

Filling-in Missing Objects in Orders

Toshihiro Kamishima Shotaro Akaho

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/>) s.akaho@aist.go.jp

Abstract

Filling-in techniques are important, since missing values frequently appear in real data. For categorical or numerical values, such filling-in techniques have been established. Though lists of ordered objects are widely used as representational forms (ex., Web search results, best seller lists), filling-in techniques for orders have received little attention. Therefore, we propose simple but effective technique to fill-in missing objects in orders. We empirically show its effectiveness by building-in this technique into our collaborative filtering system.

1 Introduction

We propose a technique for filling-in missing objects in orders. We empirically show its effectiveness by building-in this technique into our collaborative filtering system.

An *order* is a sorted sequence of objects, and only the determination which object precedes the others is meaningful. Before discussing orders, we must draw attention to a notion of *Stevens' Scales of Measurement* [22], which is a classification of scales into four levels: ratio, interval, ordinal, and nominal. A ratio (ex. length) and an interval (ex. time) indicate numerical scales with and without the origin, respectively. Nominal scale represents categories. For these three levels of scales, many analysis or processing methods have been developed. However, in terms of an ordinal scale, a few attempts have been made. Lists of ordered objects are widely used as representational forms. For example, Web search engines return page lists sorted according to their relevance to queries. Further, best seller lists, which are item-sequences sorted according to selling volume, are used on many E-commerce sites. Processing methods for orders are also useful for information retrieval [4, 7]. In order to improve search results, users' feedback is considered. To obtain feedback information, the systems request users to specify which documents are more relevant than others. In spite of their importance, the methods of process-

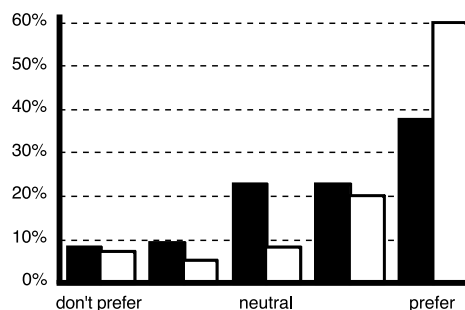


Figure 1. Distributions of rating scores obtained by the SD method

ing orders have received little attention. Filling-in missing objects in orders is one of such processing tasks. This task is important, since missing values are frequently observed in real data. Specifically, given samples of orders, we propose a method to determine the rank of the object, which is not appears in one sample orders, in the order based on the summaries of samples. This is an analogy for filling-in missing values in the other levels of scales by the modes or the means of samples.

We were motivated to develop our filling-in technique to perform better recommendation in collaborative filtering. Collaborative filtering (*CF* for short) is a framework for recommending the items anticipated to be preferred by the users [21]. It is required for performing this CF task to measure users' preference. We first show that orders are fit for measuring the quantity of sensation, such as, preference, taste. To measure users' preferences, almost all the CF methods adopt the Semantic Differential (SD) method [18]. In this method, the users' preferences are rated using a scale on which extremes are represented by antonyms. One example is a five-point-scale on which 1 and 5 indicate *don't prefer* and *prefer*, respectively. To treat the acquired ratings as interval values, the two assumptions must be introduced; "The total lengths of all the scales are equal" and "All the intervals of one scale are equal." However, it is commonly

difficult for users to quantify their own sensation so as to satisfy these assumptions. To show this difficulty, we depict the distributions of rating scores obtained by the SD method in Figure 1. The black bars show the distribution of our sushi preference survey, and the white bars show that of scores rated by Amazon.com customers¹. Under the above assumptions, these distributions should ideally be unimodal and symmetric, but the real distributions might be highly skewed and non-unimodal. Additionally, the SD method is often affected by psychological rater effects. One example is a leniency effect, which is the phenomena that some respondents tend to give more lenient (i.e., better or higher) scores than the true sensation. Such shifts in ratings were reported in [1].

One alternative is the ranking method. Users preference patterns are measured by response orders, which are lists of objects sorted according to the degrees of the users’ preferences. We previously called a CF framework adopting the ranking method by *Nantonac Collaborative Filtering*² [8], and more appropriate items were recommended by adopting this ranking method. However, it was not as advantageous when the length of response orders were short, because it becomes difficult to evaluate the correlation between users’ preferences. Such short responses can be easily collect by using a Joachims’ procedure [7], thus it is fruitful to improve the quality of recommendation for short order responses. To achieve this improvement, we were motivated to introduce the filling-in technique for orders and to test their effectiveness.

In Section 2, we describe a filling-in technique for orders. We show a task of nantonac CF in Section 3. Section 4 shows experimental results to test our filling-in technique. Section 5 summarizes our conclusions.

