Robot, Organize my Shelves!
Tidying up Objects by Predicting User Preferences

Nichola Abdo Cyrill Stachniss Luciano Spinello Wolfram Burgard

***Abstract*— As service robots become more and more capable of performing useful tasks for us, there is a growing need to teach robots how we expect them to carry out these tasks. However, learning our preferences is a nontrivial problem, as many of them stem from a variety of factors including personal taste, cultural background, or common sense. Obviously, such factors are hard to formulate or model a priori. In this paper, we present a solution for tidying up objects in containers, e.g., shelves or boxes, by following user preferences. We learn the user preferences using collaborative filtering based on crowdsourced and mined data. First, we predict pairwise object preferences of the user. Then, we subdivide the objects in containers by modeling a spectral clustering problem. Our solution is easy to update, does not require complex modeling, and improves with the amount of user data. We evaluate our approach using crowdsoucing data from over 1,200 users and demonstrate its effectiveness for two tidy-up scenarios. Additionally, we show that a real robot can reliably predict user preferences using our approach.**

Fig. 1. A robot arranging objects on shelves by predicting user preferences. First, it predicts pairwise preferences between objects, and then it assigns objects to different shelves by maximally satisfying user preferences.

I. Introduction

One of the goals in robotics research is to develop autonomous service robots that assist humans in their everyday life. Robots are envisioned to undertake a variety of tasks including tidying up, cleaning, and attending to the needs of disabled people. As robots get more and more capable of performing such tasks, there is a growing need to teach robots how their users expect them to do so. Learning user preferences, however, is a nontrivial problem. In a home scenario, each user has a preferred way of sorting and storing groceries, kitchenware items, or clothes in different shelves or other containers. Many of our preferences stem from factors such as personal taste, cultural background, or common sense, which are hard to formulate or model a priori. At the same time, it is highly impractical for the robot to constantly query users about their preferences.

This paper provides a solution for how to arrange objects in tidy-up tasks, such as organizing a shelf, or sorting objects in boxes. The key idea of our approach is to first be able to predict user preferences of pairwise object arrangements based on partially-known preferences, and then to compute the best subdivision in shelves or boxes. To achieve this, we build our approach upon the framework of collaborative filtering, which is a popular paradigm from the data-mining community. Collaborative filtering is generally used for learning user preferences in a wide variety of practical applications including suggesting movies on Netflix or products on Amazon. By leveraging this theory, we are able to encode multiple user preferences for each object. Our method does not require that all user preferences are specified for all object-pairs. Additionally, our solution is able to provide preferences even when a completely novel object, unknown to all users, is presented to the system. For this, we combine collaborative filtering with a mixture of experts that compute similarities between objects by using object hierarchies. These hierarchies consist of product categories downloaded from online shops, supermarkets, etc. Finally, we organize objects in different containers by finding object groups that maximally satisfy the predicted pairwise constraints. For this, we solve a minimum k-cut problem by efficiently applying self-tuning spectral clustering. Our prediction model is easy to update and simultaneously offers the possibility for lifelong learning and improvement.

Our approach proceeds in two phases. First, we bootstrap our learning by collecting many user preferences, e.g., by crowdsourced surveys. In this phase, we build a model for object-pair preferences for a tidy-up task. In the second step, the robot queries the user about some preferences, predicts all the remaining ones, and then sorts the objects accordingly.

We present an extensive evaluation for two relevant tidy-up scenarios, arranging toys in different boxes and grocery items on shelves, as well as a real-robot experiment. For training, we collected preferences from over 1,200 surveys. The results demonstrate that our method is able to accurately predict the preferences of the users about objects and their organization in boxes/shelves.

Nichola Abdo, Wolfram Burgard, and Luciano Spinello are with the University of Freiburg, 79110 Freiburg, Germany. Cyrill Stachniss is with the University of Bonn, Inst. of Geodesy and Geoinformation, 53115 Bonn, Germany. This work has partly been supported by the German Research Foundation under research unit FOR 1513 (HYBRIS) and grant number EXC 1086.

II. Related Work

Recent progress in the areas of perception, manipulation, and control has enabled service robots to attend to a variety of chores like cleaning and tidying up [10, 18, 24]. However, as highlighted by a number of researchers, service robots should be able to perform such tasks in a manner that corresponds to our personal preferences [3, 8, 22, 23]. For example, the results of Pantofaru et al. show that people exhibit strong feelings with respect to robots organizing personal items, suggesting the need for the robot to ask humans to make decisions about where to store them [22].

Our environments are rich with cues and information that can assist robots when reasoning about objects and their locations. In this context, several approaches addressed the problem of predicting the locations of objects by leveraging knowledge about how humans use them, typical 3D structures in indoor environments, or co-occurrences of everyday objects in a scene [2, 12, 20].

Recently, Kunze et al. presented an approach that uses object-object spatial relations to predict the locations of everyday objects (e.g., desk arrangements) by training a Gaussian mixture model [15]. However, they address an active search problem where a robot has to locate a certain object. In contrast to that, our work is concerned with tidying up everyday objects by predicting user preferences for object-object relations. In the context of service robots, Schuster et al. presented an approach for predicting the location for storing different objects (e.g., cupboard vs drawer) based on the objects observed in the environment [25]. They train classifiers that consider features for object-object relations. As one of their features, they make use of similarities between objects based on a given hierarchy or ontology. We also make use of a similarity measure based on hierarchies mined from the Web and use it for making predictions for unknown objects. However, in contrast to Schuster et al., our approach leverages collaborative filtering theory to learn organizational patterns from different users without the need for specifying relevant features. Moreover, we formulate a spectral clustering problem that allows us to satisfy as many user preferences as possible when allocating objects to specific shelves. To cope with new objects, our method combines collaborative filtering with a mixture of experts approach based on information from resources such as online stores. Similarly, Nyga et al. recently presented an ensemble approach where different perception techniques are combined in the context of detecting everyday objects [20]. The approach by Pangercic et al. also leverages information from online stores but in the context of object detection [21].

