Transferring Learned Activities in Smart Environments

Parisa RASHIDI [a,1], Diane J. COOK [a]
[a] *Washington State University, Pullman, WA, USA, 99163*

Abstract. Most commonly-used techniques in smart environments such as ADL recognition are designed and tested for a specific space and a specific person; therefore learning in each environmental situation is treated as a separate context. In this paper, we try to develop a method for recognizing and transferring learned knowledge of activities between different residents. Our method is able to map activities despite intra-subject variability and inter-subject variability, by using a discontinuous mining method and a similarity measurement method. At the end, we will provide the results of our experiments on real data obtained from a smart apartment.

Keywords. Smart Environments, Data Mining, Transfer Learning, Activity Mapping

Introduction

Smart environments provide many benefits such as comfort, security, energy efficiency, and health monitoring of elderly or people with disabilities [13], by using machine learning techniques to process data from various sensors in order to predict upcoming events [6], recognize activities [9], and automate interactions with the environment [5].

Activity recognition can be used to monitor the well being of the residents and also to respond in a context-aware way to their needs. For example, if people with memory deficits are provided with appropriate activity cues, they will be able to complete the Activities of Daily Living (ADL) [13] more easily, and as a result to live independently. There have been a number of methods for recognizing activities, such as Naïve Bayes [2], decision trees [12], Markov models, dynamic Bayes networks, and conditional random fields [7,16]. None of these approaches address the issue of transferring learned knowledge to new contexts to make the systems more scalable. All of the approaches make the assumption that a sufficient amount of data is available, and as supervised models they assume that the activity is predefined in a consistent manner without missing steps. A challenge that researchers face in modeling resident activities in real world is the variability that occurs from one individual to another (inter-subject variability) as well as across multiple executions of an activity by the same individual (intra-subject variability) [22]. We anticipate that different individuals might perform an activity in such vastly different ways that one model cannot perform well for many different users. In this paper, we introduce an unsupervised method to recognize and transfer learned activities across differ-

[1]Corresponding Author: Parisa Rashidi, Washington State University, Pullman, WA, USA, 99163; E-mail: prashidi@eecs.wsu.edu.

ent residents. We have developed a discontinuous misplaced sequential mining method (DMSM) to discover interesting patterns in data in order to capture intra-subject variability. More importantly, we also have developed an activity mapping method (AMM) to map activities from a source space to a target space to address inter-subject variability and to provide a degree of similarity between two sets of activities.

As a recent preliminary work, Kasteren et. al [21] describe a simple transfer learning method to recognize activities across different physical spaces, by simply transferring the transitional probabilities for two different Markov models. It does not address how activities in a target context can be found using knowledge from the source space. In addition, many important features of activities are ignored, except for transitional probabilities; such as mapping of the states, duration of the states, or order of the states. Our model is able to find relevant activities using our DMSM mining method by detecting variations of interesting patterns that might even be discontinuous or misplaced. In addition, we consider each state of the activity to have arbitrary attributes, such as duration or frequency which contribute to finding an appropriate mapping. Also, we do not require target activities to have the same structure as the source activities, e.g. equal number of states. Considering such general aspects, we will define a flexible method to map and measure similarity between source and target activities. In addition to reducing the required amount of data, finding a mapping between activities allows us to exploit gained knowledge in the source context, for example if we have discovered that certain types of cue detail and cue timing work best for a specific source activity, that can be used for the mapped target activity too, depending on the similarity degree.

In following sections, we first explain the general model, and then we describe each of the components in more detail. Finally we will present the results of our experiments.