1.1 Related Work

We here describe processing of orders except for filling-in techniques. Cohen et al. [4] and Joachims [7] proposed a method to sort attributed objects given in the form of ordered pairs. Kamishima and Akaho [9] and Kazawa et al. [11] studied a problem of learning from ordered item sets. Sai et al. [20] proposed association rules between order variables. Mannila and Meek [14] tried to establish the structure expressed by partial orders among a set of orders. Kamishima and Fujiki studied the clustering method for incomplete orders [10]. Lebanon and Lafferty discussed the combination method for orders using conditional probability models [13]. Dwork et al. developed a meta-search engine to combine the ordered lists of Web pages provided

¹from the invited talk “Analyzing Customer Behavior at Amazon.com” by A. S. Weigend at KDD2003

²We would like to point out that the word *nantonac* is originates from a Japanese word, *nantonaku*

by sub engines [5]. Freund et al. developed the boosting for orders [6].

2 Filling-in Missing Objects in Orders

We describe basic notations and filling-in techniques regarding orders.

2.1 Basic Notations

We first give basic notations. An object x_j corresponds to an object, entity, or substance to be sorted. The universal object set, X^* , consists of all possible objects. The order is denoted by $O = x_1 \succ x_2 \succ \dots \succ x_n$. The meaning of the order, $x_1 \succ x_2$, is “ x_1 precedes x_2 .” The object set X_i is composed of all the objects in the order O_i . Let $|A|$ be the size of the set A , then $|X_i|$ is equal to the length of the order O_i . An order of all objects, i.e., O_i s.t. $X_i = X^*$, is called a complete order; otherwise, it is called an incomplete order. The rank, $r(O_i, x_j)$, is the cardinal number that indicates the position of the object x_j in the order O_i . For example, $r(O_i, x_2), O_i = x_1 \succ x_3 \succ x_2$ is 3. For two orders, O_1 and O_2 , consider an object pair x_a and x_b , such that $x_a, x_b \in X_1 \cap X_2, x_a \neq x_b$. We say that the orders O_1 and O_2 are concordant w.r.t. x_a and x_b , if two objects are placed in the same order, i.e.,

$$(r(O_1, x_a) - r(O_1, x_b))(r(O_2, x_a) - r(O_2, x_b)) \geq 0;$$

otherwise, these are discordant. O_1 and O_2 are concordant if O_1 and O_2 are concordant w.r.t. all object pairs such that $x_a, x_b \in X_1 \cap X_2, x_a \neq x_b$.

We then describe the distance, $d(O_a, O_b)$, defined between two orders consisting of the same objects, i.e., $X_a = X_b (\equiv X)$. we use well-known *Spearman distance* $d_S(O_a, O_b)$ [12, 15]; this is defined as the sum of the squared differences between ranks. The statistical property of d_S have been well studied and its computational complexity, $O(|X| \log |X|)$, is relatively small. By normalizing the distance range to be $[-1, 1]$, the *Spearman’s rank correlation* ρ is derived.

$$\rho = 1 - \frac{6 \times d_S}{|X|^3 - |X|}. \quad (1)$$

We below describe a task of filling-in missing objects in orders. Suppose to calculate the distance between two orders,

$$O_a = x_1 \succ x_3 \succ x_6 \text{ and } O_b = x_5 \succ x_3 \succ x_2 \succ x_6. \quad (2)$$

The distances is defined between two orders consisting of the same objects, but X_a and X_b are not identical, that is to say, the both orders include missing objects; The missing objects for O_a (i.e., $\tilde{X}_a = \{x | x \notin O_a \wedge x \in O_b\}$) are

$\{x_2, x_5\}$, and those for O_b are $\{x_1\}$. Hence, it is not possible to directly calculate the distance between O_a and O_b . One way to overcome this difficulty is ignoring missing objects and the distance is calculated over common objects, i.e., $X_a \cap X_b$. For example, by ignoring missing objects, both of O_a and O_b are converted into $x_3 \succ x_6$, accordingly the Spearman's distance $d_S = 0$. However, useful information might be contained in these ignored objects, so it would lessen the precision or the confidence in the calculation of distances. Moreover, if $X_a \cap X_b = \emptyset$, all the objects are deleted and the distance can not be derived. If samples of orders were available, the ranks of such missing objects could be filled-in based on the summary statistics of the samples. Such techniques would be highly beneficial to the derivation of more appropriate distances between orders.

