Learning User's Preferred Household Organization via Collaborative Filtering Methods

Stephen Brawner
Brown University
Providence, RI
brawner@cs.brown.edu

Michael L. Littman
Brown University
Providence, RI
mlittman@cs.brown.edu

ABSTRACT

As learning robots and smart devices become common household occurrences, their users will be required to invest more time to train them on the details specific to their household and lifestyle. This *burden of personalization* may eventually become a roadblock to the adoption of smart devices and robots. We are interested in reducing the burden of personalization by leveraging learned information from other households. However, machine learning methods incorporating such data will require smart recontextualizations that can map the preferences from a collection of similar users onto the user's own household space. We present several collaborative filtering based methods to solve the problem of a robot organizing the items in a kitchen for their user: a traditional collaborative filtering method based on prior work incorporating user's item–item distance ratings, and a context-aware collaborative filtering method, which enables direct learning of item–location ratings. We present results on user-annotated kitchens.

CCS Concepts

•**Information systems** → **Recommender systems;** •**Applied computing** → *Command and control;*

Keywords

applications, robotics, context-aware, collaborative-filtering

1. INTRODUCTION

As learning robots become common household items, people will be required to train their robot on the details specific to their household and lifestyle. In the context of household automation, smart devices like the Nest thermostat can learn user's preferred heating and cooling schedules from tracked manual inputs [11]. Even more so than smart devices, robots will need to learn significantly larger amounts of household information, putting a substantial burden on the user for training. As this *burden of personalization* becomes more commonplace in home automation and robotics, users may find themselves investing more and more time into training their smart devices and robots. We are interested in reducing the burden of personalization by leveraging learned information from other households.

The challenge lies in the inability to directly transfer learned information from one household to another, due to uniqueness of household designs, schedule, demographics and composition of the household members, and the combined preferences of all users interacting with the system. Therefore, it is important to identify certain contextual facets of the learning problem to work on.

We consider one such personalization problem in the area of household robotics: the problem of a robot tasked with learning the organization of items in a user's kitchen like putting away groceries or the items from a dishwasher. Asking for the location of every item would be time consuming—possibly more so than having the person put away the items themselves. Therefore, we were motivated to design an algorithm that leverages the organization of other users' kitchens to help predict the user's organization of items.

In the context of videos, products, and news articles, collaborative filtering techniques used for recommender systems have been developed to transfer the experience of other users to help identify items a target user would find desirable. Typically, these techniques predict ratings of user-item pairs, that is, how would a given user rate an item based on their past ratings [18]. Recently, *context* has been incorporated into recommender systems to improve their performance—a crucial element for our application.

Context-aware collaborative filtering, for example as presented by Panniello et al. [12] and Rendle et al. [17], assumes that more information exists that predicts or informs a user's choice of ratings than just the item itself. Context may include information about the user or about the item being rated. In the case of household organization, the context can include the set of possible locations, the identities of the items known to be in those locations, or categorical information related to the items and locations. For this application, we envision a robot with sufficient perceptual abilities to identify these pieces of context.

One of our proposed methods employs a context-based recommender system, factorization machines (FMs) [15], to predict a user's item–location ratings—the degree of acceptability of the placement of a given item at a given location in a kitchen. In contrast, we also present an alternative collaborative filtering method that predicts locations based on the item's predicted rating distance to items already placed in that location, which builds directly on prior work [1]. The important difference is, while the previous collaborative filtering technique required explicit item–item ratings, the FM method enables direct predictions of item–location ratings, which can even be boosted by learning on the contextual features of these variables.

We present data collected from participants on Amazon Mechanical Turk who annotated kitchens with locations of items. We describe several methods for enabling a learning robot to predict item–location ratings for novel users from this data. We also discuss how these ratings can be used in an interactive system that attempts to find a trade-off between asking the user for the correct input and making a wrong choice. We demonstrate results on a collection of user-annotated, simulated kitchen examples.

2. RELATED WORK

Researchers have long argued that user interfaces can benefit from learning from their users [10]. However, devices in the home like the Nest thermostat [11] require a significant amount of learning effort. Yang et al. [21] reported that early users struggled with understanding what the Nest thermostat learned and that these users found it hard to override its learned behavior.

