the set P_C. As the probability $\Pr[x^a{\succ}x^b]$, we adopted the following Bayesian estimator with Dirichlet prior distribution in order that the probability remains at non-zero:

$$\Pr[x^a{\succ}x^b] = \frac{|x^a,x^b|{+}0.5}{|x^a,x^b|{+}|x^b,x^a|{+}1},$$

where $|x^a,x^b|$ is the number of the item pairs (x^a,x^b) in P_C. Then, for each item that appears in some order in the C, (i.e., $X_C = \bigcup_{O_i\in C} X_i$), the following μ_a are calculated:

$$\mu_a = \frac{1}{|X_C|}\sum_{x^b\in X_C}\Phi^{-1}\bigl(\Pr[x^a{\succ}x^b]\bigr),$$

where $\Phi(\cdot)$ is a normal distribution function of $N(0,1)$. By sorting the items according to these μ_a, the order mean is approximated.

The recommendation process is as follows: Before recommendation, the database D_S is divided into the partition $\pi_{D_S} = \{C_1, \ldots, C_{|\pi_{D_S}|}\}$ by the k-o'means. Given the active user's order, O_a, the system calculates the above dissimilarities between the O_a and each order mean of clusters in the π_{D_S}, and the most similar cluster is found. The system then recommends the highly ranked items in the most similar order mean. Note that since the output of the k-o'means depends on the initial partitions, we chose the partition that achieved the minimum sum of dissimilarities among 10 trials in the experiment described in Section 4.

3.3 Hybrid Methods

We examined the hybrid methods of memory-based and model-based methods. The cluster models combined with the simple rank correlation method.

The original memory-based algorithm calculates the similarities between the active user and each user in the entire user database. In this hybrid method, first the most similar cluster C^* is found by the procedure described in Section 3.2. Then active user's preference is predicted only from the user data in the C^*, and the rest of the data is ignored.

4. EXPERIMENTS

To compare the above CF methods, we applied these methods to sushi preference data.

4.1 The Data Collection Procedure

Before describing the data collection procedure, we would like to explain why we chose to use preferences in sushi (a Japanese food) as a testbed. First, there are many types of sushi, and preferences are different for each person. In addition, since users have fewer privacy concerns about expressing their preferences in sushi, the data was easily collected.

The preference data were collected using the following procedure. Before collecting the data, we surveyed menu data from 25 sushi restaurants found on the WWW. For each type of item (i.e., type of sushi sold at the restaurant), we counted the number of restaurants that offered the item. From these counts, we derived the probabilities that each item would be supplied. By eliminating unfamiliar or low frequency items, we came up with a list of 100 items. This item set corresponds to X^* in Section 2.1.

We generated two item sets, which were presented as X_i to each user. The type A set (X^A) was common for all users. We chose the following 10 popular items: Shrimp, Sea eel, Tuna, Squid, Sea urchin, Salmon roe, Egg, Fatty tuna, Tuna roll and Cucumber roll. This set was used for testing. The other type B sets (X_i^B) were different for each user. Ten items were randomly selected from X^* according to the above probability distribution. The orders in this item set were treated as user responses. Note that the X^A and the X_i^B had overlap of 2.41 items per order on average.

We collected the responses via a commercial WWW survey service. The following queries were presented:

1) We asked each user i to sort items in the X^A according to his/her preference. The user selected ranks of presented items through the WWW interface. The response order was denoted by O_i^A.

2) We asked the users i to rate their preferences in items of X_i^B by the SD method using a five point scale. The response scores were denoted by S_i^B.

3, 4) Next, two questions were irrelative to preferences. These two questions lessened the influence of query 2 on query 5.

5) We asked the user i to sort the items in the X_i^B set according to his/her preference. The response order was denoted by O_i^B.

The total number of responses collected was 1039. We eliminated the data obtained within a response time which was either too short or too long. Consequently, the data set includes 1025 tuples: (O_i^A, O_i^B, S_i^B). We performed a preliminary experiment to compare two types of responses. We derived the ratio of responses in which there exists contradiction between S_i^B and O_i^B. Here, the contradiction means that, though the item x^a precedes the x^b in O_i^B, the score of x^b in S_i^B is rated higher than that of x^a. We found such contradictions in 70% of the tuples. This result at least shows that different aspects of preference can be captured by the ranking and SD methods.