1. Model Description

In our model, we hypothesize that the data collected for one resident in the source space Φ_s can be used to recognize activities for that resident, and more importantly can be transferred to a target space Φ_t to learn activities for a different resident. The obtained mapping and similarity measure will allow us to transfer source activity's related knowledge later, such as cues, details of cues and timings, etc. In current work, as a first step, we consider transferring between different residents, however a similar method can be used for transferring knowledge between different physical spaces, or between spaces with different sets of sensors. In our model, the input consists of activities $\mathcal{A}$ from Φ_s (which we assume have already been discovered using DMSM), and a small dataset $\mathcal{D}$ from Φ_t. Using DMSM, we first identify interesting patterns $\mathcal{P}$ as sensor sequences that usually appear together in $\mathcal{D}$. Next, we cluster $\mathcal{P}$ into $|\mathcal{A}|$ clusters where $|\mathcal{A}|$ is the number of activities in $\mathcal{A}$. By comparing each cluster k's representative, c_k, to each activity in $\mathcal{A}$, and finding a similarity measure between them, we are able to generate a mapping from Φ_s to Φ_t (see Figure. 1). Note that an activity can be mapped to another activity with more or less states, and with different state attributes. In addition, during a mapping the order of states need not be preserved.

In the following sections, we will describe each one of the mining, clustering, and similarity measurement components in more detail.

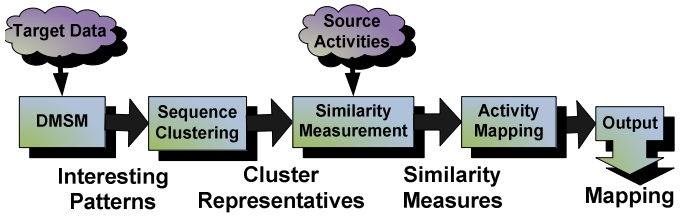


Figure 1. System's architecture and its main components.

2. Finding Interesting Patterns in Data

Many methods have been proposed for mining sequential data, such as frequent itemset mining [1], frequent pattern mining using regular expressions [4], constraint based mining [15], and simultaneous frequent-periodic pattern mining [17]. Those approach do not discover discontinuous misplaced patterns, which can appear in the daily activity data due to the erratic nature of human activities. For example, in meal preparation activity the steps do not always follow the same strict sequence; rather their order is changed and is interleaved with random actions. In smart environments it's important to be able to discover such discontinuous misplaced patterns to deal with intra-subject variability. Ruotsalainen et al. [11] introduce GAIS for detecting interleaved patterns using genetic algorithms, but it is a supervised solution that looks for matches of specified patterns. Other works proposed for mining discontinuous patterns cannot find certain continuous patterns [14,23], and the work by Chen et al. [3] to mine hybrid patterns is not able to find misplaced patterns such as $\langle a, b, c\rangle$ as a variation of for example $\langle b, a, c\rangle$. We propose a model to find a general hybrid pattern from discontinuous misplaced instances, e.g. the general pattern $\langle a, b\rangle$ from instances $\{b, x, c, a\}$, $\{a, b, q\}$, and $\{a, u, b\}$. Our approach is different from frequent itemsets mining, as we are taking into account the order of events, and for each general pattern we report a prevalent pattern and a number of variations that differ in the order of their events. We denote the i^{th} individual variation of a general pattern a as variation a_i, and the most frequent variation as a prevalent variation.

First we identify all symbols in $\mathcal{D}$ with a frequency above the minimum frequency f_{min}, and place them in a reduced dataset $\mathcal{D}_r$. We should keep f_{min} small, as an activity might not appear very frequently in a small dataset. Next we find increasing length patterns by sliding a window across $\mathcal{D}_r$. The window size initially is set to 2, and increased in each iteration. While sliding window across $\mathcal{D}_r$, all patterns that can be permutated into one another, are saved as variations of the same general pattern in a hash-table. For general pattern a, we define its frequency f_a as the sum of its variations' frequencies (the variations with frequency below a certain fraction of f_a are eliminated). We identify interesting general patterns as patterns satisfying Eq. 1, where C is the compression threshold according to minimum description length principle [18], L represents the input data length, and $\Gamma_g(a)$ represents continuity of general pattern a.

$$\frac{|a| * f_a * \Gamma_g(a)}{L} > C \tag{1}$$

To understand what the continuity factor means, consider general pattern $\langle a, b, c\rangle$ in Figure. 2, and instance "abgeqydc" with irrelevant events "geqyd" separating ab from c.

Though we will still might consider this instance as a variation of the general pattern $\langle a, b, c\rangle$, at the same time we should take into account the continuity between ab and c.

