Activity Recognition in Smart Home Environments using Conditional Random Fields and Hidden Markov Models

Caezarina Calimbahin
Department of Computer Science
College of Engineering
University of the Philippines - Diliman
Email: cfcalimbahin@up.edu.ph

Isabelle Tingzon
Department of Computer Science
College of Engineering
University of the Philippines - Diliman
Email: ibtingzon@upd.edu.ph

***Abstract*—Health monitoring is an integral part of maintaining and improving the health status of the geriatric population. Wireless sensor networks deployed in home environments are a key technology that can help assist the elderly in performing their daily activities. Activity recognition using sensor data as observations is an important component in automated health monitoring. In this study, we employ CRF and HMM for activity recognition in smart home environments and compare their respective performances to those of other machine leaning techniques such as SVM, decision trees, and Gaussian Naive Bayes classifier. Based on experimental results, we show that CRF provides high performance measures, thus proving its effectiveness in classifying activities of daily living.**

***Keywords*—Activity Recognition, Smart Environments, Wireless Sensor Networks, Machine Learning**

I. Introduction

Recently, there has been a growing interest in automating health monitoring systems. This technology is used to monitor the activities of daily living (ADLs) of individuals in need of health assistance such as elderly and disabled persons and to facilitate them in performing daily tasks. Essential to health monitoring is activity recognition, which is the recognition of actions based on a set of observation (e.g. sensor data). Activity recognition in the healthcare setting aims to help healthcare providers better assess physical and cognitive well being of their patients. In enabling health monitoring, wireless sensor networks (WSN) are proving to be a promising technology due to ease of installation and minimal intrusion [2].

Several studies have been conducted on activity recognition using sensor data. Our work is most similar to the studies conducted by Ordonez et al. [2], which uses a hybrid dicriminative/generative model for classifying human activities in smart home environments, and Kastaren et al. [8] which uses temporal probabilistic models such as Hidden Markov Models (HMM) and Conditional Random Fields (CRF) for activity recognition for health monitoring. Nazerfard et al. [3] also use CRF for activity recognition in smart environments.

In our study, we use CRF and HMM for activity recognition in smart home environments, and compare their respective performances to those of other machine leaning techniques such as SVM, decision trees, and Gaussian Naive Bayes classifier. The remaining content of this paper is organized as follows: in section 2, we give a detailed description of the sensor data, including segmentation and feature extraction methods. In sections 3 and 4, we discuss CRF and HMM respectively. In section 5, we discuss the other machine learning methods we will use for comparison. In section 6, we will discuss the experimental setup, including the datasets used, the possible feature representations, the compuatation of the performance measure, and validation. We then present the experimental results and discussion in section 7. Finally in section 8, we conclude the paper.

II. Sensor Data

Sensor data is composed of temporal data generated by a set of binary sensors installed in a home environment. The WSNs deployed in the two different environments are composed of the following: passive infrared sensors for motion detection in certain areas; reed switches to detect open/close states of doors and cupboards; and float sensors to detect the toilet being flushed [2]. An overview of the dataset as decribed by Ordonez et al. can be seen in Table 1.

TABLE I. Home settings descriptions as described in [2]

	OrdonezA	OrdonezB
Setting	House	House
Rooms	4	5
Duration	14 days	21 days
Sensors	12	12

Based on the contributions in [2], we discretize the sensor stream data (i.e. timeline) into time slices of constant length Δt. Each time slice can be associated with a set of features that can be used to classify the activity performed in the said time segment. A sensor event in time slice t is denoted as x_t^i indicating whether sensor i fired at least once between times t and $(t + \Delta t)$, with $x_t^i \in \{0, 1\}$. For a home setting with n sensors, a binary observation vector $(x_t^1, x_t^2, ..., x_t^n)$ can be defined for each time slice.

Apart from the actual sensor events, the time of day and day of the week that the events occur are also considered valuable features. The time of each sensor event is a continuous value and must be discretized as was done in [3]. The values are

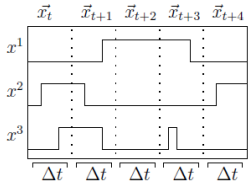


Fig. 1. Temporal segmentation and relation between sensor outputs x^i and time segments Δt based on [2]

binned into the following hour ranges: 0-5 (early morning), 5-10 (morning), 10-15 (afternoon), 15-20 (evening), 20-24 (late evening). Each bin can then be assigned a corresponding unique identifier. Similarly, each day of the week can be represented as a unique identifier. Thus, combining these features with the binary observation vector, we obtain an observation vector $\vec{x}_t = (d_t, u_t, x_t^1, x_t^2, ..., x_t^n)$ for each time slice, where d_t is the corresponding day of time slice t and u_t is the bin representing the time of the day of time slice t.

