Task Prediction in cooking activities using Hierarchical state space Markov Chain and Object based Task Grouping

Prasanth Lade, Narayanan C. Krishnan, Sethuraman Panchanathan
Center for Cognitive Ubiquitous Computing, Arizona State University,
Tempe, AZ-85282, USA
{*prasant1,narayanan.ck,panch*}*@asu.edu*

***Abstract*—Cooking activities are complex activities consisting of multiple steps or tasks. These tasks can be associated with one another based on two properties - the temporal structure that defines the sequence of occurrence of tasks and the objects that are used in the activity. This paper develops cooking activity models for the purpose of task prediction based on these two properties. The temporal structure of the sequence of tasks is captured by the novel hierarchical state space markov chain (HMC) and the object usage is represented using the object based task group (OTG) models. A probabilistic task prediction algorithm that fuses the HMC and OTG models has been developed to predict the next most probable task, given that a sequence of tasks has been completed. The proposed models and algorithms have been evaluated on two complex cooking activities - making brownies and making eggs, achieving a subject independent accuracy of 68.5% for predicting tasks, which is an improvement by an average of 6% in comparison to a Markov chain. The work done in the paper is first of its kind as it focuses on task prediction rather than task recognition. The task prediction framework described in the paper can be easily adapted to any complex activity supporting various annotation schemes and activity models.**

***Keywords*-Cooking Activties; Activity Prediction; Task Prediction, Hierarchical State Space; Markov Chain;**

I. Introduction

Cooking activities are complex activities and typically contain several sub-activities (which we call tasks) within itself. When a person starts a cooking activity, he or she accomplishes it by following a particular order of doing its constituent tasks. This is illustrated in Figure 1, where fews tasks that are involved in the activity of ***making brownie*** are shown. There exists a lot of variability in the order of occurrence of these tasks across individuals and contexts, which makes the problem of learning activity models for these complex activities challenging. Furthermore, for applications such as a prompting system for individuals with Alzheimer's disease (AD) or dementia, the activity models must also be capable of predicting the task the individual will perform even before its occurrence.

The problem of activity task prediction can be defined as follows: given that a person has started a cooking activity and has completed a set of tasks, what is the next most probable task that he or she is likely to do. In contrast, the activity recognition problem focuses on identifying the task based on some sensor data. The focus of this paper is only on activity task prediction and not on activity recognition.

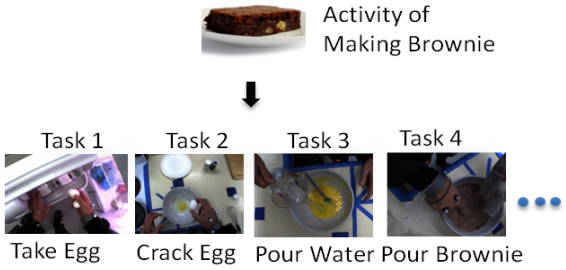

Figure 1. The Activity of ***Making Brownie*** divided into tasks

Learning an activity model that performs reliable activity task prediction is beneficial in many ways. Firstly, certain applications require activity task prediction such as prompting systems for individuals with AD or dementia. Secondly, reliable task prediction can in turn be used to reduce the search space for the recognition engine, thereby improving the recognition accuracies. Finally, task prediction also helps in designing reconfigurable systems that can save energy by switching on only the necessary sensors.

Any task prediction framework depends on the underlying activity model. In this paper the activity model has been developed using a novel methodology called the Hierarchical state space Markov Chain (HMC) that models the temporal relations between tasks. This has been further augmented with an Object based Task Grouping (OTG) model that takes into account the relations between tasks based on common object usage. The models proposed have been evaluated on complex cooking activities viz. ***making brownie*** and ***making eggs***.

Section II briefly discusses the related work. In Section III the overall framework of task prediction is presented and the data that has been used is discussed in IV. In Section V the HMC model is discussed and OTG model is described in Section VI. Section VII details the task prediction algorithm that uses both these models. In Section VIII the results are discussed followed by conclusions in Section IX.

