Feature extraction for human activity recognition on streaming data

Nawel Yala, Belkacem Fergani
LISIC Laboratory
USTHB, Faculty of Electronics and Computer Sciences
Algiers, Algeria
YNawel@hotmail.com, BFergani@gmail.com

Anthony Fleury
Mines Douai, URIA, F-59508 Douai
Univ. Lille, F-59000 Lille
France
Anthony.Fleury@mines-douai.fr

***Abstract*— An online recognition system must analyze the changes in the sensing data and at any significant detection; it has to decide if there is a change in the activity performed by the person. Such a system can use the previous sensor readings for decision-making (decide which activity is performed), without the need to wait for future ones. This paper proposes an approach of human activity recognition on online sensor data. We present four methods used to extract features from the sequence of sensor events. Our experimental results on public smart home data show an improvement of effectiveness in classification accuracy.**

Keywords—Activity Recognition, streaming data, Incremental SVM

I. INTRODUCTION

Sensor-based Activity Recognition (AR) is a key feature of many ubiquitous computing applications such as health care and elder care. It aims to identify the actions performed by a person given a set of observations in her environment. There are several projects that work on activity recognition in smart environment such as the CASAS project [1], and PlaceLab [2].

We can classify the researches on activity recognition according to the type of sensors used to collect information, the machine learning models and the nature of activities performed (simple or complex).

Independently of the sensors used, in the feature extraction step most AR systems discretize data obtained from the sensors into time slices of constant or variable length, and each time slice is labeled with only one activity. There is relatively no problem when activities are performed sequentially (one after another), but this is not the case when activities are interleaved, that is to say one time slice may contain information about more than one activity. To deal with this, techniques that work on online/streaming data are required.

There is another need for online activity recognition, when a specific application track the execution of a daily living activity, step-by-step, for delivering in-home interventions to a person or for giving brief instructions describing the way a task should be done for successful completion [3].

The present paper is an extension of a study presented in [4]. This approach proposes online activity recognition on discrete binary motion sensor data obtained from real-world smart homes. It classifies every sensor event based on the information encoded in a sliding window containing the preceding sensor events. It explores both fixed static window size and dynamic varying window size. The extension in the present paper focuses on methods based on fixed window size with the following contributions:

1. Proposing an extension of feature descriptors using mutual information based weighting of sensor events within a window.
2. Proposing a new feature descriptor using "last-state" of sensor within a window.

This paper is structured in the following way. A discussion on the different techniques to segment online or streaming data is depicted in Section II. Section III introduces the method of construction of the feature vector while our approach used to recognize activities on streaming data is presented in section IV. Section V presents the experimental setup for evaluating the proposed approaches; the results are presented and discussed in this section too. Conclusion and future works are found in Section VI.

The segmentation step aims to divide the data into segments or windows most suitable for activity recognition. On each window, features are computed and then used as an instance for learning or testing phase. It is a difficult task since humans perform activities regularly and consecutive activities cannot be clearly distinguished as the exact boundaries of an activity are difficult to define. In this section, we present briefly the most used segmentation techniques in the context of human activity recognition on streaming data.

II. WINDOWING OF STREAMING DATA

A. Activity-based windowing

This method divides streaming data events into windows at the points of detection of changes in activity [15]. Each window likely corresponds to an activity. It has however some drawbacks. Since the activities are generally not well distinct, resulting activity boundaries are not precise. Moreover, finding the pertinent separation points occur during training phase, which complicates the calculations. This technique is not suitable for online recognition since it has to wait for future data to make decision. This method is more suitable for labeling data.

B. *Time-based windowing*

Here, streaming data events are divided into fixed time windows. It is the most commonly used segmentation method for activity recognition due to its simplicity of implementation [9, 15, 18] and for well dealing with continuous data sensor. However, many of the classification errors using this method come from the selection of the window length [17]. If a small length is selected, there is a possibility that the window contains insufficient information for making decision. On the contrary, if the length is too wide, information of multiple activities can be embedded in one window. As a result, the activity that dominates the time window will be more represented compared to other activities, which badly affects the decision. Furthermore, if sensors do not have a constant acquisition rate (case of motion and door sensors that are "event-based"), it is possible that some windows do not have any sensor data in them.

