Enhancing Activity Recognition using CPD-based Activity Segmentation

Samaneh Aminikhanghahi and Diane J. Cook

s.aminikhanghahi@wsu.edu, djcook@wsu.edu

School of Electrical Engineering and Computer Science

Washington State University, Pullman, WA

Abstract

Segmenting behavior-based sensor data and recognizing the activity that the data represents are vital steps in all applications of human activity learning such as health monitoring, security, and intervention. In this paper, we enhance activity recognition by identifying activity borders. To accomplish this goal, we introduce a change point detection-based activity segmentation model which segments behavior-driven sensor data in real time, which in turn increases the performance of activity recognition. We evaluate our proposed method on data collected from 29 smart homes. Results of this analysis indicate that the method not only provides useful information about their activity boundaries and transitions between activities but also increase the average accuracy of recognizing individuals' activities while they are performing their daily routines more than 1%.

Keywords: Activity segmentation, Activity recognition, Change point detection, Smart home

1. INTRODUCTION

Learning human activity models from streaming sensor data is an important problem for the successful realization of intelligent environments. Human activity learning aims to learn and understand observed activities and situations within an environment and is useful for a wide range of applications and services such as detection of health monitoring and emergencies, early disease detection, home automation, security, and behavior intervention. Activity learning encompasses valuable capabilities such as activity recognition, activity detection, activity segmentation, and activity forecasting. In this work, we focus on the activity segmentation problem, or the problem of segmenting behavior-based sensor data into subsequences, each corresponding to a single activity.

Various strategies have been proposed for activity segmentation. While earlier works use pre-segmented event streams [1][2], others approximate activity segmentation by processing data captured in sliding windows that move linearly through sensor data offline or in real time [3]–[5]. Recognizing activities from pre-segmented data offers an advantage that the beginning and ending point of each activity occurrence is known. However, such segmentation is typically performed offline based on inspection of an entire dataset and is therefore not useful for real-time applications. While a sliding window approach can be used for online segmentation, finding the proper length of the window is difficult because the duration or the sensor sequence length of each activity occurrence may differ significantly. Recently deep RNN based neural network algorithms are introduced to solve this problem. A short length for the sliding window compared to the activity length does not nullify activity recognition results since a memory vector can be kept from one sliding window instance to the next, and allow the relevant information to be carried through time. However, these methods are supervised and require a massive amount of training data in advance which is not very useful for online activity recognition.

Identifying the start and end points of each activity is very useful for modeling human behavior, detecting activity routines, and providing activity-aware automation. Identifying activity boundaries facilitates extraction of features that reflect characteristics of the activity as a whole such as activity start times, end times, and duration. This information can improve the accuracy of activity recognition [6]. The information is also valuable for the assessment of an individual's functional health and changes in their health status [7].

Another area of research development that will benefit from activity transition detection is context-aware intervention delivery. While mobile devices have become a popular tool for notification of information news and delivery of real-time interventions, pushing notifications can

also be detrimental if they interrupt ongoing activities [8][9]. Prompting during transition periods, a period when the user is not engaged in an activity and may be transitioning between activities has been suggested as an effective prompting time [10]. In the case of adults with memory limitations, mobile delivery of activity prompts been found to support functional independence because the device acts as a type of cognitive prosthesis. While activity segmentation information finds many uses, due to the nature of human behavior and its unpredictability dividing daily routines into individual activities and providing labels for the activities is challenged by real-world complexities including activity ambiguity, interruptions, parallel activities, and sensor noise.

In this paper, we address the problem of activity segmentation. To illustrate the method and validate the approach, we apply our proposed technique in the context of smart homes that collect sensor data while individuals perform continuous, unscripted, everyday routine activities. We propose a novel transition-aware activity segmentation approach which identifies the transition between activities by detecting change points in the corresponding time series-based data. This change point detection technique is effective for behavior-driven sensor data analysis, and we show that using such change-point detection techniques can identify wherein the sensor data one activity ends and another begins. Furthermore, we hypothesize that activity segmentation can improve the performance of activity recognition on non-scripted activities by distinguishing activity borders and integrating features related to the entire activity segment. We evaluate the performance of our proposed approach using smart home data from 29 homes, collected continuously for up to 18 months per home.

The remainder of the paper is organized as follows. Section 2 explains the related work in this area. Section 3 describes the basic definitions we used throughout the paper and formulates the problem we are addressing. Our proposed approach, including segmentation and alternative methods for incorporating

activity transition detection into activity recognition, is explained in Section 4. The implementation of the work and evaluation of the approach to smart home data is presented in Section 5. Finally, Section 6 summarizes the results and discusses future work.

2. RELATED WORK

Human activity recognition has become a very popular area of research. As a result, numerous studies investigate both online and offline methods for activity recognition. Wang et al. [11], Lara et al. [12] and Bulling et al. [13] surveyed algorithms for supervised and unsupervised activity recognition from wearable sensors. In this section, we focus on algorithms that detect transitions between activities in addition to labeling the activities.

Early studies on human activity segmentation focused primarily on video segmentation or segmenting a combination of video and accelerometer data [14]–[17]. Advances in smart homes, wearable devices, and pervasive computing make the problem of activity segmentation from sensor data one of the main points of interest in the era of IoT (Internet of Things). IoT refers to the networked interconnection of everyday objects, which are often equipped with ubiquitous intelligence [18]. Here we review work that has been conducted in this emerging area of research.

To segment data from wearable sensors, Yoshizawa et al. [19] analyzed accelerometer data in the frequency domain by setting a threshold that indicates points where activities change from static (such as standing and sitting) to movement (such as walking and running). Similarly, Nyan et al. [20] proposed a wavelet decomposition model to segment accelerometer data into walking, ascending stairs, and descending stairs. He et al. [21] asked the user to give explicit ground truth segmentation points by standing for a few seconds between two activities to identify the corresponding beginning and ending points. Cho et al. [22] applied long short-term memory (LSTM), a type of recurrent neural

network, to determine whether or not an activity is terminated on the dataset collected in a scripted environment with ambient sensors. Zhu et al. [18] developed a two-step supervised classifier to recognize ambulation activities along with transitions from accelerometer data. In the first step, an artificial neural network categorizes stationary and non-stationary activities. In the second step, a hidden Markov model labels the corresponding activity. One important issue related to this work is that all the data were collected from one single individual. All of these methods either detect transitions using a supervised approach and/or operate only in scripted environments. In such settings, participants perform the activities following instructions. The beginning and ending of each activity are thus specified in advance. The resulting activity transition detection and recognition process are thus much different from analyzing activities from sensor data in real-world settings with streaming sensor data.

To address these challenges, more recent studies focus on unsupervised methods for detecting activity boundaries in real-world datasets. Wang et al. [23] combined SVM and association rule mining to design a three-level activity segmentation algorithm and applied it to CASAS datasets. The first SVM model is used for activity recognition based on raw sensor events. The second SVM model tries to find the boundary information, and finally, in the third layer, the association rule miner verifies the boundary recognition results to reduce recognition errors.

In other recent efforts, Ortiz et al. [24] designed a Transition Aware Human Activity Recognition (TAHAR) for the recognition of physical activities using smartphones. This algorithm captured body motion with inertial sensors and utilized a Probabilistic-SVM algorithm for activity recognition and transition detection. In this approach, transitions are considered as an additional class. Another approach was introduced by Noor et al. [25], who used three different decision tree classifiers to play the roles of a transitional activity detector, a non-transitional activity classifier and a transitional

activity classifier to segment physical activity based on a single tri-axial accelerometer. A transitional activity detector differentiates transitional activities from static/dynamic activities in sensor data. Depending on the result, either the transitional activity classifier or the non-transitional activity classifier is used to label the activities.

More similar to our proposed method, Ni et al. [26], Alam et al. [27] and our preliminary work [28] used an unsupervised method to detect transitions and then applied a supervised classifier to label the activities. Ni et al. [26] introduced the Multivariate Online Change detection Algorithm (MOCA) to detect transitions by comparing statistical values such as mean value as well as the variance-covariance matrices of two consecutive windows. In the next step, a supervised classifier is used to recognize static, dynamic, and transitional activities collected from accelerometers and performed by ten older adults. The experimental results show a significant improvement in labeling ambulatory activities over the existing window based method. However, the proposed method may not be as accurate in detecting activities of daily living in smart homes. The main reason is it only uses mean and variance value to detect changes or transitions. Alam et al. [27] applied the Relative unconstrained Least-Squares Importance Fitting (RuLSIF) algorithm to accelerometer data collected from a wearable smart earring to detect gesture changes and utilized a Hidden Markov Model (HMM) to label each gesture. On the other hand, our preliminary work [28] applied the same unsupervised algorithm to smart home data to detect transitions between daily activities.