II. Related Work

Beaudoin [1] have discussed extraction of common patterns from Inertial Measurement Units (IMU) data and development of motion motif graphs which represent the various actions in an activity along with their transitions and durations. Singla [2] have used data from activation, water and heat sensors in a smart home and have constructed activity models using markov models which include the time duration of events. Yang et al. in their paper [3] have used examples of plans, that have been developed in the context of AI, as their activity data. Using these plans they have constructed action model for each action within an activity. Spriggs et al. [4] have extracted the frame wise video features from a first person camera and used PCA to project the features into a low dimensional space and have used GMMs to cluster related frames and HMMs to classify 2 cooking activities of ***making brownie*** and ***omelet***. In their AutoReminder application [5], Pollack et al. have modeled activities using Quantitative Dynamic Bayesian Networks considering the activity of each client as a new plan. Activity models that consider objects involved have been detailed in [6] where they have divided each activity into stages i.e. an activity like ***making tea*** was divided into ***boil water***, ***add tea***, ***add flavor***. They have developed activity models using Dynamic Bayesian Networks by considering stages as the hidden variables and objects as observed variables.

Once activity models are developed, they are used for the purpose of automatic activity recognition. [2][4][6] have used activity models for activity recognition. Activity models can be used to validate new plans in the context of planning and scheduling in AI as presented in [3]. Isbell et al. [7] in their paper have developed activity models to predict the most probable task in the perspective of related commands while operating devices in a ubiquitous environment. In [7] they have grouped related commands into a task using KNN and used markov chains to predict next probable task. The work presented in this paper is a first of its kind, which attempts at developing the activity models based on user data for the purpose of reliable task prediction.

III. Task Prediction Framework

Developing task prediction framework requires prior knowledge about the tasks involved in an activity. This knowledge can be acquired through an external source of information such as webpages or can be derived from user labeled training data. In the work done by Philipose et al. [6], the task information has been extracted by mining web pages describing activity recipes and how-to's. In contrast, the proposed framework utilizes labeled activity data of users to derive the activity model. The annotation of tasks can be done either in a supervised manner using manual labeling or in an unsupervised way using automatic activity recognition engine that can identify and annotate unique

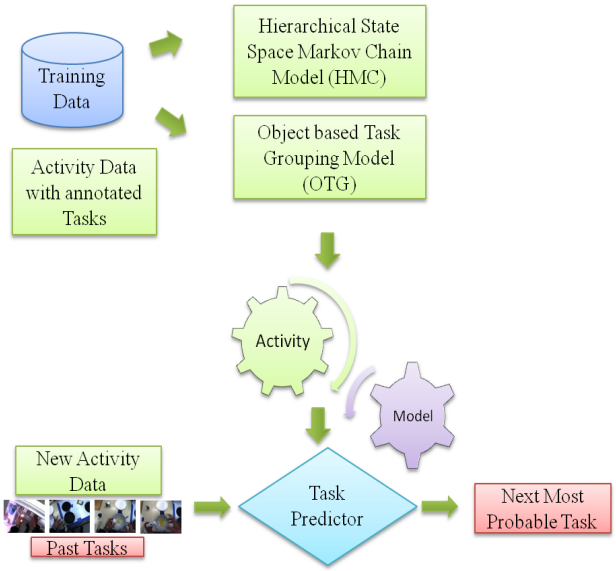

Figure 2. The proposed Task Prediction Framework that contains Training data, Activity Model, Test Data, Task Predictor and next most probable task

patterns from the activity data as tasks. This annotated activity data is considered as the training data for activity modeling. Figure 2 gives a diagrammatic representation of the Task Prediction Framework. The HMC and OTG models that define the activity model are learnt from the training data. When an activity is started and few tasks have been done, the Task Predictor algorithm uses the activity model to predict the next most probable task.