2.2 An Incomplete Order Set

In a psychological statistics literature, these missing values are commonly processed by considering a set of orders, instead of a single order. We introduce the notion of an *Incomplete Order Set*³ [15], which is defined as a set of orders that are concordant with the given order. Formally, let O be the order consisting of the object set X , and \tilde{X} be a set of missing objects. An IOS is defined as

$$\text{ios}(O, \tilde{X}) = \{O'_i | O'_i \text{ is concordant with } O, X'_i = X \cup \tilde{X}\}.$$

Unfortunately, this idea is originally designed assuming that $|X'|$ is small, typically less than 5. Therefore, this idea is not fit for large scale data sets targeted by the data mining or the information filtering techniques. Because the size of $\text{ios}(O, \tilde{X})$ is $|X'|! / |X|!$, which exponentially grows in accordance with $|X'|$. Furthermore, there are some difficulties in calculation of distances. To calculate distance between two sets of orders, $\text{ios}_a = \text{ios}(O_a, \tilde{X}_a)$ and $\text{ios}_b = \text{ios}(O_b, \tilde{X}_b)$, the definition of distance has to be extended. One possible extension is adopting arithmetic average:

$$d(\text{ios}_a, \text{ios}_b) = \frac{1}{|\text{ios}_a| |\text{ios}_b|} \sum_{O_i \in \text{ios}_a} \sum_{O_j \in \text{ios}_b} d(O_i, O_j).$$

But, since $d(\text{ios}_a, \text{ios}_a)$ may not be 0, this is not distance. There are some alternatives such as a Hausdorff distance, but the ranges of values are changed by the extension.

2.3 A Default Rank

To avoid the above difficulties in IOS, we proposed the idea of *default rank* [8]. This idea is to rank missing

³in the cited book, this notion is referred by the term "incomplete rankings", but we adopt this term to insist to be a set

objects into the middle of the filled-in orders. In the case of Equation (2), by inserting missing objects in the middle of orders, filled orders, O_a and O_b , are obtained:

$$O'_a = x_1 \succ x_3 \succ x_2 \sim x_5 \succ x_6 \text{ and } O'_i = x_5 \succ x_3 \succ x_1 \succ x_2 \succ x_6,$$

where \sim denotes a tie in rank.

In [8], the efficiency of adopting default ranks was investigated. Unfortunately, default ranks were not found to be effective. We had considered that the middle ranks in orders represent the neutral values, but this might not be the case. For example, suppose that there is an object ranked at the lowest in almost all the sample orders. If this object was ranked at the middle, the filled-in order would indicate that the object is ranked relatively higher.

2.4 A Default Order

We finally propose our idea of default orders. In the case of numerical or nominal variables, missing values can be replaced with the summary statistics of samples, such as the means or the modes. By analogy, we will fill the ranks of missing objects by using the centers of orders in sample set, S . This central order \bar{O}_S [15] is defined as

$$\bar{O}_S = \arg \min_O \sum_{O_i \in S} d(O_i, O).$$

Note that \bar{O}_S is composed of objects $\bar{X}_S = \cup_{O_i \in S} X_i$. If the Spearman's distance is adopted and all the sample orders are complete, i.e., $X^* = X_i, \forall i$, the central order can be derived by sorting objects according to their mean ranks in sample orders [15]. However, since sample orders tend to be incomplete in a practical situation, deriving the true central orders is not tractable. Hence, we employ the Thurstone's paired comparison method. This method is based on the model of the Thurstone's law of comparative judgment [23]. This model sorts objects, x_j , according to their scores, which follow the normal distribution $N(\mu_j, \sigma)$. By applying the least square method to this model [17], the μ'_j , i.e., the linear transformation of the μ_j , is derived as

$$\mu'_j = \frac{1}{|X|} \sum_{x \in \bar{X}} \Phi^{-1}(\text{Pr}[x_j \succ x]), \quad (3)$$

where $\Phi(\cdot)$ is the normal distribution function. The probability that the object x_j precedes x , $\text{Pr}[x_j \succ x]$, can be estimated by simply counting the ordered pairs appeared in the sample set. We approximate the central order by sorting objects according to corresponding μ'_j .

The *default order* is the order that is concordant with the central order and is composed of missing objects. For example, suppose the case of Equation (2) and the central order of samples is $x_1 \succ x_5 \succ x_2 \succ x_3 \succ x_4 \succ x_6$. The missing objects for O_a and for O_b are $\{x_2, x_5\}$ and $\{x_1\}$, respectively. Accordingly, the default order for the O_a becomes $\bar{O}_a = x_5 \succ x_2$. Similarly, that for the O_b is $\bar{O}_b = x_1$.

We propose to fill-in the missing objects in preference orders by using the default orders. For this purpose, the sample order and its default order have to be merged. By definition, no objects are commonly included in both of two orders. However, the existing methods utilize common objects as keys for merging. Thus, these methods cannot be applied. We therefore propose the following method based on order statistics.