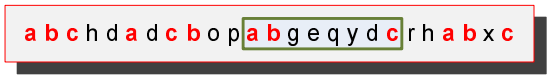


Figure 2. A small dataset containing prevalent pattern $\langle a, b, c\rangle$.

The general pattern's continuity, Γ_g, is defined as the weighted average continuity of its variations. The continuity of a variation, Γ_v, is then defined as the average continuity of its instances; and the instance continuity Γ_i shows how continuous its component events are. The continuity between component events, Γ_s, is defined for each two consecutive symbols in an instance. For each frequent symbol x, we record how far apart (s_x) it is from a preceding frequent symbol in terms of number of symbols separating them in $\mathcal{D}$ (in above example, $s_x = 5$). Then $\Gamma_s(x)$, the symbol continuity for x is defined as $\frac{1}{s_x}$; the more the separation between two frequent symbols, the less will be the symbol continuity. If s_x is zero, we define $\Gamma_s(x) = 1$ (perfect continuity). Based on this, $\Gamma_i(a_i^j)$, for an instance j of a variation a_i will be defined as in Eq. 2 where $|a_i^j|$ shows the length of a_i^j. $\Gamma_v(a_i)$ is defined as in Eq. .3, where n_{a_i} shows the total number of instances for variation a_i. Γ_g is defined according to Eq. 4, where the continuity for each a_i is weighted by its frequency f_{a_i} and n_a shows the total number of variations for general pattern a. If continuity happens to be less than a minimum threshold γ_{min}, we discard that instance (or variation or general pattern).

$$\Gamma_i(a_i^j) = \frac{1}{|a_i^j|} \sum_{k=1}^{|a_i^j|} \Gamma_s(k) \tag{2}$$

$$\Gamma_v(a_i) = \frac{1}{n_{a_i}} \sum_{j=1}^{n_{a_i}} \Gamma_i(a_i^j) \tag{3}$$

$$\Gamma_g(a_i) = \frac{\sum_{i=1}^{n_a} \Gamma_v(a_i) * f_{a_i}}{\sum_{i=1}^{n_a} f_{a_i}} \tag{4}$$

After calculating Γ_g, we can decide which general patterns are interesting according to Eq. 1. We find patterns of increasing length by increasing the sliding window's size at each iteration. We will stop increasing the window size until a certain number of user defined iterations has been reached, or if no more interesting patterns are found. A postprocessing step records attributes such as durations of events, and prunes non-maximal patterns, i.e. those patterns that are totally contained in another larger pattern.

3. Activity Mapping Method (AMM)

Next, we will cluster discovered patterns $\mathcal{P}$ into $|\mathcal{A}|$ clusters, the number of activities in Φ_s. Clustering $\mathcal{P}$ into $|\mathcal{A}|$ clusters allows us to find a 1-1 mapping between source and target activities better, as we will compare each cluster's representative c with each

activity $a \in \mathcal{A}$, instead of comparing every $p \in \mathcal{P}$ with each $a \in \mathcal{A}$. The clustering method we are using is a standard k-means clustering [10], however we need a method for defining representatives and comparing activities in order to form clusters.

Two methods for comparing similarity of sequences are "edit distance" [8] and "LCS" [20] for simple sequence of symbols. Saneifar et al. [19] have proposed a similarity measure for more complex itemset sequences based on the number of common items. Those methods do satisfy our complex definition of an activity as a sequence of events with arbitrary state attributes, and do not deal with general aspects of sequences such as temporal information, order of states, etc. Besides, in our model it's possible to map combination of two states to one state, e.g. if two states' total duration in the source context is close to duration of a single state in the target context.

To calculate similarity between two activities a and b, we need to determine similarity between their set of states $\mathcal{S}_a$ and $\mathcal{S}_b$, in addition to the order similarity. It's possible to combine several states during mapping, in case one state immediately follows another state with a sensor of the same type that provides the same functionality. To find possible combinations of states, we define $Ext(\mathcal{S})$, the extension of a state set $\mathcal{S}$, by considering possible combinations between consecutive states with sensors of the same type to form a new state. For example, for three consecutive sensors of the same type (a, b, c), it's possible to consider three new extended states, $a' = \{a, b\}$, $b' = \{b, c\}$ and $c' = \{a, b, c\}$. Note that additive attributes such as duration will be added together. We denote the union of actual and extended states for an activity a as $\Pi_a = \{\mathcal{S}_a \cup Ext(\mathcal{S}_a)\}$.