IV. Cooking Activity Data

The CMU Quality of Life Technology Center's [8] human activity data has been used to validate the methods proposed in this paper. The CMU human activity database is a collection of cooking activity data from different subjects and consists of data for activities ***Making Brownie***, ***Making Eggs***, ***Making Sandwich***, ***Making Pizza*** and ***Making Salad***. A typical task within an activity is a combination of *action-object-preposition-object* e.g *pour-oil-into-big_bowl*. The annotated data was available for ***Making Brownie*** and we have manually annotated for the activity ***Making Eggs*** using the same procedure. The task prediction results presented in this paper have been obtained by evaluating on these 2 activities.

Out of many tasks that belong to an activity, few tasks like *put-oil-into-cupboard* are not relevant in acheiving the goal of an activity as such. Therefore after ignoring such tasks, the activity ***Making Brownie*** has 32 tasks and ***Making Eggs*** has 33 tasks which have not been listed here due to the constraint of space. Due to abnormalities in activity data of few subjects, those subjects have been ignored and finally ***Making Brownie*** and ***Making Eggs*** contain data from 14 and 11 subjects respectively with one trial per each subject.

Figure 3. The clusters in the activity graph for ***Making Eggs*** after applying thresholding. The uncolored tasks belong to no cluster and indicate the uncertainity of their occurrence

V. HMC Model

A markov model is a stochastic model where the state that a system occupies during a given time step is dependent on the states it has occupied in the past time steps. The markov chain is a specific case where the number of time steps that affect the next state of the system is 1. When the process of doing an activity is considered, the task is the state of the activity. The tasks done earlier define the next probable task. If the activity is modeled using a markov chain then only the current task influences the next task, which is not a completely valid assumption.

The observation is that in human activity the next task depends on the current task as well as a group of other past tasks. In order to capture both these concepts into the model, the state space of the markov model has been transformed such that the state captures not only the immediate preceding task information but also the information about other tasks that have already occurred. The tasks of the activity form the original state space. Each task is transformed into a hierarchical state and these new states form the hierarchical state space. Section V-A and V-B give more details on this transformation.

A. Clustering of Tasks

The activity is initially modelled using a markov chain with the tasks as the states. A transition matrix for the markov chain, TM_MC, is constructed from the training data which is a set a sequences of tasks as done by various people.

$$TM\_MC(i,j) = P(s_{n+1} = T_j / s_n = T_i)$$

where $TM\_MC(i,j)$ is the probability of transiting from task T_i to task T_j and s_n is the state at time n. Among all the tasks, there are some tasks that transit mostly within themselves i.e these tasks occur in close intervals of time and are related to each other. In order to extract such groups of tasks, the TM_MC is plotted as a graph with the tasks as the nodes and the transitions as edges. Since the data consists of activity data from different people there will be many possible transitions while most of them being weak transitions. A threshold probability is appiled on the graph and all the edges whose probabilities are less than the threshold are removed. From the new graph all the weakly connected components are identified and each weakly connected component is considered as a cluster and the tasks within each connected component are assigned to that cluster. These task clusters are represented as $\{C_1, C_2, \ldots, C_M\}$. Figure 3 shows the clusters in the graph for the activity ***Making Eggs*** after applying thresholding on the graph. The colored tasks belong to clusters with more than one task. It can be observed from the graph that tasks like ***take-spatula***, ***switch-on-stove*** etc. can occur very randomly at any time during the activity and so they belong to no cluster and their prediction is more uncertain than other tasks.

B. Hierarchical State Space

Once the tasks of an activity are clustered, the activity data from each person (from the training data), which is a sequence of tasks, is considered. Every task in the sequence, T_i where i is the time step, which is a state in the original state space is mapped to a new state HSS_i which is a triplet given by

$$HSS_i = (T_i, C(T_i), Pos(T_i)) \quad (1)$$

where $C(T_i)$ is the cluster to which T_i belongs to and $Pos(T_i)$ is the count of the number of tasks $T_j \ni j \leq i$ and $C(T_j) = C(T_i)$

As an example consider the tasks ***take oil***, ***twist-on cap***, ***pour oil into bowl*** and ***stir bowl*** that are a part of the activity ***Making Brownie***. Let these tasks belong to a cluster $C = 5$. The transformation from the original state space to hierarchical state space is shown in Table I using these tasks as example.

