Using Change Point Detection to Automate Daily Activity Segmentation

Samaneh Aminikhanghahi
School of Electrical engineering and Computer Science
Washington State University
Pullman, WA, USA
s.aminikhanghahi@wsu.edu

Diane J. Cook
School of Electrical engineering and Computer Science
Washington State University
Pullman, WA, USA
cook@eecs.wsu.edu

***Abstract*—Real time detection of transitions between activities based on sensor data is a valuable but somewhat untapped challenge. Detecting these transitions is useful for activity segmentation, for timing notifications or interventions, and for analyzing human behavior. In this work, we design and evaluate real time machine learning-based methods for automatic segmentation and recognition of continuous human daily activity. We detect activity transitions and integrate the change point detection algorithm with smart home activity recognition to segment human daily activities into separate actions and correctly identify each action. Experiments with on real-world smart home datasets suggest that using transition aware activity recognition algorithms lead to best performance for detecting activity boundaries and streaming activity segmentation.**

Keywords— Activity Recognition; Segmentation; Change point detection; Smart home.

I. INTRODUCTION

Systems that are designed to serve humans benefit from real-time knowledge about the daily activities that humans perform. The goal of activity recognition (AR) is to recognize human daily activities from data collected through a variety of sensors [1]. AR is a well-researched problem that routinely finds application in a broad range of real-world situations, such as monitoring patients for health diagnosis and rehabilitation, home automation, and the automated surveillance of public places for crime prevention.

Current AR approaches either assume that sensor data is pre-segmented into non-overlapping activities or divide the data stream into windows of sequential data for labeling. Recognizing activities from pre-segmented data has the advantage that the input data only represents a single activity. Additionally, information about the length and time span of the activity segment can be used for activity classification. However, such segmentation is typically done offline based on inspection of an entire dataset.

To handle non-segmented streaming data, current approaches typically divide the entire sequence of sensor events into sliding windows with static or dynamic size. However, selecting the optimal length of the window is very important. If a very small interval is chosen, there is a possibility that it will not contain sufficient relevant activity information. If the time interval is too large, then it may contain data corresponding to multiple activities and the activity that dominates the interval will have a greater influence on the final label [1]. In both cases, the activity start and end points are difficult to define and they are typically manually annotated during the experiment.

One useful alternative approach to handling non-segmented sensor data is to detect activity transitions or boundaries that can segment the data in real time. Detecting such transitions can also be very helpful for identifying points at which users can receive notifications or interventions without interrupting critical activities and increasing cognitive load [2][3][4]. The problem is that transitions between activities are not clearly defined and there is no distinct beginning/end of an activity. Therefore, to segment and recognize a continuous activity sequence, the detection of transitions between activities is crucial [5]. Detecting activity breakpoints or transitions based on characteristics of observed sensor data from smart homes or mobile devices can be formulated as change point detection problem in a time series area. Change points are abrupt variations in time series data and they are useful in segmenting or smoothing activities, interacting with humans while minimizing interruptions, providing activity-aware services, and detecting changes in behavior that helpful in health assessments

The rest of this paper is organized as follows. Section 2 discusses related work on activity segmentation. Methods and processes used for activity segmentation along with the accompanying modifications are explained in Section 3. Section 4 presents the experimental setup for evaluating activity segmentation algorithms and presents the results and discussions. Section 5 summarizes our conclusions and provides directions for future work.

II. RELATED WORK

The different activity segmentation approaches proposed in the literature differ in terms of the sensing technology that is used, the type of activities that are modeled, and the machine learning methods that are employed. There is a wide variety of sensors such as wearable devices [6][7], cameras [5], Smartphone [8] and smart home [9] sensors that are useful for gathering information for detecting human activities.

Activity segmentation algorithm has two parts: finding activity boundaries or transition detection (ATD) and labeling the activities. There have been a number of machine learning models that have been used for ATD. These can be categorized

978-1-5090-4338-5/17/$31.00 ©2017 IEEE

into supervised or unsupervised techniques. Supervised learning algorithms are machine learning algorithms that learn a mapping from input data to a target attribute, which is usually a class label [1]. When a supervised approach is employed for ATD, it can be trained using binary or multi-class classifiers. In a multi-class approach, we utilize activity labels provided by an expert or an activity recognition algorithm. Boundaries between activity labels (the end of one activity and the beginning of the next) then provides ground truth information to train the activity transition classifier. An alternative is to treat change point detection as a binary class problem, where the positive class represents a transition and the negative class is a non-transition. While only two classes need to be learned in this case, this is a much more complex learning problem if the number of possible types of transitions is large [1]. This type of problem will also suffer from extreme class imbalance as there are typically many more within-state sequences than change point sequences. Unsupervised learning algorithms are typically used to discover patterns in unlabeled data. In the context of ATD, such algorithms can be used to find transitions based on statistical features of the data without requiring prior training for each situation.