Home organization is a good example of a task that both exhibits strong user preferences and is highly desirable to automate. Several researchers found that cleaning and organization are the top two tasks users most want robots to do [20, 4]. Pantofaru et al. [13] argue that, unlike cleaning tasks, organization is "nuanced and emotional." They also found that, even if they could afford the help, many people will not hire human assistants to organize their belongings because it is too personal of a problem. Therefore, a robot learning this task must tread carefully by weighing the cost of requiring too many user interactions and misplacing items.

Several researchers in the robotics community have begun to look at placing and organizing items. Cha et al. [5] examined methods for predicting locations of items in a user's kitchen from other items in the kitchen. Using item-related features like item-type and use, they found that a Support Vector Machine classifier (SVM) performed the best in their domain.

However, in collaborative filtering, SVM methods are often disregarded due to their poor performance on sparse datasets. We present as a baseline a Support Vector Regression method that struggles on the full data set when considering individual locations, but performs better only over item and location features. However, our proposed solutions still outperform this SVR method.

Fisher et al. [6] presented a probabilistic model that can generate plausible scenes of items from user-provided examples. Leveraging a larger scene database, they can create arrangements of novel items suitable for the user. Collaborative filtering methods extend beyond this type of generative model by incorporating preferences of other users to predict given user's preferences. Jiang et al. [7] discuss a method for placing items optimizing for stable and semantically relevant locations, but do not capture a user's preferences for locations among semantically identical locations. Schuster et al. [19] learn organizational principles to place items into meaningful locations, but do not make use of user preferences to select locations. However, a robot utilizing our algorithms could also easily incorporate these capabilities of identifying free space and stable placement poses within the specific location chosen by our prediction system.

The work by Abdo et al. [2] most closely mirrors the contributions of our paper. They use collaborative filtering techniques on item–item pair ratings and then approximately solve a minimum k-cut problem to best group the items from their predicted ratings. Using these groupings, they then place the items into semantically identical bins or shelves. We build on this work to solve the problem of predicting specific locations for items in a kitchen, as opposed to just their general groupings. We present several methods to solve this more general problem, including a method to adapt their technique over collocation data to predict suitable locations. We further present a context-aware collaborative filtering technique that leverages item–item collocation information without its explicit representation in the training data.

3. METHOD

Our proposed solutions predict a user's preferred locations for items they want organized. It consists of two components, a rating prediction collaborative filter and an algorithm for producing the optimal location given a set of location ratings produced from the collaborative filter.

There are two solutions we present here. The first builds on previous work by using a collaborative filter to predict a user's item–item ratings – whether the two items should be placed together or not. Predicting the correct location is a matter of choosing the location which contains the item with lowest distance rating to the item being placed for the evaluated user. The second uses a context-aware collaborative filter to directly predict a user's item–location ratings. The predicted location is just the location with the minimum item–location rating for the user.

We envision this work as a component in a robot's back and forth interactions with their users. To minimize the number of interactions required at the risk of misplacing objects, we also present a method for tuning the predictive success of the location-prediction system by allowing the system to choose to place based on its predicted value or ask the user for the actual location.

4. RECOMMENDER SYSTEMS

Recommender systems are concerned with the problem of predicting user–item ratings. That is, given user u, what rating $r \in \mathbb{R}$ would they assign to the item i? For the problem of household organization, we modify the problem so that the system makes predictions on item–item ratings or item–location ratings. This modification makes our problem a variant of the classical recommender problem in that users provide ratings on either item–item pairs or item–location pairs.

Collaborative filtering is a category of methods for recommender systems that seek to predict ratings for items novel to a user from ratings of the items from similar users. The typical input is a rating matrix $R \in \mathbb{R}^{M \times N}$, where rows are associated with the items, and the columns are associated with users. Factorization methods attempt to generate a lower dimensional representation to improve generalization.

As shown in Abdo et al. [2], we decompose this matrix into:

$$R = B + \bar{R} \tag{1}$$

where B is a bias matrix that encodes the global bias, item bias and user bias, and $\bar{R}$ is the residuals matrix that the collaborative filtering algorithm attempts to learn. We use the L-BFGS minimization algorithm [9] to find the factorization of the residual matrix and bias matrix that minimizes the regularized squared error.

4.1 Context-aware recommender systems

Compared to classical recommender systems, context-aware recommender systems assume that some additional information exists that relates to the user's choices of ratings. For example, a user may choose a location based on the other items already placed there or based on the location's salient features. We include information about the locations into the context-based recommender, but our chosen context-aware recommender also learns automatically information related to item–item collocation to inform its predictions.

