Published in final edited form as:
IEEE Trans Hum Mach Syst. 2015 October ; 45(5): 575–585. doi:10.1109/THMS.2014.2362529.

Automated Detection of Activity Transitions for Prompting

Kyle D. Feuz[*], **Diane J. Cook**[†] **[Fellow, IEEE]**, **Cody Rosasco**[‡], **Kayela Robertson**[‡], and **Maureen Schmitter-Edgecombe**[‡]

[*]Computer Science Department, Weber State University

[†]School of Electrical Engineering and Computer Science, Washington State University

[‡]Department of Psychology, Washington State University

Abstract

Individuals with cognitive impairment can benefit from intervention strategies like recording important information in a memory notebook. However, training individuals to use the notebook on a regular basis requires a constant delivery of reminders. In this work, we design and evaluate machine learning-based methods for providing automated reminders using a digital memory notebook interface. Specifically, we identify transition periods between activities as times to issue prompts. We consider the problem of detecting activity transitions using supervised and unsupervised machine learning techniques, and find that both techniques show promising results for detecting transition periods. We test the techniques in a scripted setting with 15 individuals. Motion sensors data is recorded and annotated as participants perform a fixed set of activities. We also test the techniques in an unscripted setting with 8 individuals. Motion sensor data is recorded as participants go about their normal daily routine. In both the scripted and unscripted settings a true positive rate of greater than 80% can be achieved while maintaining a false positive rate of less than 15%. On average, this leads to transitions being detected within 1 minute of a true transition for the scripted data and within 2 minutes of a true transition on the unscripted data.

Index Terms

Activity Recognition; Prompting Systems; Change-Point Detection; Smart Environments

I. Introduction

The world's population is aging, with the estimated number of individuals over the age of 85 expected to triple by 2050 [1]. Currently 50% of adults aged 85+ need assistance with everyday activities, and one in three households is anticipated to have at least one family member with cognitive decline within the next decade [2]. Research has demonstrated that individuals with mild cognitive impairment (MCI), defined as an intermediate state between normal aging and dementia [3], can successfully learn to use memory compensation strategies in the laboratory and clinical environment (e.g., memory notebooks or mnemonic techniques) [4]–[6]. However, after the initial training sessions there are few methods to promote or ensure that the individual remembers to use the techniques in everyday life, aside from placing additional burden on a caregiver to prompt for use.

Significant recent advances in the use of smart environments for activity recognition and prompting [7], [8] suggest that smart technologies can augment traditional rehabilitation techniques and support functional independence. Most applications of smart technology for health assistance to date have focused on a narrow set of tasks or have been performed in controlled laboratory settings [8]–[10], and do not couple smart environment contributions with other technological healthcare innovations. Furthermore, many of the technologies fail to build upon efficacious rehabilitation strategies or to incorporate cognitive rehabilitation theory when designing interventions [11]. In this work we advance the state of the art by partnering mobile electronic technologies, smart environments, and traditional clinical interventions to provide memory assistance. The long-term goal of our work is to develop technologies to support proactive health care that will successfully improve the functioning and quality of life of individuals with MCI and dementia, and reduce their reliance on caregivers. In this paper, we describe one such technology that will contribute to this overall goal. The technology is a digital memory notebook (DMN) that builds on a smart home / mobile technology partnership.

Memory impairment, one of the earliest and most problematic symptoms of MCI, can significantly impact everyday functional independence [12], [13]. Recently, evidence-based cognitive rehabilitation approaches that enhance everyday independence in traumatic brain injury [14] have been applied to older individuals with MCI [5], [6]. Unlike prior non-drug interventions used with MCI and dementia populations (e.g., cognitive stimulation or reminiscence therapy), these newer approaches aim to maintain functional independence and delay dementia diagnosis by teaching the use of practical memory strategies for everyday activities (e.g., remembering appointments and daily tasks). Prior work [6], [15] demonstrates that individuals with MCI can learn to journal in a memory notebook and effectively use the notebook to schedule, plan, and carry out daily activities independently with the assistance of a care-partner (i.e., spouse, child, or friend).

The cornerstone of the original version of this intervention was a pen-and-paper notebook with three components: 1) a daily log and to-do-today list used to record, store and retrieve information about daily activities; 2) a calendar coordinated with the daily log used to record and retrieve important appointments and upcoming events; and 3) a personal notes section to store birthdays, medications, bus schedules and other personal information. Preliminary data showed that, in comparison to standard care, MCI individuals in the intervention group were able to effectively apply note-taking and problem solving strategies post-intervention. They also self-reported fewer symptoms of depression and greater use of memory strategies in their everyday lives, while the care-partners reported better coping self-efficacy [16].

The work we describe here improves upon the paper-and pencil notebook intervention by developing a smart home / DMN partnership. Such a partnership facilitates learning and continued use of the memory notebook to support functional independence through activity recognition and context-aware prompting.

Teaching a memory-impaired individual to use a memory notebook takes time (typically 12–20 training sessions) and using the notebook is itself a memory exercise for the individual [17]. The memory impaired individual must be aware of his or her memory

difficulties, learn to integrate the memory notebook into his or her everyday routine, and develop a habit of consistently using the notebook in the appropriate context [6]. This is usually accomplished through reminder prompts, which can come from a care-partner, a time-based alarm, or aspects of the individual's routine (e.g., refer to memory notebook at meals) [17]. If an individual with memory impairments does not overlearn the process of using the memory notebook, the use of the notebook will be forgotten and the tool itself will become ineffective, or will place additional burden on a care-partner to monitor and prompt for memory notebook use.

We use machine learning and pervasive computing technologies to provide automated prompting for DMN use. Prompting technologies have been shown to increase adherence to instructions, decrease errors in everyday instrumental activities of daily living (IADLs), reduce frustration due to interruptions increase independence and increase activity engagement in individuals with cognitive impairment [18]–[21]. Some mobile prompting systems have been designed which allow for the setting of reminder prompts [22]. Such time-based prompting may help individuals use the aid more often, but requires the user to do extra work to learn how to use the system and set the prompts which may reduce overall use [23]–[25]. Furthermore, to help an individual acquire the habit of regularly looking at and entering information into a memory notebook, a time-based prompt delivered when the person is taking a nap or actively engaged in an activity will likely be ineffective [26], [27]. Similarly, a prompt delivered after the individual has already completed the task will only result in annoyance. While location-based prompting provides some contextual input, researchers have found that incorporating information about current activities reduces the limitations of current prompting systems [11], [28].

Prompting during transition periods, a period of time when the user is not engaged in an activity, has been suggested as an effective prompting time [29], [30]. Detecting and using transition periods for prompting is supported theoretically by results from dual-task studies. Prompting during an attention-demanding activity can create instances of information overload and reduce the efficiency with which the individual can complete the task [31], [32]. Our approach to prompting individuals to use their DMN is thus based on the idea of detecting activity transitions and prompting them during these times.

In this paper we develop and assess machine-learning algorithms that will detect transition periods between activities. We hypothesize that activity transitions can be detected, independent of the specific activities performed, using traditional supervised learning algorithms for activity recognition by representing it as a two-class problem (activity or transition). Furthermore, we hypothesize that activity transitions are also accompanied by corresponding changes in the data distributions of the feature-space and can thus be detected without the use of labeled data (i.e. in an unsupervised manner) by looking for changes in the data distributions. To test our hypothesis we have participants complete twelve tasks of everyday living (such as cooking, cleaning, and watching television) in a smart home. The smart home is equipped with infrared motion sensors, magnetic door sensors, temperature and light sensors, and some object shake sensors that will allow us to track the participant's movements. We then apply the machine-learning techniques to the sensor data to successfully detect activity transitions.

A. Related Work

Automated activity recognition plays an important role in the proposed technology. There have been a number of machine learning models that have been used for activity recognition. These can be broadly categorized into template matching, generative and discriminative approaches. Template matching techniques employ k-NN classifiers using the distance computed between subsequences of sensor events [33]. Generative approaches to activity recognition utilize probabilistic graphs such as naive Bayes models, hidden Markov models, dynamic Bayes networks, or Gaussian mixtures that can model activity sequences for generation of data as well as labeling of activity classes [34]–[36]. Discriminative approaches that model the boundary between the different classes have also been popular for activity recognition. These include decision trees, support vector machines, and conditional random fields [7], [37]–[39].

A key component of our approach is recognizing transitions between activities. Focusing on activity transition detection allows us to abstract away the activities being performed and simplifies the task of finding transition points. Similar work has been pursued in activity segmentation where a continuous stream of sensor data is segmented into discrete meaningful units that can be separately classified. Approaches have been tried that train supervised learning algorithms based on examples of activity starts and stops, examples of activity transitions, or examples of activity pairs [40], [41]. Lack of confidence in the activity classification has also been used to indicate a possible transition between activities [42]. Unsupervised approaches analyze statistical parameters of the sensor data to explicitly look for clusters that represent cohesive activities or changes in the parameters that indicate activity transitions [29], [30], [43].