We also need to define order similarity, to see if activities a and b have the same relative order. We define the order similarity, $s_o(i, j)$, between state $i \in \mathcal{S}_a$ and state $j \in \mathcal{S}_b$ as in Eq. 5 where $pos(s)$ shows position of state s in its corresponding activity (for extended states, total number of states will not be equal to $|\mathcal{S}|$, therefore it is replaced by corresponding new size).

$$s_o(i, j) = 1 - |\frac{pos(i)}{|\mathcal{S}_a|} - \frac{pos(j)}{|\mathcal{S}_b|}| \tag{5}$$

To measure similarity between two activities a and b, we need to find the best possible mappings between their states. We start with an initial mapping and then will resolve any possible conflicts, e.g. if two separate actual or extended states are mapped to the same state. For each state $i \in \Pi_a$, we find the best possible mapping state $j \in \Pi_b$. The state similarity, $s_s(i, j)$, between two states i and j is defined as in Eq. 6. Here attribute k found in both states i and j is denoted by k_i and k_j; w_k is a weight applied to attribute k to indicate the importance of k in calculating similarity (e.g. we might consider duration more important than frequency); and m denotes the total number of attributes. In our model, we only map sensors of the same type (e.g. motion sensors to motion sensors).

$$s_s(i, j) = 1 - (\sum_{k=1}^{m} w_k * (\frac{k_i}{max(k_i)} - \frac{k_j}{max(k_j)}) * s_o(i, j)) \tag{6}$$

Then, the best possible mapping for state $i \in \Pi_a$ is defined as in Eq. 7.

$$map(i) = argmax_{j \in \Pi_b}\{s_s(i, j)\} \tag{7}$$

The cumulative similarity, $s_c(a, b)$ between a and b is defined as in Eq. 8, note that the subset of states selected from Π_a and Π_b for mapping will be denoted by Υ_a and Υ_b.

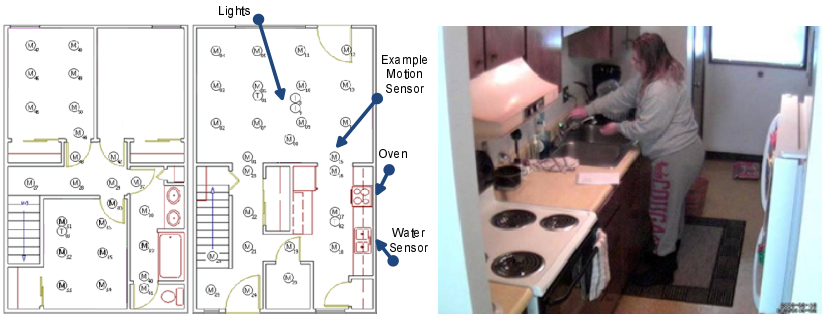
$$s_c(a,b) = \frac{1}{\Upsilon_a} \sum_{i=1}^{|\Upsilon_a|} s_s(i, map(i)) \tag{8}$$

To resolve any conflicts between a subset of states $\mathcal{S}_c \subset \Pi_a$ that map to a single state c, we will find a new mapping for each of the states $s \in \mathcal{S}_c$. The new mappings will be selected from $\Pi_b - \{c \cup \Upsilon_b\}$. The state with the least new similarity will be mapped to c, as such an assignment will cause the least amount of decrease in the cumulative similarity. The rest of the states will be assigned according to new mappings. If still there is any conflict, it will be resolved in an iterative manner. If no mappings can be found for a state, it will be mapped to a null state.