C. *Sensor-based windowing*

As for this method, data are divided into windows of equal number of sensor events. On Fig. 1 (c), the sensor windows are obtained using a sliding window of length 6 sensor events. It is clear that the resulting windows duration differs from one window to another. During the execution of activities, multiple sensors could be triggered, while during silent periods, there will not occur many sensor firings. The sensor events preceding the last event in a window define the context for the last event. This method has also some inherent drawbacks. For example, consider the segment S27, on Fig. 1 (c). The last sensor event of this segment corresponds to the beginning of activity A2. There is a significant time laps between this event and the preceding. The relevance of the use of all the sensor data in this segment with this last event might be small if the time lag is large. The method has another drawback in the case of two or multiple residents in a smart home. One segment can contain sensor events of two residents. The last event in this segment belonging to one resident while the sensor events preceding and define it belonging to two residents. Thus processing all the sensor events in a large window with equal importance for all the data might not be a good approach. This method as it is generally used may not be attractive; modifying it to account for the relationship between the sensor events is a good way to process the stream of sensor events [4]. This approach offers computational advantages over the activity-based windowing and does not require future sensor events for classifying past sensor events.

In this paper, we use this technique to deal with streaming sensor data with some modifications to overcome its drawbacks. The problem is presented in the next section.

III. FEATURES EXTRACTION

In the context of human activity recognition, some applications such as prompting systems need to know at which activity a single sensor event belongs, to provide the necessary assistance at the right time. Approach proposed in [4] aims to classify every single sensor event into a label to the best possible extent.

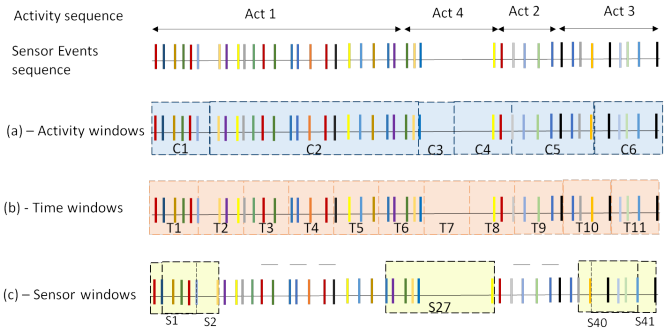


Fig. 1. Illustration of the different segmentation approaches of streaming data.

A. *Feature extraction*

In this section, we introduce four methods used to extract features from the sequence of sensor events. Two of them are proposed [4] and the other two methods are our contribution.

Let's consider $[E_1, E_2, \dots, E_N]$ a sequence of sensor events collected from a one resident smart home test-bed. An example of such sequence is depicted on Fig. 2. Each event sensor is associated with date and time, sensor ID, sensor status and activity. Sensors IDs starting with M are motion sensors and IDs starting with D are door sensors.

The segmentation technique used is the sensor-based windowing. Each window contains an equal number of events. The sensor events preceding the last event in a window define the context for the last event. Thus, from a window, we extract one feature vector that represent the last event, and is labeled with the label of the last event in the window.

If we consider $\boldsymbol{m}$ as being the number of events in a window, sensor event E_i is represented by the sequence $[E_{i-m}, E_{i-m+1}..., E_i]$. $\boldsymbol{m}$ is selected empirically. It is influenced by the average number of sensor events that span the duration of different activities.

1) *Baseline method*

Once the sensor event window E_i is defined, we can now transform this window into a feature vector. For this, we construct a fixed dimensional feature vector X_i containing the time of the first and last sensor events, the duration of the window E_i and a simple count of the different sensor events within the window. For instance, if 34 is the number of sensors installed in the smart home, the dimension of feature vector X_i will be 34 + 3. X_i is tagged with the label Y_i of E_i [4].

One of the problems with the sensor-based windowing method is the possibility for the window to contain sensor events that are widely separated in time. We can illustrate this problem by Fig. 3. This is an example of a sequence of sensor events from Aruba CASAS dataset. We can observe that there is a difference of four hours between the two last sensor events. All the sensor events that represent the last event have occurred in the 'distant' past. Thus in the absence of any weighting scheme, even though the sensor event corresponding to the 'work-end' activity occurred in the past, it has an equal influence on defining the context of the event corresponding to the activity 'sleeping-begin'.

In order to overcome this problem, [4] proposed a time-based weighting scheme that takes into account the relative difference in the triggering of each events.

Another problem appear when a window contains sensor events corresponding to the transition between two activities. Most of these events have not relation with the last event in the window and sensors from a particular activity dominate the window. This leads a wrongly description of the last event in the window. To overcome this problem, [4] defines a weighting scheme based on a mutual information measure between the sensors as described in the next section.

