2014 13th International Conference on Machine Learning and Applications

Activity Recognition using Graphical Features

Syeda Selina Akter
School of Electrical Engineering and
Computer Science
Washington State University
Pullman, WA-99163
Email: selina.akter@email.wsu.edu

Lawrence B. Holder
School of Electrical Engineering and
Computer Science
Washington State University
Pullman, WA-99163
Email: holder@wsu.edu

Abstract—**Activity Recognition is important in order to facilitate elderly residents' and their caregivers' needs. This problem has been widely investigated using different methods including probabilistic and Markovian approaches. The focus of this paper is to perform activity recognition more accurately than existing approaches using non-intrusive sensors. We represent motion sensors of smart environments in a graph and resident's movements as edges in the graph. Then graph-based features are extracted and used as input for a Support Vector Machine. These features have been combined with motion-sensor based features. This method has been compared with three other widely used approaches, Naive Bayes, Hidden Markov Model (HMM) and Conditional Random Fields (CRF) on three different datasets from three smart apartments. In all cases, the method based on graphical features outperformed one of the state of the art methods for activity recognition.**

I. Introduction

Our goal is to evaluate the use of graph representations and mining to improve performance on recognition and prediction tasks for Wireless Sensor Networks (WSN). Most WSN applications focus on monitoring living beings, sensing object interaction and tracking locations visited. WSN have been deployed for Outdoor Environmental Monitoring such as Forest Fire Detection [1], Flood Detection [2], Habitat Monitoring [3]; for Indoor Environmental Monitoring such as Reduce Energy Waste [4], Fire and Smoke Detection [4]; Support for Logistics such as Inventory Control Application [4]; for Human-Centric Applications such as HealthCare [5], Tracking and Monitoring Doctors and patients inside a hospital [4] and Tracking and Monitoring elderly residents inside their house [6]. As our first application area, we started assessing the use of graph representations in smart homes designed for independent living by elderly residents. Smart homes equipped with sensors for monitoring resident's activities are beneficial due to increase of aged population, high cost of formal health care and importance of independent living of elderly people in their own home. Automating the recognition of activities is important to monitor whether individuals are being able to complete Activities of Daily Living (ADL) at their home. Activity recognition has been identified as the topmost need for Alzheimer's patients' caregivers in a survey of assistive technologies [6].

We attempted to improve performance of activity recognition over existing approaches using a graph representation of non-intrusive sensor data. We propose a graph representation for sensor nodes deployed in smart apartments. Residents' movement information captured by these sensor nodes has also been represented in the graph. Then we extract information from the graph as features for Machine Learning techniques. Our goal is to improve accuracy of activity recognition that will facilitate better remote functional health monitoring and will enhance the ability to provide technological help more accurately.

II. Related Work

Extensive research has been done to solve the problem of activity recognition using various approaches. In previous works, the focus has been on non-graphical features such as motions sensors, temporal information during activity such as time of day, day of week, whether the day is weekend or not, activity length in time (seconds), previous/next activity, number of kinds of motion sensors involved, total number of times motion sensor events triggered and energy consumption for an activity (in Watts). Chen et al. [7] selected the most important features based on two feature selection techniques - Minimal Redundancy and Maximal Relevance (mRMR) and Mutual Information and used four different Machine Learning techniques for recognizing activities: Bayes Belief Networks, Artificial Neural Networks, Sequential Minimal Optimization and LogitBoost Ensemble. These methods have been compared with each other in this work, but have not been compared with other widely used techniques such as Naive Bayes and HMM. Singla et al. [8] used a Markov model with and without temporal information and observed improved accuracy for activity recognition. They applied Naive Bayes, Markov Model and Hidden Markov Model to interleaved and scripted Activities of Daily Living (ADL) in [9]. Nazerfard et al. [10] used Conditional Random Fields, a probabilistic approach which can capture interdependence between features and has been fed with 5 types of features: sensors, time of day, day of week, previous activity and activity length. Though HMM and CRF have been used for segmented activity recognition and are available as part of a tool on [11], there is still need of and scope to recognize resident activities more accurately. We try a graph-based approach in this paper to obtain better performance for the activity recognition task.