Our approach to unsupervised detection of activity transitions is based on an approach to detecting abrupt changes in time series data, known as change-point detection [44]. To do this, we can compare the probability distributions of time series samples drawn from different periods of time. The larger the difference between two distributions, the more likely a change-point has occurred. These differences can be computed using cumulative sum, generalized likelihood-ratio, or change finder [45]–[47].

More recently, techniques have been developed which skip the estimation of the two separate probability densities and instead just estimate the probability density ratio between the two samples [45], [48]. The intuition behind these techniques is that knowing the probability densities implies knowing the probability density ratio, although the reverse is not true. Knowing the probability density ratio does not imply knowing the probability densities, which indicates that estimating the probability density ratio should be an easier problem. Examples of direct density ratio estimation include kernel mean matching, logistic regression, Kullback-Leibler importance estimation, unconstrained least-squares importance fitting, and relative unconstrained least-squares importance fitting (RuLSIF) [44], [45], [48]. Bayesian inference techniques can be used for real-time estimation [49], but estimating the distribution function and data parameters must be performed a priori. RuLSIF has been shown to be one of the most promising techniques for direct-density estimation and change-point detection, so we use it for our transition detection problem.

II. Transition Detection

We implement two different algorithms to detect activity transitions. The first is a supervised learning algorithm and the second is an unsupervised change-point detection algorithm. The premise for detecting transitions under each model is different. The supervised learning algorithm operates under the assumption that a transition period can be detected as a distinct activity independent of what activities the user is transitioning between. In contrast, the unsupervised learning algorithm operates under the assumption that the distribution of sensor event data changes as the user transitions from one activity to another activity and hence transitions can be detected by changes in the event data distribution without supervised training. Implementation details for these algorithms are described next.

A. Supervised Learning

Our supervised activity recognition algorithm provides real-time activity labeling as sensor events arrive in a stream [50]. To do this, we formulate the learning problem as that of mapping a sequence consisting of the most recent sensor events within a sliding window of length k to a label representing the activity to the last (most recent) event in the sequence. The sensor events preceding the last event define the context for this last event. Data collected in a smart home consists of events generated by the sensors. These are stored as a 4-tuple: (Date, Time, Sensor Id, Message). For example, the following sequence of sensor events would be mapped to a *Sleep* activity label.

2011-06-15	03:38:23.271939	BedMotionSensor	ON
2011-06-15	03:38:28.212060	BedMotionSensor	ON
2011-06-15	03:38:29.213955	BedMotionSensor	ON

We extract features from data point i, where the data point corresponds to a sensor event sequence of length k. The vector x_i includes values for the 64 features summarized in Table 1. Each y_i corresponds to the activity label that is associated with the last sensor event in the sequence. A collection of data points x_i and the corresponding labels y_i are fed as training data to a classifier to learn the activity models in a discriminative manner. The classifier thus learns a mapping from the sensor event sequence to the corresponding activity label.

Our activity recognition algorithm employs a decision tree learner because this method provides consistently accurate results while imposing minimal computational requirements for training and testing the models. However, we also use the WEKA toolkit [51] to provide comparative results using naïve Bayes, logistic regression, supportvector machine and adaboost algorithms. Although these algorithms are capable of handling many class labels simultaneously, we are specifically interested in detecting transitions independent of the activities that the user is transitioning between. Therefore, we formulate the problem as a two-class problem (i.e., Transition or Non-Transition).

B. Unsupervised Learning

Our unsupervised activity transition technique uses Relative Unconstrained Least Squares Importance Fitting (RuLSIF) to estimate the probability density distribution ratio between two samples. This ratio can then be used to detect changes in the underlying data distribution. As a result, this approach detects activity transitions by detecting changes in the underlying probability distributions, using the assumption that the distribution changes as the activities change. This can be viewed as the change-point detection problem encountered in time-series data, which is formally defined as follows. Let $y(t) \in \mathbb{R}^d$ represent a series sample at time t. Let $Y(t) \coloneqq [y(t)^T, y(t+1)^T, \ldots, y(t+k-1)^T]^T \in \mathbb{R}^{dk}$ be a forwarded subsequence of the time series starting at time t with length k. By treating $Y(t)$ as a sample instead of $y(t)$ we can incorporate time-dependent information. Finally, let $\gamma(t)$ be a set of n subsequence samples starting at time t such that

$$\gamma(t) := [Y(t), Y(t+1), \ldots, Y(t+n-1)] \quad (1)$$

Given two consecutive segments $\gamma(t)$ and $\gamma(t+n)$, the problem is to determine whether a change-point has occurred between these two segments. RuLSIF determines whether a change-point has occurred by computing a dissimilarity measure for the two segments as $PE(P\|P') + PE(P'\|P)$, where P and P' are the probability distributions of the samples in $\gamma(t)$ and $\gamma(t+n)$, respectively. PE here represents the α-relative Pearson Divergence

$$PE(P\|P') := \frac{1}{2}\int p'_\alpha(Y)\left(\frac{p(Y)}{p'_\alpha(Y)} - 1\right) \mathrm{d}Y \quad (2)$$

where $p(Y)$ and $p'(Y)$ are probability density functions of P and P', respectively, and $p'_\alpha(Y) = \alpha p(Y) + (1-\alpha)p'(Y)$ is the α-mixture density. The term $p'\alpha(Y)$ places an upper bound on the density ratio of $1/\alpha$ for $\alpha > 0$. Unfortunately, $p(Y)$ and $p'(Y)$ are typically unknown, thus we are unable to compute the Pearson divergence directly. Rather than estimating these probability densities, which is a difficult problem, we instead estimate the density ratio directly [44]. The advantage of adapting RuLSIF for use in activity transition detection is that it provides an analytic solution and has been shown to perform well in detecting change-points in time-series data [44].

For detecting activity transitions, we create our times series data in the following manner. The current state of all of the sensors is defined as a single data point. This gives us a d-dimensional data point, where d is the number of sensors in the smart apartment. The current state is then updated at every time step to yield our time series data. RuLSIF is an offline change-point detection method, requiring $n + k$ time steps of data ahead of the current point $t + n$ to detect a change-point. However, in our work we redefine the change-point score as occurring at time step $t + 2n + k$ instead of $t + n$ and thus eliminate the lag at the cost of a reduction in accuracy. The difference in accuracy is examined in Section IV. All the values reported in the rest of this paper assume the change point score is for the point at $t + 2n + k$.

We can improve the accuracy of the transition detection for both the supervised and unsupervised learning techniques by recognizing the fact that activities and transitions occur

over a period of time. If the previous time-window is classified as an activity than the current time window is likely an activity also. Similarly, if the previous time-window is classified as a transition, then the current time-window is likely a transition as well. To incorporate this fact we calculate the final time-window prediction based upon the sum of the classification probabilities (or *PE* for the RuLSIF technique) for the previous *m* time-windows. If the summed probability that the time-window is a transition period (or summed *PE* score) is above a specified threshold than we output a transition label for the current time-window. Table II shows an example of how this works when $m = 3$. In addition to increasing the accuracy of the transition detection (by reducing false positives) the technique also facilitates adjusting the trade-off between true-positives and false-positives by manipulating the threshold value.

III. Methods

In order to evaluate our proposed supervised and unsupervised approaches to activity transition detection we need some ground truth labeled data by which the algorithms performance can be evaluated. However, obtaining detailed and accurate labeled activity data is itself a challenging problem. One approach is to collect data in a controlled environment. This allows for accurate and detailed labeling of the participant's activities but the performed activities may not be as realistic as if the activities had been performed in less controlled environment. Another approach is to collect data in the natural environment of the participant. This approach captures more realistic activities but it is also more difficult to obtain detailed and accurate activity labels in such an environment due to privacy concerns. In a controlled laboratory setting, live video cameras can be used to accurately label the activities but in the natural environment of the participant a live video camera is often viewed as too invasive.

The first dataset (referred to as *scripted*) represents activities performed in a scripted environment and has been collected specifically for the purpose of analyzing and detecting activity transitions in a memory notebook prompting study. In contrast, the second dataset (referred to as *unscripted*) is collected in an actual smart home installation while residents perform their normal daily routines.

We use two different performance metrics to evaluate our transition detection algorithms. For the first metric we evaluate transition detection as a type of classification problem. Correspondingly, we calculate the area under the curve (AUC) of the receiver operating characteristic curve, to compare the performance of two binary classifiers across the entire spectrum of true positive and false positive rates (TPR and FPR, respectively). As in the work of Liu et al. we consider a transition detected at time step t as correct if $t \in [t'-\delta, t'+\delta]$, where t' is a true transition point. Similarly, a transition detected at time step t is considered incorrect if $t \notin [t' - \delta, t' + \delta]$ for all possible true transition points t'. Transitions detected within δ seconds of a previously detected transition point are ignored. In our experiments we use $\delta = 20$ seconds. An ROC curve is generated for the AR and RuLSIF techniques by ordering the transitions points according to the transition score output by each algorithm. The TPR and FPR are then updated for each point until all of the transition points have been processed.