The cluster representative c_k for cluster k is defined as following: the number of states for c_k will be equal to the average number of states in k, the attributes for each state will equal the average values for that state in k, also each state's sensor type will be the most common type sensor type in k for that state. After clustering is finished, we will compare each $a \in \mathcal{A}$ to each c_k, and by finding $argmax_k\{s_c(c_k, a)\}$, we are able to provide a mapping between activities in Φ_s and Φ_t. This mapping and similarity measure can act as a guideline for transferring activity's related knowledge.

4. Experiments

The testbed is a 3-bedroom smart apartment (Figure. 3), located on Washington State University campus. Sensor data was collected from 59 sensors inclusing motion, temperature, water, burner, phone usage, and the presence of key items.



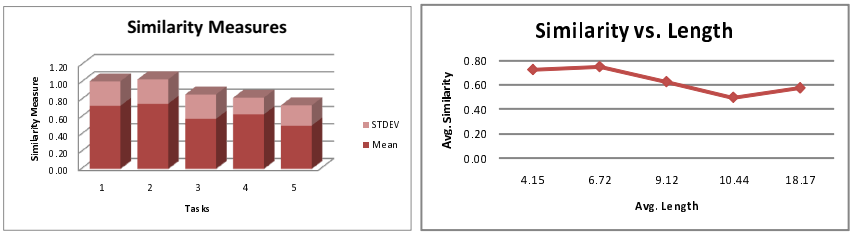
(a) Smart apartment's layout. (b) Participant performing "hand washing".

Figure 3. Smart apartment testbed.

We brought 23 participants into the smart apartment, one at a time. Each participant was asked to perform a script of five I/ADL, typically found in clinical questionnaires assessing everyday functional activities [13]; including (1) telephone usage, (2) hand washing, (3) meal preparation, (4) medication use, and (5) household cleaning. Our data sets consisted of sensor event data for the series of 5 ADLs, each repeated for about 3 times with random events injected between activities up to 50%.

Using a 10-fold cross validation approach, we assessed the ability of our algorithm to accurately recognize activities for new target participants based on models learned

from a source participant. DMSM was able to find correct patterns for each participant, despite discontinuous misplaced steps with intra-subject variability of up to 20%. In these experiments, we set $C = 0.3$, $f_{min} = 2$, and $\gamma_{min} = 0.5$.



(a) Similarity measures for all 5 tasks. (b) Similarity vs. Length

Figure 4. Similarity measure results.

Our model was also able to map and measure similarity between activities correctly, despite the fact that the activities were performed in vastly different ways, with different sensor reading's order, different durations and different activities' length (see Figure. 4). It was interesting to note that longer patterns usually had a lower similarity measure.

In another experiment, we randomly changed the order of events in two sets of datasets similar to the above datasets to simulate the effect of misplaced steps, one dataset containing shorter patterns of 1 to 14 events, and the other containing longer patterns of 14 to 47 events. Though for longer patterns the misplacement can generate much diverse patterns and therefore making it more difficult to detect patterns, still our algorithm was able to find patterns (see Fig. 5).

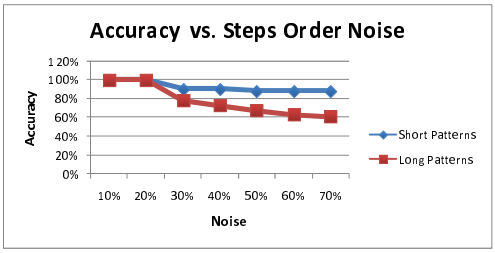


Figure 5. Accuracy vs. injected order noise for short and long patterns.

The above results confirm our hypothesis that our method is able to find patterns in data despite intra-subject variability, and to map activities despite inter-subject variability. The small datasets used for those experiments (each dataset containing an activity repeated three times with 50% random activities in between) also show how our method can use knowledge from source space to effectively recognize activities in target space.

5. Conclusion

In this paper, we introduced a model for transferring and measuring similarities between activities performed by different residents. Our DMSM method captures intra-subject

variability, and AMM method provides mappings despite inter-subject variability. In the future, we intend to transfer activities for different spaces and different sensor types. We also want to relate similarity measurement with activity's cues, to see how similarity can be used as a guideline for transferring various aspects of cues, such as timing, detail, etc.