Graph-based approaches have been applied to some wireless sensor network applications. In [12], time-dependent spatio-temporal networks such as road networks have been represented using nodes and edges of a graph where locations can be represented as nodes and road segments can be represented as edges with time-dependent network properties like network congestion as edge attributes. The authors proposed a graph representation of such a network with time series attached as attributes on nodes and edges. This representation can handle change of edge attributes and existence and disappearance of edges with time. Using such graphical representation, an algorithm also has been proposed to answer queries that might change with time. For example, the shortest route in a road network might change with time due to change of traffic. Using their approach, the shortest route for a given start time or for a range of time can be calculated in an efficient way.

Long and Holder [13] used graph-based approaches to solve the problem of activity recognition in smart environment. In [13], three graph-based methods have been used: frequent subgraph technique to generate feature vectors for various learning algorithms, SVM with a graph kernel and nearest neighbor approach with a graph comparison measure. Results of these three methods have been compared with non-graph SVM method with a goal of solving a 10-class activity recognition problem in the smart home. An ensemble of all these four methods has also been used to classify unscripted, multi-resident and interleaved activities. Though none of the graph methods showed improved accuracy, all the graph methods exhibited uncorrelated error with the baseline non-graph method and with each other indicating potential of applying more graph methods towards solving this problem.

III. Methodology: Applying Graphical Features

For representing smart home data as a graph, we consider each motion sensor in the smart apartment and represent it as a vertex in the graph. For each labeled activity, we construct a graph. When two consecutive sensors are turned on during a labeled activity, an undirected edge is added between the two corresponding vertices in the graph. Two motion sensors can be triggered consecutively multiple times, that is, an edge can be triggered multiple times. We store this count of multiple edges being triggered as edge attributes. Same motion sensor can be triggered consecutively in smart environment and hence, we allowed self-loops in our graph representation to capture this information.

We validated our approach on four datasets from CASAS project [14] that are varied in terms of apartment layout, number of residents and number of activities (both scripted and non-scripted). In Table I, we summarized data characteristics for these four smart testbeds including total number of motion sensors in each testbed, number of residents, total number of possible edges including self-loops in our proposed graph representation, total number of edges triggered and total number of activities for each dataset. In kyoto dataset, residents performed scripted activities; motion sensor data were collected from Cairo, Aruba and Tulum where residents performed unscripted real-life daily activities. Cairo and Aruba are single-floor apartments; Tulum and Kyoto are two-storey apartments.

We show example graphs to illustrate our representation on Cairo, Aruba, Tulum and Kyoto smart apartment layouts in Fig 1, 2, 3 and 4 respectively for different activities. In these figures, vertices of our graph representation are aligned with the corresponding motion sensor nodes in the apartment layout, lines connecting two motion sensors represent edges in the graph representation and the thickness of these lines signify the corresponding edge attributes.

TABLE I. Characteristics of Four Datasets

Datasets	**Cairo**	**Aruba**	**Tulum**	**kyoto**
Residents	2+pet	1	2	400
Sensors	27	31	31	27
Possible Edges	378	496	496	378
Triggered Edges	350	435	486	289
Activities	10	11	16	16

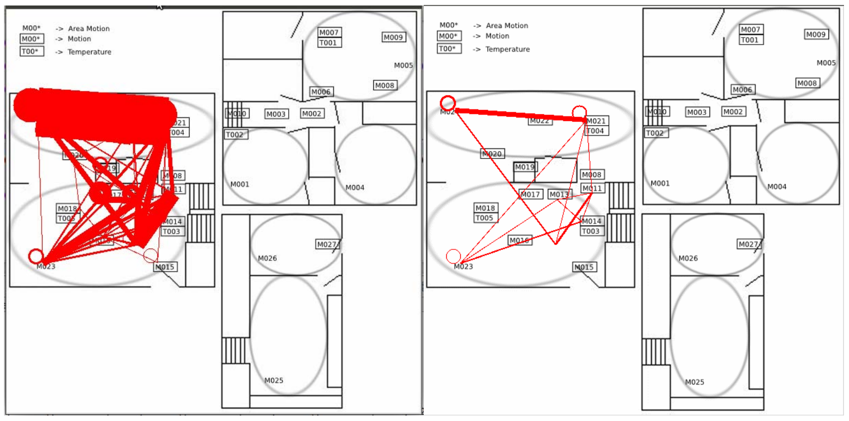
After graph construction for an activity, we extract graphical features from it. We consider each possible edge ever triggered in the data as a feature. If an edge exists in this graph for the current activity, we assign corresponding edge attributes as the value in the feature vector. If an edge does not exist in the graph, default value for that feature is zero. Edges that have never been triggered for any activity in the dataset were not included in the feature list. In our initial experiments with the Cairo dataset, the total number of graphical features extracted in this way is 350. We also experimented with a feature whose value is 1 if that edge is triggered at least once during that activity. We also tried representing the transition from one sensor to another sensor as a directed edge. After experimenting with one-edge path as features, we tried to use all possible combination of 2-edge paths as features.