Suppose the case that the preference order O and its default order \tilde{O} are merged into the filled order O' . Note that, by definition, no objects are shared between O and \tilde{O} . Additionally, O' is composed of objects in O or \tilde{O} , and therefore $|O'| = |\tilde{O}| + |O|$. Instead of directly modeling this merging process, we consider the division process. Suppose that $|O|$ of objects are randomly sampled from objects in the O' without replacement, and these are sorted so as to be concordant with the O' , so that the O is obtained. In this case, for the i -th object $x_{i:O}$ in the O , the expectation of ranks in the O' becomes

$$E[r(O', x_{i:O})] = \frac{i(|O'| + 1)}{|O| + 1},$$

according to [2]. Similarly, for the j -th object in the \tilde{O} , the expectation is $j(|O'| + 1)/(|\tilde{O}| + 1)$. We assigned these expectations of rank to the objects in the O and the \tilde{O} , then the O' are formed by sorting according these expectations. In the case of Equation (2) example, $O_a = x_1 \succ x_3 \succ x_6$ and its default order $\tilde{O}_a = x_5 \succ x_2$ are merged. To the second object x_3 in O_a , the expected rank $2(5+1)/(3+1)=3$ is assigned. These expectations are assigned to all the remaining objects in a similar way. Consequently, by sorting objects according to these expectations, we obtain the order $O'_a = x_1 \succ x_5 \succ x_3 \succ x_2 \succ x_6$. Similarly, $O'_b = x_5 \succ x_3 \succ x_1 \succ x_2 \succ x_6$. Note that this filling-in technique is very simple, thus its computational complexity is small, $O(\max(|X'|, |\tilde{X}'| \log |\tilde{X}'|))$. Hence it can be applied to process large scale data.

3 A Task of Nantonac Collaborative Filtering

We built-in the filling-in technique in the previous section into our nantonac CF method. So, we describe this CF task and the method using default orders.

3.1 Formalization of a Nantonac CF task

Before describing a nantonac CF framework, we will first describe the CF task developed as a part of the GroupLens project [19]. The aim of a CF task is to predict the preferences of a particular user (an active user) based on the preference data (a user DB) collected on other users. Formally, the task is defined as follows. Let s_{ij} be the score

of the object 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 objects which the user i rated, and $S_i = \{s_{ij} | x_j \in X_i\}$. The user DB, $D_S = \{S_1, \dots, S_{|D_S|}\}$, is a set of all S_i . Sample users are people who provided ratings in the DB. Let S_a be the set of scores rated by the active user, and X_a be the set of objects the active user has already rated. Given the S_a and the D_S , the CF task is to estimate which the objects the active user is anticipated to rate high. Such objects are then recommended.

A *nantonac collaborative filtering* task is the same as that of the GroupLens, except in terms of the representation of the users' preferences. Instead of using a set of scores acquired by the SD method, we adopted the order acquired by the ranking method. The system shows a set of objects, X_i , to the user i , and the user sorts these objects according to his/her preferences. The sorted sequences are denoted by $O_i = x_1 \succ x_2 \succ \dots \succ x_{|X_i|}$.

In a nantonac CF framework, the user DB is a set of orders sorted by all the respective users, $D_S = \{O_1, \dots, O_{|D_S|}\}$. The orders in D_S are called sample orders. Let X_a be a set of objects sorted by the active user, and O_a be the sorted order. Given the O_a and D_S , the goal of nantonac CF is estimating which the objects are anticipated to be preferred by the active user.

3.2 A Simple Correlation Method

A simple correlation method is a basic method for performing a nantonac CF task [8]. In this method, objects are recommended to active users through almost same process as that used in the GroupLens. Rating scores s_{ij} are simply substituted by ranks $r(O_i, x_j)$, which are the ranks of object j in the sample order of the user i . The prediction process of this simple correlation method is as follows. \bar{r}_i denotes the mean ranks over objects $X_{ai} = X_a \cap X_i$ in the order of the user i ; i.e., $(1/|X_{ai}|) \sum_{x_j \in X_{ai}} r(O_i, x_j)$. \bar{r}_a is the mean ranks in the active user's order. The similarities between the active and the sample user i are defined as

$$R_{ai} = \frac{\sum_{x_j \in X_{ai}} (r(O_a, x_j) - \bar{r}_a)(r(O_i, x_j) - \bar{r}_i)}{\sqrt{\sum_{x_j \in X_{ai}} (r(O_a, x_j) - \bar{r}_a)^2} \sqrt{\sum_{x_j \in X_{ai}} (r(O_i, x_j) - \bar{r}_i)^2}}. \quad (4)$$