We predict user preferences for organizing objects based on the framework of collaborative filtering, a successful paradigm used in the data mining community for addressing personalized user recommendations [13, 14]. Recently, collaborative filtering has been applied to problems in robotics and computer vision [16, 17]. For example, Matikainen et al. combine a recommender system with a multi-armed bandit formulation for selecting good floor coverage strategies to a vacuum-cleaning robot. However, they do not consider personal user preferences related to robotic tasks. Finally, to learn different user preferences, we collect data using a crowdsourcing platform. Recently, other works have also leveraged crowdsourcing to transfer human knowledge to robots in different contexts [5, 6, 11].

$$
\begin{array}{c} \\ \\ \\ \\ \\ \mathcal{P}\ \text{(pairs)} \end{array}
\begin{array}{c}
\begin{array}{cccccc} & \multicolumn{5}{c}{\mathcal{U}\ \text{(users)}} \\ & u_1 & u_2 & & u_j & & u_N \end{array} \\
\begin{pmatrix}
p_1 & * & r_{12} & & \dots & & * \\
p_2 & r_{21} & r_{22} & & \dots & & r_{2N} \\
 & * & & \ddots & & & * \\
p_i & \vdots & \vdots & & r_{ij} & \ddots & \vdots \\
 & & & & & & \\
p_M & r_{M1} & * & & \dots & & r_{MN}
\end{pmatrix}
\end{array}
$$

Fig. 2. The ratings matrix $\mathbf{R}$. Each entry r_{ij} corresponds to the rating of a user u_j for an object-pair $p_i = \{o_k, o_l\}$, which denotes whether the two objects should be placed in the same container or not. Our goal is to predict the missing ratings denoted by * and to compute a partitioning of the objects into different containers that satisfies the user preferences.

III. Collaborative Filtering for Predicting Pairwise Preferences

The problem of predicting an object-object preference for a user closely resembles the problem of suggesting items based on user tastes. This problem is widely addressed by employing recommender systems, popularly used on websites (e.g., Amazon). They suggest items to users based on their purchase history. Instead of relating items and users, our method relates pairs of objects to users. Our technique predicts a user preference, or *rating*, for an object-pair based on two sources of information: *i*) known preferences of how the user has previously organized other objects, *ii*) how other users have organized these objects in their environments.

Let $\mathcal{O} = \{o_1, o_2, \dots, o_O\}$ be a set of objects, each belonging to a known class, e.g., book, toy, pencil, etc. We assume to have a finite number of *containers* $\mathcal{C} = \{c_1, c_2, \dots, c_C\}$, which the robot can use to organize the objects, e.g., shelves, drawers, boxes, etc. We model each container as a set which could be $\emptyset$ or could contain a subset of $\mathcal{O}$. We call $\mathcal{P} = \{p_1, p_2, \dots, p_M\}$ the set of all pairs of objects. We assign a rating r_{ij} to a pair $p_i = \{o_l, o_k\}$ to denote the preference of user u_j for placing o_l and o_k in the same container. Each rating takes a value between 0 and 1, where 1 means that the user prefers the pair together and vice versa. We construct a ratings matrix $\mathbf{R}$ of size $M \times N$, where the rows correspond to the elements in $\mathcal{P}$ and the columns to the users, see Fig. 2. Note that the ratings matrix is typically missing most of its entries. This is due to the fact that each user typically rates only a small subset of object-pairs.

A. Prediction

To predict an unknown object-pair user preference $\hat{r}_{ij}$, we take from factorization-based collaborative filtering [13, 14].

First, we decompose $\mathbf{R}$ into a bias matrix $\mathbf{B}$ and a residual ratings matrix $\overline{\mathbf{R}}$:

$$\mathbf{R} = \mathbf{B} + \overline{\mathbf{R}}. \tag{1}$$

Each entry b_{ij} in $\mathbf{B}$ is formulated as follows:

$$b_{ij} = \mu + b_i + b_j, \tag{2}$$

where μ is a global bias term, b_i is the bias of the pair p_i, and b_j is the bias of user u_j. We compute μ as the mean rating over all users and object-pairs in $\mathbf{R}$. The bias b_j describes how high or low a certain user u_j tends to generally rate object-pairs compared to the average user. Similarly, b_i captures the tendency of a pair p_i to receive high or low ratings. For example, the pair $\{$*salt*, *pepper*$\}$ tends to receive generally high ratings compared to the pair $\{$*sugar*, *tuna*$\}$.

After removing the bias, the residual ratings matrix $\overline{\mathbf{R}}$ captures fine user preferences. Due to the large amount of missing ratings, we follow the factorization procedure from Koren [13] to express $\overline{\mathbf{R}}$ as the product of an object-pair factors matrix $\mathbf{S}^T$, and a user factors matrix $\mathbf{T}$ of sizes $M \times K$ and $K \times N$, respectively. Each column $\mathbf{s}_i$ of $\mathbf{S}$ is a K-dimensional factors vector corresponding to an object-pair p_i. Similarly, each column $\mathbf{t}_j$ in $\mathbf{T}$ is a K-dimensional factors vector associated with a user u_j. We compute the residual rating $\overline{r}_{ij}$ as the dot product of the factor vectors for object-pair p_i and user u_j, i.e.,

$$\overline{r}_{ij} \; = \; \mathbf{s}_i^T \cdot \mathbf{t}_j. \tag{3}$$

The vectors $\mathbf{s}$ and $\mathbf{t}$ are low-dimensional projections of the pairs and users, respectively. Pairs or users that are close to each other in that space are similar with respect to some property. For example, some users could prefer to group objects together based on their shape, whereas others do so based on their function.