Ali et al. [10] investigated a semi-supervised activity segmentation method using a Multiple Eigen space technique based on Principal Components Analysis. Results show that the method can reliably perform activity segmentation, and classification results using hidden Markov models demonstrate the practical value of the proposed technique. Nevertheless, it still needs subject-specific data for training. Ling et al. [11] proposed an automatic activity segmentation model which separates measured accelerometer data into discrete action subseries. To partition among the different activity types, the unsupervised segmentation model of minimized contrast with the sliding window autocorrelation correctness was used. Due to using only accelerometer data, this approach is not suitable for detecting daily activity breakpoints. Alam et al. [6] proposed a change point detection-based gestural activity recognition approach (GeSmart) using accelerometers which are installed on different body positions (or "parts") as smart jewelry to identify chronic psychological conditions. Relative density-ratio estimator (RuLSIF) was used to detect change points and segment the data stream, and a hybrid classifier was used to label speech gestural activities.

The most current activity segmentation approaches have been evaluated using either a scripted or pre-segmented sequence of sensor events related to activities. Another issue is that current methods are typically offline or need a huge amount of data ahead of the detection point. These constraints are not consistent with the real-time nature of the applications needing activity segmentation. No prior work is reported that focuses on segmentation of unscripted activities from sensor data. This contribution of this work is a proposed real-time activity segmentation system the processes sensor data obtained from real-world smart homes while residents perform their daily routines.

III. ACTIVITY SEGMENTATION MODEL

Figure 1 shows a schematic representation of our activity segmentation model. The segmentation steps include: 1) smart home data collection, 2) extracting event features, 3) activity transition detection, 4) extracting segment features, and 5) activity recognition.

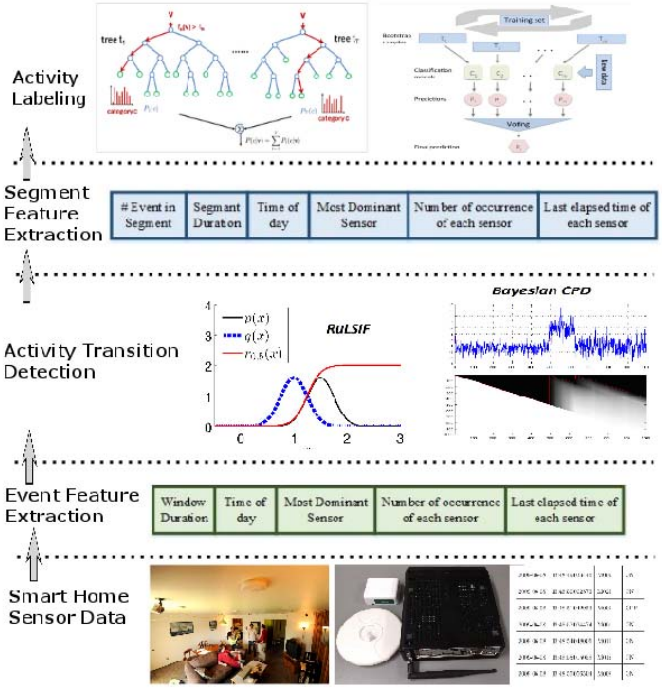

Fig. 1. An overview of the activity segmentation process.

A. Smart Home Sensor Data

The sensor data collected in a CASAS smart home consists of continuous sensor events. 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 a normal daily routine. Figure 2 shows the layout of a smart home we analyze in this paper. Sensor labels starting with "M" indicate motion sensors and "D" indicates door sensors. Infrared motion sensors on the ceilings with removable adhesive strips are placed to track the location of smart home residents. Most of the motion sensors are focused to sense area in a one-meter diameter area immediately below the sensor. However, additional motion sensors are placed in each major room, which have a much broader coverage in order to indicate whether human motion is occurring anywhere in the room. Each sensor event includes date, time, sensor ID, and a sensor message. A sequence of these events can be mapped to different daily activities. For example, the sequence of sensor events in Figure 1 is mapped to a *'Cooking'* label.