4.2 The Evaluation Procedure and Criterion

To evaluate each method, we applied the 10-fold cross validation test. The training and the test sets are denoted by $\bar{D}'$ and D', respectively. As described above, a data set D was composed of tuples: (O_i^A, O_i^B, S_i^B). The S_i^B and O_i^B were used for a traditional CF and for ours, respectively. The set, which consists of all the O_i^B (or S_i^B) in the $\bar{D}'$, was treated as a user database D_S. Each O_i^B (or S_i^B) in the D' was picked up in turn, and the picked order (or score) was considered as the active users response O_a^B (or S_a^B). Given the D_S and the O_a^B (or S_a^B), the system predicted the order of items in the X^A (the type A item set) sorted according to the active user's preference. Let $\hat{O}_i^A$ be the predicted order. The $\hat{O}_i^A$ is compared with the true response order O_i^A in the D'. To measure the quality of the predicted order, ρ between the $\hat{O}_i^A$ and the O_i^A as used in [10].

In order to investigate the changes in user database sizes, we generated three sets, sizes (denoted by $|D|$) of which are 1025, 500, and 300, respectively. Similarly, by randomly eliminating items from the X_i^B and renumbering the ranks of these items, we changed the sizes of item sets X_i^B to 10, 7, 5, 3, and 2.

4.3 Experimental Results

We applied the evaluation procedure in the previous section to each of the methods described in Section 2.1 and 3. These methods are denoted by the symbols: `SCR` (Sec. 2.1), `RCR` (Sec. 3.1.1), `CLS` (Sec. 3.2), and `HCR` (Sec. 3.3).

As described in Section 4.1, the item sets X^A and X_i^B can be overlapped. We didn't abandon these overlapped items because the preference orders have to be predicted, even if the ranks or scores of items in the X_i^B are given. These overlapped items were treated as follows. In the case of the `SCR` method, since the scores were rated with common scales, the score given in the S_i^B was used as the expected score. The scores of the other non-overlapped items were predicted by Equation (1). However, in the cases of all the other methods, all the preferences were estimated irrespective of whether or not the items were in the X_i^B, because ranks depend on the item set to sort. The numbers of clusters k have to be fixed to apply the methods, `CLS` and `HCR`, that use the k-o'means algorithms. Since we had not developed

Table 1: Means and s.d. of ρ ($|X^B|$=10, $|D|$=1025)

	SCR	RCR	CLS	HCR
mean	0.362	0.432	0.389	0.442
s.d.	0.3075	0.3034	0.3443	0.3095

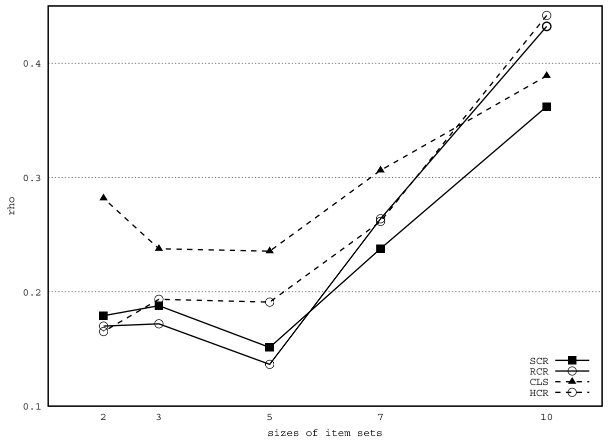

Figure 2: Changes of criteria according to the sizes of the response item sets ($|X^B|$)

a method to determine these numbers, the numbers were changed from 2 to 10, and the best result was selected.

We applied all the methods to the data set in which the size of the data set ($|D|$) is 1025 and the sizes of the response item sets ($|X^B|$) is 10. The means and standard deviations of ρ are shown in Table 1. Surprisingly, in all criteria, the traditional method was worse than any of our methods. We think that this result is due to the undesirable properties of SD methods for CF. Below, we show the ratios of each rating score selected by users:

[1] 0.082 [2] 0.095 [3] 0.226 [4] 0.224 [5] 0.372

This distribution is highly skewed, and users gave ratings within a narrow range of the scale. Therefore, we believe it is problematic to treat these ratings as values of interval scales by introducing the assumptions in Section 1.1. Note that it might appear to be effective to normalize by using the minimum and the maximum ratings of each user. By imposing this normalization on the SCR, the mean of ρ became 0.367. This normalized SCR appeared to have no significant advantage over the original SCR, and was found to be significantly inferior to the RCR.