To summarize the existing work, previous approaches can be largely categorized based on the method they used for transition detection. These prior approaches utilized either a supervised classifier [23][24][25] or a RuLSIF-based unsupervised learner to perform activity transition detection [27][28]. Supervised methods require a sufficient amount of training data which increases significantly when the number of activity classes grows. This constraint is not consistent with the

real-time nature of the applications needing activity segmentation and makes using unsupervised learning approach more suitable. In this current work, we introduce a SEP change detection algorithm for transition detection. We confirmed that SEP outperforms RuLSIF in detecting transitions and can detect transitions closer to their actual occurrence. The result of improving transition detection is an increase in the accuracy of our activity segmentation as well as subsequent activity recognition from smart home data.

3. BACKGROUND

3.1 Definitions

To begin, we provide definitions that we will use throughout the rest of the paper. An activity learner receives data from sensors that are used to perceive the state of an individual or environment. We specify input data to an activity learner as a sequence of *sensor events*. Each sensor event, e, takes the form $e=< t, s, m >$ where t denotes the timestamp when the sensor message was collected, s denotes the identifier of the sensor generating the message, and m denotes the content of the sensor message. We refer to an *activity* as a complex behavior consisting of a sequence of actions that can be performed by a single individual or several individuals interacting with each other. We posit that many such activities can be detected and recognized from sensor events that are collected while the activity is performed. In order to perform activity recognition from continuously collected data, subsequences of sensor events need to be considered, processed and mapped to an activity label. In this context, we define an *activity window* as a sequence of n sensor events, $< e_1\ e_2 \ldots e_n >$, which is processed for activity analysis. Continuous-time activity recognition learns a mapping of the features to an activity label where the label represents that activity corresponding to the last event in the window [29]. In this approach, a feature vector is extracted from the activity window and input to an activity

classifier. The feature vector can be mapped to a label that indicates the activity that is occurring during the entire window or at the beginning, middle, or end of the window [29].

Our approach to activity segmentation is based on detecting change points in time series data. A *time series* data stream is an infinite sequence of elements $TS=\{x_1,\ldots,x_i,\ldots\}$, where each x_i is a d-dimensional data vector arriving at time stamp i [30]. Given a time series TS, we assume time stamp t is a *change point* if the probability density function f created from sliding-window data observed before t is sufficiently different from the data observed immediately after t, either based on the type of the function or the parameters characterizing the function change. In the context of activity-based change point detection, a detected change point is also referred to as an *activity transition* (time point at which an individual transition from one activity to another). A ***c**hange point view* is defined as a set of features describing a fixed-length sequence of sensor events before and after the candidate change point t. A change point detection algorithm compares two change point views to determine if a change point occurs between them. Finally, an *activity segment* is defined as the sequence of sensor events that occurs between two detected activity transitions in activity-based sensor data.

3.2 Problem Setup

Next, we formalize the problem of activity segmentation from sensor data. Activity segmentation identifies activity boundaries and segments observed data into individual activities. Given a sequence of n sensor events, $E=< e_1 \ldots e_n>$, where e_j corresponds to the j^{th} sensor event, an activity segmentation algorithm partitions the sequence into a set of k nonoverlapping subsequences $P=<E_1, \ldots, E_k>$, such that each $E_i \subseteq E$ and the order of the elements in E_i is preserved. Furthermore, the set of subsequences is nonempty, nonoverlapping, and $\bigcup_{i=1}^{i=k} S\_E_i = E$. In the case of activity-based sensor data, the beginning and ending of each subsequence denotes the beginning and ending of the

corresponding activity. Additionally, the time between the end of one subsequence and the beginning of the next can be considered an activity transition.

With the availability of an activity segmentation algorithm we can then distinguish window-based real-time activity recognition from segmentation-based real-time activity recognition. Both approaches perform activity recognition in near real-time, as the activities occur. Let $A = \{a_1, a_2, \ldots, a_T\}$ be a set of T activities, where a_i corresponds to the i^{th} activity class. Given the segmented sequence of n observed sensor events, $<e_1\ e_2\ ..\ e_n>$, a feature vector X is extracted that provides relevant information about the corresponding activity segment. In the case of window-based recognition, the segment is defined by a fixed rule specifying the window size. In the case of activity segmentation, the segment is defined by detected transitions in the data. An activity recognition algorithm then learns a function, h, that maps the feature vector onto an activity label, $h:X \rightarrow A$.

4. PROPOSED APPROACH

Although the goals of both window-based activity recognition and segmentation-based activity recognition are to recognize human activities from sensor data, the activity segmentation will provide additional information beyond activity recognition, because it will also identify the start and end points of the activity rather than just assign a label to a manually-segmented or arbitrarily-positioned window.

Figure 1 provides an overview of our proposed activity segmentation and recognition process. Our proposed approach contains three steps, as the figure shows. The first step is to collect ambient sensor data in smart home environments. The second step identifies activity boundaries, or performs transition detection (ATD), yielding activity segments. Finally, the third step is to recognize and label the corresponding activity (AR). As shown in the chart, we consider four alternative approaches

to activity recognition based on the collected and partitioned data. The first approach, AR-W, represents a traditional sliding window-based activity recognition that does not incorporate any information from transition detection. The second approach, AR-WT, also recognizes activity labels based on a sliding window but the additional feature from the most recent detected transition is added to the feature vector. AR-SM is the third approach which is similar to AR-W except for the label of activities between two transitions is now determined to be the majority activity label for the window. Finally, AR-SS represents our proposed method, which labels all events between two transitions as a single activity. The main contribution of this work is introducing a method to improve both activity segmentation and activity recognition using an innovative change point detection algorithm.

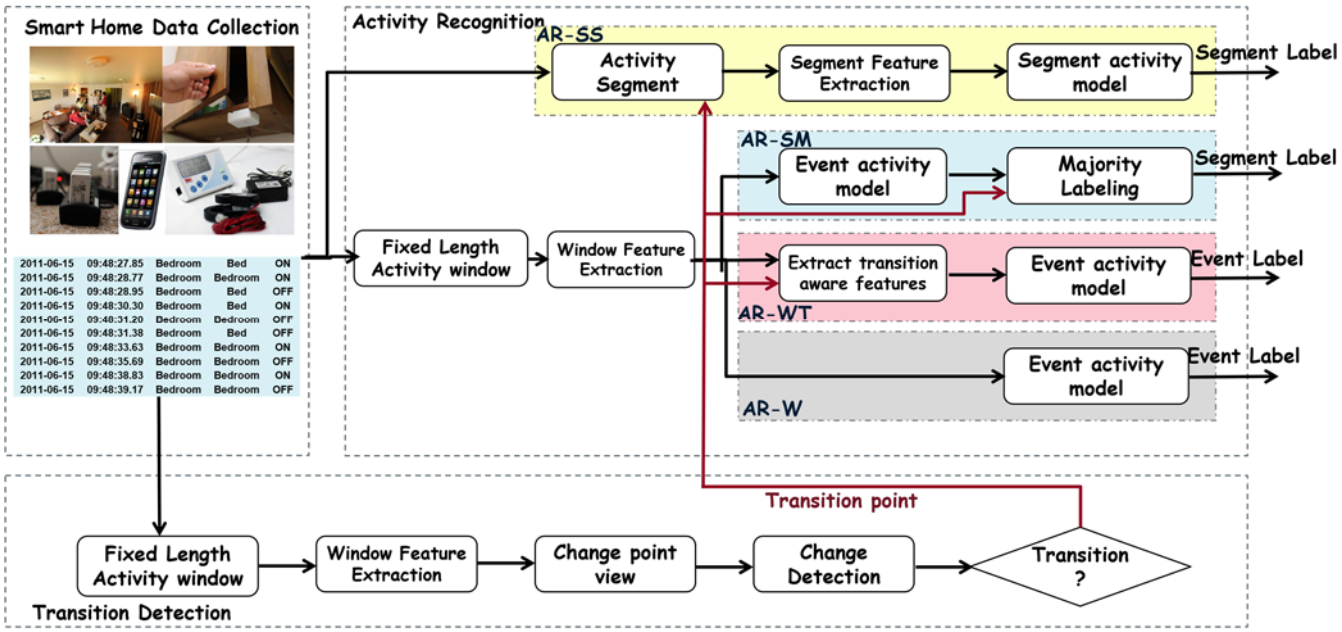

Figure 1. Overview of basic activity segmentation.

4.1 Segmentation

Real-time activity segmentation or detection of transitions between activities *as they occur* based on sensor data is a valuable but somewhat untapped challenge. We consider the problem of detecting activity transitions as a change point detection problem in time series data. This assumption is based on the hypothesis that activity transitions can be detected without knowing the activity categories a priori, by only assessing changes in the data distributions of the feature space.