B. Feature Extraction

We slide a window over the sensor data that looks at 30 events and extracts a corresponding feature vector. These features include general information such as window time duration, event time, and most dominant sensor in the window which is most frequent triggered sensor, number of occurrence of each sensor in current window and the time when each sensor last fired till the end of current window. Thus the feature vector dimension varies depending on home floorplans and sensor placements. The feature space is then updated when a new event occurs to yield our time series data.

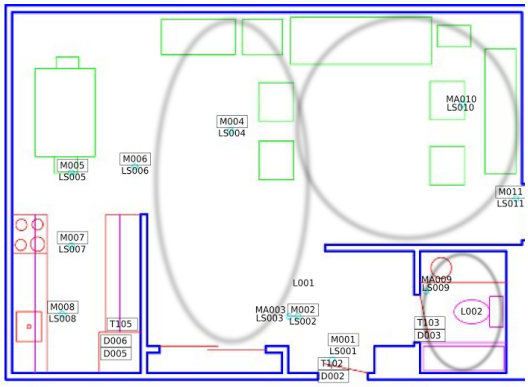

Fig. 2. Smart home floorplan and sensor positions (circles show motion sensors coverage area).

The duration between two consecutive change points is considered as an activity and segment features are calculated based on activity characteristics. These features include number of sensor events in each window, window duration, time of the day when first event of activity occurred, most dominant sensor during the activity, number of occurrences of each sensor during activity, and the time that elapsed from when each sensor last fired till the end of activity.

C. *Activity Transition Detection*

The work presented in this paper attempts real time activity transition detection (ATD) on discrete binary sensor data obtained from real-world smart homes. We hypothesize that activity transitions can be detected independent from the type of activities performed by a human by only using the changes in the data distributions of the feature space. We use two different unsupervised algorithms to detect activity transitions without the use of labeled data. The first is Relative Unconstrained Least-Squares Importance Fitting (RuLSIF), which detects transitions by looking for changes in the data distributions between two consecutive windows of sensor events. The second method is Bayesian Change Point Detection (BCPD), which estimates probability distributions of the new window based on the data that has been observed since the previous candidate change point.

1) RuLSIF. We first use RuLSIF [12] as an unsupervised algorithm to estimate the relative density ratio between two samples. We compare the computed ratio to a threshold value to detect changes in the underlying data distribution.

The rationale of this estimation is that knowing the two densities implies knowing the density ratio. However, the inverse is not true, and direct density-ratio estimation is substantially simpler than density estimation. RuLSIF models the density ratio between two consequent windows χ and χ' by a non-parametric Gaussian kernel model as:

$$g(\chi) = \frac{p(\chi)}{p'(\chi')} = \sum_{l=1}^{n} \theta_l K(X, X_l); K(X, X') = \exp(-\frac{\| X - X' \|^2}{2\sigma^2})$$

In these equations, $p(\chi)$ is the probability distribution of interval χ, $\theta=(\theta_1,.., \theta_n)^T$ are parameters to be learned from data samples, X is a sliding window, and $\sigma>0$ is the kernel parameter. In the training phase, the parameters θ are determined so that the dissimilarity measure is minimized. Given a density-ratio estimator $g(\chi)$, an approximator of the dissimilarity measure between two samples χ_t and χ_{t+n} is calculated in the test phase. The higher the dissimilarity measure is, the more likely the point is a change point [13][12].

RuLSIF uses α–relative Pearson (PE) divergence for $0 \le \alpha < 1$ as a dissimilarity measure, shown as:

$$PE_\alpha[p(x) \| p'(x)] = PE(p(x) \| \alpha p(x) + (1-\alpha) p'(x))$$

$$PE[p(x) \| p'(x)] = \int p'(x) (\frac{p(x)}{p'(x)} - 1)^2 dx$$

The original RuLSIF method calculates a change score between two retrospective intervals instead of two consecutive intervals which makes the algorithm completely offline. To detect activity transitions, we process a stream of feature vectors. We therefore state that RuLSIF is an *n*-real time algorithm, meaning that the algorithm requires *n* sensor events of data ahead of the current point to make a calculation, where *n* is the length of each sample [14].

