HomeActivity: Recognizing home activities using sensor data

Stephen Lee, Patrick Pegus and Dong Chen
ML Final Project

December 19, 2015

1 Abstract:

In this project, we user sensors deployed in homes to identify activities. The actuation of sensors provides a unique set of feature representation that allows us to identity the type of activity being undertaken by the user. The goal of our project is to use these sensor information to infer activities under different methodologies and analyze their performance. We use datasets from three different houses that are annotated manually to get the 'ground truth'. To capture the underlying model, we use different feature representations from the sensors. We present our results and discuss our findings.

2 Introduction

According to a recent study, it is estimated that almost 25+ billion Internet of Things (IoT) devices will be deployed across homes. Moreover, around 50 trillion GBs of data will be collected using these sensors. The information collected can be used to enrich the interaction between users and the physical devices. For example, NEST thermostat, an intelligent programmable thermostat, learns the occupancy of humans in a home using sensors to automatically turn on/off heating or air conditioning and save energy costs. These sensor information can be used to learn other information such as monitoring the activities of a person (e.g. elderly home) and being able to detect anomalies in behavior over time. In this project, we focus on recognizing the activities of a person using the sensor information available. i.e. We study different techniques to identify when the person is sleeping, eating etc.

Activity recognition has been a widely studied area and provides a set of challenges in itself. First, ambiguity exists if there are more than one activity happening concurrently at any given time, such as watching TV and eating a snack. Also, this muddles the distinction between the start and end of an activity. Second, activation of a sensor may represent doing a similar activity. For example, opening the fridge could represent 'getting a snack' activity or 'cooking' activity'. Third, the information could be noisy e.g. with misfiring sensors or a mistakes made by humans such as unintentionally opening a cupboard or entering a room. Fourth, recognizing activities has class imbalance as certain activity labels tend to be longer than others. For example, lying on bed or couch, staying idle, or 'not at home' could easily represent the dominant classes in these datasets.

Clearly, activity recognition is a challenging problem. Thus, we chose a dataset wherein the homes was occupied by a single user. We construct features using the binary sensors installed in these homes that we will describe later in section. In fact, we construct multiple feature representations to study the problem and follow closely the approaches studied by Kasteren [7] among others. In the following section, we will be describing our approach, and discuss its performance and lessons learned from undertaking this project.

3 Related Works

Home occupant activities recognition problem has been studying for several years. In [5],Tapia et al. deployed low-cost sensors to monitoring occupant activities at home. As reported, using these sensors information, they can detect events in toilet and bashroom in real residencial places with accuracy ranging from 25 89%. In [4], Patterson et al. collected object data using a glove outfitted with a Radio Frequency Identification (RFID) antenna. Then, they used three models, including training HMMs separately, coupling HMMs into a one large system and using execution history of those objects, to recognize object-interaction based activity in a more realistic setting. And they found that utilizing object history could gains the best accuracy-81.2%. In [7], Kasteren et al. designed easy-to-install wireless sensor network (WSN) and a low-cost but accrete annotation method to automatically recognize activities.Raw, Changepoint, and Last methods are employed to represent sensor information. And utilizing HMM and CRF models for activities recognizing are studied as well. In [1], Duong et al. argue that instead of only using duration pattern, it would be more beneficial to combine inherent hierarchical structures. A switching hidden semi-Markov model is used to empirically compare performance with traditional models. Interestingly, the experiments are tested on a tracking missing and activities overlapping dataset. In [2], Sebastian et al. point out the Hierarchical Hidden Markov Model can capture the natural hierarchy information presentation in home activities when generating models. As shown, they are able to learn two simple activity sequences and capture the hierarchical structure represented in the data. In [3], Oliver et al. use a Layered Hidden Markov Models to infer user activity status from online stream data (e.g. video, autio, keyboard and mouse clicks). Using this cascade of HMMs, they can do sensing, training and predicting at different office data granularity. In [**?**], Subramanya et al. design a hierarchical model in order to monitoring the motion status and context status for a real person. As shown in the work, breaking a big complex activity into smaller sub-activities to build models is very useful for recognizing person activities. Our project follows closely the approaches presented by Kasteren in work [7] others. We describe the work undertaken and highlight our findings, issues and limitations that are still required to be overcome in this report.