References

[1] Rakesh Agrawal and Ramakrishnan Srikant. Mining sequential patterns. In *international conference on data engineering*, pages 3–14, 1995.

[2] Oliver Brdiczka, Jérôme Maisonnasse, and Patrick Reignier. Automatic detection of interaction groups. In *Proceedings of the 7th international conference on Multimodal interfaces*, pages 32–36, 2005.

[3] Yen-Liang Chen, Shih-Sheng Chen, and Ping-Yu Hsu. Mining hybrid sequential patterns and sequential rules. *Inf. Syst.*, 27(5):345–362, 2002.

[4] Minos N. Garofalakis, Rajeev Rastogi, and Kyuseok Shim. Spirit: Sequential pattern mining with regular expression constraints. In *International Conference on Very Large Data Bases*, pages 223–234, 1999.

[5] L.B. Holder G.M. Youngblood, D.J. Cook and E. Heierman. Automation intelligence for the smart environment. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2005.

[6] Karthik Gopalratnam and Diane J. Cook. Online sequential prediction via incremental parsing: The active lezi algorithm. *Intelligent Systems, IEEE*, 22(1):52–58, Jan.-Feb. 2007.

[7] Henry Kautz, Dieter Fox, Oren Etzioni, Gaetano Borriello, and Larry Arnstein. An overview of the assisted cognition project. In *In AAAI Workshop on Automation as Caregiver*, 2002.

[8] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8, 1966.

[9] Lin Liao, Dieter Fox, and Henry Kautz. Location-based activity recognition using relational markov networks. *In Advances in Neural Information Processing Systems (NIPS)*, 2005.

[10] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, 1967.

[11] Ari Visa Marja Ruotsalainen, Timo Ala-Kleemola. Gais: A method for detecting discontinuous sequential patterns from imperfect data. In *ICDM*, pages 530–534, 2007.

[12] U. Maurer et al. Activity recognition and monitoring using multiple sensors on different body positions. *Workshop on Wearable and Implantable Body Sensor Networks*, page 116, April 2006.

[13] M.B. Patterson and J.L. Mack. The cleveland scale for activities of daily living (csadl): Its reliability and validity. *Journal of Clinical Gerontology*, 7:15–28, 2001.

[14] Jian Pei et al. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *In Proc. Int. Conf. Data Engineering*, pages 215–224, 2001.

[15] Jian Pei, Jiawei Han, and Wei Wang. Constraint-based sequential pattern mining: the pattern-growth methods. *J. Intell. Inf. Syst.*, 28(2):133–160, 2007.

[16] M. Philipose, K.P. Fishkin, M. Perkowitz, D.J. Patterson, D. Fox, H. Kautz, and D. Hahnel. Inferring activities from interactions with objects. *Pervasive Computing, IEEE*, 3(4):50–57, Oct.-Dec. 2004.

[17] Parisa Rashidi and Diane J. Cook. An adaptive sensor mining model for pervasive computing applications. In *KDD Workshop on Knowledge Discovery from Sensor Data*, pages 41–50, 2008.

[18] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[19] Hassan Saneifar, Sandra Bringay, Anne Laurent, and Maguelonne Teisseire. S2mp: Similarity measure for sequential patterns. In *AusDM*, volume 87, pages 95–104, 2008.

[20] Karlton Sequeira and Mohammed Zaki. Admit: anomaly-based data mining for intrusions. In *Proceedings of the international conference on Knowledge discovery and data mining*, pages 386–395, 2002.

[21] G. Englebienne T.L.M. van Kasteren and B.J.A. Krose. Recognizing activities in multiple contexts using transfer learning. In *AAAI AI in Eldercare Symposium*, 2008.

[22] Robert E. Wray and John E. Laird. Variability in human behavior modeling for military simulations. In *In Proceedings of Behavior Representation in Modeling and Simulation Conference*, 2003.

[23] Mohammed J. Zaki, Neal Lesh, and Mitsunori Ogihara. Planmine: Sequence mining for plan failures. In *In Intl. Conf. Knowledge Discovery and Data Mining*, pages 369–373, 1998.