4.2 Factorization Machines

Rendle et al. [15] developed factorization machines to make predictions in a model with all multi-degree interactions among the context variables with sparse data in linear time. For a system that models only interactions between two variables at a time, the model equation is:

$$\hat{y}(\mathbf{x}) := w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} (\mathbf{v_i} \cdot \mathbf{v_j})\, x_i x_j, \tag{2}$$

where $\mathbf{x}$ is the input variable vector, w_0 is the global bias, $\mathbf{w}$ is the weights over $\mathbf{x}$. $(\mathbf{v_i} \cdot \mathbf{v_j})$ is the dot product of $\mathbf{v_i}$ and $\mathbf{v_j}$ and models the interaction of the i-th and j-th variable, where $\mathbf{v_{i/j}} \in \mathbf{V}$ is a factorized representation of the weighting of $x_{i/j}$.

Instead of estimating an individual weight parameter ($w_{i,j} \in \mathbb{R}$) for higher-order interactions ($d \geq 2$), the factorization of $\mathbf{v}$ enables modeling even under sparsity. Here, w_0, $\mathbf{w}$ and $\mathbf{V} \in \mathbb{R}^{\mathbf{n \times k}}$ are the parameters to be estimated.

Context does not need to be explicitly represented because it can be learned through the factorization of $\mathbf{v_i}$. Though we show that the item–item collocation has very powerful explanatory power, we do not need to represent that information in the context variables. Instead, we include users, items, locations and features, and the optimization of the model will seek to explain these ratings appropriately.

Though the model presented in equation 2 suggests a solution would require a run time of $\mathcal{O}(kn^2)$ where k is the number of dimensions and n is the number of context variables, they show this equation can be reformulated into a linear solution with runtime complexity $\mathcal{O}(kn)$. Indeed, for models with higher degrees of interactions, it can still be refactored into a linear time solution.

5. LOCATION SELECTION

One difficulty of choosing a correct location for an item is that many locations in a kitchen are interchangeable and are justifiable placement locations ignoring the current organization. A user's preferred organization therefore requires the robot to identify the most likely location from existing item placements and physical suitability of the location for the item.

For our work, we chose to code ratings as 0.0 being a positive example and 1.0 as a negative example, though the reverse – 0.0 for negative and 1.0 for positive – is equally valid. For item–location pairs, a rating, $r(i, l) = 0.0$ indicated the item is in the preferred location and 1.0 indicated it is not. For item–item pairs, a rating, $r(i, i') = 0.0$, indicated the items should be located together. Rating predictions produced by the collaborative filtering methods are real valued approximately in the range of $[0, 1]$.

5.1 Choosing a Location from Item–Item Pairs

In our first approach, item–item ratings $r(i, i')$ are learned from training input via a context-unaware collaborative filtering technique [2]. Finding a suitable location for an item in the kitchen then requires finding the location that results in the best match to the item–item ratings. If a location is empty, the method takes the mean pair rating as its default.

Formally, for items $i, i', j, j' \in I = \{i_1, i_2, \cdots, i_m\}$, and location $l \in L = \{l_1, l_2, \cdots, l_n\}$ where $i \neq i'$ and $j \neq j'$, n is the number of locations and m is the number of items, we want to find the item–location rating $R_{\text{item–item}}$ summarized from item–item ratings $r(i, i')$.

$$R_{\text{item–item}}(i, l) = \begin{cases} \min_{i' \in l} r(i, i') & \text{if } \exists i' \in l, \\ \frac{1}{m^2} \sum_{j \in I} \sum_{j' \in I} r(j, j') & \text{otherwise.} \end{cases} \tag{3}$$

We then select the location with the lowest distance rating. We denote the event of item i being placed in location l as $i \in l$ and write the selected location as

$$\hat{l} = \operatorname*{argmin}_{l} R_{\text{item–item}}(i, l). \tag{4}$$

5.2 Predicting Location Ratings through FMs

We use the factorization machine's model to predict item–location ratings. Each row in the input matrix includes context variables for the user, item, location, and a location category. For each provided example of an item–location pairing in the input data, we assign a distance rating of 0. We produce $n - 1$ other ratings of value 1 for this same item in all the other locations in the kitchen. Each location variable is encoded uniquely for the user, even if in experiments different users annotated the same kitchens.