The second performance metric we use calculates the root mean squared error (RMSE) for the classification threshold value yielding a TPR > 80%. RMSE is defined as

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} SE(i)} \quad (3)$$

where n is the number of transitions detected and we define $SE(i)$ as

$$SE(i) = \begin{cases} 0 & \text{if } t_i \in [t' - \delta, t' + \delta] \\ (t_i - t')^2 & \text{if } t_i \notin [t' - \delta, t' + \delta] \end{cases} \quad (4)$$

where t_i is the time step of the i^{th} detected transition and t' is the closest true transition point.

A. Scripted Data

1) Participants—For the scripted data, 15 healthy adults were recruited as participants. The participants ranged in age from 18 to 36 years old with a mean age of 21.6 years old. 9 females participated in the study and 6 males also participated in the study.

2) Procedure—Data was collected in the WSU smart apartment (SA) on campus (See Figure 1). The SA is equipped with infrared motion sensors, door sensors, light switch sensors, and power usage sensors. There were also nine shake (vibration) sensors attached to items related to the activities participants were asked to perform in the SA. These items included a broom, dustpan, duster, oatmeal container, puzzle box, television remote, picnic basket, and water filter. All sensor events are transmitted wirelessly to a central server and stored in a relational database.

There were twelve total activities used in this project. These activities were selected as part of a larger ongoing effort to perform assessment and intervention design for older adults. The selected activities represent instrumental Activities of Daily Living that have been shown in the literature to be disrupted by individuals who experience cognitive difficulties [52]. These activities included: sweeping the kitchen, dusting the house, making oatmeal, watching television, doing a math worksheet, playing with a handheld game, putting an outfit together, collecting items from around the house, collecting ingredients, completing a puzzle, reading a magazine, and copying a recipe. While participants were completing each task, they were observed from a separate room by the experimenter. The experimenter used a Real-time Annotation Tool (RAT) [53] to record into a database what the participant was doing, when activity transitions occurred, and when memory notebook prompts were delivered.

Participants were introduced to the SA and oriented to the general layout, as well as given a brief description of what they would be expected to do. The experimenter delivered instructions to the participants through an intercom while observing the participants' activities from a room upstairs via a web camera (see Figure 3), using a Wizard of Oz experiment design. The tasks given to the participant were counterbalanced in that half of the participants were given similar tasks side by side (two back-to-back inactive tasks followed by two back-to-back active tasks) and the other half were given tasks that varied by

the location in the apartment where they took place. This counterbalancing strategy was applied so that the machine learning algorithm would not recognize activity transitions as a function solely of the participant location or solely based on activity level. Table 2 lists the order of the tasks for the two conditions. Directions for each task were delivered at set times during the preceding task so as to facilitate smooth transitions between activities. To ensure that each transition between activities happened naturally, participants were informed of the next task before they completed the current task. In this way, the participant did not have to pause and wait for directions between tasks.

For the RuLSIF algorithm we set $k = 16$ and $n = 60$ with a step size of .5 seconds. We found these values to yield good results without being too computationally expensive. The number of previous windows to consider, m, is set to a value of 10 which represents a reasonable trade-off between incorporating previous time-window classifications and keeping the classification relevant to the current time-window.

3) Experimental Design and Analysis—We use a leave-one out approach for the training and testing of the supervised learning algorithms. The supervised learning algorithm is trained on all of the annotated data from the other fourteen participants and the algorithm is then tested on the data from the fifteenth participant. This process is repeated for all fifteen participants and the average performance is then reported.

The unsupervised techniques do not require any training data. Each algorithm is tested on data collected for each participant and the average performance is reported.

For both performance metrics, we perform a repeated measures one-way analysis of variance (ANOVA) test to verify the difference between algorithms is significant. The performance metric score is the dependent variable and the learning algorithm used is the independent variable.

B. Unscripted Data

1) Participants—For the unscripted data, 8 older adults in an assisted care facility were recruited as participants. The participants ranged in age from 73 to 91 years old with a mean age of 83.25 years old. 2 males participated in the study and 6 females also participated in the study.

2) Procedure—We also evaluate our approach using data collected in smart home testbeds that were installed in 8 apartments at a retirement community in Seattle, Washington. Each of the apartments houses a single older adult (age 75+) resident who performs a normal daily routine while sensors in the apartment generate and store events. The sensors include combination wireless motion/light intensity sensors and combination door/temperature sensors. There is an average of 25 sensors installed in each apartment. For each apartment, we analyze approximately three hours of data starting when the participant first wakes up. The collected data has been annotated with 30+ activity labels corresponding to activities of daily living (ADLs) and other activities of interest. Since this annotated data does not specifically include activity transition annotations, we define transition periods in the following manner. Transitions are defined as the last two sensor events of one activity and

the first two sensor events of the second activity if the two activities are not labeled as "other". "Other" activities lasting less than 1 minute are labeled as transitions and "other" activities lasting more than 1 minute are treated as actual activities. The resulting problem is a two class, or binary, learning problem to distinguish activities from transitions.

For the RuLSIF algorithm we set $k = 8$ and $n = 30$ with a step size of 1 second. As can be noted, the timing is the same between the scripted and the unscripted settings with the only difference being the time step granularity. The scripted setting has many more sensors than the unscripted setting which necessitates the smaller step sizes to better capture the current state of all of the sensors. The value of m is again set to 10.

3) Experimental Design and Analysis—We use a leave-one out approach for the training and testing of the supervised learning algorithms. The supervised learning algorithm is trained on all of the annotated data from the other seven participants and the algorithm is then tested on the data from the eighth participant. This process is repeated for all eight participants and the average performance is then reported.

The unsupervised techniques do not require any training data. Each algorithm is tested on data collected for each participant and the average performance is reported.

For both performance metrics, we perform a repeated measures one-way analysis of variance (ANOVA) test to verify the difference between algorithms is significant. The performance metric score is the dependent variable and the learning algorithm used is the independent variable.

IV. Results

We first examine the loss in accuracy due to the redefinition of the location of the change-point in the RuLSIF algorithm. RuLSIF is an offline change-point detection method, requiring $n + k$ time steps of data ahead of the current point $t+n$ to detect a change-point. However, in our work we need an online algorithm and thus we redefine the change-point score as occurring at time step $t + 2n + k$. Figure 4 shows the effect this has on performance. In the scripted setting, the online method causes the accuracy to drop by approximately 0.09 units while the RMSE increases by about 12 units. Similar results are seen in the unscripted with the accuracy dropping by 0.02 units and the RMSE increasing by approximately 42 units.

Next, we compare our results against two different baselines, Fixed and Random. The Fixed baseline detects transitions at fixed-time intervals of length δ. In order to generate an ROC curve for this method, the transitions are ranked in the following order: $n/2$, $n/4$, $3n/4$, $n/8$, $3n/8$, $5n/8$, $7n/8$ … n where n is the total number of transitions detected. The Random baseline repeatedly selects a random time-step at which to detect a transition. As with the other techniques, transitions detected within δ time-steps of a previously-detected transition are ignored. An ROC curve is generated for the Random baseline by updating the TPR and FPR after each sample is drawn.

The results of the supervised learning techniques are shown in Figures 5 and 6. AR-DT, AR-NB, AR-LR, AR-BOOST and AR-SVM represent the Decision Tree, Naïve Bayes, Logistic Regression, AdaBoost and Support Vector Machines algorithms, respectively. Error bars indicate the standard error of the mean. For scripted data, the AR-DT, AR-LR, AR-BOOST and AR-SVM algorithms outperform the baseline classifiers. A repeated measures ANOVA with a Greenhouse-Geisser correction determined that mean AUC scores differed significantly between algorithms ($F(2.227, 31.172) = 46.866$, $P < 0.0005$). The AR-NB algorithm has a similar performance to the Fixed baseline on the scripted data and both perform better than the Random baseline. The results are similar for the RMSE metric with AR-SVM giving the best performance and the other AR techniques still performing better than the baseline techniques. Again, the differences are significant on the scripted data ($F(3.386, 47.405) = 33.154$, $P < 0.0005$).

Figures 5 and 6 also show the results of the supervised learning algorithms when applied to the unscripted setting. AR-DT, AR-LR and AR-NB outperform the baseline techniques but AR-BOOST and AR-SVM do not. The differences are still significant ($F(3.046, 21.322) = 16.082$, $P < 0.0005$). The RMSE results show greater variation in the unscripted setting and the differences between algorithms are not significant.