Each time slice corresponds to a single data instance, based on [2]. The class label of each data instance is defined by the activity label of the corresponding time slice. The activity in time slice t is denoted by $y_t \in \{1, ..., m\}$ for m possible states. Thus, the sequential predicition task is to find the appropriate mapping between a sequence of observations $\mathbf{x} = \{\vec{x}_{t1}, \vec{x}_{t2}, ..., \vec{x}_T\}$ and a sequence of activity labels $\mathbf{y} = \{y_{t1}, y_{t2}, ..., y_T\}$ for a T time slices.

Note that another important feature in activity recognition is the previous activity y_{t-1}, or the activity that occurred immediately before the current activity y_t [3]. However, most machine learning methods cannot implicitly handle dependencies between states without additional feature engineering. Classifiers such as SVM, ANN, and decision trees treat each state independently of other states. CRF, on the other hand, differ from most machine learning methods in that they are able to capture state-to-state dependencies in the model. In the next section, will see that CRF specialize in handling data of sequential nature seamlessly.

III. Conditional Random Fields

Conditional random fields (CRF) are discriminative undirected probabilistic graphical models that find the posterior probability $P(\mathbf{y}|\mathbf{x})$ for a given sequence of observations $\mathbf{x}$ and a sequence of class labels $\mathbf{y}$. That is, CRF use log-linear models to encode a distribution over label sequences given some observation sequence. CRF can be seen as a supervised learning approach to predicting class labels $\mathbf{y}$ based on observations $\mathbf{x}$. However, as previously mentioned, unlike most classifiers, CRF are implicitly capable of modelling state-to-state dependencies. Because of this, CRF are becoming a more popular method for natural language processing and sequence prediction tasks.

Lafferty et al. [4] described CRF as a series of state feature functions $\mathbb{s}(y_t, \mathbf{x}, t)$ with corresponding weights λ a transition feature function of the form $\mathbb{t}(y_{t-1}, y_t, \mathbf{x}, t)$ with corresponding weights μ.The conditional probability $P(\mathbf{y}|\mathbf{x})$ can be expressed as:

$$P(\mathbf{y}|\mathbf{x}) \propto \Big(\sum_j \lambda_j \mathbb{s}_j(y_t, \mathbf{x}, t) + \sum_k \mu_k \mathbb{t}_k(y_{t-1}, y_t, \mathbf{x}, t) \Big)$$

Collapsing the state feature functions and transition feature function together into one notation style for all feature functions:

$$f(\mathbf{y}, \mathbf{x}, t) = \begin{cases} \mathbb{s}(y_t, \mathbf{x}, t), & \text{if } f \text{ is a state function} \\ \mathbb{t}(y_{t-1}, y_t, \mathbf{x}, t), & \text{if } f \text{ is a transition function} \end{cases}$$

We merge values of associated μ and λ into a single weight vector λ. Features from $\mathbf{x}$ and label sequence $\mathbf{y}$ can now be represented as a feature vector:

$$F_j(\mathbf{y}, \mathbf{x}) = \sum_{t=1}^{T} f_j(\mathbf{y}, \mathbf{x}, t) \tag{1}$$

where T is the total number of labels. The log-linear model can thus be defined as

$$P(\mathbf{y}|\mathbf{x}; \lambda) = \frac{1}{Z(\mathbf{x}; \lambda)} \exp \sum_j \lambda_j F_j(\mathbf{y}, \mathbf{x})$$

where Z is the normalization factor.

$$Z(\mathbf{x}; \lambda) = \sum_{\mathbf{y}} \exp \sum_j \lambda_j F_j(\mathbf{y}, \mathbf{x})$$

The task is to find $\mathbf{y}^*$ that maximizes the conditional probability

$$\mathbf{y}^* = \arg\max_y P(\mathbf{y}|\mathbf{x}) = \arg\max_y \exp \sum_j \lambda_j F_j(\mathbf{y}, \mathbf{x})$$

In this study, we employ the CRF++ toolkit [9], an open-source implementation of conditional random fields, for the construction of CRF. The toolkit provides a number of attractive features, including fast training based on L-BFGS and less memory usage both in training and testing.