In Equation 5, we show the context variable vector we used in our dataset. Each context variable is assigned a value of 1 when active and 0 otherwise. In our notation, u is a user, i_i is an item, l_j is location, and $\Phi(l)$ are the feature values over the active location variable l:

$$\mathbf{x} = \{u_1, \cdots, u_U, i_1, \cdots, i_m, l_1, \cdots, l_n, \Phi(l)\}. \tag{5}$$

Because the model directly learns item–location ratings $R_{\text{item–location}}(i, l) = r(i, l)$, we can find the location with the lowest distance rating:

$$\hat{l} = \operatorname*{argmin}_{l} R_{\text{item–location}}(i, l). \tag{6}$$

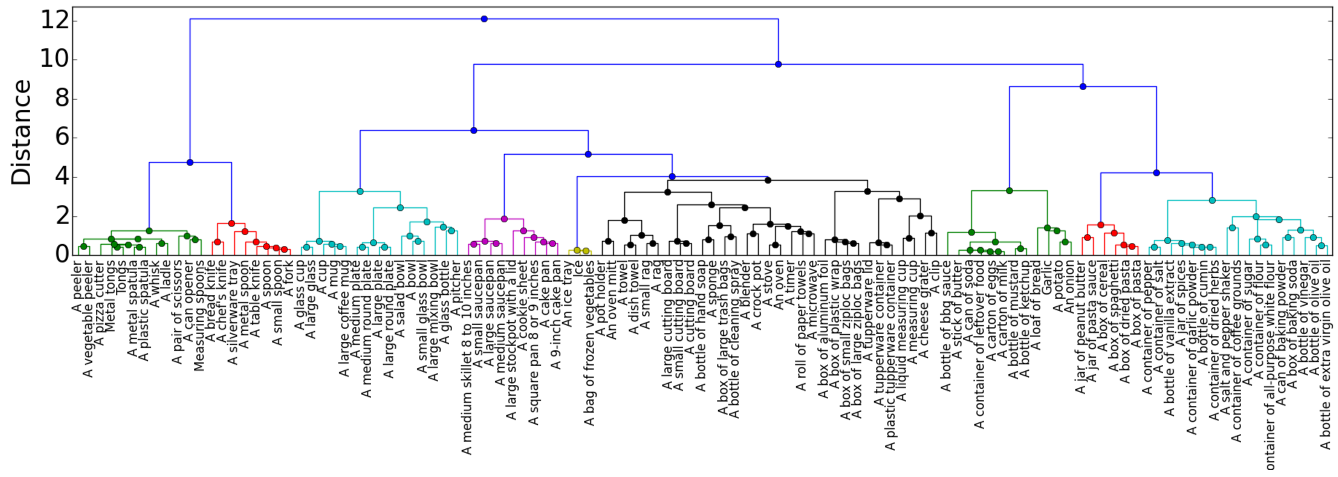

Figure 1: A hierarchical clustering of 111 kitchen items and their mean pair ratings. Groups were colored for distances less than or equal to 4

5.3 Combination Method

Both techniques, predicting via item–item pairs and item–location pairs, provide ratings for an item in a location. We can combine the two values in a number of ways. We chose harmonic mean to capture information provided by both ratings without being unduly influenced by the relative offset biases between the two types of ratings.

$$\hat{l} = \underset{l}{\text{argmin}} \left(\frac{2.0 R_{\text{item–location}}(i,l) * R_{\text{item–item}}(i,l)}{R_{\text{item–location}}(i,l) + R_{\text{item–item}}(i,l)} \right). \tag{7}$$

6. DATA COLLECTION

To create a data set with maximum opportunities for generalizing between users, we were interested in finding a collection of items that many people would be familiar with and would have in their households.

6.1 Finding common kitchen items

To create our item set, we began by pulling a list of recommended cooking items from 'How to Cook Everything' by Bittman [3]. Their recommended items consisted of cooking tools and basic foods useful for cooking a variety of recipes. We also examined the list of items identified by Cha et al. [5] in their CMU kitchen dataset. They surveyed several households and annotated all the items found in their participants' kitchens. We merged the items across these sources and removed duplicates—items that we judged to be the same as an item already in the list—and were left with 398 items. We replaced items commonly found in a container, like seasonings, oils or other liquids, with the appropriate container of that item ('olive oil' became 'a bottle of olive oil').

We randomly split the selected items into surveys of 10 items, and one survey of 8 item and asked participants on Amazon Mechanical Turk if they had any of the items in their household kitchen. For each survey, we collected 20 responses. Participants were allowed to answer as many different surveys as they chose (min: 1, max: 30, mean: 10.5). No one participant completed all the surveys.