Note that the objects missing in the other order are ignored, but the ranks are not renumbered. For example, for $O_a = x_1 \succ x_2 \succ x_3$ and $O_i = x_3 \succ x_1$, the object x_2 in the O_a is missing in the O_i , so the x_2 is ignored. However, the rank of x_3 remains $r(O_a, x_3) = 3$, not 2. The system estimates the active user's preferences for the object j by the function:

$$\tilde{r}_a + \frac{\sum_{i \in I_j} R_{ai} (r(O_i, x_j) - \bar{r}_i)}{\sum_{i \in I_j} |R_{ai}|}, \quad (5)$$

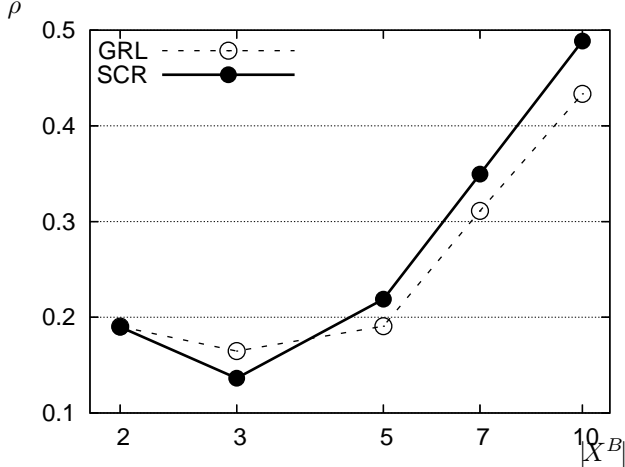


Figure 2. Means of ρ derived by the GroupLens and the nantonac CF methods in Section 3.2

where \tilde{r}_a is the mean ranks in the active user’s order over all objects in X_a ; that is, $(1/|X_a|) \sum_{x_j \in X_a} r(O_a, x_j)$. I_j is a set of indices of sample users who evaluated the object j ; i.e., $\{i | O_i \in D_S \text{ s.t. } x_j \in X_i\}$. The objects are sorted in ascending order of these estimated preferences, and highly ranked objects are recommended.

3.3 Nantonac Collaborative Filtering Using Our Filling-in Technique

We first describe the reason why filling-in technique is beneficial to the performance of a nantonac CF task. Figure 2 shows the results derived by the GroupLens’ method (GRL) and the nantonac CF method in Section 3.2 (SCR). The experimental procedure and evaluation criteria are described in Section 4. The figure shows the changes of the mean ρ according to the length of the response orders ($|X^B|$) when the sizes of the data sets ($|D|$) are fixed to 5000. If $|X^B|$ is larger than 5, the SCR method was superior to the GRL, and the differences were statistically significant at the significance level of 1% by paired t -test. Note that we will use this test condition throughout this paper. However, if the length of response orders are short, recommendations by the SCR were not so advantageous.

An improvement of recommendation for short response orders, especially the length of two, is promising. One of the obstacle to perform CF is that users tend to take the trouble of representing their preferences. One way to overcome this obstacle is to collect users’ responses implicitly exposed in histories of users’ actions. Joachims proposed such a method [7]. Suppose that an ordered list, $x_1 \succ x_2 \succ \dots$, of retrieved or recommended objects is displayed to a user.

The user scan the list from the top to the bottom, and selected the third object, x_3 . From this action, the system can get the user’s implicit responses, $x_3 \succ x_1$ and $x_3 \succ x_2$. Because the user has already checked x_1 and x_2 , but chose x_3 , so it can be concluded that the user prefers x_3 to x_1 or x_2 . This technique will make it easier to collect preference data.

To perform better recommendations for a few responses in the case of the SD-based CF, Breese et al. proposed to fill-in missing scores before calculating the correlation between users’ responses [3]. We introduce this idea into the nantonac CF case. In accordance with the decrease of $|X_i|$ relative to $|X^*|$, the frequency of the event $X_a \cap X_i = \emptyset$ increases. Because the R_{ai} are always 0 in cases in which no commonly evaluated objects exist, the similarities between users can no longer be precisely measured. By filling-in the missing objects, the similarities can be calculated over $X_a \cup X_i$, not $X_a \cap X_i$; thus, more appropriate similarities will be derived. Note that filling-in the missing objects incidentally brings a theoretical merit. The similarity of Equation (4) is heuristically defined, if no filling technique is used and ranks are not renumbered. Therefore, the meaning of this similarity is mathematically unclear. By filling-in, this similarity can be interpreted as Spearman’s rank correlation ρ (see Equation (1)).