We can improve the accuracy of the transition detection techniques by recognizing the fact that activities usually have blurred boundaries and the transition between activities occurs over a period of time. If the previous sensor event is detected as a transition, then the current sensor event is likely a transition also. To incorporate this fact, we calculate the final transition score based upon the sum of the PE values for the previous two sensor events. If the summed score is above a specified threshold than we output a transition label for the current sensor event. 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 [9].

Determination of RuLSIF parameters is very influential on ATD results. A sensitivity analysis of these parameters is not only critical to model validation but also serves to guide future research efforts. We use random search on one month of the collected data for parameter tuning of the RuLSIF algorithm. Random search samples algorithm parameters from a random distribution for a fixed number of iterations. A model is constructed and evaluated for each combination of parameters chosen. These parameters include window size (*n*), minimum distance between two transitions (δ), and the threshold value for identifying a transition from the RuLSIF score (Threshold), respectively. Increasing the window size does not affect performance, and since the larger window size incurs a greater computational cost, the optimal size is two sensor events in each window. Although a smaller δ value increases the accuracy of ATD, it will decrease the accuracy of rejecting non-transition events. Thus, depending on the system objective and application, a different δ value can be used. Random search on alternative threshold values shows larger thresholds decrease the true positive rate (TPR). Using the random search results and considering the fact that we prefer lower computational cost for an online activity segmentation process, we choose n = 2, δ = 1, and Threshold = 0.9. Therefore, the activity segmentation algorithm is 2-real time, which means we need a two sensor events look ahead to detect transitions.

2) Bayesian Change Point Detection. Compared to RuLSIF which only considers pairs of consecutive samples, BCPD [15] compares new sliding window features with an estimation based on all previous intervals from the same state. BCPD estimates the posterior distribution by defining an auxiliary variable run-length (r_t), which represents the time that elapsed since the last change point. Given the run length at time t, the run length at the next time point can either reset back to 0 (if a change point occurs at this time) or increase by 1 (if the current state continues for one more time unit). The run length distribution based on Bayes' theorem is:

$$P(r_t \mid x_{1:t}) = \frac{\sum_{r_{t-1}} p(r_t \mid r_{t-1}) p(x_t \mid r_{t-1}, x_t^{(r)}) p(r_{t-1}, x_{1:t-1})}{\sum_{r_t} p(r_t, x_{1:t})}$$

Where $x_t^{(r)}$ indicates the set of observations associated with the run r_t and $p(r_t|r_{t-1})$, $p(x_t|r_{t-1},x_t^{(r)})$, and $p(r_{t-1},x_{1:t-1})$ are prior, likelihood, and recursive components of the equation. The conditional prior is nonzero at only two outcomes $(r_t=0)$ or $(r_t= r_{t-1}+1)$ and simplifies the equation.

$$p(r_t \mid r_{t-1}) = \begin{cases} H(r_{t-1}+1) & r_t = 0 \\ 1 - H(r_{t-1}+1) & r_t = r_{t-1}+1 \\ 0 & Otherwise \end{cases}$$

In this equation, $H(r)$ is a hazard function which is defined as the ratio of probability density over the run to the total value of probability densities. The likelihood term represents the probability that the most recent datum belongs to current run. This is the most challenging term to calculate and it tends to be most computationally efficient when a conjugate exponential model is used. After calculating the run length distribution and updating the corresponding statistics, change point prediction is performed by comparing probability values. If r_t has the highest probability in the distribution, then a change point has occurred and the run length is reset to zero. If not, the run length is incremented by one [15]. We are using the Bayesian change point detection method as an online method (1-real time), which means it will calculate the probability for every single sensor event. We assume change point has occurred if the probability starts to decrease.

D. Activity Recognition

The activity recognition step provides real-time activity labels for each segment as new activity transition is detected in a stream. After segmenting the data, we extracted segment features and applied different multi class classifiers on real data traces to detect the activity labels. These include Decision Tree (DT), Random Forest (RF), Extra Trees (ET), Gradient Boosting (GB), Bagging, and AdaBoost. Towards the classification, we split data stream into three equal parts: the first two parts are used for training, and last part is used for testing. After labeling a segment, detected label was assigned to all events in the segment, and event labels are used to evaluate the activity recognition performance.

IV. Experiments and Results

We implement the aforementioned segmentation model and test it on real time unscripted smart home data to detect transitions between activities and recognize activity labels.