Figure 2 shows the changes of the means of criteria according to the sizes of the response item sets ($|X^B|$), when the sizes of the data sets ($|D|$) are fixed to 1025. When comparing the SCR method and the other order-based methods, the larger the $|X^B|$ becomes, the more the order-based methods overcome the SCR. Among the order-based methods, the CLS method is rather characteristic. This method performed even if the $|X^B|$ is small, but is inferior to the other methods when the $|X^B|$ is large. We think this is due to the fact that while the other methods are designed to purely predict the personal preferences of an active user, the CLS method is designed to predict the preferences of the group to which the active user belongs. Therefore, if less personal information is supplied, the CLS method can make a better recommendation based on rather generalized preferences.

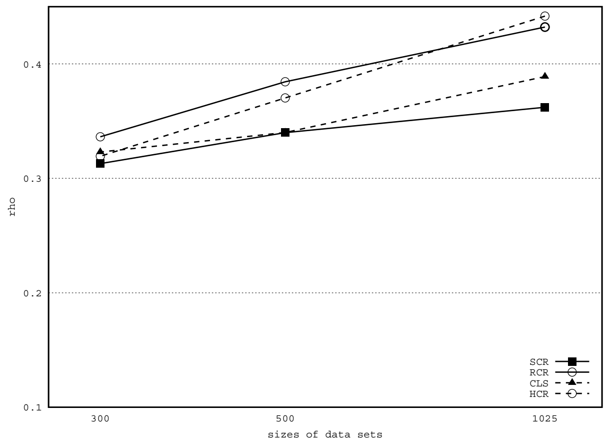

Figure 3: Changes of criteria according to the sizes of the data sets ($|D|$)

Table 2: t-values of ρ between the SCR and each of the other methods

(a) sizes of item sets ($|X^B|$)

$\lvert X^B \rvert$	10	7	5	3	2
RCR	$\oplus$ 7.390	$\ominus$ 2.306	−1.275	$\triangle$−2.019	$\triangle$−1.772
CLS	$\ominus$ 2.031	$\oplus$ 4.803	$\oplus$ 5.675	$\oplus$ 3.572	$\oplus$ 8.358
HCR	$\oplus$ 7.865	$\ominus$ 2.017	$\oplus$ 3.024	0.724	$\times$−2.560

(b) sizes of data sets ($|D|$)

$\lvert D \rvert$	1025	500	300
RCR	$\oplus$ 7.390	$\oplus$ 3.234	1.280
CLS	$\ominus$ 2.031	0.006	0.404
HCR	$\oplus$ 7.865	$\ominus$ 1.982	0.329

Figure 3 shows the changes of the means of criteria according to the sizes of the data sets ($|D|$), when the $|X^B|$ are fixed to 10. Again, the order-based method achieves superiority over the SCR method if more information is available.

To stringently examine the difference between the SCR method and the other methods, we applied a paired t-test (Table 2). The positive t-values indicate that order-based methods are superior. The $\oplus$ and $\ominus$ ($\times$ and $\triangle$) indicate that the order-based methods are superior (inferior), and that the difference is statistically significant at the significance level of 1% and 5%, respectively. We first discuss results in Table 2(a). As described above, except for the CLS method, the order-based methods are superior to the SCR method, if the $|X^B|$ is large. However, the SCR is superior when the $|X^B| = 2$ or 3. Overall, if the system is designed to request to sort 7 or more items, the order-based method surpasses the traditional methods. In the case of Table 2(b), the larger the $|D|$ grows, the more superior the order-based methods, except for the CLS, become.

To test the efficiency of hybridization in Section 3.3, a hybrid method and its source method were compared. Table 3 shows the t-values between the ρ of the hybrid method HCR and of its source method, RCR and CLS. The positive t-values indicate the superiority of the hybrid methods. When com-

Table 3: Comparison with hybrid methods and their source methods

$\|X^B\|$	10	7	5	3	2
RCR	1.464	-0.301	$\oplus$ 4.422	$\oplus$ 2.933	$\triangle -1.767$
CLS	$\oplus$ 5.112	$\times -3.627$	$\times -3.303$	$\times -3.295$	$\times -9.207$

$\|D\|$	1025	500	300
RCR	1.464	-1.446	-1.316
CLS	$\oplus$ 5.112	$\ominus$ 1.896	-0.175

Table 4: t-values between the ρ with and without default voting ($|D| = 1025$)

$\|X^B\|$	10	7	5	3	2
t-val	$\oplus$ 2.681	-1.053	$\times -9.412$	$\times -31.520$	$\times -8.891$
#	1.768	0.868	0.445	0.161	0.072

paring with the RCR, there were no clear characteristics according to the changes of the $|D|$ or the $|X^B|$. However, none of the methods are worse than the RCR if the $|X^B|$ is greater than 3. By clustering the user database in advance, it is possible to save the number of the sample users to which similarities had to be calculated. Since this result shows that the performance was not depressed by using clustering, it is possible to save time required for recommendation to be made by using hybridization. When comparing with CLS, by hybridization, the characteristics of the CLS method seem to have been lost. Therefore, similar to other memory-based methods, the hybrid methods are superior if the $|X^B|$ is large.