Change point detection is the problem of identifying time points when the probability distribution of a time series changes. We use an approach introduced in our previous work for activity transition detection called SEP [31]. SEP is a non-parametric sequential change detection which calculates a change score between two consecutive change point views to indicate the amount of change that occurs from the first view (a sequence of sensor events) to the next. As demonstrated in Figure 1, we used a fixed length sliding window to extract features for change point detection. To distinguish this sliding window from one used for activity recognition; we will refer to this window as a change point view. Extracted features can be categorized into three groups: 1) time features which represent the time of the last event in the view such as day of week and time of day, 2) view features which represent information about the view as a whole such as view duration, most frequent sensor, complexity, and activity change, and 3) sensor features which represent the number of occurrences of each sensor event in the view and the last time each sensor fired within the view. Table 1 provides a complete explanation of these features as well as features used in other algorithms. As a result, the dimension of the time series data feature space depends on the home layout and number of sensors. After completing feature extraction, we apply SEP change detection to determine whether there is an activity transition between the two consecutive change point views.

SEP utilizes separation distance as a dissimilarity measure and estimates the ratio of probability distributions. Considering two estimated probability densities, $f_t(x)$ and $f_{t-1}(x)$, corresponding to two consecutive change point views, each with length n, the separation distance S between them can be calculated as shown in Equation 1.

$$S = Max(1 - \frac{f_{t-1}(x)}{f_t(x)}) \quad (1)$$

The Max operator returns the maximum over all estimation of probability ratios. In order to calculate Separation distance as change point score, similar to our previous work [31], we estimate the ratio of the two probability densities by a kernel function $g(x_i)$ and calculate the change point score as Equation 2. Here, the result is compared with 0 to avoid negative values.

$$\widehat{SEP} = \text{Max}(0, \left(1 - \frac{1}{n}\sum_{i=1}^{n} g(x_i)\right)) \quad (2)$$

The score calculated in Equation 2 can be used to detect change points. Considering the fact that a larger SEP score means that the probability of a change point is higher, we reject all candidate points whose SEP values are lower than a threshold value. To reduce the chance of false alarms and avoid double change points (two change points in quick succession that are part of the same transition), we only consider local peak score values as change points. The threshold value (Th) is chosen based on optimal performance for a particular time series. In our experiments, we identify a threshold value that optimizes a tradeoff between TPR and FPR based on a sample of the data. Another important parameter in the SEP algorithm is n, the length of the view. As with the threshold value, we vary the view size for each dataset in order to find the best window length regarding both acceptable accuracy and real-time detection. Through empirical evaluation [31], we identified a view length of two and a threshold value of 0.3 to be effective and efficient for ambient sensor data analysis.

Table 1. Feature sets used for activity segmentation and activity recognition algorithms throughout this paper.

Domain	Feature name	ATD/ AR-W /AR-SM	AR-WT	AR-SS
Time features	day of week	■	■	■
	hour of day	■	■	■
	seconds past midnight	■	■	■
	time of transition		■	■
Window features	number of sensor events		■ from last transition	■ for activity segment between two transitions
	window duration (time)	■ fixed length window	■ fixed length window and from last transition	■ for activity segment between two transitions
	most recent sensor	■	■	■
	last sensor location	■	■	■
	most frequent sensors	■ fixed length window	■ fixed length window and from last transition	■ for activity segment between two transitions
	previous most frequent sensor	■ fixed length window	■ fixed length window	■ for activity segment between previous two transitions
	dominant location		■ from last transition	■ for activity segment between two transitions
	previous dominant location			■ for activity segment between previous two transitions
	entropy-based data complexity	■ fixed length window	■ fixed length window and from last transition	■ for activity segment between two transitions
	activity level change	■ between two halves of fixed length window	■ between two halves of fixed length window	■ for activity segment between two transitions
	number of transitions	■ fixed length window	■ fixed length window	■ for activity segment between two transitions
Sensor features	count of events	■ for each sensor in fixed length window	■ weighted sensor count in fixed length window based on the position of transition; counts of events from last transition	■ for each sensor in activity segment between two transitions
	elapsed time for each sensor since last event	■	■	■

Recognizing activity labels in real time is very important in evaluating activity recognition algorithms. Here we are using the term *ε-real time* to compare different approaches. An activity recognition algorithm can be said to perform in *ε-real time* when the algorithm makes a decision about a label of sensor events after a delay of ε sensor event.

Here, we note that ε, the delay required for change point detection, is based on the number of sensor events rather than time. This is due to the fact that our example application utilizes ambient sensors, which generate sensor events as a change in state is detected, rather than at a steady sample rate. For situations in which data arrives at a steady rate, this delay could be represented by t units of time rather than a number of reported sensor events. To give an indication of the time delay that is represented by a single sensor event in these testbeds, the average time delay between sensor events is listed for each smart home in Table 2. Based on this view, a complete real-time activity recognition algorithm can be viewed as 0-real-time because for every sensor event; it can predict the label immediately.

Table 2. Average time gap between two sensor events for each home.

Home	Average gap (Sec)	Home	Average gap (Sec)	Home	Average gap (Sec)	Home	Average gap (Sec)	Home	Average gap (Sec)
hh101	23.3	hh107	12.3	hh113	18.7	hh119	29.1	hh126	28.6
hh102	17.1	hh108	17.6	hh114	19.8	hh120	25.5	hh127	42.6
hh103	44.4	hh109	13.5	hh115	16.1	hh121	7.2	hh128	12.9
hh104	15.6	hh110	22.6	hh116	13.0	hh122	17.5	hh129	16.0
hh105	22.5	hh111	21.4	hh117	24.1	hh123	25.4	hh130	21.6
hh106	22.4	hh112	17.8	hh118	14.9	hh125	23.5	Average	20.9

Our proposed transition detection is not completely real time because it needs two sensor events at times t_{i+1} and t_{i+2} to decide if there is a transition at time t. We, therefore, refer to this algorithm as operating in 2-real time. The output of the algorithm is a tag attached to each sensor event indicating whether the event is a transition or not. This tag can be sent to an activity recognition algorithm to use as a source of additional information. The accuracy of transition detection has a considerable impact on the performance of an activity segmentation model and thus on activity recognition. A larger false positive rate will split the time series into more segments and thus shorter activities.

4.2 Activity Recognition

We consider and compare three different approaches to CPD-driven activity recognition. We also implement a baseline activity recognition method that does not employ CPD but uses a fixed length sliding window to label activities. As demonstrated in Figure 1, three of the activity recognition approaches are event based, meaning each sensor event is labeled individually with a corresponding activity class. The fourth approach is segment based, meaning a single label is assigned to the segment as a whole. For all of these cases, we use a Random Forest (RF) classifier with 100 decision trees. In previous work, we have found this method to be robust in labeling human activities from sensor data and provides a computationally efficient training algorithm [32].

4.2.1 AR-W

Our first activity recognition algorithm, AR-W, adopts a fixed-length sliding window to move over the data and extract features. AR-W maps the extracted feature vector to a label that indicates the activity occurring at the end of the window. These features as demonstrated in Table 1 include all of the time, window, and sensor features that are used by the activity transition detection algorithm. As we can see in Figure 1, AR-W provides a label for each event immediately and thus it is a 0-real-time algorithm. At the same time, it will not use any change point information and cannot recognize activity boundaries.

4.2.2 AR-WT

A primary challenge for AR-W is choosing an appropriate window length. If the window is too small, it may not contain sufficient relevant information to recognize the activity. On the other hand, if it is too large, then it may contain multiple activities. To address this issue, the AR-WT algorithm adds detected transition information to the feature space, resulting in transition-aware window-based activity recognition. In this design, after identifying activity transitions, we extract transition-aware features

for each sensor event and again use the traditional sliding window approach for labeling. In this case, we modify previous features by adding weights to the number of occurrences of each sensor in the window. Sensors that fired (generated a sensor event) after a change point (activity transition) are weighted more heavily in the feature vector than sensors that fired before the change point. We also add some transition features such as the time of the transition, the number of events that occurred since the transition, the most frequent sensor and its location since the transition, complexity after transition, and activity level change after transition. These differences are explained in Table 1.

After extracting a transition-aware feature set, we again employ an event activity model to label the last event. This approach labels each sensor event based on a fixed-length window. The approach is not 0-real-time anymore because detecting a change point incurs a delay. Thus AR-WT operates in 2-real-time which means it has a delay of two sensor events, but by using more information about activity transitions, we expect it can label the events more accurately. Note that although AR-WT utilizes transition-aware features, it still labels each event separately.