The results of the unsupervised learning technique (RuLSIF) are shown in Figures 7 and 8. For comparison purposes we also include the top performing supervised technique, AR-LR, as well as the two baselines, Fixed and Random. The AR-LR is not included in the tests for statistical significance since it is not an unsupervised algorithm. As can be seen in the figures, on the scripted data, the AUC score for RuLSIF is not as high as AR-LR but it matches the performance of the Fixed baseline and outperforms the Random baseline. On the unscripted dataset RuLSIF performs much better, nearly matching the performance of the supervised technique and clearly outperforming the Fixed and Random baselines. A repeated measures ANOVA with a Greenhouse-Geisser correction determined that mean AUC scores on the scripted dataset differed significantly between algorithms ($F(1.845, 25.833) = 10.06$, $P = 0.001$). The AUC score differences are also significant for the unscripted datasets ($F(1.5, 10.5) = 19.027$, $P = 0.001$). The acceptable performance of the Fixed algorithm is probably due to the nature of the scripted data. Although the exact timing of the transitions is usually left to the participant, the activities have been structured in such a way as to last between 3–7 minutes. This structure benefits the Fixed algorithm as it chooses transition points at fixed-time intervals.

The results are similar for the RMSE metric. RuLSIF is, not surprisingly, unable to match the performance of the supervised learning algorithm in the scripted setting but still outperforms the baseline techniques. These differences are significant ($F(1.108, 15.509) = 5.357$, $P = 0.032$). In the unscripted setting, RuLSIF actually outperforms all the other techniques including the supervised learning algorithms. The RMSE differences are significant ($F(1.019, 7.135) = 5.713$, $P = 0.047$).

To better understand how the AUC score relates to the TPR and the FPR we also include the average ROC for each algorithm in Figure 9. As can be seen, a higher AUC score generally translates into higher TPR at comparable FPRs, but which algorithm is better may depend on

the desired trade-off between the TPR and the FPR. In almost every case both the supervised and the unsupervised algorithms are better than the baseline algorithms indicating that transitions can indeed be detected using these techniques. In both the scripted and unscripted settings a true positive rate of greater than 80% can be achieved while maintaining a false positive rate of less than 15%.

The RMSE can be thought of as the average time between when a transition occurs and when the transition is detected. For the scripted setting this value is close to one minute. For the unscripted setting it is approximately two minutes. Ideally, we would like to minimize these times to reduce task interruption. On the other hand, a minute or two into an hour-long task may not be that significant. Additionally, the RMSE has been measured at the 80% detection rate of all transitions. This may be a much higher accuracy than is needed for most prompting systems and selecting a point lower on the ROC curve may yield better RMSE values as the confidence in the prediction increases.

The final question remaining to be addressed is what these results indicate for the automatic detection of transitions to improve prompting and notification technologies. In a separate experiment with 42 participants, prompts to record recent events in a digital memory notebook were delivered manually by experimenters either during activity transitions (transition-based condition) or every five minutes (time-based condition), with half of the group being assigned to each condition. Participants had the option to respond to each prompt either immediately, after a second prompt (given one minute later) or not at all if the prompt felt unnatural and inconvenient.

We evaluate the effectiveness of the prompting based on four measures: the percentage of participants that responded to the first prompt for each group, prompt compliance (use of the digital memory notebook) for each group, number of task interruptions that occurred as a result of the prompt for each group, and the mean time spent on each task. Participants that were assigned to the transition-based condition responded to the first prompt 68% of the time (SD=0.29), while those in the time-based condition responded to the prompt only 45% of the time (SD=0.23). Prompt compliance was 90% for the transition-based condition (SD=0.15) in comparison with 59% for the time-based condition (SD=0.28). The number of task interruptions that occurred from the transition-based condition was 9% (SD=0.15) as opposed to 40% (SD=0.28) for the time-based condition. Finally, the mean task time for the transition condition is 193.47 seconds (SD=81.48) and for the time-based condition is 228.35 seconds (SD=86.46). In each of the cases, the difference between the measure for the group is significant ($p < 0.01$). This study provides some evidence that providing prompts during activity transitions is a more effective intervention mechanism that has less impact on other activities the individuals perform in their normal routine.

V. Conclusions

Our results from this study indicate that both supervised learning and change-point detection are valid techniques for detecting activity transitions. The next step is to integrate this work with a smart home prompting system to test the effectiveness of the techniques in real-time. We are currently conducting a study with this goal in mind. Our future studies will also

evaluate the technology for use by older adults. We anticipate that usage will differ substantially based on the types of conditions that these adults face, and will consider these factors individually.

In the future, some limitations of the current approaches need to be addressed. In real-world situations, transitions are not always clearly defined. For example, a user may interleave two or more activities with no clear transitions between them. Also, a user may pause for a large amount of time in the middle of completing an activity. We will need to adapt our algorithms to handle these types of situations. Factors such as the length of an activity, the similarity of different activities, the number of possible activities, and the transition time between activities may also influence performance and can be evaluated separately. We also plan to assess the amount of time which can elapse between a transition and a notification without becoming a burden to the user. Furthermore, we plan to continue to investigate new supervised and unsupervised algorithms to better detect activity transitions.

Transition detection is an important problem for prompting systems, notification management systems and other applications, but automatically detecting transition periods using ambient sensors has received little attention in the literature. We have adapted two techniques to accurately detect activity transitions, supervised activity recognition and RuLSIF. Both algorithms have been tested in the context of time series data, but the application of these algorithms to a context-aware prompting system is novel. To our knowledge this is the first work showing that one can use the same features used for activity recognition to detect activity transitions independent of the activities being performed. Additionally it is the first attempt to use change-point detection to detect activity transitions in a smart environment. Finally, this work is novel because we are providing the first design and real-world implementation of a digital memory notebook with automated prompts for notebook use.

A memory notebook can be an effective intervention for individuals with MCI. However, the initial training to use the memory notebook requires a constant delivery of prompts and reminders. Traditionally, these are delivered by a care-partner or a time-based alarm. By automating the detection of transition periods we can increase the effectiveness of the prompts and simultaneously decrease the workload of the care-partner.

Our results indicate that both techniques are able to detect transitions and each technique has associated trade-offs. The supervised learning technique performs well when enough training data is available to accurately model activity transitions. The RuLSIF technique does not require any label data and generalizes well but does not match the performance of a well-trained supervised learning model. The development of accurate transition detection algorithms will lead to more efficient prompting and notification systems which will in turn result in less burdensome interruptions to the end-user.

References

1. Vincent GK, Velkoff VA. The next four decades: The older population in the united states: 2010 to 2050. US Census Bureau, Tech. Rep. 2010
2. Association A. Alzheimers disease: facts and figures. Alzheimers Dementia. 2012; 8:131–168.

3. Winblad B, Palmer K, Kivipelto M, Jelic V, Fratiglioni L, Wahlund L-O, Nordberg A, Bckman L, Albert M, Almkvist O, Arai H, Basun H, Blennow K, De Leon M, DeCarli C, Erkinjuntti T, Giacobini E, Graff C, Hardy J, Jack C, Jorm A, Ritchie K, Van Duijn C, Visser P, Petersen R. Mild cognitive impairment beyond controversies, towards a consensus: report of the international working group on mild cognitive impairment. Journal of Internal Medicine. 2004; 256(3):240–246. [Online]. Available: http://dx.doi.org/10.1111/j.1365-2796.2004.01380.x. [PubMed: 15324367]
4. Buschert VC, Friese U, Teipel SJ, Schneider P, Merensky W, Rujescu D, Müller H-J, Hampel H, Buerger K. Effects of a newly developed cognitive intervention in amnestic mild cognitive impairment and mild alzheimers disease: a pilot study. Journal of Alzheimer's Disease. 2011; 25:679–694.
5. Greenaway M, Hanna S, Lepore S, Smith G. A behavioral rehabilitation intervention for amnestic mild cognitive impairment. American Journal of Alzheimers Disease & Other Dementias. 2008; 23:451–461.
6. Schmitter-Edgecombe M, Howard JT, Pavawalla SP, Howell L, Rueda A. Multidyad memory notebook intervention for very mild dementia: A pilot study. American Journal of Alzheimer's Disease and Other Dementias. 2008; 23(5):477–487.
7. Cook D. Learning setting-generalized activity models for smart spaces. Intelligent Systems, IEEE. 2012; 27(1):32–38.
8. Acampora G, Cook D, Rashidi P, Vasilakos A. A survey on ambient intelligence in healthcare. Proceedings of the IEEE. 2013 Dec; 101(12):2470–2494. [PubMed: 24431472]
9. Barger T, Brown D, Alwan M. Health status monitoring through analysis of behavioral patterns. IEEE Transactions on Systems Man and Cybernitics Part A. 2005; 35(1):2227.
10. Hoey J, Bertoldi AV, Craig T, Poupart P, Mihailidis A. Automated handwashing assistance for persons with dementia using video and a partially observable Markov decision process. Computer Vision and Image Understanding. 2010; 114(5):503519.
11. Seelye A, Schmitter-Edgecombe M, Das B, Cook D. Application of cognitive rehabilitation theory to the development of smart prompting technologies. Biomedical Engineering. 2011; 99:1–18.
12. Farias S, Mungas D, Reed B, Harvey D, Cahn-Weiner D, DeCarli C. Mci is associated with deficits in everyday functioning. Alzheimer Disease and Associated Disorders. 2006; 20:217223.
13. Schmitter-Edgecombe M, Woo E, Greeley D. Characterizing multiple memory deficits and their relation to everyday functioning in individuals with mild cognitive impairment. Neuropsychology. 2009; 23:168177.
14. Cicerone KD, Dahlberg C, Malec JF, Langenbhn DM, Felicetti T, Kneipp S. Evidence-based cognitive rehabilitation: updated review of the literature from 1998 through 2002. Archives of Physical Medicine and Rehabilitation. 2005; 86:16811692.
15. Langill M, Schmitter-Edgecombe M. Memory notebook training for very mild dementia: a case study. Annual Meeting of the Association for Behavioral Analysis International. 2007
16. Schmitter-Edgecombe, M.; Sanders, C.; Low, C.; Warren, L.; Norell, D.; Wu, L.; Dyck, D. An integrated cognitive rehabilitation multi-family group intervention for individuals with mild cognitive impairment and their care partners: preliminary data; Alzheimers Association International Conference; 2012.
17. Schmitter-Edgecombe, S.; Pavawalla, LHM.; Howard, JT.; Rueda, A. Dyadic interventions for persons with early-stage dementia: A cognitive rehabilitative focus. In: Bougham, R., editor. New Directions in Aging Research: Health and Cognition. Nova Science Publishers; 2009.
18. Adamczyk, PD.; Bailey, BP. If not now when? the effects of interruptions at different moments within task execution; ACM Conference on Human Factors in Computing Systems; 2004. p. 271278
19. Boger J, Mihailidis A. The future of intelligent assistive technologies for cognition: Devices under development to support independent living and aging-with-choice. Neurorehabilitation. 2011; 28:271–280. [PubMed: 21558632]
20. Bailey BP, Konstan JA. On the need for attention award systems: Measuring effects of interruption on task performance, error rate, and affective state. Journal of Computers in Human Behavior. 2006; 22(4):709732.