We predict the rating $\hat{r}_{ij}$ of an object-pair p_i by a user u_j

$$\begin{aligned} \hat{r}_{ij} &= b_{ij} + \overline{r}_{ij} \\ &= \mu + b_i + b_j + \mathbf{s}_i^T \cdot \mathbf{t}_j. \end{aligned} \tag{4}$$

We learn the biases and factor vectors from all available ratings in $\mathbf{R}$ by formulating an optimization problem. The goal is to minimize the difference between the observed ratings r_{ij} made by users and the predictions $\hat{r}_{ij}$ of the system over all known ratings. Let the error associated with rating r_{ij} be $e_{ij} = r_{ij} - (\mu + b_i + b_j + \mathbf{s}_i^T \cdot \mathbf{t}_j)$. We jointly learn the biases and factors that minimize the error over all known ratings, i.e.,

$$\underset{b_*, \mathbf{S}, \mathbf{T}}{\arg\min} \sum_{p_i, u_j} (e_{ij})^2 + \frac{\lambda}{2}(b_i^2 + b_j^2 + \|\mathbf{s}_i\|^2 + \|\mathbf{t}_j\|^2), \tag{5}$$

where b_* denotes all object-pair and user biases, and λ is a regularizer. To do so, we use L-BFGS optimization [19]. At every step of the optimization, we update the value of each variable based on the error gradient with respect to that variable derived from Eq. (5).

After learning the biases and factor vectors for all users and object-pairs, we use Eq. (4) to predict the requested rating $\hat{r}_{ij}$.

B. Probing

To learn the biases and factors for a user u_j for computing a prediction $\hat{r}_{ij}$ as explained in Sec. III-A, at least one entry in the jth column of $\mathbf{R}$ is required. The set of known preferences for a certain user are sometimes referred to as probes in the recommender system literature; we use *probing* to refer to the process of eliciting knowledge about a user.

In a tidy-up service robot context, we envision two strategies to do so. In the first probing approach, the robot observes $\mathcal{C}$ in the environment, detects objects in them and sets the probe ratings based on whether two objects are in the same container or not:

$$r_{ij} = \begin{cases} 1, & \text{if } o_l, o_k \in c_m \\ 0, & \text{if } o_l \in c_m, o_k \in c_n, m \neq n. \end{cases} \tag{6}$$

We compute Eq. (6) for all object-pairs that the robot observes in the environment.

In the second probing approach, we rely on actively querying the user about her preferences for a set of object-pairs. On the robot, we implemented this with a simple text-based interface. Let P be the maximum number of probe ratings that the robot queries the user. One trivial solution is to acquire probe ratings by randomly querying about P object-pairs. However, we aim at making accurate predictions with as few probes as possible. Thus, we propose an efficient strategy based on insights into the factorization of Sec. III-A. The columns of the matrix $\mathbf{S}$ can be seen as a low dimensional projection of the rating matrix describing only object-pair similarities. We cluster the columns of $\mathbf{S}$ in P groups, randomly take one column from each cluster, and query the user about the associated pair. For clustering, we use k-means with P clusters. In this way, the queries to the users are selected to capture the complete spectrum of preferences.

The nature of a collaborative filtering system allows us to continuously add probe ratings for a user, either through observations of how objects are organized in the environment or by active querying as needed.

IV. Mixture of Experts for Predicting Preferences of Unknown Objects

Thus far, we presented how our approach can make predictions when objects are included in the object database. Now, we introduce how to compute predictions when objects are not present in $\mathcal{O}$. There, we cannot rely on standard collaborative filtering to find similarities between pairs, as no user has rated pairs related to the new object yet.

The idea is to leverage the known ratings in $\mathbf{R}$ as well as mined object information from the internet. The latter consists of object hierarchies provided by popular websites, including online supermarkets, stores, dictionaries, etc. (see Fig. 3 for an example of a grocery scenario). Formally, we adopt a *mixture of experts* approach where each expert $\mathcal{E}_i$ makes use of a mined hierarchy that provides information about similarities between different objects. The idea is to query the expert about an unknown object o_*

and retrieve all the object-pair preferences related to it. The hierarchy is a graph or a tree where a node is an object and an edge represents an "is-a" relation.

As a first step, we ignore the new object and follow our standard collaborative filtering approach to estimate preferences for all the missing object-object entries of the user column, i.e., Eq. (4). To make predictions for object-pairs related to the new object, we compute the similarity ρ of o_* to other objects using the hierarchy graph of the expert. For that, we employ the wup similarity [27], a measure between 0 and 1 used to find semantic similarities between concepts

$$\rho_{lk} = \frac{depth(LCA(o_l, o_k))}{0.5(depth(o_l) + depth(o_k))}, \tag{7}$$

where $depth$ is the depth of a node, and LCA denotes the lowest common ancestor. In the example of Fig. 3, the lowest common ancestor of *Canned Tuna* and *Canned Beans* is Canned Foods, and their wup similarity is 0.4.