4.2.3 AR-SM

As mentioned earlier, both the AR-W and AR-WT methods label each sensor event separately and they do not indicate the activity start and end points. A straightforward approach to providing activity labels along with distinct activity boundaries is to consider the sensor events between two change points (transitions) as a complete activity. The AR-SM method performs this segmentation step and labels each event in the segment with the majority class for the segment. In this approach, the event-based activity model is similar to AR-W. Whenever we detect a new change point (activity transition), we assign the majority label to all of the events that occurred since the previous activity transition. Using this method, we will have activity labels with distinct boundaries, but the

segmentation process is not 0-real time since it needs to detect a transition (a 2-real-time process) to label the segment. Thus, AR-SM is 2-real time.

4.2.4 AR-SS

Finally, we introduce a transition-aware activity segmentation (AR-SS) method of activity recognition. Instead of labeling each sensor event individually, AR-SS considers and labels an activity segment as a whole. After detecting a new transition, we consider the sensor events between two consecutive transitions as an activity segment. This allows AR-SS to analyze variable-sized segments in which the beginning and end points are determined by the activity transitions. We then extract segment features as explained in Table 1. Here the features are similar to previous window based methods except instead of extracting them from a fixed-length window, we extract them from the activity segment. We expect that extracting features from an activity segment will provide the classifier with more information such as activity duration, dominant sensor, and dominant location which helps to improve the activity labeling performance. Additionally, this approach generates less noisy activity label sequences compared to the window-based approach.

As with the other methods, the segmentation-based activity model utilizes an RF classifier to provide activity labels for each segment once the activity transition representing the end of the segment in a stream. Similar to the previous transition-aware approaches, this approach is also not 0-real time. The delay of this algorithm is $\varepsilon = \Omega+2$, which Ω is the length of the activity. Although this approach incurs a greater delay than the others, because it utilizes all transition and activity segment information, we expect it to be more accurate.

5. EXPERIMENTAL RESULTS AND EVALUATION

We evaluate our algorithm to segment and recognize daily activities using data collected in actual smart home installations while residents perform their regular daily routines.

5.1 Data Sets

The data used during this research was collected by the CASAS (Center for Advanced Studies in Adaptive Systems) smart home system [33] developed at Washington State University. The CASAS smart homes use embedded sensors to collect information about the state of the home and the resident(s) to monitor and analyze daily activities. Sensors generate continuous "events" to report their state. An event contains a date, time, sensor identifier, and message sent from the sensor.

Each of the CASAS smart homes has at least one bedroom, a kitchen, a dining area, a living area, and at least one bathroom. All CASAS smart homes have different sizes and layouts, yet they all include the standard sensor setup. Each of the smart apartments is equipped with a network of wireless motion and door sensors and houses a single older adult resident who performs regular daily routines. Figure 2 shows the floorplan and sensor layout for two sample apartments. Sensor labels starting with "M" indicate motion sensors and labels starting with "D" indicate door sensors.

The primary sensor found in CASAS smart homes is an overhead passive infrared (PIR) motion sensor. Motion sensors are used to determine when motion is occurring in the area covered by the sensor. The motion sensor reports an ON message when motion is detected, followed by an OFF message when the movement stops. In cases when the resident is walking under the motion sensor to some other location, the motion sensor has a gap between the ON and OFF messages that is roughly 1.25 seconds. However, if the activity results in continuous movement under the motion sensor, (e.g., dancing near the motion sensor), the sensor will not generate an OFF message until 1.25 seconds after the activity has stopped. Another sensor used in the CASAS smart home system

is the magnetic door sensor. Door sensors use a magnetic switch to determine whether the doors are opened or closed. These are usually mounted on the external doors of the smart home to indicate when the resident enters or leaves home, though some door sensors are also placed in strategic locations such as doors to cabinets that hold medicine dispensers.

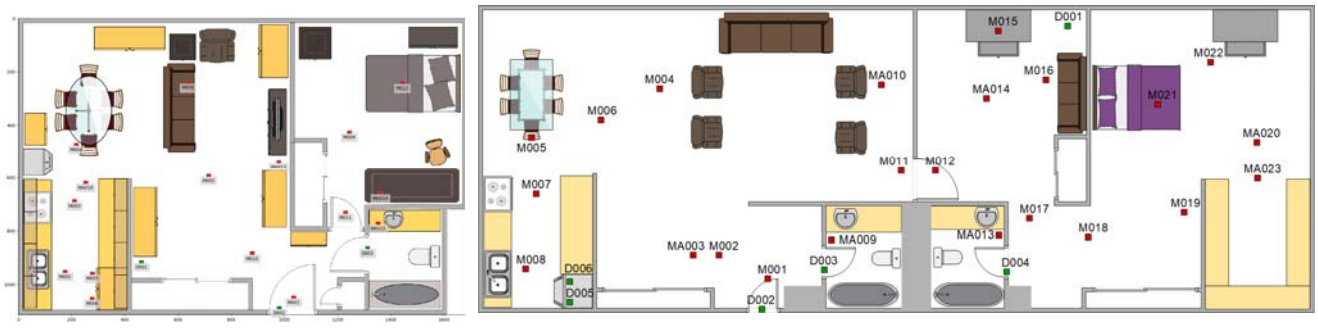

Figure 2. Sample smart home floorplans and sensor positions.

We implement our activity recognition and segmentation algorithms using data collected in smart home testbeds that were installed in 29 apartments [33]. Each of the apartments house a single older adult (age 75+) resident who performs a regular daily routine while sensors in the apartment generate and store events. To provide ground truth activity labels, annotators are given the house floor plan, the positions of the sensors, a resident-completed form indicating when and where they typically perform daily activities, and the sequence of sensor events. Multiple annotators provide consistent labels with the beginning and ending of activity occurrences, and the inter-annotator agreement is κ=0.80. The activity classes that we use for our analyses are Bathe, Enter Home, Wash Dishes, Personal Hygiene, Relax, Work, Sleep, Leave Home, Cook, Bed Toilet Transition, Eat, and Other Activity. Because all events that do not fit into the 11 predefined activity classes are labeled as "Other Activity", the activities are skewed toward this activity class. Table 3 lists characteristics of these data sets. This table lists a data set identifier, the number of sensors and number of sensor events in each apartment, the total number of days data was collected in each apartment, the

number of times each activity occurred in each apartment, and the average number of sensor events per activity occurrence.

Table 3. Characteristics of the data sets used for this study.

Apt ID	# Sensors	# Sensor Events	# Days	Activity Occurrences / Average number of sensor event per occurence										
				Bathe	Enter Home	Wash Dishes	Relax	Work	Sleep	Leave Home	Cook	Eat	Personal Hygiene	Bed Toilet Transition
hh101	18	222,876	60	57/202	173/10	102/73	568/99	4/36	76/95	174/13	196/106	205/33	588/36	20/32
hh102	37	281,370	58	36/121	118/13	262/59	404/35	193/34	119/28	118/16	487/65	352/32	995/55	49/23
hh103	25	112,477	58	23/105	165/5	143/36	191/13	33/16	173/31	166/4	220/135	186/9	591/33	115/27
hh104	38	344,706	61	24/83	207/5	141/93	170/25	408/79	530/62	207/5	261/219	178/142	780/32	353/24
hh105	36	147,043	40	17/166	219/6	124/89	242/31	123/30	95/12	221/8	223/135	107/28	401/40	52/23
hh106	48	202,479	55	48/11	248/6	146/72	240/70	337/67	131/16	244/4	247/111	183/69	798/8	51/8
hh107	26	203,948	31	104/14	214/4	928/20	1949/14	508/14	405/35	215/4	1724/20	891/19	1123/9	198/4
hh108	35	285,756	58	48/5	284/6	427/41	712/21	436/20	167/45	274/9	733/46	440/31	1707/5	69/5
hh109	26	399,155	61	17/12	407/6	771/34	312/31	887/35	193/30	420/7	1887/31	576/22	1030/29	92/15
hh110	26	98,660	27	9/16	73/6	45/57	164/67	203/28	115/41	72/7	42/65	84/30	255/37	68/13
hh111	41	218,474	59	42/18	169/6	126/24	295/51	441/89	121/84	169/8	205/33	284/24	1320/10	90/7
hh112	24	468,488	100	53/6	285/9	267/34	672/37	1170/59	300/165	295/12	499/37	633/22	2294/4	156/2
hh113	38	2,282,124	494	448/125	1640/10	1659/81	1504/46	2369/95	941/56	1652/10	2199/120	1373/39	6093/86	161/71
hh114	25	135,102	31	5/13	71/5	267/53	166/17	263/17	78/59	71/6	490/56	351/17	342/25	9/32
hh115	32	1,598,037	298	104/11	1141/6	929/87	2716/51	1011/119	1022/119	1130/8	1648/115	1222/36	5252/36	865/9
hh116	26	386,427	59	2/9	147/8	213/134	149/25	110/7	140/23	148/9	162/166	148/24	564/190	25/154
hh117	23	732,207	203	31/14	850/11	568/65	1817/16	791/20	718/15	851/11	639/71	847/33	2134/25	286/16
hh118	35	116,389	20	3/14	67/8	137/42	202/64	136/23	41/23	67/10	305/66	169/57	249/34	21/18
hh119	23	92,740	31	15/6	85/8	46/60	89/11	194/12	115/8	85/11	170/84	140/25	552/36	82/48
hh120	26	219,381	64	14/19	216/9	80/60	343/74	276/60	205/34	217/12	82/104	94/22	586/49	125/30
hh121	39	122,648	11	7/3	104/6	1603/8	600/7	758/7	99/9	94/9	1427/12	322/13	1425/12	37/23
hh122	27	129,818	27	20/101	75/5	111/111	171/47	202/26	122/38	74/7	146/121	120/25	495/25	80/13
hh123	15	106,836	31	9/113	102/10	137/106	210/80	8/9	81/51	102/10	144/132	170/46	205/34	21/23
hh125	33	216,217	59	25/50	235/10	482/56	437/15	255/16	97/20	235/11	593/78	338/14	726/48	29/25
hh126	16	112,780	37	9/35	168/9	151/31	465/9	221/7	89/15	166/15	464/28	377/11	501/22	39/5
hh127	23	50,053	26	20/16	88/6	115/15	425/28	36/20	49/11	89/9	162/20	101/6	293/20	12/4
hh128	27	415,140	61	11/1	253/10	511/40	572/38	604/11	138/32	253/10	823/42	742/13	1901/7	49/7
hh129	18	160,672	30	0/0	110/6	242/58	258/30	22/4	168/18	106/9	463/60	247/25	653/18	110/8
hh130	17	121,598	30	12/11	141/9	4/38	343/35	101/30	89/113	147/9	11/26	69/14	428/37	52/32
Average	28	344,262	75	42/45	278/8	370/58	565/38	417/34	228/44	278/9	574/79	378/30	1182/35	114/24