2010-11-04	05:40:51.303739	M004	ON	Bed_to_Toilet	begin
2010-11-04	05:40:52.342105	M005	OFF		
2010-11-04	05:40:57.176409	M007	OFF		
2010-11-04	05:40:57.941486	M004	OFF		
2010-11-04	05:43:24.021475	M004	ON		
2010-11-04	05:43:26.273181	M004	OFF		
2010-11-04	05:43:26.345503	M007	ON		
2010-11-04	05:43:26.793102	M004	ON		
2010-11-04	05:43:27.195347	M007	OFF		
2010-11-04	05:43:27.787437	M007	ON		
2010-11-04	05:43:29.711796	M005	ON		
2010-11-04	05:43:30.279021	M004	OFF	Bed_to_Toilet	end
2010-11-04	05:43:45.7324	M003	ON	Sleeping	begin
2010-11-04	05:43:52.044085	M003	OFF		
2010-11-04	05:43:53.185335	M002	ON		
2010-11-04	05:43:53.253809	M003	ON		
2010-11-04	05:43:59.493281	M002	OFF		
2010-11-04	05:44:04.048766	M003	OFF		
2010-11-04	05:44:06.14204	M003	ON		
2010-11-04	05:44:11.229146	M003	OFF		

Fig. 2. Sample raw and activity annotated sensor data.Sensors IDs starting with M are motion sensors and IDsstarting with D are door sensors.

2010-11-04	17:57:35.956698	M026	OFF		
2010-11-04	17:57:45.71581	M027	ON		
2010-11-04	17:57:46.296167	M026	ON		
2010-11-04	17:57:47.39778	M027	OFF	Work-end	
2010-11-05	00:00:11.187975	M003	ON	Sleeping-begin	

Fig. 3. Time dependency

2) Sensor Dependency method

As described earlier, when a window contains sensor events coming from two different activities, it is likely that the sensors that dominate the window do not really participate in the evaluation of the activity that induced the last event in the window. Such case is illustrated on Fig. 4.

2010-11-04	16:21:22.954497	M009	OFF		
2010-11-04	16:21:23.337588	M009	ON		
2010-11-04	16:21:27.492024	M009	OFF		
2010-11-04	16:21:53.261314	M009	ON		
2010-11-04	16:21:56.224795	M020	ON		
2010-11-04	16:21:57.076538	M013	ON		
2010-11-04	16:21:58.47367	M009	OFF	Relax	end
2010-11-04	16:21:58.868956	M014	ON		
2010-11-04	16:21:59.792013	M018	ON	Meal_Preparation	begin

Fig. 4. Sensor dependency

To reduce the impact of such sensor events on the description of the last sensor event, [4] uses a mutual information based measure between the sensors.

Mutual information measures how much one random variable tells us about another. In the current context, each individual sensor is considered to be a random variable that has two outcomes, namely 'ON' and 'OFF'. The mutual information or dependence between two sensors is then defined as the chance of these two sensors occurring consecutively in the entire sensor stream. If S_i and S_j are two sensors, then the mutual information between them, denoted MI (i, j), is defined as:

$$MI(\mathrm{i,j}) = \frac{1}{N}\sum_{k=1}^{N-1}\delta(S_k, S_i)\delta(S_{k+1}, S_j) \quad (1)$$

$$\text{s.t. } \delta(S_k, S_i) = \begin{cases} 0 & if\ S_k \neq S_i \\ 1 & if\ S_k = S_i \end{cases} \quad (2)$$

The term δ takes a value of 1 when the current sensor is S_i and the next sensor is S_j. The value of this mutual information is linked to the proximity of both sensor events.

The mutual information matrix is computed offline using the training sensor sequence. It is then used to add a weigh defining the influences of the sensor events in a window while constructing the feature vector. Each event in the window is weighted with respect to the last event in the window. Thus instead of counting the different sensor events, it is the sum of the contributions of every sensor event based on mutual information that defines the feature vector. The approach is denoted as Sensor Windows Mutual Information (SWMI) for future reference.