The idea is to use the known ratings of objects similar to o_* in order to predict the ratings related to it. For example, if $salt$ is the new object, we can predict a rating for $\{salt, coffee\}$ by using the rating of $\{pepper, coffee\}$ and the similarity of $salt$ to $pepper$. We compute the expert rating $r_{\mathcal{E}_i}(o_*, o_k)$ for the pair $\{o_*, o_k\}$ as the sum of a baseline rating taken as the similarity ρ_{*k} and a weighted mean of the residual ratings for similar pairs, i.e.,

$$r_{\mathcal{E}_i}(o_*, o_k) = \rho_{*k} + \eta_1 \sum_{l \in \mathcal{L}} \rho_{*l} \ (r(o_l, o_k) - \rho_{lk}), \tag{8}$$

where $\eta_1 = 1/\sum_{l \in \mathcal{L}} \rho_{*l}$ is a normalizer, and $\mathcal{L}$ is the set of object indices such that the rating of pair $\{o_l, o_k\}$ is known. Each expert computes Eq. (8) by using their associated hierarchy. The final prediction is a combined estimate of all the experts:

$$\hat{r}_{\mathcal{E}_*}(o_*, o_k) = \eta_2 \sum_i w_i \, r_{\mathcal{E}_i}(o_*, o_k), \tag{9}$$

where $w_i \in [0, 1]$ represents the confidence of $\mathcal{E}_i$, $\mathcal{E}_*$ denotes the mixture of experts, and $\eta_2 = 1/\sum_i w_i$ is a normalizer. We compute the confidence by a leave-one-out cross-validation of known object-object ratings as in Eq. (8) and set it to zero if it is below a threshold of 0.6. We disregard the rating of an expert if o_* cannot be found in its hierarchy or the associated ρ similarities are very small.

V. Object Grouping with Predicted Preferences

Now that it is possible to compute pairwise object preferences about known or unknown objects, we can sort the objects into different containers. In general, finding a partitioning of objects such that all pairwise constraints are satisfied is a non-trivial task. For example, the user can have a high preference for $\{pasta, rice\}$ and for $\{pasta, tomato\ sauce\}$, but a low preference for $\{rice, tomato\ sauce\}$. Therefore, we aim at satisfying as many of the preference constraints as possible when grouping the objects into $C' \leq C$ containers, where C is the total number of containers the robot can use.

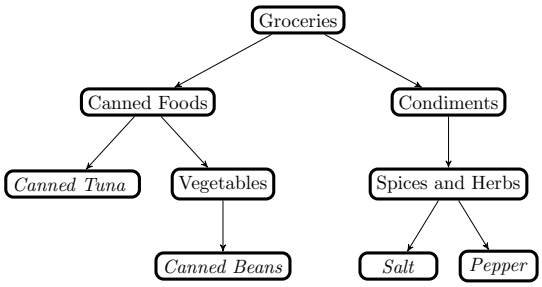

Fig. 3. Example of an hierarchy used by an expert to compute similarities across grocery items. We use this to make predictions related to a new object given its similarity to known objects in the system.

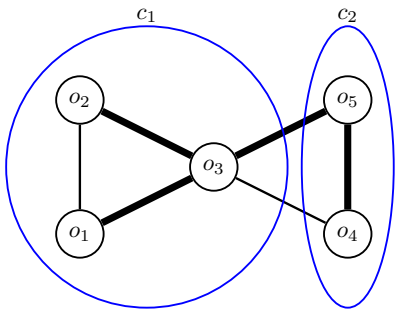

Fig. 4. A graph depicting relations between objects. Each node corresponds to an object, and the weights (different edge thickness) are the pairwise ratings. We partition the graph into subgraphs by spectral clustering. All objects in the same partition are assigned to the same container.

First, we construct a weighted graph where the nodes represent the objects, and each edge weight is the rating of the corresponding object-pair, see Fig. 4. The subdivision of objects into C' containers can be formulated as a partitioning of the graph into C' subgraphs such that the cut (the sum of the weights between the subgraphs) over all pairs of subgraphs is minimized. This is called the minimum k-cut problem [9]. Unfortunately, finding the optimal partitioning of the graph into $C' \leq C$ subgraphs is NP-hard. In practice, we efficiently solve this problem by using a spectral clustering approach [4]. The main idea is to partition the graph based on the eigenvectors of its Laplacian matrix, L, as this captures the underlying connectivity of the graph.

Let V be the matrix whose columns are the first C' eigenvectors of L. We represent each object by a row of the matrix V, i.e., a C'-dimensional point, and apply k-means clustering using C' clusters to get a final partitioning of the objects. To estimate the best number of clusters (containers) to use, we implement a self-tuning heuristic which takes the number of clusters as the number of eigenvalues of L that are equal to 0. This is a good approximation of the biggest eigen-gap, which typically indicates a reliable way to split the graph based on the similarities of its nodes [26, 28].

VI. Experimental Evaluation

We tested our method on two tidy-up scenarios: organizing toys in boxes, and organizing grocery items on shelves. We demonstrate that: *i*) our approach can accurately predict personalized user preferences for organizing objects *ii*) our mix-

Fig. 5. We consider two tidy-up scenarios: organizing toys in different boxes, and grocery items on different shelves.

ture of experts approach enables predictions for previously unknown objects, *iii*) our approach improves at the increase of the amount of user ratings, *iv*) our approach is applicable on a real tidy-up robot scenario. In all experiments, we set the number of factor dimensions to $K = 3$ and $\lambda = 0.01$.

A. Organizing Toys

In this experiment, we asked 15 people to sort 26 different toys in boxes, see Fig. 5-top. This included some plush toys, action figures, a ball, cars, a flashlight, books, and different building blocks. Each participant could use *up to* six boxes to sort the toys. Overall, four people used four boxes, seven people used five boxes, and four people used all the six available boxes to sort the toys.