To illustrate the average of activity duration, Table 4 provides a sample of activities from one of our testbed smart homes with the corresponding range of activity durations.

Table 4. Example lengths of activity occurrences in the hh101 smart home.

Activity	Range	Activity	Range
Sleep	5.13 ± 1.49 hours	Cook	4.95 ± 4.63 min
Relax	1.25 ± 0.98 hours	Eat	8.12 ± 8.70 min
Personal Hygiene	3.12 ± 2.85 min	Wash Dishes	3.83 ± 3.07 min
Bed-toilet transition	4.53 ± 4.10 min	Work	6.02 ± 3.78 min
Enter Home, Leave Home		Quite short	

5.2 Evaluation Metrics

Due to the sequential nature of both transition detection and segmentation, the typical cross-validation process of evaluating the algorithm is not applied. The window-based or CPD-based segments are temporally related and data points in the cross-validation training and testing sets may appear close together in time, biasing the results. Alternatively, the entire dataset could be partitioned into 2/3 training (the first 2/3 of the sequential data) and 1/3 testing. Once again, however, the points are related temporally and separating training too far in time from testing may artificially deflate the results. In such cases, seasonal behavior changes and concept drift in the data may cause behavioral routines and activity patterns to be quite different between training and testing. Thus, to validate the performance of our model, we partition the data into multi-day segments that contain both training and testing data. In these experiments, we group the sensor data into 6-day partitions. We use the four days of data from each partition for training and the following two days for testing. Training and testing are repeated for each following 6-day partition in the smart home dataset. This approach maintains the temporal order of the data while also avoid drift in the data set.

As a point of comparison, Figure 3 plots the overall activity recognition accuracy of the AR-W baseline algorithm for all 29 homes using 3-fold cross-validation and multi-day segments. As expected using cross-validation, the accuracy is much higher than using multi-day segment evaluation. As seen in Table 5, the accuracy is over 90% for all of the homes. However, in order to use the activity recognition algorithm in real-world applications such as intervention or health monitoring, we need to evaluate our model in a sequential manner that does not introduce bias into the evaluation process. Thus, in the rest of the paper, we only report the multi-day segment results.

Table 5. Accuracy of AR-W using 3-fold cross validation for each home.

Home	Accuracy	Home	Accuracy	Home	Accuracy	Home	Accuracy	Home	Accuracy
hh101	98.26%	hh107	95.75%	hh113	99.13%	hh119	97.74%	hh126	96.35%
hh102	97.69%	hh108	96.03%	hh114	97.46%	hh120	98.50%	hh127	97.08%
hh103	98.78%	hh109	96.88%	hh115	98.33%	hh121	93.00%	hh128	97.44%
hh104	97.86%	hh110	98.04%	hh116	99.55%	hh122	97.62%	hh129	97.05%
hh105	97.61%	hh111	97.21%	hh117	98.13%	hh123	98.77%	hh130	97.60%
hh106	97.95%	hh112	96.90%	hh118	97.64%	hh125	97.20%	Average	97.50%

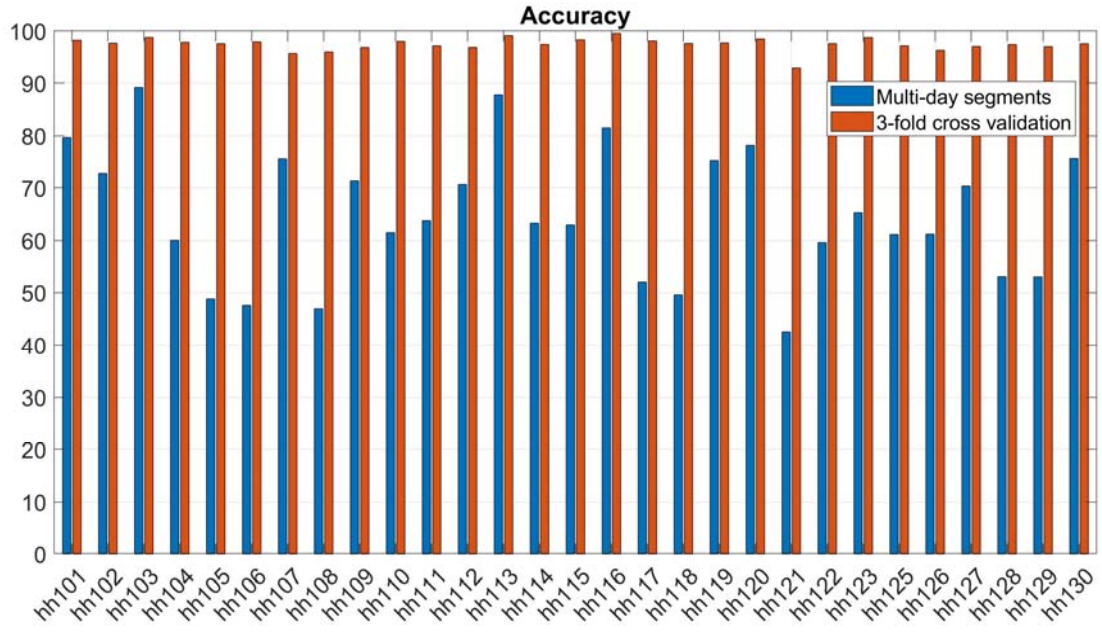

Figure 3. The overall accuracy (%) of AR-W using cross validation and multi-day segment evaluation.

As shown in Figure 1, a critical step in CPD-based activity recognition is the detection of the activity transitions (change points) themselves. To measure the performance of activity transition detection using change point detection, we use the True Positive Rate (TPR) and False Positive Rate (FPR). The TPR measure indicates how effectively a CPD algorithm will detect true activity changes. The FPR measure reflects how many false alarms are generated by a CPD algorithm. To measure the performance of our activity recognition algorithm, we use overall accuracy and per-class accuracy. Overall accuracy calculates the ratio of the number of correctly-labeled sensor events to the total number of sensor events. For each activity, per-class accuracy calculates the ratio of correctly-labeled sensor events within the activity class to the total number of sensor events for that activity.

To evaluate the performance of segmentation we use Normalized Edit Distance (NED) [34]. NED uses the Levenshtein distance [35] to measure how different a predicted activity sequence is from the actual sequence by counting the minimum number of operations that is required to make them equivalent as shown in Equations 3 and 4.

$$NED = \frac{lev(predicted\ sequence, actual\ sequence)}{length\ of\ actual\ activity\ sequence} \tag{3}$$

$$lev(i,j) = \begin{cases} \max(i,j) & \min(i,j) = 0 \\ min\begin{cases} lev(i-1,j)+1 \\ lev(i,j-1)+1 \\ lev(i-1,j-1)+1_{i \neq j} \end{cases} & otherwise \end{cases} \tag{4}$$

Equations 3 and 4 compare two activity sequences and compute the corresponding distance based on the number of operations that needs to be applied to one sequence that will make it equivalent to the other. Here, three different sequence change actions are considered. The first is the deletion of an element from the set, the second is the insertion of a new activity, and the third is a change in

activity label. This metric allows us to evaluate how similar the sequence of detected activities is to the actually performed activities.