3) Sensor Dependency extension method

Mutual information of two sensors as previously described depends on the order of occurrence of a couple of sensors in the entire data stream. For instance, we can consider 4 sensors installed in a tight place of a smart home and that participate in the performance of a specific activity. Person can take the path that fires in the following order the sensors: $S_1 \rightarrow S_2 \rightarrow S_3 \rightarrow S_4$ or in a second way: $S_1 \rightarrow S_3 \rightarrow S_2 \rightarrow S_4$ to perform this activity. Assuming that the first path is statistically less used than the second path, but also that the two paths lead to the same activity, we can clearly see that there is a dependency between sensors S_1 and S_2 whatever path is used. If we adopt the previous way for computing the mutual information between sensors S_1 and S_2, we will lose some dependency quantity between these sensors.

Furthermore, there are activities that are often performed in parallel, and sensor events of an activity can be descriptive for the other and traditional mutual information cannot take into consideration this situation.

Based on these assumptions, we propose to compute mutual information between two sensors S_i and S_j by computing their frequency of occurrence in space of ***n*** sensor events along the entire data stream, as defined by the following equation:

$$MI(\mathrm{i,j}) = \frac{1}{K}\sum_{k=1}^{K-1}\{(S_i, S_j) \in [E_{k*n+1} \cdots E_{k*n+n}]\} \quad (3)$$

Such that ***K*** is the window, **n** is the number of events in the window that is selected empirically. Feature vector is then computed as the original method. We denote this approach as Sensor Windows Mutual Information extension (SWMIex) for future reference.

4) Last-state sensor method

In the window defined from the event sensor E_i, a sensor can be trigged several times. Sometimes, last state of the sensor, with respect to E_i, can be more descriptive than its

frequency of occurrence in a window. To interpret this idea, we propose to compute the feature vector X_i as follow: for each sensor S_i, if its last state within a window is ON/OFF then it will be represented by respectively 1/-1 in the feature vector X_i, otherwise it will be represented by 0 (if absent). We denote this approach as Sensor Windows Last State (SWLS) for future reference.

IV. Experiments and result

A. Dataset

To test the proposed techniques, we tried to select a dataset as close as possible to the dataset used in the study on which our work is built. For this, we chose Aruba and Tulum real-world datasets from the Washington State University CASAS smart-home project [20], which were obtained using PIR motion sensors and door sensors. The details of datasets appear in Table 1 and Table 2. The data is represented as a sequence of time stamped sensor data, as shown in Fig. 2.

TABLE 1. ARUBA DATASET DETAILS

Dataset	Gender & age	Number of Sensors	Time Interval
Aruba	Elderly/female	34	7 months
Tulum	2 married residents	16	4 months

TABLE II. ACTIVITIES AND STATISTICS

Aruba dataset		Tulum dataset	
Activity	Number of events	Activity	Number of events
Other activity	1 144 887	Other activity	85 915
Bed_to_Toilet	1 330	Cook_Breakfast	11 343
Eating	16 153	Cook_Lunch	5 350
Enter_Home	2 018	Enter_Home	11 998
Housekeeping	10 583	Group_Meeting	23 787
Leave_Home	1 922	Leave_Home	1 200
Meal_Preparation	293 334	R1_Eat_Breakfast	10 395
Relax	72 717	R1_Snack	216 178
Resperate	542	R2_Eat_Breakfast	12 312
Sleeping	32 682	Wash_Dishes	24 392
Wash_Dishes	10 464	Watch_TV	50 280
Work	16 321	-	-

"Other activity" class contains events with missing labels. It covers 71 % of the entire sensors events sequence in Aruba dataset. This class constitutes 19% of Tulum dataset. Due to the very large quantity of data to process with a normal computer (1.5 GHz machine with 8GB RAM), we used only the first six weeks of data in Aruba dataset and 3 months data in Tulum dataset. Statistics of this data are depicted in Table 3.

TABLE III. STATISTICS ON THE FIRST SIX WEEKS OF ARUBA DATASET.

Aruba dataset		Tulum dataset	
Activity	Number of events	Activity	Number of events
Other events (OE)	81735	Other events (OE)	76073
Bed_to_Toilet	442	Cook_Breakfast	11343
Eating	4495	Cook_Lunch	5350
Enter_Home	526	Enter_Home	11998
Housekeeping	7569	Group_Meeting	23787
Leave_Home	492	Leave_Home	1200
Meal_Preparation	31201	R1_Eat_Breakfast	10395
Relax	37291	R1_Snack	101425
Resperate	204	R2_Eat_Breakfast	6667
Sleeping	9310	Wash_Dishes	15049
Wash_Dishes	3795	Watch_TV	40572
Work	4714	-	-

To evaluate our methods, classification accuracy and F-measure (F-score) are used. The accuracy shows the percentage of correctly classified instances, the F-score shows the average percentage of correctly classified instances per classes.