After constructing the graphical feature vector, we provided this feature vector to the Support Vector Machine (SVM) for classifying 10 activities of the resident from the Cairo testbed. Libsvm from Weka [15] with default settings was used to run this experiment with 10-fold cross-validation. Libsvm uses one against one method for multi-class classification by SVM. [15]

We used a non-graph method as the baseline for comparing with our graphical features based approach. If a motion sensor has been on during an activity we represent it with a 1 and if that motion sensor is never on during an activity then we represent the value of this motion sensor as zero. We compute these values for all motion sensors during an activity and construct a feature vector. We provide this feature vector to the Support Vector Machine learning algorithm. We also computed how many times each motion sensor has been turned on during an activity and used this count as a feature for non-graph method.

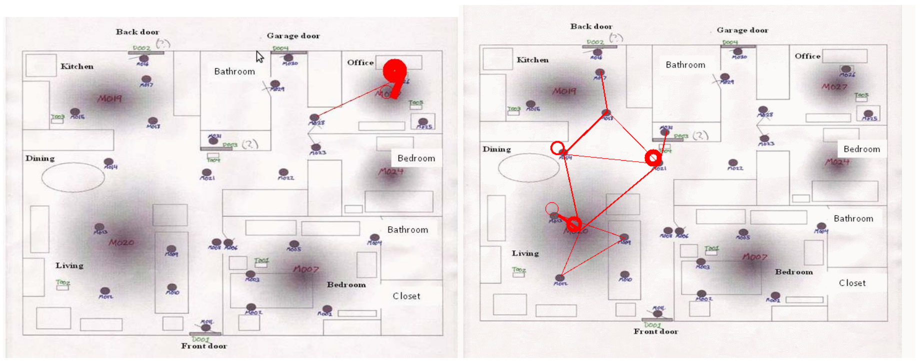
We compared our graphical feature based approach with three widely used approaches: Naive Bayes, Hidden Markov Model (HMM) and Conditional Random Fields (CRF). We used the Activity Recognition (AR) tool, ar1.3 that is available at [11] and that implemented all these techniques for recognizing segmented activities. For these three AR algorithms, there are five feature options that can be used. Those are the list of all sensors with status on/off (numeric), time of day (categorical), day of week (categorical), previous activity (categorical) and activity length in terms of number of sensors (numeric). For these three baseline methods, we used default settings for these features as set in the AR tool.

Initially we experimented on the Cairo dataset which is a multi-resident smart apartment with 27 motions sensors. Ten activities have been labeled in the Cairo Dataset. These are Laundry, Sleep, Night wandering, Lunch, Dinner, Leave Home, Breakfast, Work in office, Bed to toilet and Take medicine. We applied the Support Vector Machine on both non-graph and graphical feature vectors for classifying the 10 activities. We handled interleaved activities as separate activities here. After



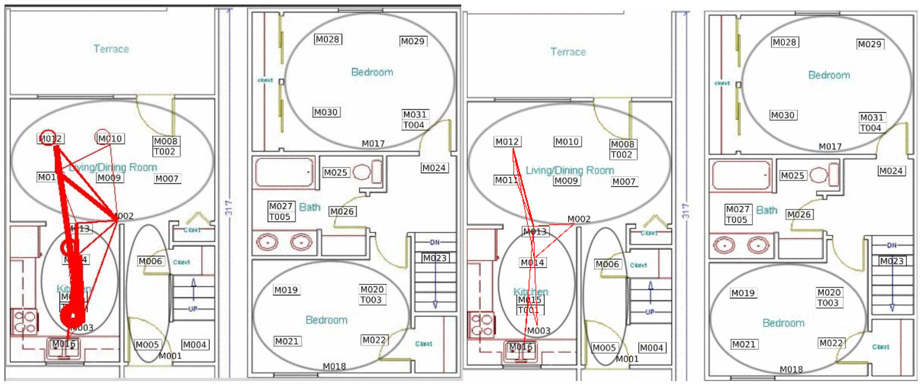
(a) Graph Representation for 'breakfast' (b) Graph Representation for 'take medicine'