A. Experimental Conditions

We evaluate our algorithm to segment daily activities using data collected over six months in smart home testbeds that were installed in three apartments. Each of the apartments house 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 wireless motion / ambient light intensity sensors and door/temperature sensors. There is an average of 25 sensors installed in each apartment.

External staff members provide activity labels for training data based on the home floorplan, a layout of the sensors in the home, and information from residents on when and where activities are typically performed. The collected data has been annotated with 12 activity labels corresponding to activities of daily living (ADLs) and "other activity" if the event was not part of an activity of interest. These activities for each apartment are shown in Table 1.

Table 1. Characteristics of the three experimental datasets.

Activity	Apt 1	Apt 2	Apt 3
Bathing	7198	16295	5151
Bed Toilet Transition	4170	14641	4346
Cook	101020	55240	44842
Eating	20771	24417	39453
Enter Home	3711	2440	996
Leave Home	4305	2476	1246
Personal Hygiene	39190	42704	37237
Relax	39934	16996	8207
Sleep	33213	10477	20693
Take Medicine	15300	2524	1248
Wash Dishes	3280	12971	0
Work	0	0	188763
Other Activities	377031	351073	246577
Total # of activity events	148,826	67,125	100,212

B. Performance Measures

A number of different measures are commonly used to evaluate the performance of activity segmentation methods. Among these, we use three different performance metrics to evaluate the ability of transition detection algorithms in detecting both transitions and non-transition sensor events. These are true positive rate (TPR, fraction of correctly-recognized activity transitions), G-mean (Geometric mean, utilizes both true positive and true negative measures to assess the overall performance of the algorithm in terms of transition and non-transition accuracy), and mean absolute error (MAE) calculated as the elapsed time between the detected and true transition. In addition, we assume detected activity transition is true if it is between the last λ seconds of one activity and the first λ seconds of the second activity. Due to average length of activity in our dataset (12 minutes), we use λ=15 seconds.

To measure the performance of our activity recognition and classification algorithms, we used two performance metrics, overall accuracy and normalized edit distance (NED). NED uses the Levenshtein distance to measure how dissimilar a predicted activity sequence is from the actual sequence by

counting the minimum number of operations that is required to make them equivalent.

C. ATD Performance

We compare our results against two different baselines, Fixed and Random. The Fixed baseline detects *m* transitions at fixed-time intervals where *m* is the total number of actual transitions in each dataset. The Random baseline repeatedly selects *m* random detected transition points. Transitions detected within 2δ sensor events of a previously-detected transition are ignored. Figure 3 plots TPR values for RuLSIF, Bayesian, and baselines for three apartments, respectively.

Both RuLSIF and Bayesian have a higher TPR than the baselines. A t-test indicates that the difference between RuLSIF and baselines or Bayesian and baselines are significant at the ($p < .05$) level. In the next step, we calculate G-Mean scores to evaluate the accuracy and the ability to accurately detect both transition and non-transitions. Presented in Figure 4, results show RuLSIF is a reliable method for both cases. Although the Bayesian method works much better than baselines, it cannot compete RuLSIF on detecting activity transitions. A t-test indicates that there is a significant difference between RuLSIF and baselines or Bayesian and baselines at the ($p < .05$) level.

Finally, the mean value of MAE for 3 apartments are depicted in Figure 5. The MAE can be thought of as the average time between when a transition occurs and when the transition is detected. For the RuLSIF algorithm, the average value is close to 3.5 minutes, which is significantly different than baselines ($p < .05$ based on t-test). For the Bayesian algorithm, the MAE is almost two times larger than RuLSIF but still significantly different than baselines ($p < .05$ based on t-test). Ideally, we would like to minimize these times to have more accurate activity boundaries. By considering the length of activities in the real world (12 minutes is the average for our datasets), both methods have acceptable MAE. From the above results, we can see that the best performance for ATD can be achieved if we use RuLSIF algorithm.

D. Activity Recognition Performance

We begin by studying the performance of different classifier model as a predictor of activity label for each segment. After detecting activity label of each segment, the label is assigned to all sensor events inside the segment and overall accuracy is calculated. The results for this experiment are summarized in Figure 6. It can be observed that the overall accuracy of activity recognition is almost the same in the case of using RuLSIF or Bayesian algorithm for ATD. Although there is not a big difference between different classifier models, Extra Trees (ET) has slightly better accuracy. We thus use ET as the classifier for the remainder of the paper.