In the end, since we are testing the proposed algorithms on 29 different smart homes, we need to compare the results statistically and determine whether there is any significant difference between the performance of the alternative algorithms. To make this comparison, we utilize one way ANOVA which is a common statistical method for comparing two or more classifiers on multiple data sets [36].

5.3 Results and Discussion

We begin by analyzing the performance of our transition detection algorithm. We assume a detected change point is correct if there exists a change point in the data that occurs soon before or after the detected change point. In other words, a detected change point at time t^* is correct if an actual change point occurs in the time interval $[t^* - \lambda, t^* + \lambda]$. In our experiments, we consider $\lambda=10$ seconds for evaluation of change point detection with a small time offset. Figure 4 shows activity transition detection TPR and FPR for SEP as well as for RuLSIF, a popular alternative CPD algorithm. Like SEP, RuLSIF estimates a ratio of probability densities before and after candidate change points, although RuLSIF uses Pearson divergence as a dissimilarity metric. Within-10 seconds change point detection is calculated for the 29 smart homes. The median values for each case are shown in the figure. Since transition detection is the heart of our segmentation method, its accuracy has a high impact on the activity segmentation performance and segmentation-based activity recognition performance. As we can see, the median of true transitions for SEP algorithm is ≥88% while the result is closer to 87% for the RuLSIF algorithm. The false positive rate for SEP is almost 12% which again outperforms the RuLSIF result of 19%. A larger false positive rate will split the data into more segments which may lower activity recognition performance due to a lack of

relevant activity information. A one-way ANOVA test indicates that the differences between algorithms TPR and FPR scores are significant at the ($p < .05$) level.

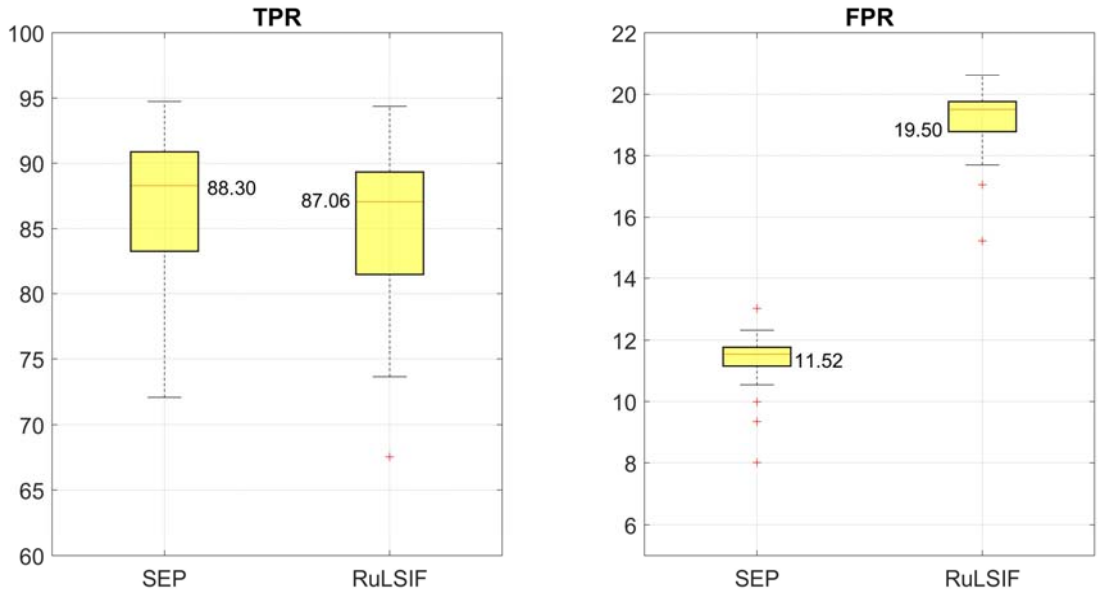

Figure 4. Box and whisker plots of the activity transition detection performance. The horizontal line within the box indicates the median, boundaries of the box indicate the 25th and 75th percentile, and the whiskers indicate the highest and lowest values of the results.

Figure 5 plots the overall activity recognition accuracy and F_Measure of the four alternative models for all 29 homes. The median value of overall accuracy for the AR-W baseline method is near 63.3% while the median of F_Measure is 68.9%. By adding transition features and still labeling each event separately, these values increase to 67.1% and 72.9% for AR-WT. For the two segmentation approaches, AR-SM and AR-SS, we generate one label per segment. In the case of AR-SM, we generate labels for each individual sensor event, but the same label is used throughout the segment. For AR-SM model, the accuracy of activity recognition decreases to 50.6% and the F_Measure decreases to 60.6%. However, if we use the AR-SS model and train the random forest to label the segment as a single data point, we observe the highest accuracy at 68.1% and highest F_Measure at 73.5%. The performance improvement for both accuracy and F_Measure of AR-SS is significant ($p <$

0.05) and is most likely due to the addition of transition information. Furthermore, AR-SS provides distinct activity boundaries. These boundaries allow activity-aware applications to calculate important information including activity start times, end times, and durations.

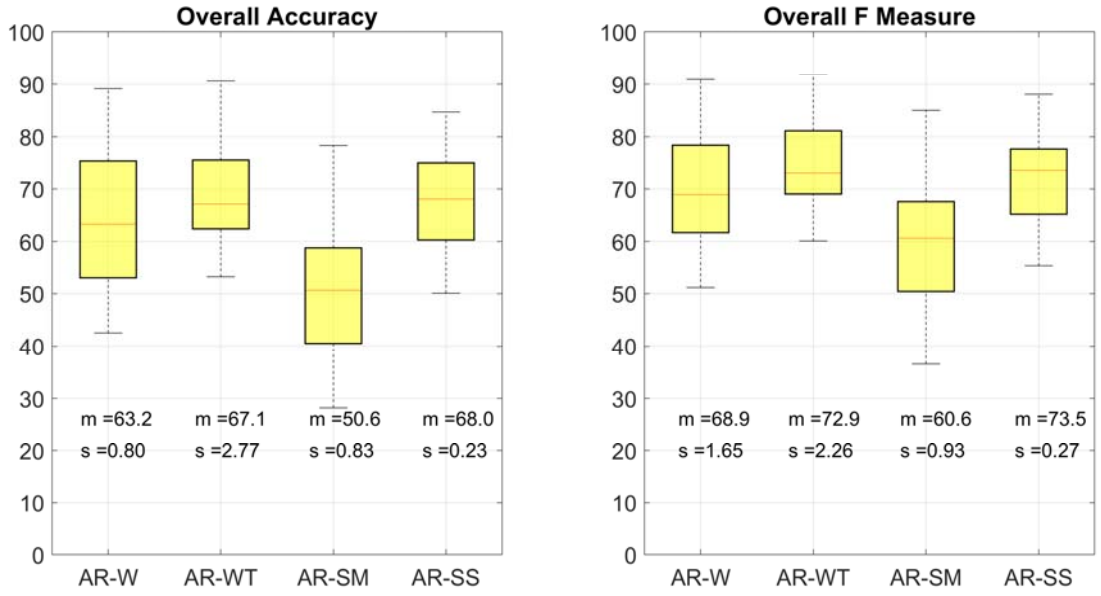

Figure 5. Box and whisker plots of the overall accuracy and F_Measure of alternative activity recognition methods. The horizontal line within the box indicates the median, boundaries of the box indicate the 25th and 75th percentile, and the whiskers indicate the highest and lowest values of the results. m is the median value and s is the standard deviation between multiple folds.

The average per-class accuracy for all homes along with the standard deviation value is graphed in Figure 6. We can see for some activities like “Enter Home”, “Leave Home”, and “Bed Toilet Transition” the basic activity recognition AR-W offers the higher accuracy. The reason is the nature of these activities. These activities are quite short (typically under a minute) and share quite similar movements with other activities. Since the length of these activities is small, the corresponding detected segments and extracted features would not be accurate, thus assigning a segment to them is not effective

for activity recognition. For the remaining activities, AR-SS outperforms the other methods or is similar.