B. Results and discussion

We conducted two sets of experiments to evaluate the effectiveness of the approaches presented in this paper.

In the first series of experiments, the system was trained using standard SVM once using a sensor events sequence containing the "other events" and once on data excluding "other events". We employed the LibSVM [19] library, and we determined the penalty parameter **C** and RBF kernel parameters via cross-validation on training data.

1) Learning on data excluding the 'other' activity

We begin our experiment by testing Baseline method with different number of events per window. We obtained the best performances (classification) with a number that is lower than the average number of sensor events that span the duration of the different activities. Then, we tested our proposed approaches SWMIex and SWLS. Table 4 shows a summary of the results of this experiment, while Table 5 shows more detailed results.

In Table 4, we can observe that the accuracy are close to each other when any events with missing labels were removed, while there is a significant improvement of the F-score. This is due to the bad affect of imbalanced dataset on the result, where an improvement of performance of the minority activities does not much increase the accuracy but it does in F-score.

By comparing Baseline approach with our methods, F-score increases respectively by 4%, 5% and 7% when using SWMI, SWMIex and SWLS approaches in Aruba dataset and by 5%, 7% and 1% in Tulum dataset.

Fig.5.(a) shows the F-score of each activity. Confusion Matrixes obtained by testing different methods show that there is an important confusion in recognition of activities that share the same functional area such kitchen activities in the datasets and activities that occur in bedroom ("Bed_to_Toilet", 'Sleeping') in Aruba dataset. The "Bed_to_Toilet" in Aruba dataset is often classified as a 'Sleeping' activity. Similarly, most instances of "Wash_Dishes" activity are classified as "Meal_Preparation", "Eating" or "Housekeeping" activity. As for the "Resperate" activity, classifier cannot identify any of the instances. The SWMI method improves the F-score of "Bed_to_Toilet" activity, while it continues to classify incorrectly both "Resperate" and "Wash_Dishes" activities. However, when using SWMIex and SWLS methods, classifier successfully identify respectively 19% and 14% of "Resperate" activity instances and 4% and 11% of "Wash_Dishes" activity.

Concerned Tulum dataset, Fig5 (b) shows that our method surpasses all methods in term of accuracy and F-score classification. The advantage provided by our SWLS method is that it could recognize "Wash_Dishes" activity at a time BaseLine and SWMI failed to identify it.

TABLE IV. LEARNING WITHOUT CONSIDERING "OTHER ACTIVITY" CLASS

Training without 'other events'				
Feature extraction Method	Aruba dataset		Tulum dataset	
	Acc	F-score	Acc	F-score
Baseline	87.23	63.29	63.46	35.60
SWMI	87.71	65.56	64.18	39.30
SWMIex	87.71	68.68	65.57	41.29
SWLS	87.55	69.24	63.95	36.91

2) Learning on data containing "other" activity

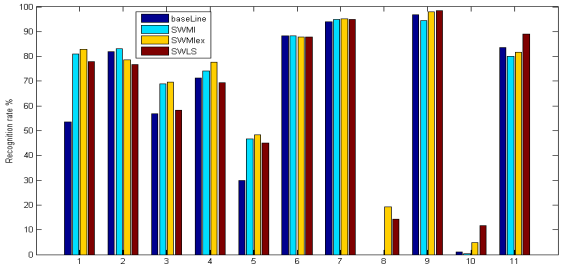
This second set of experiments aims to evaluate our methods on the data containing the "other" activity. This activity covers 45% of the original dataset. The results are presented in Table 7. The activity classification accuracy is defined by:

$$\text{Accuracy} = \sum_{m=1}^{nb_A} \frac{TP_{Am}}{N_{Am}} \qquad (4)$$

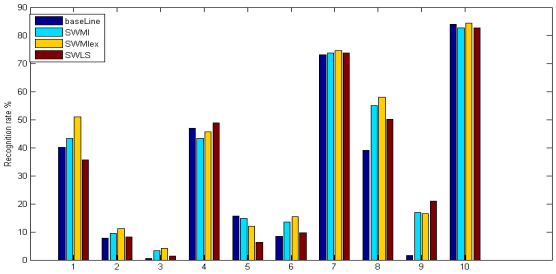
Such that nb_A is the total number of activities excluding the "other" activity.