To evaluate the performance of our activity segmentation model in activity recognition, we compare it with traditional fixed length sensor windows based activity recognition (AL) [1]. In this method, data was divided into non-overlapping windows with 30 sensor events. Then feature extraction was applied on each window and a decision tree model was built to detect activity labels. Accuracy and NED of RuLSIF-ET, Bayesian-ET and AL are showed in Figures 7 and 8.

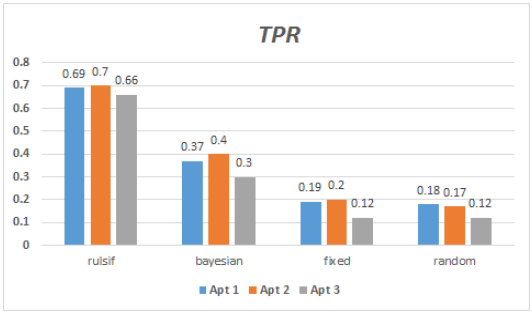

Fig. 3. TPR scores for ATD algorithms and baselines.

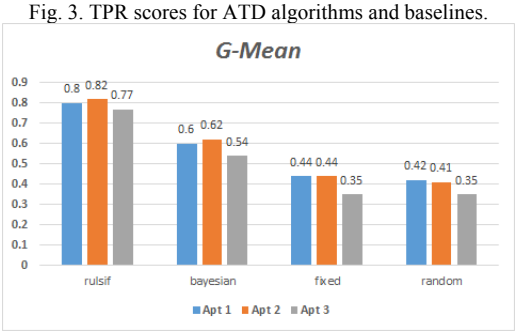

Fig. 4. G-Mean scores for ATD methods and baselines.

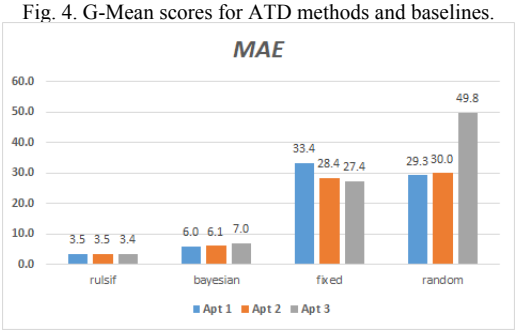

Fig. 5. MAE scores for ATD algorithms and baselines.

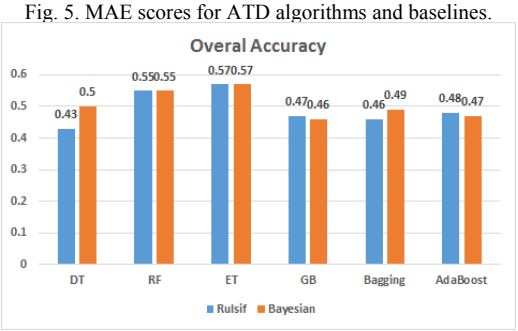

Fig.6. Accuracy of alternative activity classifiers.

It is interesting to note that there is no significant difference between overall accuracy of different methods. Although using the proposed activity segmentation, we will have additional information about activity boundaries, which is highly valuable and extensively useful in different applications. Normalized edit distance values show considerable differences between activity segmentation and the traditional method. As we know, a smaller NED value means the detected activity sequence is more similar to the actual one. Since in traditional methods, classifier predicts label for each sensor events, the sequence

will be 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 AL.

When comparing RuLSIF and Bayesian ATD algorithms, Bayesian has smaller NED value. After recalling that RuLSIF has better performance in ATD, we can conclude, depending on the final objective of segmentation algorithm, both methods can be useful. If finding more accurate activity transition is needed such as finding the best time for prompting, or detecting the exact time of a specific activity, RuLSIF would be a better choice. On the other hand, if the main objective is related to overall trend of activities, such as health assessment or lifestyle change detection, Bayesian method is more reliable.