Fig. 1. Cairo: Graph representation on the apartment layout



(a) Graph Representation for 'work' (b) Graph Representation for 'housekeeping'

Fig. 2. Aruba: Graph representation on the apartment layout



(a) Graph Representation for 'watch TV' (b) Graph Representation for 'meal preparation'

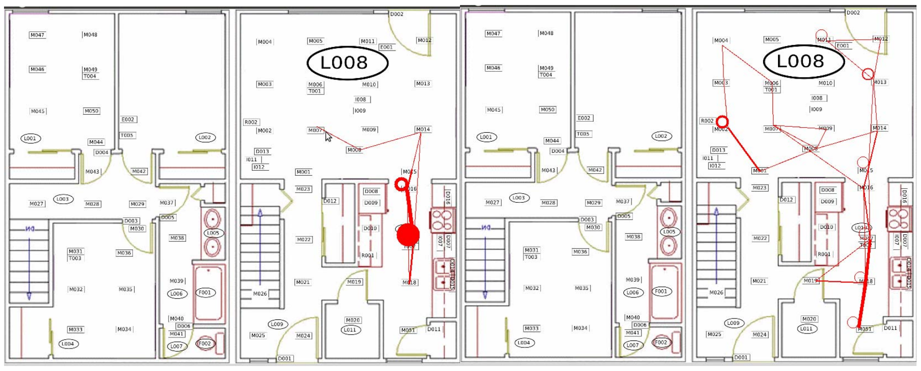
Fig. 3. Tulum: Graph representation on the apartment layout

getting promising results from Cairo, we applied the graphical feature method on other smart home datasets, namely, Aruba, Tulum and Kyoto, and compared the results with the baseline methods mentioned above.

IV. Results

A. *Graph Features Vs Non-Graph Features*

For the non-graph method, we provided the set of all sensors being on/off as the feature vector to the SVM with RBF kernel. For the graph method, we provided graphical features extracted using the approach described in the methodology



(a) Graph Representation for 'cook'

(b) Graph Representation for 'sweep kitchen and dust living room'

Fig. 4. Kyoto: Graph representation on the apartment layout

TABLE II. CAIRO DATASET: SVM (RBF KERNEL)

Features	Accuracy in percentage
Non-graph features	65.86
Graph features (Existence of Edge)	50.1
Graph features (Edge Attributes)	71.31

TABLE III. CAIRO DATASET: SVM (LINEAR KERNEL)

Features	Accuracy in percentage
Non-graph features	70.10
Graphical Features (Existence of Undirected Edge)	71.31
Graphical Features (Count of Undirected Edge)	76.57
Graphical Features (Directed Edge)	74.14

TABLE IV. CAIRO DATASET: SVM (LINEAR KERNEL) WITH ATTRIBUTE SELECTION

Filtering Techniques	Accuracy in percentage	Selected Attributes (Total Attributes)
Consistency Subset Eval with Search Method: best First Search	75.96	23 (350)
Chisquared attribute Evaluation with Search method: Ranker with Threshold 1	77.17	143 (350)

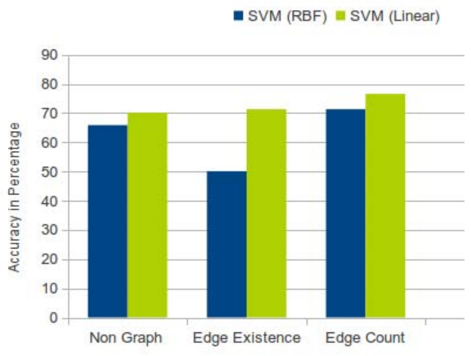


Fig. 5. Cairo: SVM (RBF) Vs SVM (Linear Kernel)

section. We present the results in Table II. From Table II, performance in terms of accuracy has been improved using edge attributes as graphical features compared with non-graph features. We tried both existence of edges (1/0) and count of edges during an activity and the later approach provided better performance of 71.31% accuracy whereas accuracy of activity recognition from the former method is 50.1%.