By comparing the results obtained in the first series of experiments with that obtained by this series, classification accuracy drops significantly from 87% to 69% on Aruba dataset. However, in Tulum dataset the performance degradation is less severe.

By testing our methods on Aruba dataset, SWMI approach loses its superiority over the Baseline. Best results in term of classification accuracy are obtained by SWLS approach while SWMIex approach surpasses all methods when using F-score as metric. Fig.6 (a) shows that none of the methods is able to identify the "Resperate" and "Wash-Dishes" activities, due to the imbalanced problem intensified by the presence of "other" activity that dominates the dataset. We also observe that the system loses its capability to recognize the 'Housekeeping' activity, where most of instances are classed as "other activity" class. About Tulum dataset, the behavior of classifier does not change and our method SWMIex still surpass all other feature extraction methods.



(a) Aruba dataset [1: Bed_to_Toilet. 2: Eating. 3:Enter_Home.4:Housekeeping.5-Leave_Home. 6:Meal_Preparation. 7:Relax. 8:Resperate. 9:Sleeping. 10:Wash_Dishes. 11:Work. 12: other activity.]



(b) Tulum dataset [1-Cook_Breakfast 2-Cook_Lunch 3-Enter_Home 4-Group_Meeting. 5-Leave_Home 6-R1_Eat_Breakfast 7-R1_Snack 8-R2_Eat_Breakfast. 9-Wash_Dishes 10-Watch_TV]

Fig. 65. Learning without "other activity" class: F-score of activities.

The best results are obtained by using our proposed SWMIex approach, where there is a significant improvement in the recognition of the most activities over Baseline approach.

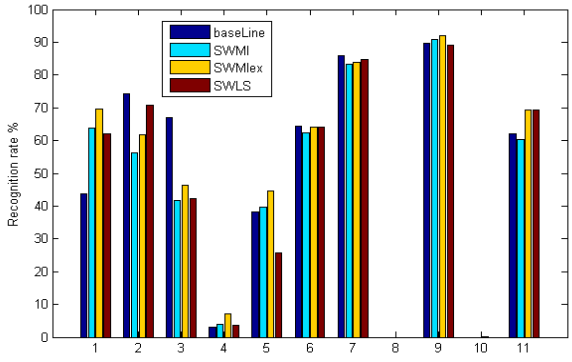
TABLE VI. LEARNING WITH "OTHER ACTIVITY" CLASS

Training with 'other events'				
Feature extraction Method	Aruba dataset		Tulum dataset	
	Acc	F-score	Acc	F-score
Baseline	67.82	49.52	63.32	35.75
SWMI	64.18	47.54	63.48	36.71
SWMIex	67.38	50.39	65.26	39.01
SWLS	69.09	47.38	63.90	34.81

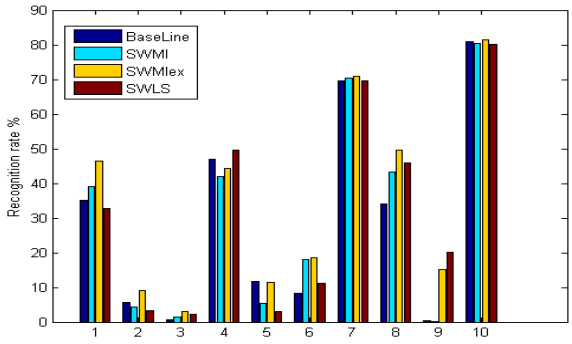
V. CONCLUSION AND FUTURE WORK

In order to provide an automated monitoring system for different human needs, an online system that is performing activity recognition from sensor readings is required. Most of the techniques used in the literature are not suitable to build an

online system. In this paper we propose and evaluate an extension of a sensor window approach to perform activity recognition in an online/streaming setting; recognizing activities when new sensor events are recorded.



a. Aruba dataset



b. Tumum dataset

Fig. 6. Learning on data containing "other activity" class: F-score of activities.

As different activities can be better characterized using different window length, mutual information based weighting of sensor events within a window is incorporated in this paper. A modification of how is computed the mutual information is proposed. To account for the fact that some activities do not require many movements to be performed, we propose a last-state of sensor feature set within the window to characterize activities. These techniques are evaluated on Aruba and datasets over six weeks and on 3 months of data. These techniques show an improvement over the Baseline technique when any events with missing labels were removed. Only one of these techniques shows a significant improvement over Baseline when we incorporate events with missing labels in data.