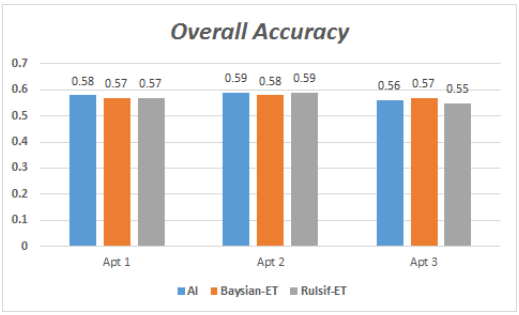

Fig. 7. Accuracy of alternative activity labeling methods.

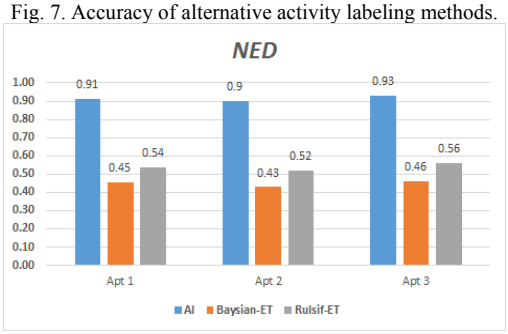

Fig. 8. NED of alternative activity labeling methods.

V. Conclusion

Nowadays, real world application can benefit from robust and accurate activity related information. Real time activity segmentation can provide insights on human behavior, indicate opportune times for requesting user attention, and assist with activity segmentation. In this paper we investigate and evaluate automate daily activity segmentation using change point detection techniques in an on line or streaming fashion, use unscripted data from smart homes and compare their performance using pre-defined metrics.

The experiments conducted to evaluate our activity segmentation techniques on real-world smart home datasets show that both RuLSIF and Bayesian algorithms are helpful in different types of activity segmentation applications. Also, while the overall accuracy of proposed automate activity segmentation is similar to the accuracy of regular fixed length window based traditional activity recognition algorithm, providing additional information about activity boundaries and more accurate detected activity sequences make it highly valuable and extensively useful.

Acknowledgment

This work was supported in part by National Science Foundation grant 1064628 and National Institutes of Health grant R25AG046114. The authors would like to thank Tinghui (Steve) Wang for his valuable help with the infrastructure.

References

[1]D. J. Cook and N. C. Krishnan, *Activity Learning: Discovering, Recognizing, and Predicting Human Behavior from Sensor Data*. Wiley, 2015.

[2]R. Fallahzadeh, S. Aminikhanghahi, A. Gibson, and D. Cook, "Toward personalized and context-aware prompting for smartphone-based intervention," in *International Conference of Engineering in Medicine and Biology Society*, 2016.

[3]T. Okoshi, J. Ramos, H. Nozaki, J. Nakazawa, A. K. Dey, and H. Tokuda, "Attelia: Reducing user's cognitive load due to interruptive notifications on smart phones," in *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2015, pp. 96–104.

[4]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.

[5]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.

[6]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.

[7]S. A. Rokni and H. Ghasemzadeh, "Autonomous sensor-context learning in dynamic human-centered internet-of-things environments," in *Proceedings of the 35th International Conference on Computer-Aided Design - ICCAD '16*, 2016, pp. 1–6.

[8]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.

[9]K. D. Feuz, D. J. Cook, C. Rosasco, K. Robertson, and M. Schmitter-Edgecombe, "Automated Detection of Activity Transitions for Prompting," *IEEE Trans. Human-Machine Syst.*, vol. PP, no. 99, pp. 1–11, 2014.

[10]A. Ali, R. C. King, and Guang-Zhong Yang, "Semi-supervised segmentation for activity recognition with Multiple Eigenspaces," in *2008 5th International Summer School and Symposium on Medical Devices and Biosensors*, 2008, pp. 314–317.

[11]Y. Ling and H. Wang, "Unsupervised Human Activity Segmentation Applying Smartphone Sensor for Healthcare," in *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing*, 2015, pp. 1730–1734.

[12]S. Liu, M. Yamada, N. Collier, and M. Sugiyama, "Change-point detection in time-series data by relative density-ratio estimation.," *Neural Netw.*, vol. 43, pp. 72–83, Jul. 2013.

[13]Y. Kawahara and M. Sugiyama, "Sequential Change-Point Detection Based on Direct Density-Ratio Estimation," in *SIAM International Conference on Data Mining*, 2009, pp. 389–400.

[14]S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowl. Inf. Syst.*, pp. 1–29, Sep. 2016.

[15]R. P. Adams and D. J. C. MacKay, "Bayesian Online Changepoint Detection," Machine Learning, Oct. 2007.