We also compared each method using a linear kernel for the SVM instead of the RBF kernel. We show the result in Table III. We observe in Fig. 5 that the SVM linear kernel performs better than the RBF kernel for each method and use of graphical features with edge count provided better performance than baseline of using count of motion sensor as non-graph features. The result of using directed edges from one node to another node representing the sequence of sensor nodes triggering in time is also shown in Table III. But representing transitions with directed edges resulted in less accuracy than the undirected edge representation.

We used filtering techniques to assess the effect of feature selection on the accuracy of classification. In Table IV, we mentioned the attribute evaluators used from Weka with "full training set" as the selection mode, corresponding search methods and results. Graphical features along with these feature selection techniques produced similar performance, but they reduced the total number of features needed for classification significantly while still providing the improved result.

B. Two-edge Transitions

We observed two edge transitions between motion sensors as features and applied SVM on this feature vector. For the Cairo dataset, the possible number of sensor-node triples is 19,683 among which 4,163 triples are triggered in the dataset, resulting 4,163 features in the feature vector. Accuracy for this method was 16.22%. Because including this additional edge information led to worse performance, we did not try generalizing to k-edge paths for each feature.

C. Graph Features to Aruba, Tulum and Kyoto datasets

Next we applied non-graph features and graphical features to other testbeds. We collected datasets for the Aruba, Tulum and Kyoto testbeds from Washington State University's CASAS Project website [14]. For each dataset, our implementation discovers all activities of residents from label in each test

TABLE V. NON-GRAPH METHOD: MOTION SENSOR ON/OFF VS COUNT OF MOTION SENSOR

Dataset	Motion sensor On/Off	Count of motion sensor
Cairo	70.10	72.73
Aruba	91.86	93.31
Tulum	71.48	80.03
Kyoto	67.15	83.20

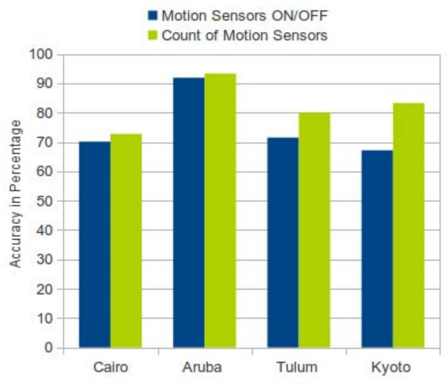


Fig. 6. Non-graph features: Motions sensors on/off Vs Count of Motion Sensors

TABLE VI. GRAPH METHODS: EXISTENCE OF EDGE VS COUNT OF EDGE

Dataset	Existence of edge	Edge attribute
Cairo	71.31	76.57
Aruba	93.29	93.41
Tulum	76.11	76.86
Kyoto	77.85	83.22

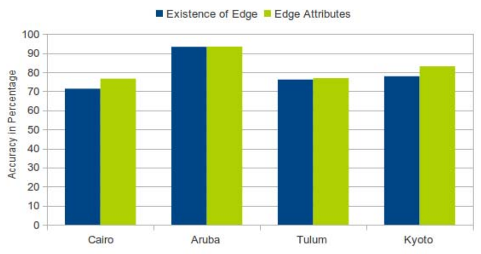


Fig. 7. Graphical features: Existence of edges Vs Count of edges

apartment as it goes through the dataset and constructs non-graph and graphical feature sets. We applied two non-graph methods to these datasets as described in the methodology section: one is using off/on status of each motion sensor as a non-graph feature and another is count of each motion sensor being triggered on as a non-graph feature. Result of this experiment is shown in Table V and is plotted in Fig. 6. We observe that the count of motion sensors as features improved the accuracy over motion sensors being on/off as features. For graph methods, we used two approaches as well for selecting graphical features: existence of edges (0/1) and count of edges. From Table VI and plot in Fig. 7, we see that later method provided better performance compared to the former method.