Using the above transformation, all the tasks in the training data are converted into hierarchical states. Considering the activity as a markov chain with the hierarchical state space (HMC), a new transition matrix TM_HSS is constructed using the new states. If there are M clusters with an average size of P tasks each, then the maximum number of states will be MP^2 when compared to MP states in a simple markov chain. But in reality this is not the case because within each cluster the tasks occur with a certain

Table I
TRANSFORMATION OF ORIGINAL STATE SPACE TO HIERARCHICAL STATE SPACE. (***take oil***= TO,***pour oil into bowl***= $POiB$,***twist-on cap***= ToC AND ***stir bowl***= SB)

Time Step	Original State	Hierarchical State
1	TO	(TO,5,1)
.	.	(.,.,.)
.	.	(.,.,.)
6	$POiB$	($POiB$,5,2)
7	ToC	(ToC,5,3)
8	SB	(SB,5,4)

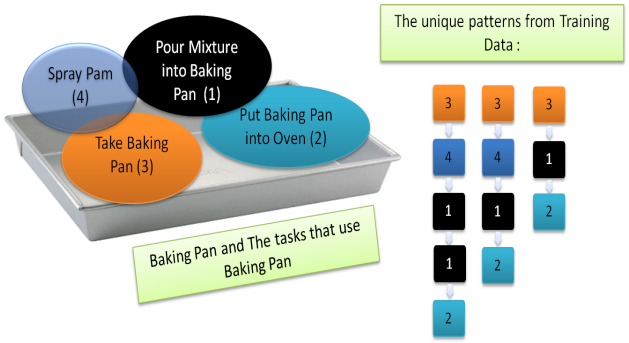

Figure 4. Illustration of Object based Task Grouping for the object *baking pan* in the activity *Making Brownie*. To the left is the group of tasks sharing *baking pan* and to the right are the unique sequences of these task numbers from training data

amount of order so that the tasks may not occur at more than 4 different positions within a cluster and thus reducing the state space to a maximum of $4MP$.

VI. OTG MODEL

Even though the HMC model discussed in Section V includes a higher order information about the tasks that have occurred earlier, there may be cases when it may predict a set of next probable tasks but some of them may be invalid at that point of time. As an example consider the activity ***Making Brownie*** where we have 2 tasks ***pour brownie mixture into baking pan*** and ***put baking pan in the oven***. It may happen at some point of time that the HMC model may consider ***put baking pan in the oven*** to be amongst the tasks that occur next. But if the task ***pour brownie mixture into baking pan*** has not yet occurred then there is no possibility of ***put baking pan in the oven*** occurring. In order to learn such relationships between tasks so that the HMC model can be augmented with extra information that could help in dropping invalid tasks, the Object based Task Grouping (OTG) has been proposed.

Most of the tasks involve interaction with one or more objects. The objects involved in a task can be either recognized using video data or RFID data. While, extracting object information from video data is very challenging, RFID data helps us to find the objects that have been used within the time frame of a given task. In this research we have manually identified the objects that belong to a particular task. Let $\{O_1, O_2, \ldots, O_K\}$ be all the objects involved in the activity and let $T_{O_i} = \{T_{i1}, T_{i2}, T_{i3}, \ldots\}$ be the tasks that are involved with the object O_i. $\{T_{O_1}, T_{O_2}, \ldots, T_{O_K}\}$ shall be called the object clusters. Considering only the tasks T_{O_i} in all the training sequences, transition matrices TM_O_i are constructed for each object O_i. Each of these transition matrices, TM_O_i, model the relations between tasks that share the object O_i. Figure 4 illustrates this grouping of tasks for the object *baking pan* in the activity *Making Brownie*.