To build-in our filling-in method in Section 2.4, the SCR method is modified. The procedures are the same as the original SCR except for filling-in missing objects of O_a and O_i before calculating the correlation Equation (4). The central order \tilde{O} over the user DB, D_S , is derived in advance. Before calculating the correlation of Equation (4), missing objects in the response orders, O_a and O_i , are filled-in. Regarding O_a , the missing objects $\tilde{X}_a = \{x | x \notin O_a \wedge x \in O_i\}$ are sorted so as to be concordant with the central order, \tilde{O} , then the default order \tilde{O}_a is derived. The default order \tilde{O}_a and the active user’s response order O_a are merged into the filled-in order O'_a . The filled-in order for the O_i is similarly derived. The rank correlations R_{ai} between filled orders, O'_a and O'_i , are then calculated. Note that the expected preference is derived by Equation (5), using these R_{ai} and the original rank $r(O_i, x_j)$, not the filled-in rank $r(O'_i, x_j)$.

4 Experiments

To test the efficiency of using default orders, we compared the original simple correlation method in Section 3.2 and the method using default orders in Section 3.3.

4.1 The Experimental Procedure

To test the efficiency of using default orders, the CF methods were applied to preference data in sushi. This data set was collected using the following procedure. 100 objects (i.e., types of sushi) were first selected, and composed

the set X^* . We generated two object 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 types of sushi:

Shrimp, Sea eel, Tuna, Squid, Sea urchin, Salmon roe,
Egg, Fatty tuna, Tuna roll, and Cucumber roll.

This set was used for testing. The type B sets (X_i^B) were different for each user. 10 objects were randomly selected from X^* according to their probability distributions, which were estimated from menu data of 25 sushi restaurants found on the Web. We collected the responses via a commercial Web survey service. We asked each user i to sort objects in the X^A and X_i^B according to his/her preference. The response orders were denoted by O_i^A and O_i^B , respectively. Consequently, this data set was composed of tuples: (O_i^A, O_i^B) . By eliminating the data obtained within a response time that was either too short or too long, we extracted 5000 data.

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. The set, which consisted of all the O_i^B in the \bar{D}' , was treated as a user DB D_S . Each O_i^B in the D' was sequentially picked up, and the picked order was considered as the active user's response O_a^B . Given the D_S and the O_a^B , the system predicted the order of objects in the object set X^A 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' . The measuring criteria will be describe in the next section.

In order to investigate the changes in user DB sizes, we generated six sets, the sizes (denoted by $|D|$) of which were 5000, 3000, 1000, 500, and 300, respectively. Similarly, by randomly eliminating objects from the O_i^B and renumbering the ranks of these objects, we changed the lengths of response orders $|X_i^B|$ to 10, 7, 5, 3, and 2.

4.2 The Evaluation Criteria

To measure the quality of the recommendation, we compared the true order O_i^A and the predicted order \hat{O}_i^A on the three criteria used in [16]. In all the criteria, a larger value indicates better quality.

ρ : **the Spearman's ρ between the \hat{O}_i^A and the O_i^A**

Rank correlation is suited for evaluating the overall quality of orders. We adopted the Spearman's ρ between \hat{O}_i^A and the O_i^A in Equation (1).

Acc: Positive/Negative Accuracy

This measures the quality of the basic determination of whether the recommended object is really preferred. The term *positive* indicates the objects ranked higher than the middle of the order, and the term *negative* indicates its com-

plement. This criterion represents the accuracy of the prediction, whether positive or negative. Formally,

$$\text{Acc} = \frac{1}{|X^A|} \left(\left| \left((r(\hat{O}_i^A, x) \leq \frac{|X^A|}{2}) \wedge (r(O_i^A, x) \leq \frac{|X^A|}{2}) \right) \right| + \left| \left((r(\hat{O}_i^A, x) > \frac{|X^A|}{2}) \wedge (r(O_i^A, x) > \frac{|X^A|}{2}) \right) \right| \right),$$

where $|cond|$ is the number of objects $x \in X^A$ that satisfy the condition *cond*.