We collected these results in a ratings matrix with 15 user columns and 325 rows representing all pairs of toys. Each entry in a user's column is based on whether he/she placed the corresponding objects in the same box or not, see Sec. III-B. For a fine quantification, we used these ratings to generate a bigger ratings matrix. For this, we randomly selected 78 ratings out of 325 from each column. We repeated this operation 50 times for each column and constructed a ratings matrix of size 325×750 where 76% of the ratings are missing.

We computed a factorization of the ratings matrix as described in Sec. III-A. Fig. 6-left shows the user factors $\mathbf{T}$ projected to the first two dimensions. This gives a visualization of the user tastes.

1) Predicting User Preferences for Pairs of Toys: We evaluated our approach for predicting the preferences of the 15 participants by using the partial ratings in the matrix we constructed above. For each of the participants, we queried for the ratings of P probes as described in Sec. III-B. We hid all other ratings from the user's column and predicted them using the ratings matrix and our approach. We rounded each prediction to the nearest integer on the rating scale [0,1] and compared it to the ground truth ratings. We evaluated our results by computing the precision, recall, and F-score of our predictions with respect to the two rating classes: *no* ($r = 0$), and *yes* ($r = 1$).

We set the number of probes to P = 50, 100, ..., 300 known ratings, and repeated the experiment 20 times for each value, selecting different probes each run. The mean F-scores of both rating classes averaged over all runs are shown in Fig. 6-middle, where our approach is indicated by CF. Additionally, we compare our results to three approaches: *i*) CF-rand selects probes randomly and then uses our collaborative filtering approach to make predictions; *ii*) Baseline-I uses our probing approach in Sec. III-B and then predicts each unknown pair rating as the mean rating over all users who rated it; *iii*) Baseline-II selects probes randomly and then predicts each unknown pair rating as the mean rating over all users.

Our collaborative filtering technique outperforms all baselines. On average, CF maintains an F-score between 0.98 and 0.99 over all predicted pair ratings. Using CF-rand, we are also able to achieve an average F-score of 0.98 over all runs. On the other hand, Baseline-I and Baseline-II only achieve an F-score of 0.89. These baselines are only able to make good predictions for object-pairs that have a unimodal rating distribution over all people and cannot generalize to multiple tastes w.r.t. an object-pair.

2) Sorting Toys into Boxes: We evaluated our approach (Sec. V) for grouping toys into different boxes based on the predicted ratings in the previous experiment. For each user, we partitioned the objects into boxes based on the probed and predicted ratings and compared that to the original arrangement. We computed the success rate, i.e., achieving the same number and content of boxes, see Fig. 6-right. Our approach has a success rate of 80% at $P = 300$. As expected, the performance improves with the number of known probe ratings. On the other hand, even with $P = 300$ known ratings, Baseline-I and Baseline-II have a success rate of only 56% and 58%. Whereas CF-rand achieves a success rate of 82% at $P = 300$, it requires at least 200 known probe ratings on average to obtain over 50%. On the other hand, CF achieves a success rate of 55% with only 100 known probe ratings. The probes chosen by our approach tend to correspond to pairs with multi-modal rating distributions, which is precious information to distinguish a user's taste.

3) Predicting Preferences for New Objects: We evaluated the ability of our approach to make predictions for object-pairs that no user has rated before (Sec. IV). For each of the 26 toys, we removed all ratings related to that toy from the ratings of the 15 participants. We predicted those pairs using a mixture of three experts and the known ratings for the remaining toys. We evaluated the F-scores of our predictions as before by averaging over both *no* and *yes* ratings. We based our experts on the hierarchy of an online toy store (toysrus.com), appended with three different hierarchies for sorting the building blocks (by size, color, or function). The expert hierarchies contained between 165-178 nodes. For one of the toys (flash light), our approach failed to make predictions since the experts found no similarities to other toys in their hierarchy. For all other toys, we achieved an