VII. TASK PREDICTION ALGORITHM

Algorithm 1 Task Prediction Algorithm

Input:
Task Clusters: $\{C_1, C_2, \ldots, C_M\}$
Object Clusters: $\{T_{O_1}, T_{O_2}, \ldots, T_{O_K}\}$
Hierarchical states from training data: $\{HSS_1, HSS_2, \ldots, HSS_P\}$
Transition Matrices: TM_HSS, TM_O_i's
Past tasks: $\{T_1, T_2, T_3, \ldots, T_n\}$ // n − curr time step
Output:
$(T_{n+1}, P(T_{n+1}))$ // Next task, Next task probability
procedure PREDICT NEXT TASK
$HSS_n \leftarrow$ TransformCurrentTask( ) ;
//The function transforms the task to hierarchical state
Let $FutStates = \{HSS_1{}^*, HSS_2{}^*, \ldots, HSS_{F1}{}^*\}$
$\ni TM\_HSS(HSS_n, HSS_i{}^*) > 0, \forall i \leftarrow 1, F1$;

for $t \leftarrow 1, F1$ **do**
$(T_t, C(T_t), Pos(T_t)) \leftarrow HSS_t{}^*$;
// Remove the task if it has already occurred
if $T_t \in \{T_1, T_2, T_3, \ldots, T_{n-1}\}$ **then**
Remove $HSS_t{}^*$ from $FutStates$;
end if
end for

$FutStates \leftarrow$ ObjectBasedTaskFiltering($FutStates$) ;
// This function filters invalid tasks using object info
Let $FutStates = \{HSS_1{}^*, \ldots, HSS_{F2}{}^*\}$;

Normalize($P(HSS_1{}^*/HSS_n), \ldots, P(HSS_{F2}{}^*/HSS_n)$) ;

$(T_{n+1}, C(T_{n+1}), Pos(T_{n+1})) \leftarrow HSS_N{}^*$
$where\ N = \underset{i}{argmax}(P(HSS_i{}^*/HSS_n))$;

$P(T_{n+1}) \leftarrow P(HSS_N{}^*/HSS_n)$;

return $(T_{n+1}, P(T_{n+1}))$;
end procedure

Once the transition matrices TM_HSS and TM_O_i are obtained from the HMC and OTG models, they are

used by the task prediction algorithm that is explained in Algorithm 1. At each time step, n, the algorithm returns the next most probable task T_{n+1} and its probability of occurrence $P(T_{n+1})$. The task at current time step T_n is transformed into a hierarchical state using the procedure $TransformCurrentTask$. Its working has been described in Section V-B using eq(1). Once the hierarchical state is obtained all the states to which the system can transit from this state can be obtained from TM_HSS and they become the future states. After removing the states corresponding to tasks that have already occurred, the function $ObjectBasedTaskFiltering$, in which object based clusters are used, helps to filter out invalid tasks and its working is described below.

For each of the future states ${HSS_i}^*$ considered, the object cluster $T^*_{O_j}$ to which the corresponding task T^*_i belongs to is considered (see eq(1)). Using the $TM\_O^*_j$ the tasks that always occur before T^*_i, named as *predecessors*, are extracted. Also the tasks that always occur after T^*_i , named as *successors*, are extracted. A task is considered as a possible task only if:
1) All of the *predecessors* have occurred and
2) None of the *successors* have occurred.
All the future states ${HSS_i}^*$ that do not satisfy this criteria are filtered and not considered by the task prediction algorithm. Finally, using the TPM_HSS the probabilities of the remaining future states from the current state are normalized and the one with maximum probability is selected. Using the task information in the hierarchical state using eq(1), the most probable task T_{n+1} is obtained and the normalized probabilities from TPM_HSS give $P(T_{n+1})$.

VIII. Results

The proposed methodology has been tested on the data discussed in Section IV. The average length of a sequence (i.e number of tasks per subject) for the activity ***Making Brownie*** is 30 and for the activity ***Making Eggs*** it is 28. A multi-fold cross validation strategy has been adopted i.e. data for a given activity from all subjects except one has been considered as training data and the left out subject data has been used for testing. The training data is used to build HMC and OTG models and the task prediction algorithm is used for testing. 2 testing strategies have been adopted to objectively test the methodology, which are discussed below.