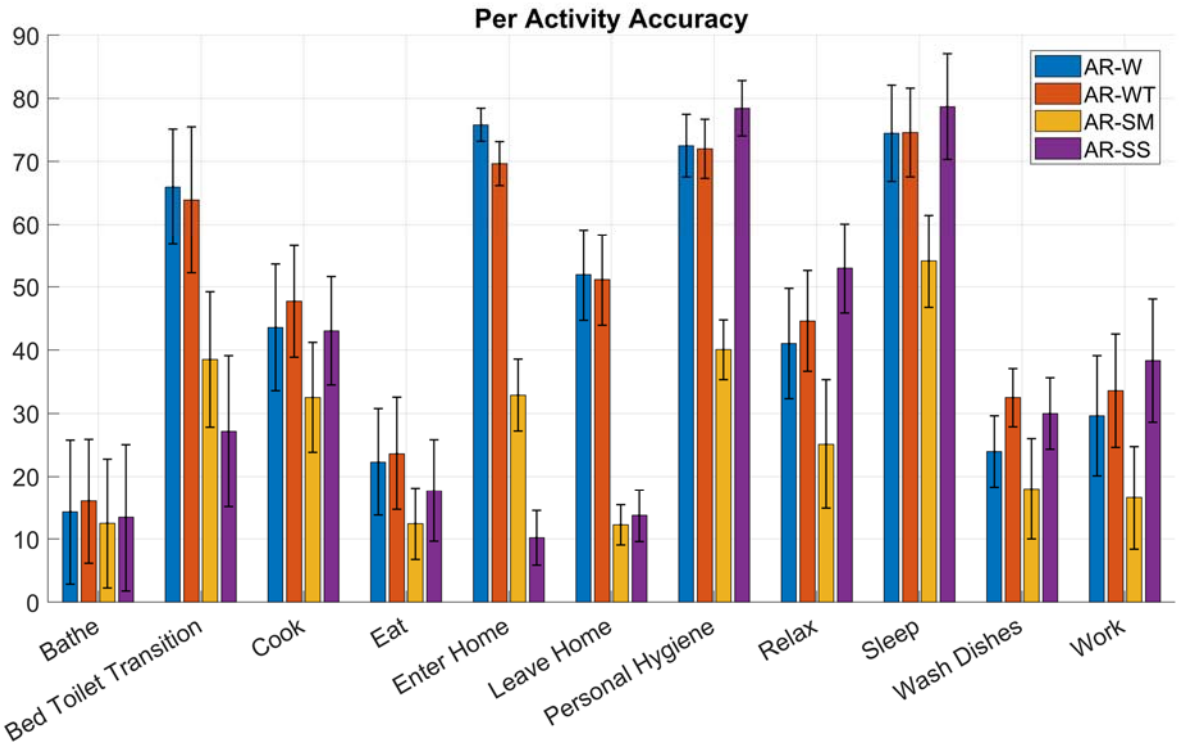

Figure 6. Per activity accuracy (%) for the four alternative methods.

Normalized edit distance values for the alternative methods are plotted in Figure 7. We can see there are considerable differences between the activity segmentation methods and the traditional event-based recognition. As we know, a smaller NED value means the detected activity segment is more similar to the actual activity segment. As the figure shows, the segmentation-based methods, particularly AR-SS, significantly outperform the others in terms of NED. In traditional methods, a real-time activity classifier predicts a label for each sensor event; thus the sequence is very different from the original one. Generating a sequence of labels is valuable for applications such as health

assessment, which makes the proposed activity segmentation method more suitable than traditional activity recognition.

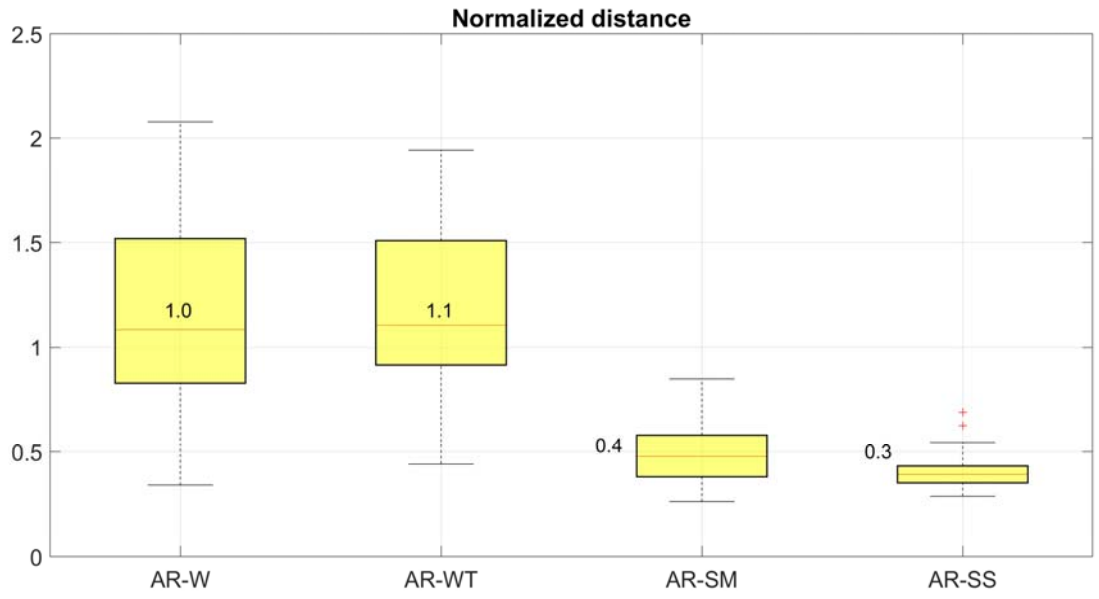

Figure 7. Box and whisker plots of the NED of alternative activity recognition methods. The horizontal line within the box indicates the median, boundaries of the box indicate the 25th and 75th percentile, and the whiskers indicate the highest and lowest values of the results

We next examine the learned models to understand the role that transition-based features play in recognizing smart home activities. Specifically, we rank the Random Forest choice of features based on their Gini Importance value [37] to investigate whether adding transition features change the structure of RF or not. The Gini Importance of each feature is the average of the total decrease in node impurity weighted by the probability of reaching that node over all trees in the ensemble. Table 6 shows the three most important features for the alternative algorithms. Since AR-W and AR-SM are using similar models, the most important features for both are the number of times each sensor fired during the fixed-length window. When we add transition features in AR-WT, the most important features are now the sensor ID at the change point before the current event in addition

to the current sensor ID. This result highlights the importance of activity transition detection because the location of the sensor event at the transition point plays such a critical role. Another feature that was identified as important for this algorithm is the weighted count of events for each sensor in the window. This feature weighs features more heavily after transitions, again stressing the impact of transition detection on activity recognition. For the AR-SS approach, the top features are the dominant sensor ID and the number of times each sensor fired in the segment. While these features are included in the other methods, they were not among the top-ranked features because they are based on a fixed-length window rather than an entire activity. As the results indicate, when we add transition aware features, the features that represent the activity as a whole are more influential in identifying the activity.

Table 6. Most important features for the alternative activity recognition methods.

Method	1st Important Feature	2nd Important Feature	3rd Important Feature
AR-W	The number of times each sensor fired during the fixed-length window		
AR-WT	Sensor ID of the last event	Sensor ID of the last transition	Weighted sensor counts in the fixed window
AR-SM	The number of times each sensor fired during the fixed-length window		
AR-SS	Dominant sensor in segment	The number of times each sensor fired during the segment	

Finally, we discuss the main problems and limitations of the proposed activity segmentation approaches. The first problem which affects each step of activity recognition is noise. The noise in the dataset may have different sources including sensor noise, data collection problems, incorrect ground truth labeling of activities, or even environmental noise. Noise can be very harmful because it may lead to false alarm change detection or error in training the classifier.

Another challenge is concept drift. Both activity segmentation and transition detection are sequential processes. In the proposed approaches we tune the change detection algorithm

parameters and train the classifier models based on the part of data and test the models on the remainder. Human behaviors, lifestyles, and physical environments may change between the periods of time used for training and test, rendering the model as inappropriate. One possible solution is to add drift detection to the model to update algorithms after drift is detected. Alternatively, additional features and methods can be explored to help the models generalize even more effectively over multiple conditions including lifestyle changes and physical environment changes.

Yet another limitation of our proposed CPD-driven activity recognition strategies is how to define change points and activity transitions. In particular, complex activities may consist of different disparate steps. Because the proposed change detection is an unsupervised algorithm and detects changes in the feature space, it may consider different steps of an activity as a transition. Similarly, transitions between two activities which do not exhibit much difference may be more difficult to detect.