21. Iqbal, ST.; Baile, BP. Leveraging characteristics of task structure to predict costs of interruption; ACM Conference on Human Factors in Computing Systems; 2006. p. 741750
22. Modayil J, Levinson R, Harman C, Halper D, Kautz HA. Integrating sensing and cueing for more effective activity reminders. AAAI Fall Symposium: AI in Eldercare: New Solutions to Old Problems, ser. AAAI Technical Report. AAAI. 2008:60–66.
23. Ferguson H, Myles BS, Hagiwara T. Using a personal digital assistant to enhance the independence of an adolescent with Asperger Syndrome. Education and Training in Developmental Disabilities. 2005; 40:6067.
24. Lancioni GE, Coninx R, Manders N, Driessen M, Dijk JV, Visser T. Reducing breaks in performance of multihandicapped students through automatic prompting or peer supervision. Journal of Developmental and Physical Disabilities. 1991; 3:115128.
25. Wilson BA, Evans JJ, Emslie H, Malinek V. Evaluation of NeuroPage: a new memory aid. Journal of Neurology Neurosurgery and Psychiatry. 1997; 63(1):113115.
26. Hayes T, Cobbinah K, Dishongh T, Kaye J, Kimel J, Labhard M, Leen T, Lundell J, Ozertem U, Pavel M, Philipose M, Rhodes K, Vurgun S. A study of medication taking and unobtrusive, intelligent reminding. Telemedicine and Ehealth. 2009; 15:770–776.
27. Lundell, J.; Hayes, T.; Vurgun, S.; Ozertem, U.; Kimel, J.; Kaye, J.; Guilak, F.; Pavel, M. Continuous activity monitoring and intelligent contextual prompting to improve medication adherence; Engineering in Medicine and Biology Society, 2007. EMBS 2007. 29th Annual International Conference of the IEEE; 2007. p. 6286-6289.
28. Barbeau SJ, Winters PL, Georggi NL, Labrador MA, Perez R. Travel assistance device: utilising global positioning system-enabled mobile phones to aid transit riders with special needs. Intelligent Transport Systems. 2010; 4(1):1223.
29. Ho, J.; Intille, SS. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems ser. CHI '05. New York, NY, USA: ACM; 2005. Using context-aware computing to reduce the perceived burden of interruptions from mobile devices; p. 909-918.[Online]. Available: http://doi.acm.org/10.1145/1054972.1055100
30. Iqbal, ST.; Bailey, BP. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ser. CHI '07. New York, NY, USA: ACM; 2007. Understanding and developing models for detecting and differentiating breakpoints during interactive tasks; p. 697-706.[Online]. Available: http://doi.acm.org/10.1145/1240624.1240732
31. Byrne MD, Anderson JR. Serial modules in parallel: The psychological refractory period and perfect time-sharing. Psychological Review. 2001; 108:847–869. [PubMed: 11699122]
32. Strayer DL, Drews FA, Johnston WA. Cell phone-induced failures of visual attention during simulated driving. Journal of Experimental Psychology. 2003; 9:23–32. [PubMed: 12710835]
33. Alon J, Athitsos V, Yuan Q, Sclaroff S. A unified framework for gesture recognition and spatiotemporal gesture segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 2009; 31(9):1685–1699.
34. Ravi, N.; D, N.; Mysore, P.; Littman, ML. Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence. IAAI: AAAI Press; 2005. Activity recognition from accelerometer data; p. 1541-1546.
35. Amft O, Troster G. On-body sensing solutions for automatic dietary monitoring. Pervasive Computing, IEEE. 2009; 8(2):62–70.
36. Singla G, Cook D, Schmitter-Edgecombe M. Recognizing independent and joint activities among multiple residents in smart environments. Journal of Ambient Intelligence and Humanized Computing. 2010; 1(1):57–63. [Online]. Available: http://dx.doi.org/10.1007/s12652-009-0007-1. [PubMed: 20975986]
37. Liao, L.; Fox, D.; Kautz, H. Proceedings of the 19th International Joint Conference on Artificial Intelligence, ser. IJCAI'05. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc; 2005. Location-based activity recognition using relational markov networks; p. 773-778.[Online]. Available: http://dl.acm.org/citation.cfm?id=1642293.1642417
38. Chen, C.; Das, B.; Cook, D. A data mining framework for activity recognition in smart environments; Intelligent Environments (IE), 2010 Sixth International Conference on; 2010. p. 80-83.

39. van Kasteren, T.; Krose, B. Bayesian activity recognition in residence for elders; Intelligent Environments, 2007. IE 07. 3rd IET International Conference on; 2007. p. 209-212.
40. Ali A, Aggarwal J. Segmentation and recognition of continuous human activity. IEEE Workshop on Detection and recognition of events in video. 2001
41. Hong X, Nugent C. Segmenting sensor data for activity monitoring in smart environments. Personal and Ubiquitous Computing. 2013; 17:545–559.
42. Alon, J.; Athitsos, V.; Sclaroff, S. Proceedings of the 2005 International Conference on Computer Vision in Human-Computer Interaction, ser. ICCV'05. Berlin, Heidelberg: Springer-Verlag; 2005. Accurate and efficient gesture spotting via pruning and subgesture reasoning; p. 189-198.[Online]. Available: http://dx.doi.org/10.1007/11573425_19
43. Gu T, Chen S, Tao X, Lu J. An unsupervised approach to activity recognition and segmentation based on object-use fingerprints. Data and Knowledge Engineering. 2010 Jun; 69(6):533–544.
44. Liu, S.; Yamada, M.; Collier, N.; Sugiyama, M. Change-point detection in time-series data by relative density-ratio estimation. In: Gimelfarb, G.; Hancock, E.; Imiya, A.; Kuijper, A.; Kudo, M.; Omachi, S.; Windeatt, T.; Yamada, K., editors. Structural, Syntactic, and Statistical Pattern Recognition, ser. Lecture Notes in Computer Science. Vol. 7626. Springer Berlin Heidelberg; 2012. p. 363-372.[Online]. Available: http://dx.doi.org/10.1007/978-3-642-34166-3_40
45. Basseville, M.; Nikiforov, IV. Detection of Abrupt Changes: Theory and Application. Upper Saddle River, NJ, USA: Prentice-Hall, Inc; 1993.
46. Gustafsson F. The marginalized likelihood ratio test for detecting abrupt changes. Automatic Control, IEEE Transactions on. 1996; 41(1):66–78.
47. Yamanishi, K.; Takeuchi, J-i. Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ser. KDD '02. New York, NY, USA: ACM; 2002. A unifying framework for detecting outliers and change points from non-stationary time series data; p. 676-681.[Online]. Available: http://doi.acm.org/10.1145/775047.775148
48. Kawahara, Y.; Sugiyama, M. Proceedings of the SIAM International Conference on Data Mining, SDM 2009, April 30 – May 2, 2009, Sparks, Nevada, USA. SIAM; 2009. Change-Point Detection in Time-Series Data by Direct Density-Ratio Estimation; p. 389-400.
49. Adams RP, MacKay DJ. Bayesian online changepoint detection. arXiv preprint arXiv:0710.3742. 2007
50. Krishnan N, Cook D. Activity recognition on streaming sensor data. Pervasive and Mobile Computing. 2014; 10:138–154. [PubMed: 24729780]
51. Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. The weka data mining software: An update. SIGKDD Explor. Newsl. 2009 Nov.11(1):10–18. [Online]. Available: http://doi.acm.org/10.1145/1656274.1656278.
52. Peres K, Chrysostome V, Fabrigoule C, Orgogozo J, Dartigues J, Barberger-Gateau P. Restriction in complex activities of daily living in MCI. Neurology. 2006; 67:461466.
53. Feuz KD, Cook DJ. Real-time annotation tool (rat). AAAI Workshop on Activity Context-Aware System Architectures. 2013