As count of motion sensors as non-graphical features and count of edges as graphical features provided better result, we selected these two approaches for comparing between non-graph and graph-based method on each dataset. Moreover, we combined non-graph features and graphical features into one set of feature vector and provided it to SVM linear to evaluate the effect of this combination. We presented the comparison among non-graph, graph-based and combination of these two methods in Table VII and plotted in Fig. 8. We observe that the graph method improved accuracy over the non-graph method for each of these datasets except Tulum. However, we got improved accuracy for combination of both kind of features for Tulum and Kyoto dataset compared to both non-graph features only and graphical features only. We can conclude from this result that improved performance have been obtained for all of these datasets using either graphical features or combination of non-graph and graphical features compared with the baseline of non-graph features only.

TABLE VII. COMPARISON: NON-GRAPH METHOD VS GRAPH METHOD

Dataset	Non-Graph Method (Count of Motion Sensors)	Graph Method (Edge Attributes)	Combination of Non-Graph and Graphical Features
Cairo	72.73	76.57	75.76
Aruba	93.31	93.41	93.24
Tulum	80.03	76.86	81.63
Kyoto	83.20	83.22	85.07

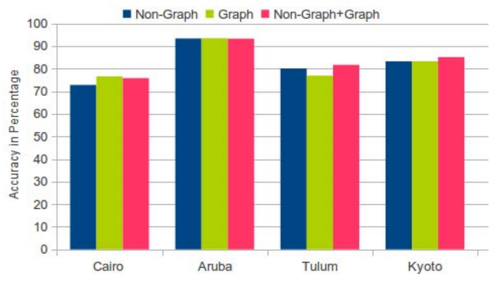


Fig. 8. Non-Graph Vs Graph Method on different datasets

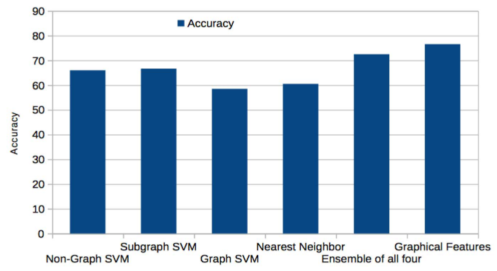


Fig. 9. Graphical Features vs Other Graph Approaches

D. Graph Method vs Other Approaches

Here, we compare the Graphical Feature method with Non-graph SVM, Graph SVM, Subgraph SVM, Nearest Neighbor and an ensemble of the these four as proposed in [13] and present the comparison in Fig. 9. These results show that the graphical feature based method outperforms the previous results from [13].

We also compare the graph-based method with other base-line methods used widely in smart home research, namely, Naive Bayes, Hidden Markov Model and Conditional Random Field. We used AR tool from [11] to run Naive Bayes, HMM

TABLE VIII. Comparison of Graphical Features with other Methods

Dataset	NB	HMM	CRF	Non-graph features	Graphical features	Non-graph + graphical features)
Cairo	65.04	73.57	69.67	72.73	76.57	75.76
Aruba	91.18	88.87	91.83	93.31	93.41	93.24
Tulum	62.30	61.68	38.28	80.03	76.86	81.63
Kyoto	NA	NA	NA	83.20	83.22	85.07

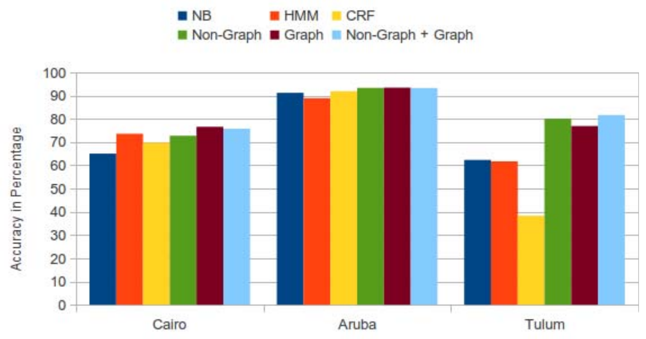


Fig. 10. Graph Methods Vs other baseline methods

TABLE IX. Comparison:Execution Time

Dataset	CRF	Non-graph features	Graphical features	Non-graph+graphical features
Cairo	358m 32s	55s	8s	8s
Aruba	415m 32s	25m 11s	16m 43s	21m 33s
Tulum	1352m 56s	13m 4s	30m 43s	48m 9s
Kyoto	NA	43s	47s	1m 15s

and CRF on these four datasets. Along with other three datasets, we chose kyoto dataset to assess performance on data that is collected from many participants. Kyoto (400 participants) is different from other three test datasets as many individuals participated in scripted activity experiments separately at this testbed on different days and times. This dataset does not contain day of week information which is one of the features used by AR in default settings resulting unavailability of the result and mentioned as ’NA’ in Table VIII. We extracted non-graph and graph features though for this dataset and presented the result in Table VIII.