IV. Hidden Markov Models

Hidden Markov Models (HMMs) are popular generative temporal probabilistic models. Generative models fully define the joint probability $p(y_{1:T}, x_{1:T})$ and can be used to generate data from this distribution, or to perform inference given a sequence of observations [8], the latter being the objective of this study.

The two dependence assumptions that define HMMs are:

- The hidden variable at time t, namely y_t, depends only on the previous hidden variable y_{t-1}.
- The observable variable at time t, namely x_t, depends only on the hidden variable y_t at that time slice.

The joint probability distribution factorizes as follows

$$p(\mathbf{y}_{1:T}, \mathbf{x}_{1:T}) = p(y_1) \prod_{t=1}^{T} p(x_t|y_t) \prod_{t=2}^{T} p(y_t|y_{t-1})$$

The initial distribution $p(y_1)$ represents the probability of starting in state y_1, the observation distribution $p(x_t|y_t)$ represents the probability that the state y_t would generate observation x_t, and the transition distribution $p(y_t|y_{t-1})$ represents the probability of going from one state to the next [8].

HMM can be effectively used for recognizing human activities, however, modeling the observation probabilities when observable variables are a collection of binary values (i.e from each sensor) can reach a high degree of complexity. To exactly model the distribution of the observation vector, all possible combinations of values in the feature space have to be considered, resulting in a large number of parameters and requiring accordingly large numbers of training elements [2].

In this study, we employ Matlab's HMM tool which is under the Statistics and Machine Learning Toolbox [11]. For training, it uses Baum-Welch algorithm to estimate the transition and observation probabilities while for testing, it uses Viterbi algorithm to compute for the most likely sequence of states the model would go through to generate a given sequence seq of observations.

V. Other Machine Learning Methods

We also employ three other machine learning methods, namely: Support Vector Machines (SVM), Gaussian Naive Bayes classifier, and decision trees (DT), for comparison with CRF and HMM. This section presents a brief description of the classifiers.

1) Support Vector Machines: Support vector machines are known to be among the most robust of classification algorithms [5]. SVM involve mapping input vectors to higher dimensional feature space where an optimal hyperplane can be computed by finding the maximal margin. In this study, we used the open-source machine learning tool in Python scikit-learn [10] for the construction of SVMs.

2) Decision Trees: Decision trees were also chosen as a preference for classification due to their natural capability to handle multi-class classification problems with ease. The most widely known algorithms for building decision trees include ID3/C4.5/C5.0, and Classification and Regression Trees (CART) [7]. Decision trees are constructed using features and threshold that yield the largest information gain at each node [6], [7]. New data is tested by starting at the root node and following down to the leaf nodes, and at internal nodes a decision is performed based on a certain extracted feature. The overall objective of constructing a decision tree is to produce a good generalization of the data. For activity recognition, we constructed decision trees with the aid of scikit-learn [10], which uses an optimized version of the CART algorithm.

3) Gaussian Naive Bayes: The Gaussian Naive Bayes classifier which computes the likelihood of features using the following formula:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(\frac{x_i - \mu_y}{2\pi\sigma_y^2}\right)$$

where σ_y and μ_u are estimated using maximum likelihood. Similar to DT, the Naive Bayes classifier was employed with the aid of the open source machine learning tool in Python, scikit-learn [10].

TABLE II. Percentage of instances per class for each dataset as described in [2].

Activity	OrdonezA	OrdonezB
Leaving	8.32	17.41
Toileting	0.76	0.55
Showering	0.54	0.24
Sleeping	39.1	35.58
Breakfast	0.63	1.02
Dinner	0	0.38
Idle	5.61	11.73
Lunch	1.59	1.30
Snack	0.05	1.33
Spare Time/TV	42.7	29.98
Grooming	0.73	1.42

VI. Experimental Setup

A. Datasets

In this work, we have employed two separate datasets originally generated by Ordonez et al. [2] for a study on activity recognition in home environments using hybrid generative/discriminative models. The datasets are comprised of the ADLs performed by two different users (one dataset per user) in their respective homes for a certain number of days. Each dataset is composed of binary temporal data generated by a set of simple state-change sensors installed in two different home environments. The datasets are publicly available for download from the University of California - Irvine (UCI) Machine Learning Repository [1].