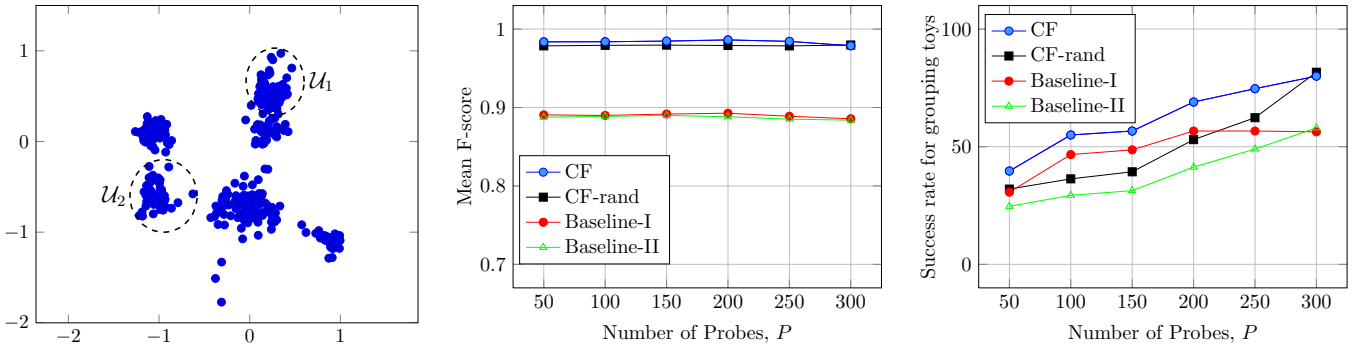

Fig. 6. Left: a visualization of user tastes with respect to organizing toys by plotting the user factor vectors projected to the first two dimensions. For example, the cluster $\mathcal{U}_1$ corresponds to users who grouped all building blocks together in one box. Cluster $\mathcal{U}_2$ corresponds to users who separated building blocks into standard bricks, car-shaped blocks, and miscellaneous. Middle: the mean F-score of the predictions of our approach (CF) in the toys scenario for different numbers of known probe ratings. We achieve an F-score of 0.98-0.99 on average over all predicted ratings. CF-rand selects probes randomly and then uses our approach for predicting. It is able to achieve an F-score of 0.98. On the other hand, baselines I and II are unable to adapt to multimodal user preferences. Right: the percentage of times our approach is able to predict the correct arrangement of boxes for sorting different toys. We outperform both baselines and improve with more probe ratings as expected, reaching a success rate of 80%. By selecting probes based on object-pair factor vectors, we are able to achieve high success rates with less probes compared to CF-rand.

average F-score of 0.91 and predicted the correct box to place a new toy 83% of the time.

B. Organizing Groceries

In this scenario, we considered the problem of organizing different grocery items on shelves. We collected data from over 1,200 users using a crowdsourcing service [1]. We consider a set of 22 common grocery item types, e.g., cans of beans, flour, tea, etc. We asked each user about her preferences for a subset of pairs related to these objects. For each pair, we asked the user whether she would place the two objects together on the same shelf. Each user could answer with *no*, *maybe*, or *yes*, which we translated to ratings of 0, 0.5, and 1, respectively. The resulting $\mathbf{R}$ has dimensions 179×1284. Each user column contains between 28 and 36 known ratings, and each of the 179 object-pairs was rated between 81 to 526 times. Only around 16% of the matrix is filled with ratings. Due to the three possible ratings and the noise of crowdsourced surveys, the ratings are largely multi-modal.

1) Predicting User Preferences for Pairs of Grocery Items: We tested our approach for predicting user ratings of pairs through 50 runs of cross-validation. In each run, we randomly sampled 50 user columns and queried them with P of their known ratings. We hid the remaining ratings from the matrix and predicted them using our approach. We rounded each prediction to the closest rating (*no-maybe-yes*) and evaluated our results by computing the precision, recall, and F-score. Additionally, we compared the predictions of our approach (CF) to CF-rand, Baseline-I, and Baseline-II as in Sec. VI-A.1. The average F-scores over all runs and rating classes are shown in Fig. 7-top for $P = 4, 8, \ldots, 20$. Both collaborative filtering approaches outperform the baseline approaches, reaching a mean F-score of 0.63 at $P = 20$ known probe ratings. Baseline-I and Baseline-II are only able to achieve an F-score of 0.45 by using the same rating of a pair for all users. Note that by employing our probing strategy, our technique is able to achieve an F-score of 0.6 with only 8 known probe ratings. On the other hand, CF-rand needs to query a user at least 12 times on average to achieve the same performance. Furthermore, we found interesting similarities based on $\mathbf{S}$. For example, users tend to rate $\{coffee, honey\}$ similarly to $\{tea, sugar\}$. Also, the closest pairs to $\{pasta, tomato\ sauce\}$ included $\{pancakes, maple\ syrup\}$ and $\{cereal, honey\}$, i.e., people often group objects based on whether they can be used together or not.

Additionally, we analyzed the average error in the predictions of each pair over all users and runs. Fig. 7-bottom shows the total number of pairs with a prediction error over 0.25 with increasing values of P. With only four known probe ratings, our approach results in an error of 0.25 or more for only 80 of the 179 object-pairs, dropping to 21 given 16 or more probes. This shows that most false predictions made by our approach correspond to mixing *no* or *yes* with *maybe*, but not *no* with *yes*. Note that using our probing method, we are able to select probes in a way that reduces the prediction error across more pairs compared to CF-rand. With a high number of probes, a random strategy offers slight advantages. Random probing is not biased to selecting samples from specific P modes, as our approach does. Note that this happens after probing with almost all pairs. On the other hand, the performance of the baselines does not improve with more probes due to multi-modal user preferences.

2) Predicting Preferences for New Objects: We defined three experts by mining the hierarchies of the groceries section of three large online stores (amazon.com, walmart.com, target.com). This includes up to 550 different nodes in the object hierarchy. For each of the 22 grocery objects, we removed ratings related to all of its pairs from $\mathbf{R}$. We used the mixture of experts to predict those ratings using the remaining ratings in each column and the expert hierarchies as explained in Sec. IV. The mean F-score over all users for three grocery objects is shown in Fig. 8-top, where the mixture of experts is denoted by $\mathcal{E}_*$. We also show the individual expert results ($\mathcal{E}_1$-$\mathcal{E}_3$) and their corresponding baseline predictions ($\mathcal{E}'_1$-$\mathcal{E}'_3$), which take only the *wup* similarity of