A. Matching Accuracy

The test sequence is provided to the task prediction algorithm one task at a time in the order of occurrence. The task prediction algorithm predicts the next probable task with some probability. If the predicted task is the same as the actual next task in the test sequence the matching accuracy is increased. Thus the final matching accuracy is the ratio of the total number of matches to the total tasks predicted.

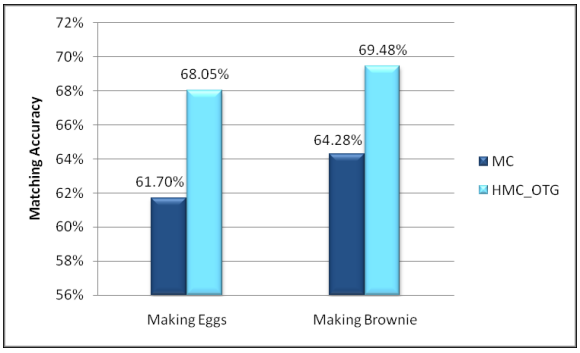

Figure 5. The subject independent multi-fold cross validation results using a Markov Chain (MC) and the proposed algorithm (HMC_OTG) on 2 cooking activties. The average accuracies are calculated using of number of tasks correctly predicted.

The final accuracies have been calculated by taking an average of accuracies obtained from 14 test sequences for ***Making Brownie*** and 11 sequences for ***Making Eggs***. The optimal matching accuracies acheived on the 2 activities ***Making Brownie*** and ***Making Eggs*** using the proposed HMC_OTG model are 69.48% and 68.05% respectively. Figure 5 compares these accuracies with the accuracies achieved using a Markov Chain with the tasks as the states and by ignoring tasks that have already occurred during prediction. As shown in the figure there has been an improvement of around 5% and 7% in accuracies for each of the activities. As dicussed in Section V-A, to cluster the tasks in HMC model, a threshold is applied on the transition matrix. For the activities ***Making Brownie*** and ***Making Eggs*** the thresholds that produced these optimal accuracies are 0.012 and 0.02 respectively.

B. DTW Distance

While calculating matching accuracy, the current task from the original test sequence is provided and using all the tasks of the original test sequence till then, only the next task is predicted. But in order to test how realistically the model can predict and model rest of the entire activity itself this methodology has been used. Let $\{TT_1, TT_2, TT_3, \ldots, TT_{27}\}$ be the actual test sequence of length 27 where TT denotes Test task. A partial sequence $\{TT_1, TT_2, TT_3, TT_4\}$, say of length 4 is considered and the proposed algorithm is used to predict the rest of the sequence and let $\{PT_5, PT_6, \ldots, PT_{27}\}$ be the predicted sequence where PT denotes Predicted task. Using this strategy, by considering only the actual first task, the rest of the activity can be generated and thereby the recipe of the activity can automatically be generated. Now the actual sequence $\{TT_5, TT_6, \ldots, TT_{27}\}$ and the predicted sequence $\{PT_5, PT_6, \ldots, PT_{27}\}$ are compared and the distance between these sequences is calculated using Dynamic Time Warping (DTW). The distance between any matched tasks

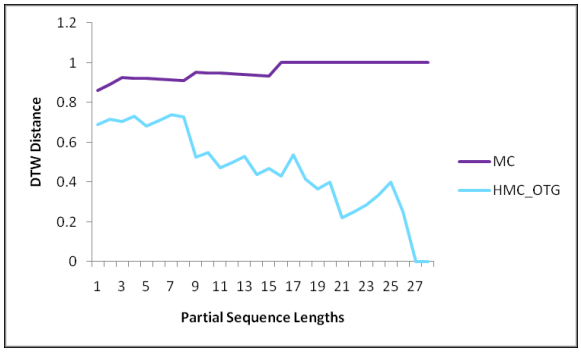

Figure 6. Normalized DTW distances for the sequence of a subject doing the activity ***Making Eggs***. The MC model shows an increase in distance as sequence lengths increas whereas the HMC_OTG model shows a decreasing trend.

is considered to be 0 and unmatched task as 1. For the above mentioned test sequence since 26 partial sequences are possible 26 DTW distances are calculated and their average has been considered.