From the full set, we selected the items at least 85% of respondents indicated were in their households, producing a list of 111 items to be used for location annotations.

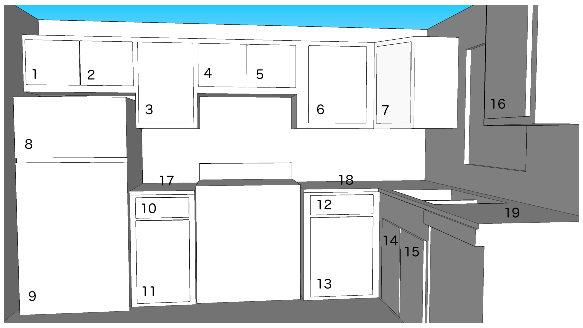

Figure 3: Diagram of a kitchen with numeric labels.

6.2 Kitchen annotations

We were provided four kitchen layouts from the authors of the CMU Kitchen Dataset. We labeled each drawer, cabinet, refrigerator, freezer and counter with a unique numerical identifier. Figure 6.2 shows one of the labeled kitchen images we provided for a survey. We created four different surveys, each with a different labeled kitchen. We asked participants to assign locations in their labeled kitchen for all 111 items. Participants were asked not to respond to more than one survey. For the participants who took multiple surveys, we only kept responses to their first. We received 25 responses to each survey. We removed three responses from two groups due to duplication of users. Ten participants from the previous task also completed this one.

To summarize patterns in how objects were placed closely to one another, we built the hierarchical clustering of item–item pair distances shown in Figure 1. Over all users, and for each item–item pair, we computed the fraction of times the items were not placed together. Groups of items with distances less than or equal to 4 are given unique colors in the diagram. The figure illustrates that items fall into clear categories in their placement around the kitchen. For example, 'a glass cup', 'a large glass', 'a cup', 'a mug', and 'a large coffee mug' were commonly placed together.

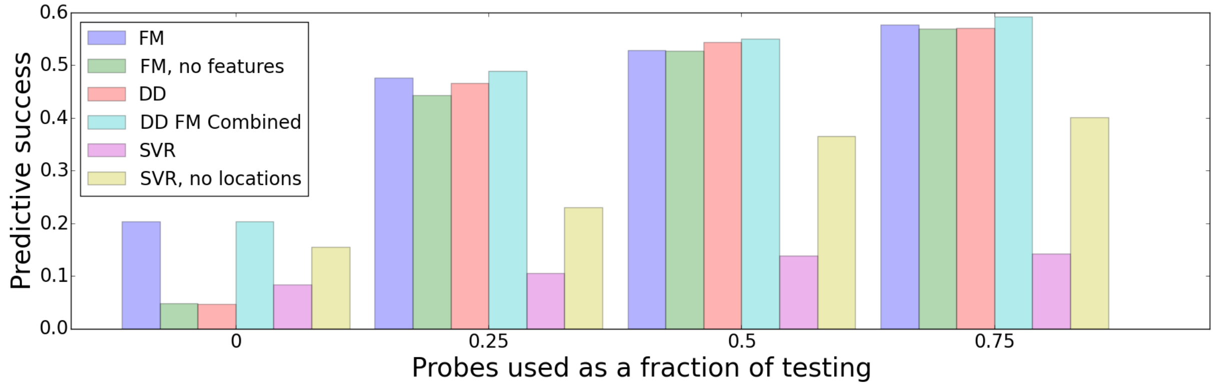

Figure 2: Prediction accuracy versus the number of probes for several different learning methods.

7. EXPERIMENTS

From the data presented in Section 6.2, we created four sets of training and testing data by assigning data collected for three kitchens to training and the fourth to testing. Even though many people annotated the same kitchen, we used a variable for a location unique to each user. For each item and location in a kitchen, we created a collocation data set by indicating for each item–location pair in the original data set whether it was found with one of the other 111 items. That is, we changed the data lines from rating–item–location to rating–item–item–location. We set the associated rating to 0.0 if the two items were found together in the location and 1.0 otherwise.

For the item–location dataset, probes were drawn from the testing data. For each user in the testing data, we selected a random set of items and moved the ratings of that user for those items to the training dataset. We used the same sets of items for the collocation dataset and created a set of probes for all item pairings in this set of items.