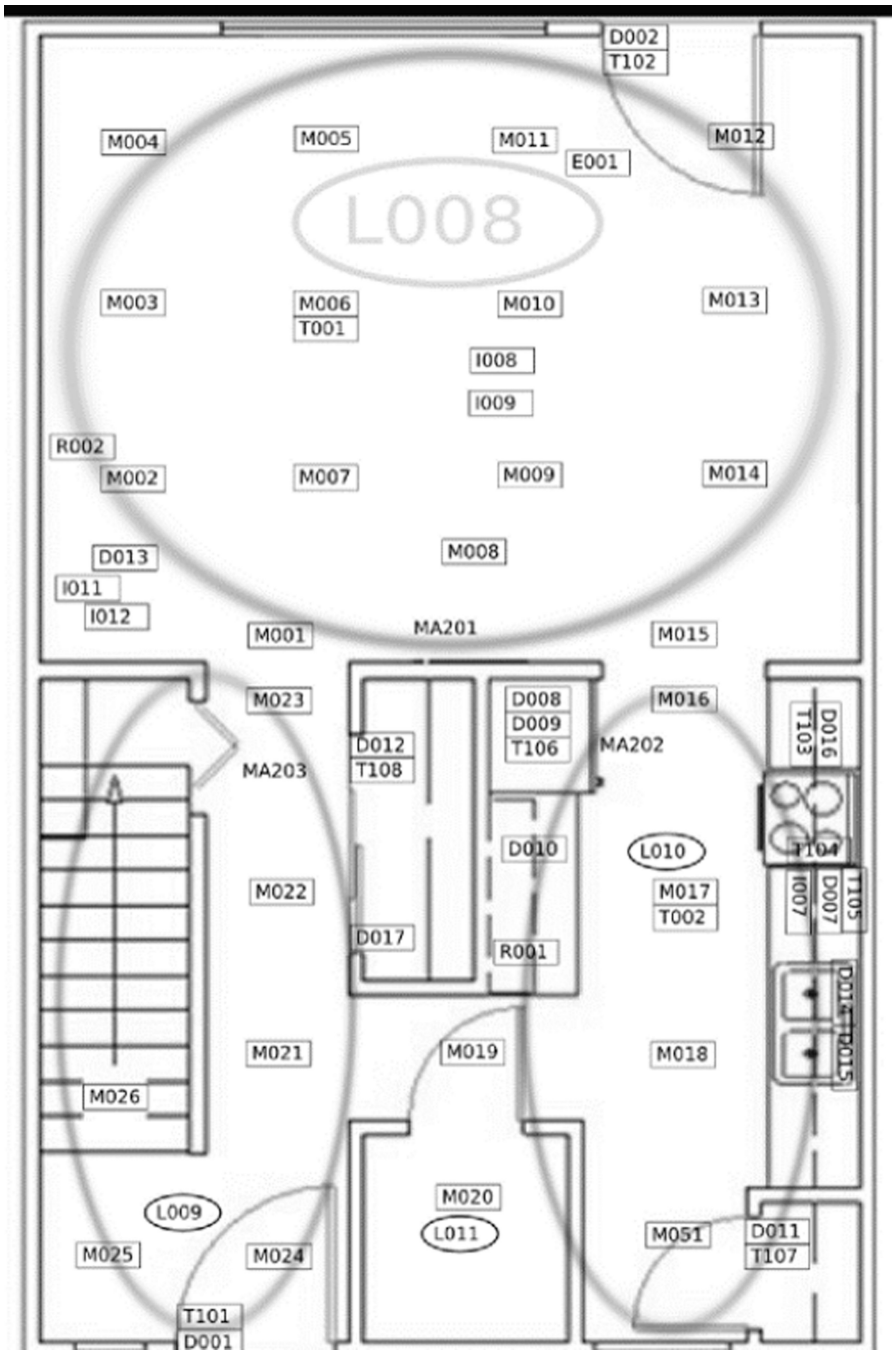

Fig. 1.

The layout of the smart apartment and sensor placement. Sensor labels starting with “M” indicates motion sensors, “L” indicates light sensors, “D” indicates door sensors, “T” indicates temperature sensors, and “I” indicates item sensors.

Fig. 2.
An individual accesses the Digital Memory Notebook.

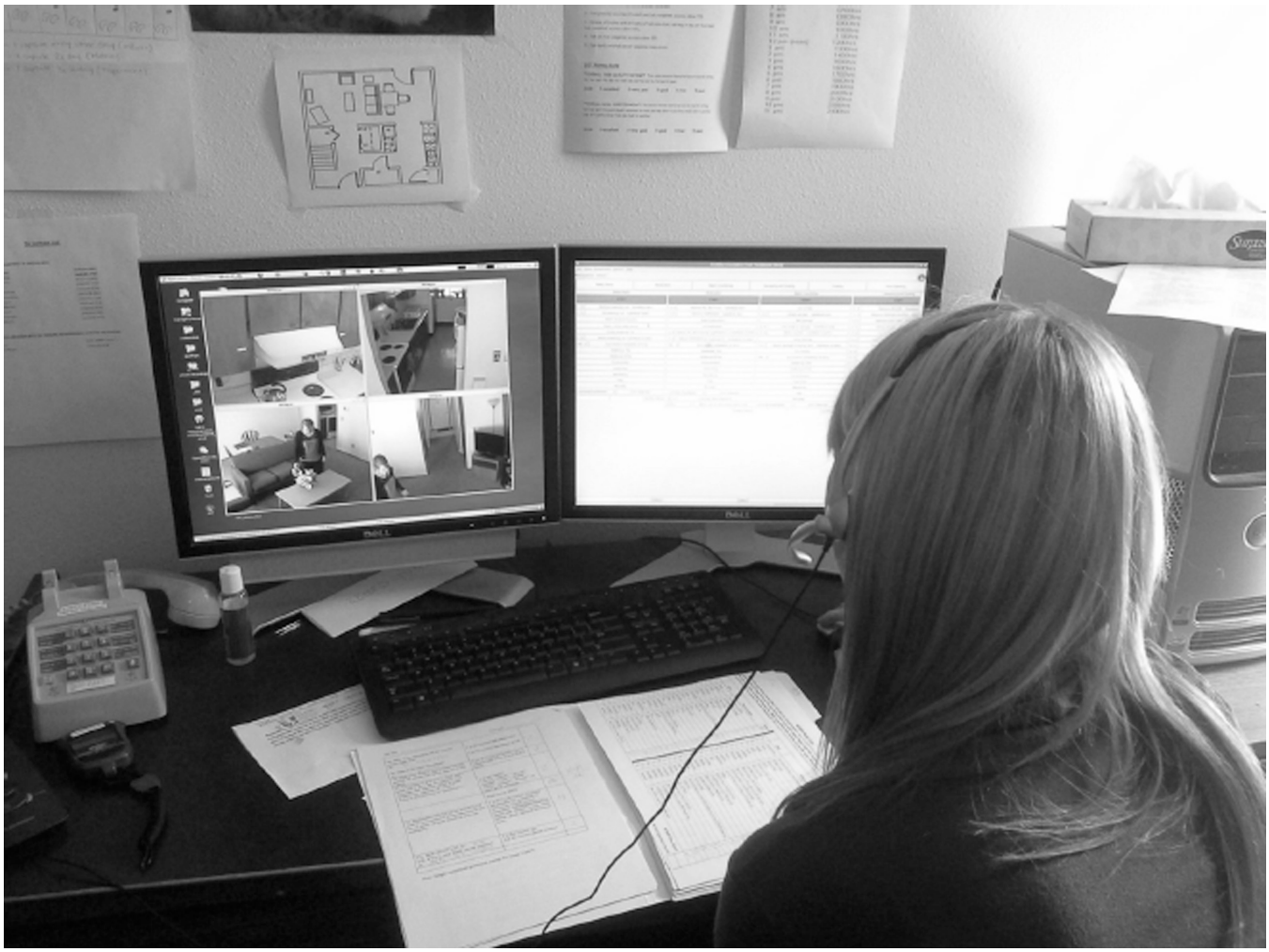

Fig. 3.
An experimenter observes a participant performing everyday activities via web cameras and logs information using the RAT.