In both datasets, ten different ADLs were included as labels, namely: "Leaving", "Toileting", "Showering", "Sleeping", "Breakfast", "Lunch", "Dinner", "Snack", "Spare Time/TV", "Grooming". Time slices with no corresponding activity label are referred to as "Idle". Table 2 shows the percentage of instances per class for each dataset as described in [2]. As mention in a previous section, the sensor data stream were divided into time slices of length $\Delta t = 60$ seconds, based on [2], [8]. After segmentation, there were a total of 20,455 time slices for dataset "OrdonezA" and 30,469 time slices for dataset "OrdonezB". In this experiment, we also compare performance of using both unlabeled and labeled data to using only labeled data as was done in [8]. After removing unlabelled time slices (i.e. time slices with activity label "Idle"), we have 19,629 time slices for dataset "OrdonezA" and 27,235 time slices for dataset "OrdonezB".

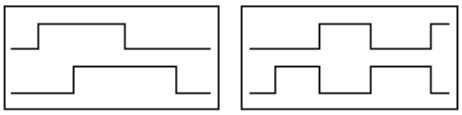
B. Feature Representation

The raw data streams produced by the sensors can be processed into different representation forms [2]. In this experiment, sensors streams are employed using two different representations:

- Raw: The raw sensor representation produces sensor data as it was received from the WSN, i.e. value is 1 if sensor is active, else 0.
- Last Sensor: The last sensor representation produces sensor data indicating which sensor fired last. The sensor that changed its state last continues to produce a

1 and only changes to 0 when another different sensor fires.

Fig. 2. Feature representations. (a) Raw; (b) Last Sensor. based on [2]



C. Performance Measure

Based on the confusion matrix in Fig. 3, we can obtain the performance measures to evaluate the effectiveness of the classifiers.

Fig. 3. Confusion Matrix showing true positives (TP), total true labels (TT) and total inferred labels (TI) for each class based on [2].

	Inferred			
True	**1**	**2**	**3**	
1	TP_1	ϵ_{12}	ϵ_{13}	TT_1
2	ϵ_{21}	TP_2	ϵ_{23}	TT_2
3	ϵ_{31}	ϵ_{32}	TP_3	TT_3
	TI_1	TI_2	TI_3	*Total*

The usual choice of performance measure is in machine learning tasks is *accuracy*, which can be computed as follows:

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} TP_i}{Total}$$

where N is the number of possible activity labels.

However, based on Tabled 2, there is a severe class imbalance in the dataset, i.e. activities "Sleeping" and "Spare time/TV" dominate the datasets while equally important activities like "Toileting" and "Showering" only take up a very small percentage of the datasets. Therefore, accuracy may not be the most suitable performance measure since a trivial classifier that predicts every instance of the majority class correctly could achieve very high accuracy [2]. Since the classification of each class is equally important, we evaluate the models using F-Measure instead as was done in [2], [8].

The F-Measure can be calculated using the confusion matrix in Figure 3 as follows:

$$\text{F-Measure} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{2}$$

where precision and recall are computed as follows:

$$\text{precision} = \frac{1}{N} \sum_{i=1}^{N} \frac{\text{TP}_i}{\text{TI}_i}$$

$$\text{recall} = \frac{1}{N} \sum_{i=1}^{N} \frac{\text{TP}_i}{\text{TT}_i}$$

D. Validation

The models were validated using a "leave-one-day-out" approach as was done in [2], [8]. This approach takes one full day of sensor readings as the test set and uses the remaining sensor readings as the training set. This process is repeated for all the days and the average F-Measure is reported over the total number of days, i.e. dataset "OrdonezA" reports the average F-Measure over a period of 14 days, and dataset "OrdonezB" reports the average F-Measure over a period of 21 days (see Table 1).

VII. Results and Discussion

We compare the performance of CRF and HMM with other machine learning methods: SVM, Decision Trees, and Gaussian Naive Bayes classifier. We also compare the results for raw sensor representation and last sensor representation. Lastly, we compare the performance of using both unlabelled and labelled data to using only labelled data. The results are shown in Tables 3-6.

TABLE III. Experimental results for dataset OrdonezA using labelled and unlabelled data (expressed in %)

	SVM	Decision Trees	Gaussian Naive Bayes	CRF	HMM
Raw					
F-Measure	66.15	63.62	64.37	71.22	54.49
Accuracy	92.94	88.83	93.58	91.95	93.66
Last Sensor					
F-Measure	58.41	58.28	53.94	58.94	56.66
Accuracy	94.52	91.62	71.85	92.03	95.98

TABLE IV. Experimental results for dataset OrdonezA using labelled data only (expressed in %).