7.1 Model configuration

We trained the factorization-machine model (FM) using libFM [16] on the training data to produce predictions for each user–item–location in the testing dataset. We used adaptive stochastic gradient descent with a learning rate of 0.03, a sampling standard deviation of 0.1 and 30 for the number of dimensions of the model. The item–location tuple with the lowest predicted rating was chosen as the predicted location of the item.

For the data-driven factorization method (DD), we trained the collaborative filter on the collocation training dataset without the additional probes. The probes were then used to update the model through the new-user update method presented by Abdo et al. [2]. As done in their work, we also built our model with three dimensions for the factorization. The limited memory BFGS minimization algorithm from SciPy [8] was used to find the optimal variables, with a stopping criteria 'factr' set to 10.

We included a support vector regression (SVR) model to predict ratings from the dataset. The SVR model was trained on the item–location data set. We used the implementation provided in scikit-learn [14]. The user columns of the item–location dataset were removed to reduce sparseness. We used a radial basis function kernel, with the default gamma of $\frac{1}{N}$ where N is the number of features. The stopping tolerance was set to 0.0001, and no shrinkage was used.

We trained both SVR models and one FM model with features. For both, we included features about the locations describing the location type: drawer, low cabinet, high cabinet, counter, refrigerator, freezer. Additionally, for the SVR models, we also included item features related to the use of the item: edible, drinkable, food storage, etc.

For the combined model, we used the FM model trained with features and the DD method discussed above.

7.2 Results

Figure 2 presents the predictive success of the non-probe items in the testing dataset for two FM models, a data-driven factorization model over item collocation data (DD), a model combining the FM model and DD model (DD FM Combined), and two SVR models. We measured predictive success as the fraction of times the model predicted the correct location a user chose to place an item.

For the two FM models, we present a model trained as we described in previous sections (FM), and also a model trained on these variables but without location features (FM, no features). For the two SVR methods, we show the results of one model trained using location variables (SVR) and one without (SVR, no location). We show results for different fractions of items used as probes, specifically 0, $\frac{1}{4}$, $\frac{1}{2}$ and $\frac{3}{4}$, corresponding to 0, 28, 56, and 83 probes out of the 111 total items.

As seen in this graph, the proposed methods perform quite strongly for this difficult task. Even with no information about a current user (0 probes), the FM and DD FM combined model can correctly predict a significant fraction of locations for that user. The FM, no features and the DD methods perform strongly when probes are available, but reduce to random guessing when 0 probes are used. This is due to the limitations of collaborative filtering without additional contextual information. Without any information from the user, the base performance is generally quite poor.

The FM models both performed strongly over a wide range of probes. Both of the FM models were trained without explicit collocation data, yet both performed as well as the data-driven model, implying that the FM model is correctly learning the importance of item–item pairings for prediction. Indeed, the success of the DD method to predict locations when probes were available illustrates the importance of the inherent item–item affinities presented earlier.

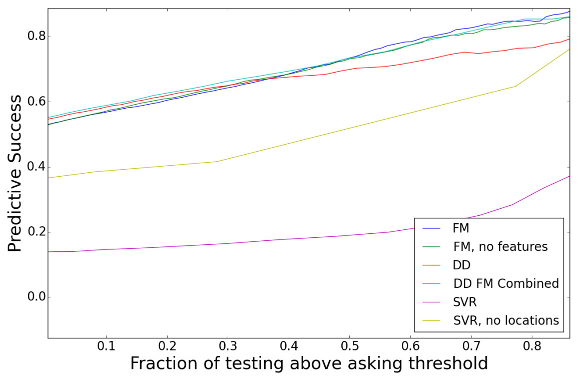

Figure 4: Predictive success versus the fraction of testing that did not meet the minimum required threshold for placement. Lower fractions imply more locations were chosen, with less placement confidence.

The DD FM Combined model outperformed the FM models and the DD model by themselves when probes are provided. It combined the learning of the FM model on location features with the high accuracy of item–item ratings provided by the DD model.

Both SVR methods performed worse than the proposed solutions regardless of the number of probes presented. The SVR method trained with locations performed especially poorly. The locations were uniquely encoded for each individual user, which created a highly sparse input that posed difficulty to this method. By excluding locations, the SVR no locations method performed better because this input is significantly less sparse. Regardless, the collaborative filtering methods strongly outperformed the SVR methods.