Figure 6 shows the DTW distances generated for a test sequence of length 29 using markov chain (MC) and proposed algorithm (HMC_OTG). So there are 28 partial sequences and the 28 corresponding DTW distances are normalized by the lengths of the partial sequences. It can be observed that the DTW distance of the sequences generated by the HMC_OTG models are less throughout when compared to MC model. In fact the DTW distances for MC increase as the sequence lengths increase whereas the DTW distances for HMC_OTG models decrease because, as the lengths of partial sequences increase there is more knowledge about the tasks that are completed and the object information predominantly helps in filtering possible future tasks. We also observe that for HMC_OTG models, the DTW distances do not decrease monotonically i.e. there are spikes in the distances. The reason is that even though the difference between 2 immediate partial sequences is only 1 task, that single task gives a completely different path to the completion of the activity which results in larger distances. The average DTW distances on all test sequences of the activities ***Making Brownie*** and ***Making Eggs*** are 0.94, 0.81 respectively using MC model and 0.35, 0.34 respectively using HMC_OTG model. These distances clearly indicate the better performance of the proposed model.

IX. Conclusions

This paper focuses on the problem of task prediction for complex cooking activities that consist of multiple steps or tasks. These cooking activities have been modeled using the temporal structure of the sequence of tasks and common object usage using the proposed hierarchical state space Markov chain (HMC) and object based task groupings (OTG) models respectively. A task prediction algorithm that fuses the HMC and OTG models has been developed to predict the next most probably task, given a sequence of tasks. The models and algorithms have been evaluated on two complex cooking activities that are a part of the CMU kitchen dataset resulting in a subject independent task prediction accuracy of 68.5%, which is better than the traditional Markov chain model by over 6% and is far superior over a random guess that gives a prediction accuracy of 7% only. The work presented in the paper is the first of its kind focusing on activity models for task prediction. The models proposed in the paper can further be used for automatically creating recipes from past cooking data customized to individuals for assisting in activity completion. In this research the objects that belong to a task have been manually identified but in future we would like to automatically extract that information from timestamped RFID data.

Acknowledgment

The authors would like to thank the researchers Moritz Tenorth from Technical University of Munich and Ekaterina Taralova from CMU for their help regarding the annotation of data. The data used in this paper was obtained from [8] where the data collection was funded in part by the National Science Foundation under Grant No. EEEC-0540865.

References

[1] P. Beaudoin, S. Coros, M. van de Panne, and P. Poulin, *Motion-motif graphs*. ACM SIGGRAPH, Eurographics Symposium on Computer Animation, 2008.

[2] G. Singla, D. Cook, and M. Schmitter, *Incorporating temporal reasoning into activity recognition for smart home residents*. Proceedigs of AAAI Workshop on Spatial and Temporal Reasoning, pages 53-61, 2008.

[3] Q. Yang, Kangheng, Wu, and Y. Jiang, *Learning action models from plan examples using weighted MAX-SAT*. Artificial Intelligence, 171, 2-3 , 107-143, Feb 2007.

[4] E. H. Spriggs, F. De La Torre, and M. Hebert, *Temporal segmentation and activity classification from first-person sensing*. IEEE Computer Vision and Pattern Recognition Workshops, June 2009.

[5] M. E. Pollack, C. E. McCarthy, S. Ramakrishnan, I. Tsamardinos, L. Brown, S. Carrion, D. Colbry, C. Orosz and B. Peintner,*Autominder: A Planning , Monitoring , and Reminding Assistive Agent*. 7th International Conference on Intelligent Autonomous Systems, 2002.

[6] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson , D. Fox, H. Kautz, and D. Hahnel, *Inferring Activities from Interactions with Objects*. IEEE Pervasive Computing 3, 4, 50-57, Oct 2004.

[7] C. L. Isbell, O. Omojokun, and J. S. Pierce, *From devices to tasks: automatic task prediction for personalized appliance control*. Personal Ubiquitous Computing 8, 3-4, 146-153, Jul 2004.

[8] http://kitchen.cs.cmu.edu/, *CMU Multi-Modal Activity Database*