Results show that the impact of the proposed methods compared to Baseline is reduced when is taken into account "other activity" class to learning, especially on Aruba dataset. There are a large confusion between "other activity" class and the different known activities. Our future work will include finding a way to reduce confusion between this activity and the others.

REFERENCES

[1] D. Cook, M. Schmitter-Edgecombe, A. Crandall, C. Sanders, B. Thomas, Collecting and disseminating smart home sensor data in the casas project, in:

[2] S. Intille, K. Larson, E.M. Tapia, J. Beaudin, P. Kaushik, J. Nawyn, R. Rockinson, Using a live-in laboratory for ubiquitous computing research, Proceedings of Pervasive (2006) 349–365.

[3] M.E. Pollack, L.E. Brown, D. Colbry, C.E. McCarthy, C. Orosz, B. Peintner, S. Ramakrishnan, I. Tsamardinos, Autominder: an intelligent cognitive orthotic system for people with memory impairment, Robotics and Autonomous Systems 44 (2003) 273–282.

[4] Krishnan, N. & Cook, D. J. Activity recognition on streaming sensor data. Pervasive Mob. Comput. 10, 138–154 (2014).

[5] Duong T., Phung D., Bui H., Venkatesh S. Efficient duration and hierarchical modeling for human activity recognition. Artif. Intell. 2009;173:830–856.

[6] Patterson D.J., Fox D., Kautz H., Philipose M. Fine-Grained Activity Recognition by Aggregating Abstract Object Usage. Proceedings of the Ninth IEEE International Symposium on Wearable Computers; Osaka, Japan. 18–21 October 2005; pp. 44–51.

[7] Huynh T., Schiele B. Towards Less Supervision in Activity Recognition from Wearable Sensors. Proceedings of the 2006 10th IEEE International Symposium on Wearable Computers; Montreux, Switzerland. 11–14 October 2006; pp. 3–10.

[8] Corchado J.M., Bajo J., Tapia D.I., Abraham A. Using heterogeneous wireless sensor networks in a telemonitoring system for healthcare. Trans. Inf. Tech. Biomed. 2010;14:234–240.

[9] Tapia E.M., Intille S.S., Larson K. Activity recognition in the home using simple and ubiquitous sensors. Pervasive Comput. 2004;3001:158–175.

[10] Wilson D., Atkeson C. Simultaneous Tracking and Activity Recognition (STAR) Using Many Anonymous, Binary Sensors Pervasive Computing. Volume 3468. Springer; Berlin/Heidelberg, Germany: 2005. pp. 329–334.

[11] Andreas Bulling, Jamie A. Ward, Hans Gellersen, and Gerhard Tr¨oster. 2008. Robust Recognition of Reading Activity in Transit Using Wearable Electrooculography. In Proc. of the 6th International Conference on Pervasive Computing (Pervasive 2008). Springer, 19–37.

[12] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kr¨ose. 2008. Accurate activity recognition in a home setting. In Proc. UbiComp. 1–9.

[13] D. Patterson, D. Fox, H. Kautz, and M. Philipose. 2005. Fine-grained activity recognition by aggregating abstract object usage. In Proc. ISWC. 44–51.

[14] T. Huynh, U. Blanke, and B. Schiele. 2007. Scalable Recognition of Daily Activities with Wearable Sensors. In Proc. LoCa. 50–67.

[15] L. Bao and S. Intille. Activity recognition from user-annotated acceleration data.Lecture Notes in Computer Science, pages 1-17, 2004.

[16] J. Lester, T. Choudhury, N. Kern, G.Borriello, and B. Hannaford. 2005. A hybriddiscriminative/generative approach for modeling human activities. In Proc. 19th International JointConference on Artificial Intelligence. 766–772.

[17] T. Gu, Z. Wu, X. Tao, H. Pung, and J. Lu. epSICAR: An Emerging Patterns based approach to sequential, interleaved and Concurrent Activity Recognition. In Proceedings of the 2009 IEEE International Conference on Pervasive Computing and Communications-Volume 00, pages 1-9. IEEE Computer Society, 2009.

[18] L. Wang, T. Gu, X. Tao, J. Lu, A hierarhical approach to real-time activity recognition in body sensor networks, Journal of Pervasive and Mobile Computing (2011).

[19] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (2011) 1–27.

[20] Aruba, Tulum Datasets from WSU CASAS smart home project. http://ailab.wsu.edu/casas/datasets/.