From Table VIII we observe that graph-based SVM outperformed other widely used baseline methods for each of these datasets. Moreover, a drawback of CRF is that it has long execution times which is shown in Table IX comparing with non-graph and graphical approaches. AR execution time on kyoto is unavailable (NA) due to the same reason mentioned for Table VIII. We observe from Table IX that it needed 6 to 22 hours of execution time for CRF on Aruba and Tulum dataset while other methods executed in range of minutes.

V. Conclusion

In this work, we validated our graph-based approach on three different testbeds of single resident, multi-resident and multi-resident with a pet, with residents performing non-scripted activities. The graphical feature based approach improved the accuracy of activity recognition for all of these datasets compared with other widely used baseline methods for motion sensor data. We applied this method on segmented data with labeled activities. We plan to apply similar methods on streaming data from smart environments and try to recognize activities. We also plan to incorporate temporal information from time-based sensor stream in the graph to further improve the performance. Being inspired by evaluation of graphical features method in smart home data, we are going to assess usefulness of this method and other graph methods on datasets from other wireless sensor networks that are widely being used for tracking and monitoring.

VI. Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 1318913.

References

[1] L. Yu, N. Wang, and X. Meng, “Real-time forest fire detection with wireless sensor networks.” in *Wireless Communications, Networking and Mobile birdsComputing*, vol. 2, 23–26, 2005, pp. 1214–1217.

[2] D. Basha and E. Rus., “Design of early warning flood detection systems for developing countries.” in *Information and Communication Technologies and Development (ICTD)*, Dec. 15–16, 2007.

[3] A. Mainwaring, J. Polastre, R. Szewczyk, and D. Culler., “Wireless sensor networks for habitat monitoring.” in *In ACM Workshop on Sensor Networks and Applications*, 2002.

[4] T. Arampatzis, J. Lygeros, and S. Manesis., “A survey of applications of wireless sensors and wireless sensor networks.” in *Proceedings of the 13th Mediterranean Conference on Control and Automation*, 2005.

[5] H. Alemdar. and C. Ersoy., “Wireless sensor networks for healthcare: A survey.” *Computer Networks*, vol. 54:15, pp. 2688–2710, 2010.

[6] D. J. Cook, “Learning setting-generalized activity models for smart spaces.” *IEEE intelligent systems*, vol. 2010.99, p. 1, 2010.

[7] C. Chen, B. Das, and D. J. Cook., “A data mining framework for activity recognition in smart environments.” in *Intelligent Environments (IE)*, Springer Berlin Heidelberg, 2010, pp. 85–99.

[8] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe., “Incorporating temporal reasoning into activity recognition for smart home residents.” in *Proceedings of the AAAI workshop on spatial and temporal reasoning.*, 2008.

[9] G. Singla, D. J. Cook, and M. Schmitter-Edgecombe, “Tracking activities in complex settings using smart environment technologies.” *International journal of biosciences, psychiatry, and technology (IJBSPT)*, vol. 1:1, p. 25, 2009.

[10] E. Nazerfard, B. Das, L. B. Holder, and D. J. Cook, “Conditional random fields for activity recognitionin smart environments,” in *Proceedings of the 1st ACM International Health Informatics Symposium. ACM*, 2010.

[11] Washington state university casas project tools. [Online]. Available: http://ailab.wsu.edu/casas/tools/ar/

[12] G. Betsy and S. Shekhar, “Time-aggregated graphs for modeling spatio-temporal networks,” in *Advances in Conceptual Modeling-Theory and Practice*, Springer Berlin Heidelberg, 2006, pp. 85–99.

[13] S. S. Long and L. B. Holder, “Using graphs to improve activity prediction in smart environments based on motion sensor data,” in *Toward Useful Services for Elderly and People with Disabilities*, Springer Berlin Heidelberg, 2011, pp. 57–64.

[14] Washington state university casas project dataset. [Online]. Available: http://ailab.wsu.edu/casas/datasets.html

[15] C. Chang and C. J. Lin, “Libsvm: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2:3, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.