6. CONCLUSIONS

The IoT environment requires robust activity learning technology to provide proper services to its residents. Activity segmentation improves the robustness of these technologies by providing information about activity transitions and thus insights on activity start/end times and durations. Many applications can make use of the additional information and improved activity learning to monitor trends in resident behavior and provide activity-aware services. In this paper we proposed a human daily activity segmentation based on change point detection techniques in an online or streaming fashion, using unscripted data from smart homes. We evaluated the performance of alternative segmentation and window-based activity recognition algorithms using pre-defined metrics. The experiments conducted on real-world smart home datasets provide evidence that detecting activity

transitions and utilizing segment features in activity recognition improve recognition performance while also providing activity boundary and transition insights.

From this research, some ideas arise as future work. They include improving the performance of the model to identify very short activities like Enter home and Leave home. Another future direction can be changing the investigation of real-time algorithms from event based to time unit based. Furthermore, we will generalize the model to consider drift during the time and retrain the activity recognition algorithm when it is needed.

7. ACKNOWLEDGMENT

This research has been supported in part by National Science Foundation grant 1543656 and National Institutes of Health grant R01EB009675.

8. REFERENCES

[1] E. M. Tapia, S. S. Intille, and K. Larson, “Activity Recognition in the Home Using Simple and Ubiquitous Sensors,” Springer, Berlin, Heidelberg, 2004, pp. 158–175.

[2] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, “Accurate activity recognition in a home setting,” in *Proceedings of the 10th international conference on Ubiquitous computing - UbiComp ’08*, 2008, p. 1.

[3] P. Palmes, H. K. Pung, T. Gu, W. Xue, and S. Chen, “Object relevance weight pattern mining for activity recognition and segmentation,” *Pervasive Mob. Comput.*, vol. 6, no. 1, pp. 43–57, Feb. 2010.

[4] H. Junker, O. Amft, P. Lukowicz, and G. Tröster, “Gesture spotting with body-worn inertial sensors to detect user activities,” *Pattern Recognit.*, vol. 41, no. 6, pp. 2010–2024, Jun. 2008.

[5] L. Atallah, B. Lo, R. King, and G.-Z. Yang, “Sensor Positioning for Activity Recognition Using Wearable Accelerometers,” *IEEE Trans. Biomed. Circuits Syst.*, vol. 5, no. 4, pp. 320–329, Aug. 2011.

[6] B. Chen, Z. Fan, and F. Cao, “Activity Recognition Based on Streaming Sensor Data for Assisted Living in Smart Homes,” in *2015 International Conference on Intelligent Environments*, 2015, pp. 124–127.

[7] P. Dawadi, D. Cook, and M. Schmitter-Edgecombe, "Automated Cognitive Health Assessment from Smart Home-Based Behavior Data.," *IEEE J. Biomed. Heal. informatics*, Aug. 2015.

[8] T. Okoshi, J. Ramos, H. Nozaki, J. Nakazawa, A. K. Dey, and H. Tokuda, "Reducing users' perceived mental effort due to interruptive notifications in multi-device mobile environments," in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '15*, 2015, pp. 475–486.

[9] B. P. Bailey and J. A. Konstan, "On the need for attention award systems: Measuring effects of interruption on task performance, error rate, and affective state," *J. Comput. Hum. Behav.,* vol. 22, no. 4, pp. 709–732, 2006.

[10] K. Robertson, C. Rosasco, K. Feuz, M. Schmitter-Edgecombe, and D. Cook, "Prompting technologies: A comparison of time-based and context-aware transition-based prompting.," *Technol. Health Care*, vol. 23, no. 6, pp. 745–56, Jan. 2015.

[11] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A Survey," *Pattern Recognit. Lett.*, Feb. 2018.

[12] O. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Commun. Surv. Tutorials*, vol. 15, no. 3, pp. 1192–1209, 2013.

[13] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Comput. Surv.*, vol. 46, no. 3, pp. 107–140, 2014.

[14] A. Ali and J. K. Aggarwal, "Segmentation and recognition of continuous human activity," in *Proceedings IEEE Workshop on Detection and Recognition of Events in Video*, 2001, pp. 28–35.

[15] E. B. Fox, M. C. Hughes, E. B. Sudderth, and M. I. Jordan, "Joint modeling of multiple time series via the beta process with application to motion capture segmentation," *Ann. Appl. Stat.*, vol. 8, no. 3, pp. 1281–1313, Sep. 2014.

[16] P. Bojanowski *et al.*, "Weakly Supervised Action Labeling in Videos under Ordering Constraints," Springer, Cham, 2014, pp. 628–643.

[17] F. Sener and A. Yao, "Unsupervised Learning and Segmentation of Complex Activities from Video," Mar. 2018.

[18] F. Xia, L. T. Yang, L. Wang, and A. Vinel, "Internet of Things," *Int. J. Commun. Syst.*, vol. 25, no. 9, pp. 1101–1102, Sep. 2012.

[19] M. Yoshizawa, W. Takasaki, and R. Ohmura, “Parameter exploration for response time reduction in accelerometer-based activity recognition,” in *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication - UbiComp ’13 Adjunct*, 2013, pp. 653–664.

[20] M. N. Nyan, F. E. H. Tay, K. H. W. Seah, and Y. Y. Sitoh, “Classification of gait patterns in the time–frequency domain,” *J. Biomech.*, vol. 39, no. 14, pp. 2647–2656, Jan. 2006.

[21] Z. He and L. Jin, “Activity recognition from acceleration data based on discrete consine transform and SVM,” in *2009 IEEE International Conference on Systems, Man and Cybernetics*, 2009, pp. 5041–5044.

[22] M. Cho, Y. Kim, and Y. Lee, “Contextual Relationship-based Activity Segmentation on an Event Stream in the IoT Environment with Multi-user Activities,” in *Proceedings of the 3rd Workshop on Middleware for Context-Aware Applications in the IoT - M4IoT 2016*, 2016, pp. 7–12.

[23] Y. Wang, Z. Fan, and A. Bandara, “Identifying activity boundaries for activity recognition in smart environments,” in *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1–6.

[24] J.-L. Reyes-Ortiz, L. Oneto, A. Samà, X. Parra, and D. Anguita, “Transition-Aware Human Activity Recognition Using Smartphones,” *Neurocomputing*, vol. 171, pp. 754–767, Jan. 2016.

[25] M. H. M. Noor, Z. Salcic, and K. I.-K. Wang, “Adaptive sliding window segmentation for physical activity recognition using a single tri-axial accelerometer,” *Pervasive Mob. Comput.*, vol. 38, pp. 41–59, Jul. 2017.

[26] Q. Ni, T. Patterson, I. Cleland, and C. Nugent, “Dynamic detection of window starting positions and its implementation within an activity recognition framework,” *J. Biomed. Inform.*, vol. 62, pp. 171–180, Aug. 2016.

[27] M. A. U. Alam, N. Roy, A. Gangopadhyay, and E. Galik, “A smart segmentation technique towards improved infrequent non-speech gestural activity recognition model,” *Pervasive Mob. Comput.*, 2016.

[28] S. Aminikhanghahi and D. J. Cook, “Using Change Point Detection to Automate Daily Activity Segmentation,” in *13th Workshop on Context and Activity Modeling and Recognition*, 2017.

[29] N. C. Krishnan and D. J. Cook, “Activity recognition on streaming sensor data,” *Pervasive Mob. Comput.*, vol. 10, pp. 138–154, Feb. 2014.

[30] D.-H. Tran, “Automated Change Detection and Reactive Clustering in Multivariate Streaming Data,” Nov. 2013.

[31] S. Aminikhanghahi, T. Wang, D. J. Cook, and I. Fellow, "Real-Time Change Point Detection with application to Smart Home Time Series Data," *Submitt. to IEEE Trans. Knowl. data Eng.*, 2018.

[32] K. D. Feuz and D. J. Cook, "Modeling Skewed Class Distributions by Reshaping the Concept Space - Semantic Scholar," in *AAAI* , 2017.

[33] D. J. Cook, A. S. Crandall, B. L. Thomas, and N. C. Krishnan, "CASAS: A Smart Home in a Box," *Computer (Long. Beach. Calif).*, vol. 46, no. 7, pp. 62–69, Jul. 2013.

[34] Q. Ni, A. B. García Hernando, and I. P. de la Cruz, "The Elderly's Independent Living in Smart Homes: A Characterization of Activities and Sensing Infrastructure Survey to Facilitate Services Development.," *Sensors (Basel).*, vol. 15, no. 5, pp. 11312–62, May 2015.

[35] L. Yujian and L. Bo, "A Normalized Levenshtein Distance Metric," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1091–1095, Jun. 2007.

[36] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *J. Mach. Learn. Res.*, vol. 7, no. Jan, pp. 1–30, 2006.

[37] L. Breiman, *Classification and Regression Trees*. Routledge, 1984.