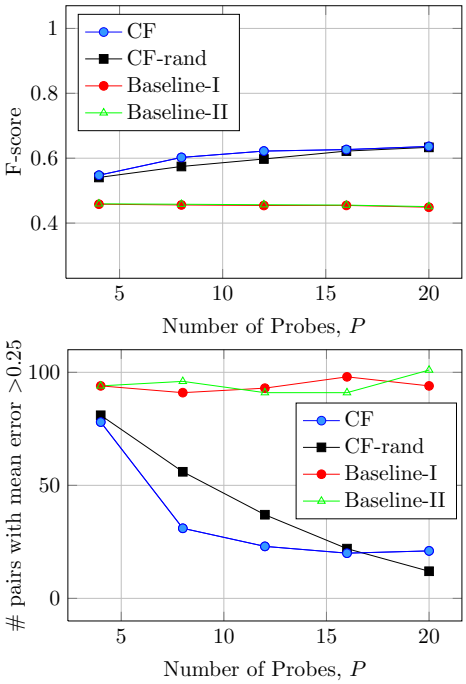

Fig. 7. Results for the scenario of organizing grocery items on different shelves. Top: the mean F-score of our predictions averaged over all rating classes *no*, *maybe*, and *yes*. Despite the large degree of multi-modality and noise in the user preferences we collected through crowdsourcing, our approach (CF) is able to achieve an F-score of 0.63 with 20 known probes and to outperform the baselines. Moreover, our performance improves with more knowledge about user preferences as expected. Bottom: the number of object-pairs where the average prediction error is higher than 0.25 on the rating scale of [0,1]. Most false predictions made by our approach correspond to only confusing *no* or *yes* with *maybe*, and not *no* with *yes*. Moreover, by selecting probes using our approach, we are able to achieve a better performance with less probes compared to CF-rand.

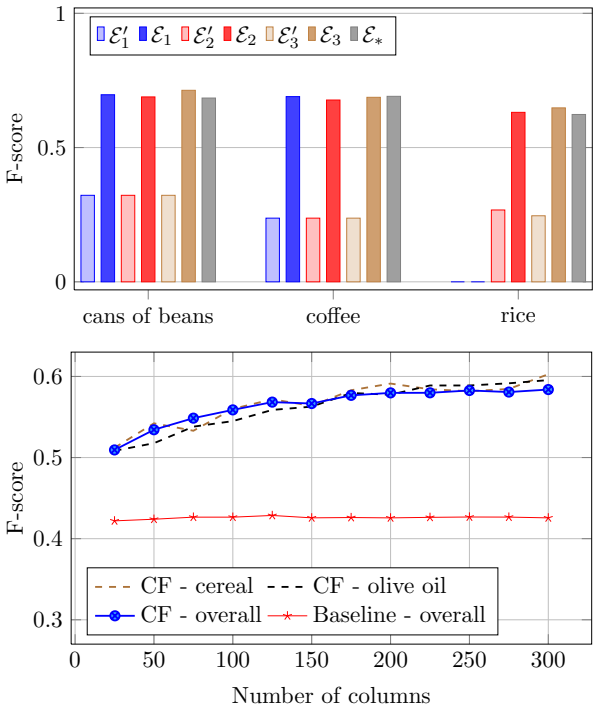

Fig. 8. Top: we predict preferences related to new objects by using a mixture of experts approach. The experts $\mathcal{E}_1$-$\mathcal{E}_3$ are based on the hierarchies of three online grocery stores. The mixture of experts $\mathcal{E}_*$ is a merged prediction of all three experts based on their confidence for a specific user. Therefore, it is able to recover if a certain expert cannot find similarities for a new object, as in the case of *rice*. The predictions $\mathcal{E}'_1$-$\mathcal{E}'_3$ make predictions based only on the similarity of two items without considering the ratings of similar pairs rated by the user, see Sec. IV. Bottom: as soon as some users have rated pairs related to a new object, our collaborative filtering approach is able to make predictions about it. The performance improves with more users rating pairs related to a certain object.

two objects as the rating of the pair but do not consider the ratings of similar pairs. The results of each individual expert significantly outperform the baseline predictions. Note that $\mathcal{E}_*$ is able to overcome the shortcomings of the individual experts, as in the case of *rice*. There, $\mathcal{E}_1$ is unable to find similarities between *rice* and any of the rated objects, whereas $\mathcal{E}_2$ and $\mathcal{E}_3$ are able to relate it to *pasta* in their hierarchies. For two of the objects (*bread* and *candy*), we were unable to make any predictions, as none of the experts found similarities between them and other rated objects. For all other objects, we achieve an average F-score of 0.61.

3) Improvement with Number of Users: We evaluated our approach with respect to the number of users in the system. For each object, we removed from $\mathbf{R}$ all columns with ratings related to that object. Over 20 runs, we randomly sampled 10 different user columns from these and hid their ratings for pairs related to the object. We predicted those ratings using our approach (Sec. III-A) by incrementally adding more columns of the other users who rated that object to the ratings matrix in increments of 25. We evaluated the mean F-score for the predictions for the 10 users. The results are shown in Fig. 8-bottom averaged over 20 different types of objects (those where we had at least 300 user ratings). We also show the improvement with respect to two of the objects individually. The performance of our approach improves steadily with the number of users who rate pairs related to a new object, as opposed to a baseline that updates the mean rating over all users and uses that for predicting. This shows that collaborative filtering is suitable for lifelong and continual learning of user preferences.