	SVM	Decision Trees	Gaussian Naive Bayes	CRF	HMM
Raw					
F-Measure	74.33	73.23	74.85	82.83	61.15
Accuracy	97.52	94.60	97.57	96.46	97.48
Last Sensor					
F-Measure	74.21	71.90	65.04	79.34	64.39
Accuracy	98.09	96.71	74.80	98.03	98.29

TABLE V. Experimental results for dataset OrdonezB using labelled and unlabelled data (expressed in %).

	SVM	Decision Trees	Gaussian Nave Bayes	CRF	HMM
Raw					
F-Measure	58.98	54.83	33.38	66.55	49.18
Accuracy	81.21	75.02	53.25	87.28	83.55
Last Sensor					
F-Measure	64.75	65.25	47.62	64.21	49.57
Accuracy	87.90	87.43	71.41	87.91	86.20

Based on the overall F-Measure results, CRFs are found to outperform all other machine learning methods for both

TABLE VI. EXPERIMENTAL RESULTS FOR DATASET ORDONEZB USING LABELLED DATA ONLY (EXPRESSED IN %).

	SVM	Decision Trees	Gaussian Nave Bayes	CRF	HMM
Raw					
F-Measure	63.38	61.41	35.37	73.84	49.03
Accuracy	89.21	83.21	58.58	92.33	88.01
Last Sensor					
F-Measure	69.50	67.83	55.36	72.74	51.77
Accuracy	94.44	93.33	80.77	94.93	89.39

datasets "OrdonezA" and "OrdonezB". In particular, the highest F-Measure for the "OrdonezA" dataset is found to be 82.83% using labelled data in raw sensor representation.

For dataset "OrdonezA", the raw sensor representation produces better results than the last sensor representation for all classifiers. On the other hand, for the "OrdonezB" dataset, the last sensor representation produces better results for all classifiers except CRF.

We also note that higher performance measures are achieved for datasets that use only labelled data than datasets that use both labelled and unlabelled data. This implies that the unpredictable nature of unlabelled data may have a negative effect on the performance of the models.

VIII. CONCLUSION

In this study we use Conditional Random Fields (CRF) and Hidden Markov Models (HMM) for activity recognition in smart home environments using state-change sensor streams. In particular, we examine the effectiveness of CRF and HMM in recognizing activities of daily of living using different feature representations and compare their performances to machine learning methods such as SVM, decision trees and Gaussian Naive Bayes classifier. Based on experimental results, we conclude that CRF is effective in classifying activities of daily living based on binary sensor stream data.

REFERENCES

[1] UCI Machine Learning Repository: Activities of Daily Living (ADLs) Recognition Using Binary Sensors Data Set. Avaliable Online: https://archive.ics.uci.edu/ml/machine-learning-databases/00271/ Accessed: 24 Nov. 2016.

[2] Ordonez, Fco Javier, Paula de Toledo, and Araceli Sanchis. "Activity recognition using hybrid generative/discriminative models on home environments using binary sensors." *Sensors* 13.5 (2013): 5460-5477.

[3] Nazerfard, Ehsan, *et al*. "Conditional random fields for activity recognition in smart environments." *Proceedings of the 1st ACM International Health Informatics Symposium*. ACM, 2010.

[4] Lafferty, John, Andrew McCallum, and Fernando Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data." *Proceedings of the Eighteenth International Conference on Machine Learning, ICML*. Vol. 1. 2001.

[5] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273297, 1995.

[6] Mohamed Aly. Survey on multiclass classification methods. 2005.

[7] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. Classification and regression trees. CRC press, 1984.

[8] Kasteren, T. L. M. "Activity recognition for health monitoring elderly using temporal probabilistic models." (2011).

[9] Tako, Kudo. 2005. "CRF++: Yet Another CRF toolkit." Taku910.github.io. 14 Mar. 2015. Web. 22 Oct. 2016. Available Online: https://taku910.github.io/crfpp/#templ. Accessed 26 Nov 2016.

[10] Scikit-learn: Machine Learning in Python, Pedregosa *et al*., *Journal of Machine Learning Research, JMLR*. 12, pp. 2825-2830, 2011. Available Online: http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html. Accessed: 26 Nov 2016.

[11] "Statistics and Machine Learning Toolbox" http://www.mathworks.com/help/stats/hidden-markov-models-hmm.html. Accessed: 30 Nov 2016.