7.3 Placement-versus-asking trade-off

An important design consideration for an interactive system is the trade-off between placing an item incorrectly and asking the user for the correct placement. We observed that each algorithm places higher confident answers closer to the ranges of their output. Therefore, we can enable systems to ask questions for uncertain items by requiring a threshold for ratings.

Figure 4 plots the success of each algorithm against the fraction of items in the testing dataset that do not meet the threshold for placement with 50% of probes already placed. We normalized the predictions of each algorithm between 0 and 1 to simplify the comparison.

Data points at a fraction of 0 correspond to data presented in Figure 2 for 50% probes. The DD FM Combined method remains among the strongest overall. Data for fractions above 0.85 were not included because there are too few data points for accurate predictive success.

8. CONCLUSIONS

In this work, we presented several methods for reducing the burden of personalization that comes from users supplying a learning system household organization preferences. We collected data about the existence, pairing and location of items in participants' kitchens. We found that people have a lot of items in common but also many unique items not found in other kitchens. A system that recognizes a location suitable for an item could easily still fail to match the user's preference for its location. In response, we presented a solution that incorporates a user's preferences but enables an autonomous system to make use of perceivable features of these locations when learning.

We showed that a context-based collaborative filtering model called factorization machines is well suited to predict item–location ratings. Choosing the appropriate location is a simple task of finding the location with the best rating. For interactive systems, we show the value in trading off asking for a location versus placing it incorrectly.

Though household robotics entails a sizable burden of personalization, this issue also plagues many other modern smart-household devices like thermostats and lighting. Machine-learning methods that use data from other households in learning a user's preferences will require smart recontextualizations that can map the preferences drawn from a collection of different users onto the user's own household space. We envision these topics as bright areas of future research.

References

[1] N. Abdo, C. Stachniss, L. Spinello, and W. Burgard. Collaborative filtering for predicting user preferences for organizing objects. *arXiv preprint arXiv:1512.06362*, 2015.

[2] N. Abdo, C. Stachniss, L. Spinello, and W. Burgard. Robot, organize my shelves! tidying up objects by predicting user preferences. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 1557–1564. IEEE, 2015.

[3] M. Bittman. *How to Cook Everything: 2,000 Simple Recipes for Great Food.* John Wiley & Sons, 2011.

[4] M. Cakmak and L. Takayama. Towards a comprehensive chore list for domestic robots. In *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*, pages 93–94. IEEE Press, 2013.

[5] E. Cha, J. Forlizzi, and S. S. Srinivasa. Robots in the home: Qualitative and quantitative insights into kitchen organization. In *HRI*, pages 319–326, 2015.

[6] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan. Example-based synthesis of 3d object arrangements. *ACM Transactions on Graphics (TOG)*, 31(6):135, 2012.

[7] Y. Jiang, C. Zheng, M. Lim, and A. Saxena. Learning to place new objects. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3088–3095. IEEE, 2012.

[8] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. [Online; accessed May 19, 2016].

[9] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

[10] P. Maes et al. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):30–40, 1994.

[11] Nest. Home | nest. https://nest.com/, 2016. Accessed: 2016-4-1.

[12] U. Panniello, A. Tuzhilin, M. Gorgoglione, C. Palmisano, and A. Pedone. Experimental comparison of pre-vs. post-filtering approaches in context-aware recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, pages 265–268. ACM, 2009.

[13] C. Pantofaru, L. Takayama, T. Foote, and B. Soto. Exploring the role of robots in home organization. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pages 327–334. ACM, 2012.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[15] S. Rendle. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 995–1000. IEEE, 2010.

[16] S. Rendle. Factorization machines with libFM. *ACM Trans. Intell. Syst. Technol.*, 3(3):57:1–57:22, May 2012.

[17] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 635–644. ACM, 2011.

[18] F. Ricci, L. Rokach, and B. Shapira. *Introduction to recommender systems handbook*. Springer, 2011.

[19] M. J. Schuster, D. Jain, M. Tenorth, and M. Beetz. Learning organizational principles in human environments. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3867–3874. IEEE, 2012.

[20] C.-A. Smarr, T. L. Mitzner, J. M. Beer, A. Prakash, T. L. Chen, C. C. Kemp, and W. A. Rogers. Domestic robots for older adults: attitudes, preferences, and potential. *International journal of social robotics*, 6(2):229–247, 2014.

[21] R. Yang and M. W. Newman. Learning from a learning thermostat: lessons for intelligent systems for the home. In *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pages 93–102. ACM, 2013.