We compared the RCR methods with and without default voting in Section 3.1.2. The results of the comparison are shown in Table 4. The positive t-values indicate the superiority of the method with default voting. The row labeled "#" shows the mean numbers of items shared between two preference orders, O_a and O_i. As the sizes of item sets decrease, the number of shared items also decrease. We had expected default voting to be effective when the number of shared items was small, but default voting depressed the performance in such cases, i.e., $|X^B| \leq 5$. This phenomenon was also observed if $|D|$ is 500 or 300. We currently are not able to explain these phenomena, but one possible hypothesis is that default ranking described in Section 3.1.2 does not work as neutral evaluation.

5. CONCLUSIONS

We advocate the new nantonac CF, by using which items are recommended based on the order responses. We showed that the prediction performance of nantonac CF was clearly superior to those of the traditional method.

The current framework cannot handle a large universal item set X^*, since it is difficult for users to sort items if $|X_i|$ is larger than about 10. Thus, to collect a more user preference data, we will develop a method that allows each user to bring multiple responses in terms of distinct item sets.

6. ACKNOWLEDGMENTS

A part of this work is supported by the grant-in-aid for exploratory research (14658106) of the Japan society for the promotion of science.

7. REFERENCES

[1] C. Basu, H. Hirsh, W. W. Cohen, and C. Nevill-Manning. Technical paper recommendation: A study in combining multiple information sources. *Journal of Artificial Intelligence Research*, 14:231–252, 2001.

[2] J. S. Breese, D. Heckerman, and C. Kadie. Emprical analysis of predictive algorithms for collaborative filtering. In *Uncertainty in Artificial Intelligence 14*, pages 43–52, 1998.

[3] W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.

[4] T. Joachims. Optimizing search engines using clickthrough data. In *Proc. of The 8th Int'l Conf. on Knowledge Discovery and Data Mining*, pages 133–142, 2002.

[5] T. Kamishima and S. Akaho. Learning from order examples. In *Proc. of The IEEE Int'l Conf. on Data Mining*, pages 645–648, 2002.

[6] T. Kamishima and J. Fujiki. Clustering orders. In *Proc of The 6th Int'l Conf. on Discovery Science*, 2003. (submitted).

[7] H. Kazawa, T. Hirao, and E. Maeda. Ranking SVM and its application to sentence selection. In *Proc. of 2002 Workshop on Information-Based Induction Sciences*, 2002. (in Japanese).

[8] M. Kendall and J. D. Gibbons. *Rank Correlation Methods*. Oxford University Press, fifth edition, 1990.

[9] W. S. Lee. Collaborative learning for recommender systems. In *Proc. of The 18th Int'l Conf. on Machine Learning*, pages 314–321, 2001.

[10] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *ACM SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*, 1999.

[11] A. Nakamura and N. Abe. Collaborative filtering using weighted majority prediction algorithms. In *Proc. of The 15th Int'l Conf. on Machine Learning*, pages 395–403, 1998.

[12] C. E. Osgood, G. J. Suci, and P. H. Tannenbaum. *The Measurement of Meaning*. University of Illinois Press, 1957.

[13] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: An open architecture for collaborative filtering of Netnews. In *Proc. of The Conf. on Computer Supported Cooperative Work*, pages 175–186, 1994.

[14] P. Resnick and H. R. Varian. Recommender systems. *Communications of The ACM*, 40(3):56–58, 1997.

[15] Y. Sai, Y. Y. Yao, and N. Zhong. Data analysis and mining in ordered information tables. In *Proc. of The IEEE Int'l Conf. on Data Mining*, pages 497–504, 2001.

[16] J. B. Schafer, J. A. Konstan, and J. Riedl. E-commerce recommendation applications. *Journal of Data Mining and Knowledge Discovery*, 5:115–153, 2001.

[17] S. S. Stevens. Mathematics, measurement, and psychophysics. In S. S. Stevens, editor, *Handbook of Experimental Psychology*. John Wiley & Sons, Inc., 1951.