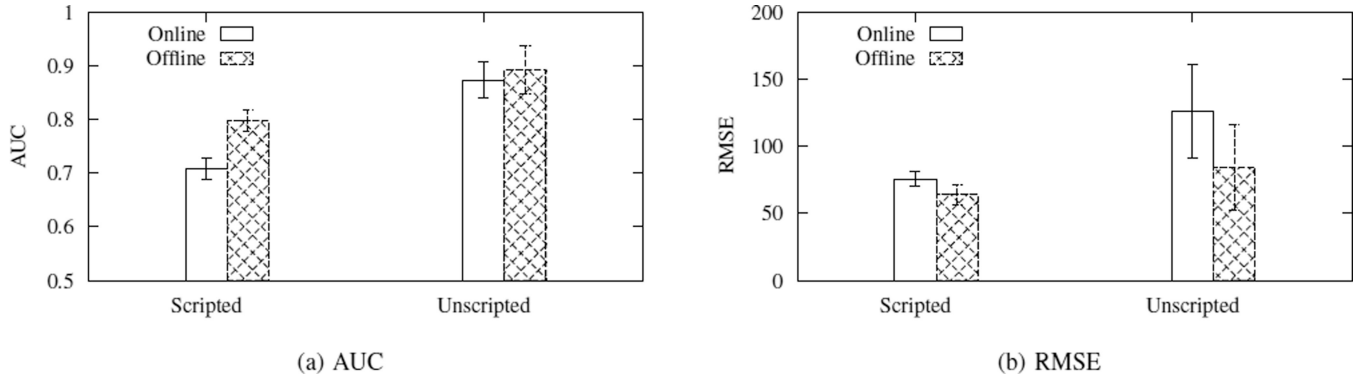

Fig. 4.

Comparison of a) AUC and b) RMSE between RuLSIF Online vs. Offline mode. As expected the Online mode suffers a small decrease in performance. Error bars represent the standard error of the mean.

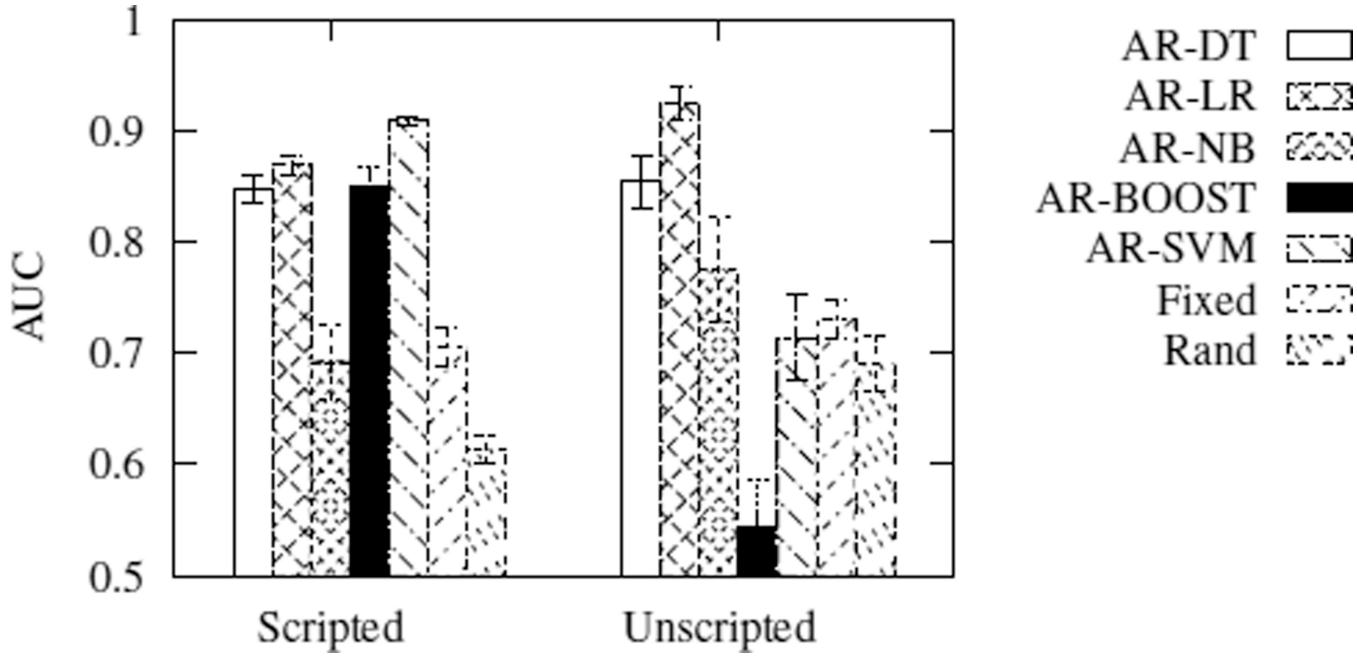

Fig. 5.

AUC scores for supervised learning on the scripted and unscripted datasets. AR-DT, AR-LR, AR-BOOST and AR-SVM show significant improvement over the baseline techniques of Fixed and Random in the scripted setting while only AR-DT and AR-LR outperform the baselines in the unscripted setting. Error bars represent the standard error of the mean.

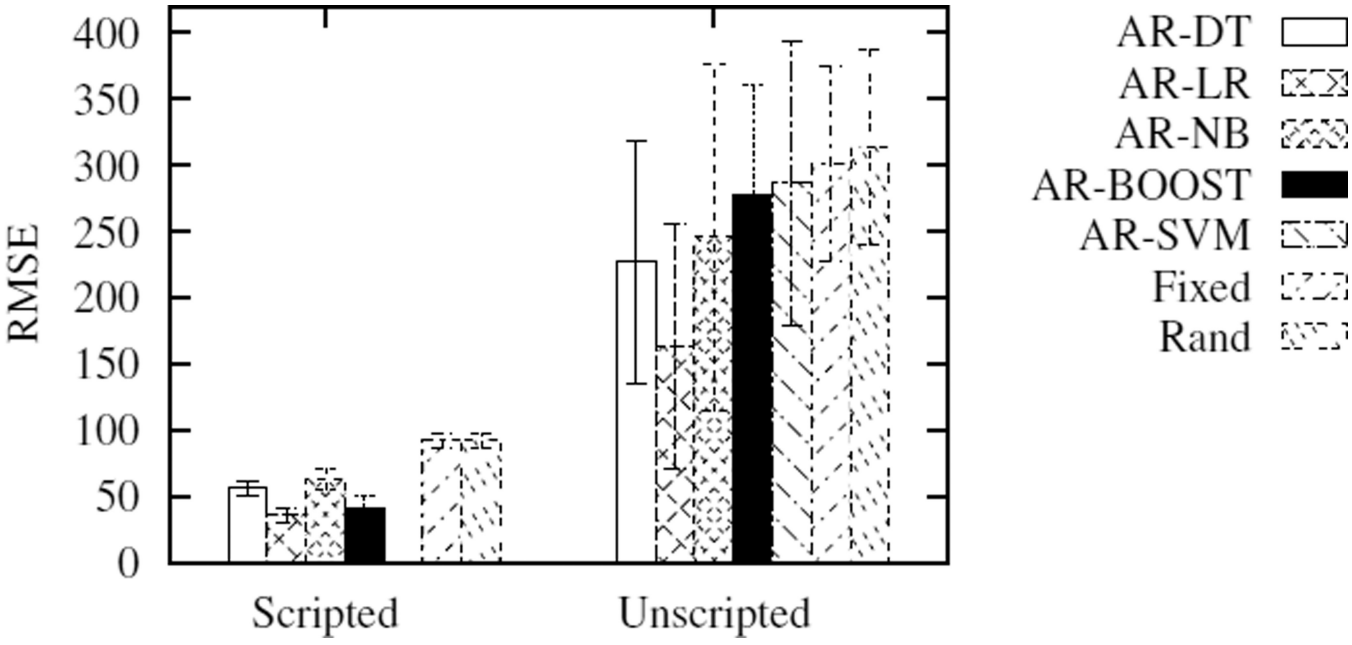

Fig. 6.

RSME scores for supervised learning on the scripted and unscripted datasets. All of the AR techniques outperform the baseline techniques of Fixed and Random. Error bars represent the standard error of the mean.