4) Real Robot Experiments: We tested our approach on a real tidy-up robot scenario, see Fig. 9. We asked 15 people to organize 17 different grocery items according to their preferences, using *up to* six shelves. Four people grouped the items on four shelves, three people used five shelves, and eight people used all six shelves. We constructed the corresponding user columns and added them to the ratings matrix from the crowdsourcing surveys. We then conducted 25 experimental runs where we selected a random user column and organized the shelves as he/she did in the survey. We randomly selected two objects, removed them from their shelves, and placed them on a table. The task of the robot is to fetch those objects and place them back on the shelves based on the predictions of our approach, see Fig. 1 for an example where the robot successfully placed *coffee* on the same shelf as *tea*. In this work, we rely on existing techniques for object recognition. In each run, we used fiducial markers to recognize the objects on the table and

Fig. 9. The robot has to assign the two objects that are on the table to shelves according to predicted user preferences. In this example, the robot places *coffee* on the same shelf as *tea*, and *rice* next to *pasta*.

provided the robot with information about the objects on the shelves to use as probe ratings, see Sec. III-B. For navigating between the table and the shelves, we relied on a state-of-the-art planner [7] that generates a sequence of actions for the robot, and formulated goals based on the target shelf of each object. For manipulation, we used an out-of-the-box motion planner. Overall, our approach predicted the correct shelf assignment for 82% of the objects, either by placing them in empty shelves or together with other objects. A video showing parts of the experiments can be found at: `http://www.informatik.uni-freiburg.de/%7Eabdon/videos/icra15/abdo_icra15.mp4`.

VII. Conclusions

This paper presents a novel approach that enables robots to predict user preferences with respect to tidying up objects in containers, such as shelves or boxes. In our approach, we first predict pairwise object preferences of the user by formulating a collaborative filtering problem. Then, we subdivide the objects in containers by modeling and solving a spectral clustering problem. Our technique allows for easily updating knowledge about user preferences, does not require complex modeling, and improves with the amount of user data, allowing for lifelong learning of user preferences. We trained the system by using surveys from over 1,200 users through crowdsourcing, and thoroughly evaluated the effectiveness of our approach for two tidy-up scenarios: arranging groceries on shelves and sorting toys in boxes. Additionally, we presented an experiment with a real robot that has to arrange groceries on shelves. Our technique is accurate and is able to sort objects into different containers according to user preferences.

References

[1] Crowdflower crowdsourcing service. http://crowdflower.com/.

[2] A. Aydemir and P. Jensfelt. Exploiting and modeling local 3d structure for predicting object locations. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2012.

[3] M. Cakmak and L. Takayama. Towards a comprehensive chore list for robots. In *Int. Conf. on Human-Robot Interaction (HRI)*, 2013.

[4] F. R. K. Chung. Spectral graph theory. In *CBMS Regional Conference Series in Mathematics*, 1996.

[5] M. J.-Y. Chung, M. Forbes, M. Cakmak, and R. P. Rao. Accelerating imitation learning through crowdsourcing. In *Int. Conf. on Robotics & Automation (ICRA)*, 2014.

[6] J. Deng, J. Krause, and L. Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[7] C. Dornhege and A. Hertle. Integrated symbolic planning in the tidyup-robot project. In *AAAI Spring Symposium*, 2013.

[8] J. Forlizzi and C. DiSalvo. Service robots in the domestic environment: A study of the roomba vacuum in the home. In *Int. Conf. on Human-Robot Interaction (HRI)*, 2006.

[9] O. Goldschmidt and D. S. Hochbaum. A polynomial algorithm for the k-cut problem for fixed k. *Mathematics of Operations Research*, 1994.

[10] J. Hess, G. D. Tipaldi, and W. Burgard. Null space optimization for effective coverage of 3D surfaces using redundant manipulators. In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2012.

[11] A. Jain, D. Das, and A. Saxena. Planit: A crowdsourcing approach for learning to plan paths from large scale preference feedback. Technical report, 2014.

[12] Y. Jiang, M. Lim, and A. Saxena. Learning object arrangements in 3d scenes using human context. In *Int. Conf. on Machine Learning (ICML)*, 2012.

[13] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Int. Conf. on Knowledge Disc. and Data Mining (SIGKDD)*, 2008.

[14] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. In *ACM Trans. on Knowledge Disc. from Data (TKDD)*, 2010.

[15] L. Kunze, C. Burbridge, and N. Hawes. Bootstrapping probabilistic models of qualitative spatial relations for active visual object search. In *AAAI Spring Symposium Series*, 2014.

[16] P. Matikainen, R. Sukthankar, and M. Hebert. Model recommendation for action recognition. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[17] P. Matikainen, P. M. Furlong, R. Sukthankar, and M. Hebert. Multi-armed recommendation bandits for selecting state machine policies for robotic systems. In *Int. Conf. on Robotics & Automation (ICRA)*, 2013.

[18] S. Miller, J. Van Den Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel. A geometric approach to robotic laundry folding. *Int. J. of Robotics Research (IJRR)*, 2012.

[19] J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 1980.

[20] D. Nyga, F. Balint-Benczedi, and M. Beetz. PR2 looking at things: Ensemble learning for unstructured information processing with markov logic networks. In *Int. Conf. on Robotics & Automation (ICRA)*, 2014.

[21] D. Pangercic, V. Haltakov, and M. Beetz. Fast and robust object detection in household environments using vocabulary trees with sift descriptors. In *Workshop on Active Semantic Perception and Object Search in the Real World at the Int. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[22] C. Pantofaru, L. Takayama, T. Foote, and B. Soto. Exploring the role of robots in home organization. In *Int. Conf. on Human-Robot Interaction (HRI)*, 2012.

[23] C. Ray, F. Mondada, and R. Siegwart. What do people expect from robots? In *Int. Conf. on Intelligent Robots and Systems (IROS)*, 2008.

[24] A. Saxena, J. Driemeyer, and A. Y. Ng. Robotic grasping of novel objects using vision. *Int. J. of Robotics Research (IJRR)*, 2008.

[25] M. Schuster, D. Jain, M. Tenorth, and M. Beetz. Learning organizational principles in human environments. In *Int. Conf. on Robotics & Automation (ICRA)*, 2012.

[26] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 2007.

[27] Z. Wu and M. Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*, 1994.

[28] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.