Pr3: Precision at Top 3

Because users tend to consider highly ranked objects to be more preferable, it is important to precisely predict the preferences of these objects. This criterion is defined as the ratio of the true positive objects to the objects ranked higher than the top 3 in the predicted order. Formally,

$$\text{Pr3} = \frac{1}{3} \left| \left((r(\hat{O}_i^A, x) \leq 3) \wedge (r(O_i^A, x) \leq \frac{|X^A|}{2}) \right) \right|.$$

4.3 Comparison of Methods with and without Default Orders

To test the method in Section 3.3, we compared it with two baseline methods. One is the simple correlation method, without using default orders, described in Section 3.2. The other is non-personalized recommendation; for all active users, the objects in test set X^A were sorted so as to be concordant with the central orders of the user DB. That is to say, the central orders were treated as a best-seller list, and popular objects were recommended. Comparison results are shown in Figure 3. SDO, SCR, and BAS denote a method using default orders, a simple correlation method, and a non-personalized method, respectively. The first, second, and third rows show the means of criteria ρ , Acc, and Pr3, respectively. The left column (a) of the figure shows the changes according to the lengths of the response orders $|X^B|$ when fixing $|D|=5000$. On all criteria, the SDO method was clearly superior to the SCR, and the differences were statistically significant, if $|X^B| \leq 7$. As mentioned above, the shorter the response orders were, the less objects were commonly ranked between an active user and a sample user. Therefore, our filling-in technique was, in spite of its simplicity, remarkably effective for the shorter response orders.

The right column (b) of the figure shows the changes according to the sizes of the data sets $|D|$ when fixing $|X^B|=10$. On all criteria, the SDO is superior to the SCR if $|D| \leq 500$. The differences between criteria are statistically significant on ρ of $|D|=300, 500$ and Pr3 of $|D|=500$. The difference became greater when $|D|=100$, but was not statistically significant because the sample size was too small to confirm significance. Though the default orders were introduced to improve the recommendation when response orders were short, the SDO method is secondarily effective when sample sizes are small.