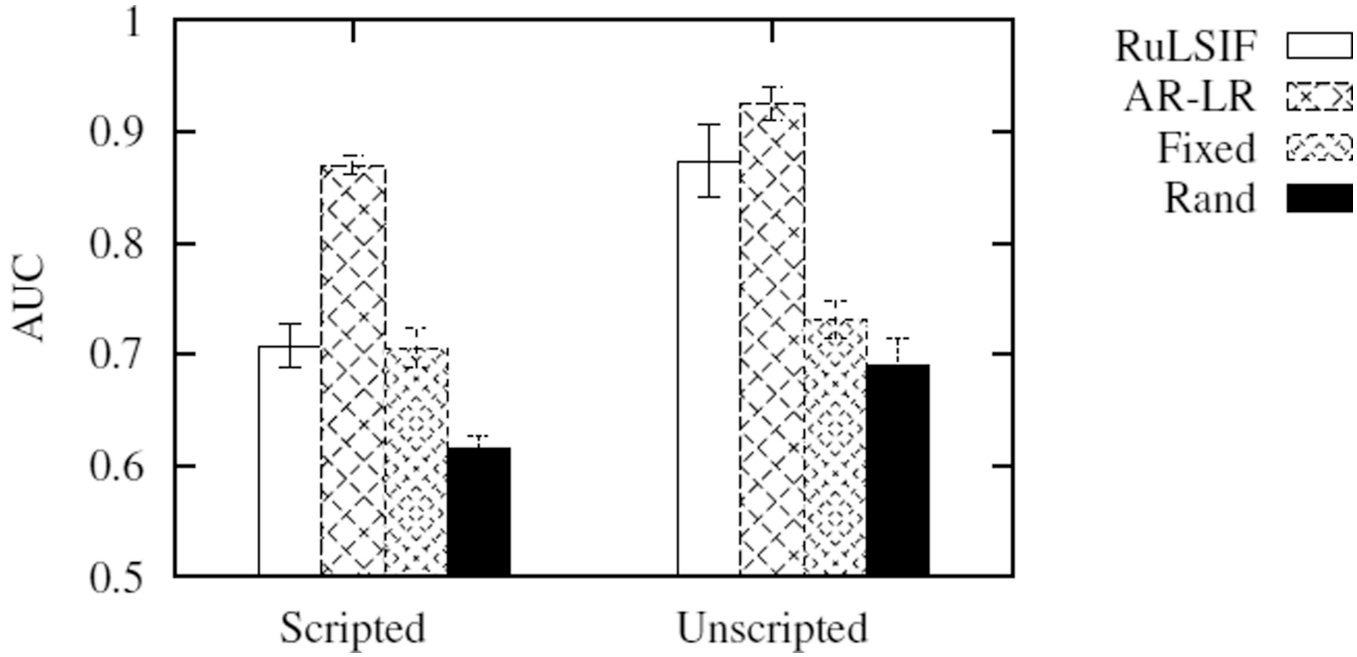

Fig. 7.

AUC scores for the unsupervised learning technique on the scripted and unscripted datasets. AR-LR is a supervised learning technique but is included for comparison purposes. RulSIF outperforms the baseline techniques. Error bars represent the standard error of the mean.

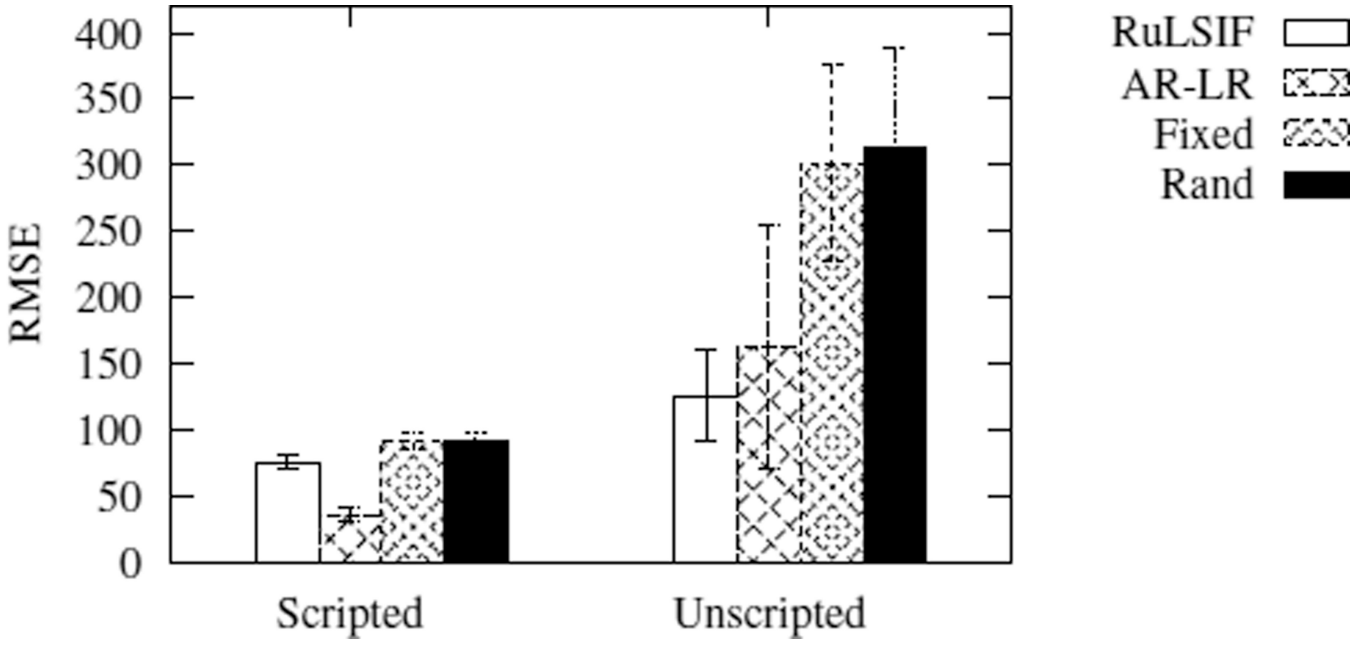

Fig. 8.

RMSE scores for the unsupervised learning technique on the scripted and unscripted datasets. AR-LR is a supervised learning technique but is included for comparison purposes. RulSIF outperforms the baseline techniques and even outperforms AR-LR in the unscripted setting. Error bars represent the standard error of the mean.

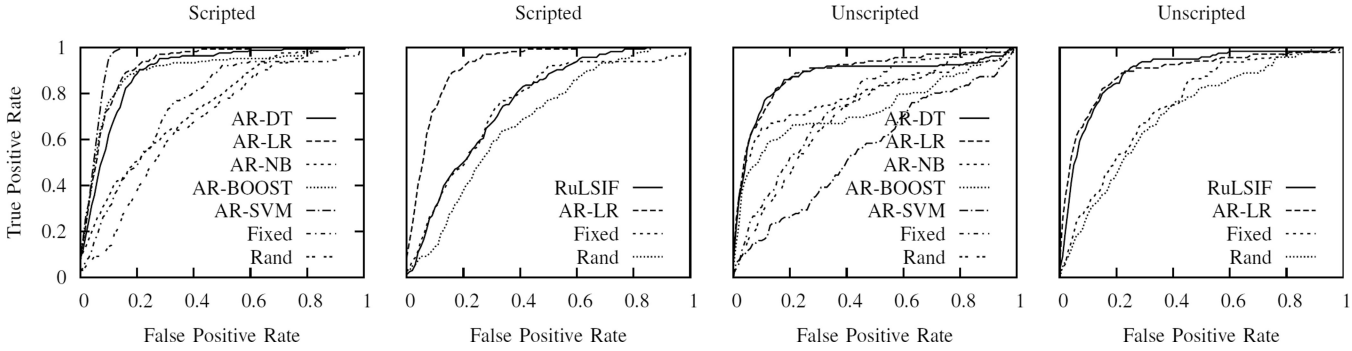

Fig. 9.

ROC curves for both the supervised and unsupervised learning algorithms on the unscripted and scripted data. Which algorithms has the best performance depends on the desired trade-off between the true positive and false positive rates. In both the scripted and unscripted settings a true positive rate of greater than 80% can be achieved while maintaining a false positive rate of less than 15%.

TABLE I

The feature vector describing a data point

Feature #	Value
1–30	# Times each sensor generated an event in the sequence (30 unique sensors)
31	Elapsed time (in seconds) since previous sensor event occurred
32	Time duration of window (in seconds)
33–34	The identifier of the sensor generating the most events for the previous window (and for the window prior to the previous window)
35–64	Elapsed time (in seconds) since each sensor previously generated an event

TABLE II

Example of summing probabilities with $m = 3$. Here the first column identifies a particular data instance being classified, the middle column represents the probability that the instance is a transition (generated by the classifier), and the last column represents the summed probability over $m = 3$ instances.

i	$P(y_i = \text{Transition}\|x_i)$	Summed Probability
0	0.10	
1	0.13	
2	0.78	1.01
3	0.89	1.80
4	0.95	2.62
5	0.56	2.40
6	0.31	1.82

TABLE III

Order of activities performed in the smart apartment for each condition

Order	Condition A	Condition B
1	Read Magazine	Dust Apartment
2	Watch TV	Copy Recipe
3	Build Puzzle	Get Items From Closet
4	Do Math Problems	Read Magazine
5	Copy Recipe	Get Ingredients for Recipe
6	Make Oatmeal	Watch TV
7	Play Handheld Game	Get Items for Day Out
8	Get Items From Closet	Build Puzzle
9	Get Ingredients for Recipe	Sweep Kitchen
10	Sweep Kitchen	Do Math Problems
11	Get Items for Day Out	Make Oatmeal
12	Dust Apartment	Play Handheld Game