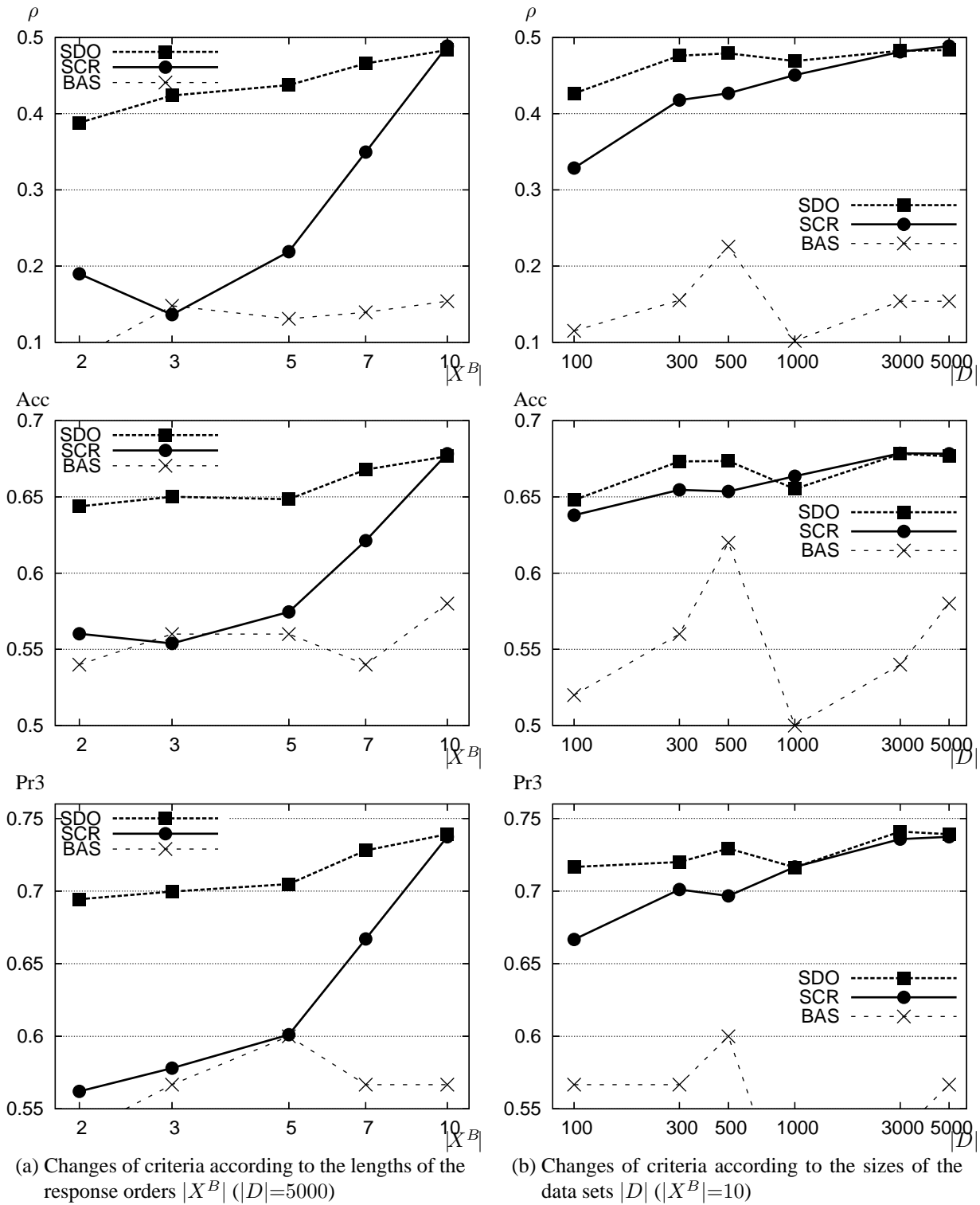


Figure 3. Comparing the SDO method with two base-line methods

We then observe results for the other baseline, the BAS method. If all users had a shared preference in sushi, the central order itself would have provided better recommendations. However, this non-personalized method was apparently worse than the SDO method. This indicates that the advantage of the SDO was not due to the specific characteristics of the users' sharing common preferences, but arose from the ability of the SDO method to provide well personalized recommendations.

5 Conclusions

In this paper, we have proposed the notion of default orders to fill-in missing objects in orders. These default orders were used for collaborative filtering. We empirically showed that the prediction performance could be improved by using default orders.

The current framework cannot handle a large universal object set X^* , because it is difficult for users to sort items if $|X_i|$ is larger than about 10. To overcome this limitation, we will collect multiple response orders for different object sets and develop methods to deal with this type of multi-response data.

Acknowledgments

A part of this work is supported by the grant-in-aid 14658106 and 16700157 of the Japan society for the promotion of science.

References

- [1] C. C. Aggarwal, J. L. Wolf, K.-L. Wu, and P. S. Yu. Horting hatches an egg: A new graph-theoretic approach to collaborative filtering. In *Proc. of The 5th Int'l Conf. on Knowledge Discovery and Data Mining*, pages 201–212, 1999.
- [2] B. C. Arnold, N. Balakrishnan, and H. N. Nagaraja. *A First Course in Order Statistics*. John Wiley & Sons, Inc., 1992.
- [3] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Uncertainty in Artificial Intelligence 14*, pages 43–52, 1998.
- [4] W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.
- [5] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *Proc. of The 10th Int'l World Wide Web Conf.*, pages 613–622, 2001.
- [6] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.
- [7] 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.
- [8] T. Kamishima. Nantonac collaborative filtering: Recommendation based on order responses. In *Proc. of The 9th Int'l Conf. on Knowledge Discovery and Data Mining*, pages 583–588, 2003.
- [9] T. Kamishima and S. Akaho. Learning from order examples. In *Proc. of The IEEE Int'l Conf. on Data Mining*, pages 645–648, 2002.
- [10] T. Kamishima and J. Fujiki. Clustering orders. In *Proc. of The 6th Int'l Conf. on Discovery Science*, pages 194–207, 2003. [LNAI 2843].
- [11] H. Kazawa, T. Hirao, and E. Maeda. Order SVM: A kernel method for order learning based on generalized order statistic. *The IEICE Trans. on Information and Systems*, pt. 2, J86-D-II(7):926–933, 2003. (in Japanese).
- [12] M. Kendall and J. D. Gibbons. *Rank Correlation Methods*. Oxford University Press, fifth edition, 1990.
- [13] G. Lebanon and J. Lafferty. Crankng: Combining rankings using conditional probability models on permutations. In *Proc. of The 19th Int'l Conf. on Machine Learning*, pages 363–370, 2002.
- [14] H. Mannila and C. Meek. Global partial orders from sequential data. In *Proc. of The 6th Int'l Conf. on Knowledge Discovery and Data Mining*, pages 161–168, 2000.
- [15] J. I. Marden. *Analyzing and Modeling Rank Data*, volume 64 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, 1995.
- [16] 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.
- [17] F. Mosteller. Remarks on the method of paired comparisons: i — the least squares solution assuming equal standard deviations and equal correlations. *Psychometrika*, 16(1):3–9, 1951.
- [18] C. E. Osgood, G. J. Suci, and P. H. Tannenbaum. *The Measurement of Meaning*. University of Illinois Press, 1957.
- [19] 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.
- [20] 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.
- [21] J. B. Schafer, J. A. Konstan, and J. Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5:115–153, 2001.
- [22] S. S. Stevens. Mathematics, measurement, and psychophysics. In S. S. Stevens, editor, *Handbook of Experimental Psychology*. John Wiley & Sons, 1951.
- [23] L. L. Thurstone. A law of comparative judgment. *Psychological Review*, 34:273–286, 1927.