4 Approach

4.1 Feature Representation

We describe the feature representation we use for classification purpose:

1. Raw: This uses the binary sensors readings directly. For each time step t, it takes the value of 1 or 0 depending on whether the sensor is actuated.
2. Change: The feature contains the sensor value as 1 only when there is a change in the state.
3. Last: The feature continues to have the sensor value as 1 until another sensor changes its state.
4. Last + Change: This feature combines the Last+Change representations.

4.2 Models

We explore and implement the following techniques to recognize activities:

4.2.1 Naive Bayes

Naive Bayes model is a simple probabilistic model that assumes independence between the features. Under the conditional independence assumption, and using the Bayes rule, the conditional probability can be represented as :

$$P(y_t \mid x_{t1}, \ldots, x_{tn}) \propto P(y) \prod_{i=1}^{n} P(x_{ti} \mid y_t) \tag{1}$$

where y_t is the activity label, and x_t is the vector of sensor readings at time t. The model can then be used to classify the labels as follows,

$$\hat{y_t} = \arg\max_{y_t} P(y) \prod_{i=1}^{n} P(x_{ti} \mid y_t), \tag{2}$$

In our approach we assume that the distribution of $P(x_{ti} \mid y_t)$ is modeled by independent bernoulli distribution, given by

$$p(x_t \mid y_t = i) = {\mu_{ni}}^{x_t} (1 - \mu_{ni})^{1-x_t} \tag{3}$$

4.2.2 Hidden Markov Models (HMM)

HMM is a generative probabilistic model that models the joint distribution of both the observed and the latent states. It is commonly used in recognizing temporal patterns such as handwriting and speech recognition, where the future state is dependent on the previous state. In our project, the latent variable($\mathbf{y}$) is the activity performed by the user, and the observed variable($\mathbf{x}$) is the vector of sensor readings.

At each time step t, the latent variable y_t depends only on the previous hidden variable y_{t-1} and variables before $t-1$have no influence on it (Markov Property). The observed variable x_t, at time t, depends only on the latent variable y_t. The parameters of the HMM model are the transition probabilities $p(y_t \mid y_{t-1})$ i.e. the probability of going from one state to another; and the emission probability $p(x_t \mid y_t)$ i.e. the probability that the state y_t would generate observation x_t. In order to learn the parameters of the distribution, one can maximize the joint probability $P(x, y)$ of the observed and latent sequences in the training data. The joint probability can be factorized as follows:

$$P(x, y) = \prod_{t=1}^{T} p(y_t \mid y_{t-1}) p(x_t \mid y_t), p(y_1) = p(y_1 \mid y_0) \tag{4}$$

Since the data is discrete in nature, frequency counting can be used to learn the parameters. We use Viterbi algorithm to infer the sequence of activity labels from the observed sensor reading sequences.

4.3 Conditional Random Fields (CRF)

We use linear-chain CRF to represent the latent and observed variables as it closely resembles the HMM in terms of structure. Since, CRF is a discriminative probabilistic model, we can model the conditional probability distribution $P(y \mid x)$, to predict y from x. Thus, the parameters are learnt by maximizing the following conditional probability distribution $P(y \mid x)$,

$$p(y \mid x) = \frac{1}{Z(x)} exp \left[\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t) \right] \tag{5}$$

where K is the number of feature functions used to parameterize the distribution, λ_k is the weight parameter and $f_k(y_t, y_{t-1}, x_t)$ is a feature function. The product of the parameters and the feature function $\lambda_k f_k(y_t, y_{t-1}, x_t)$ is called the energy function, while the exponential representation is the potential function. The partition function *Z(x)* is a normalization constant that sums over all the potential functions and is given as follows:

$$Z(x) = \sum_{y} exp \left[\sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x_t) \right] \tag{6}$$

In our approach, we use the BFGS method to learn the parameters of the model.

4.3.1 Support Vector Machines (SVM)

SVM is a discriminative classifier that builds a hyperplane that can be used for classification or regression purposes. It constructs this hyperplane such that the distance between the hyperplane and the nearest point is maximized. In particular, the problem can be seen as minimizing a loss function that can be represented as following:

$$\min_{w,\beta} L(w) = \frac{1}{2}||w||^2$$
$$\text{subject to } y_i(w^T x_i + \beta) \geq 1 \; \forall i,$$

where x_i are the training examples, y_i represent the labels, w represents the weight vector and β is the bias. In our approach, we use a Linear SVM for classification.

4.3.2 Structured Support Vector Machines (SSVM)

Structured SVM, as the name suggests, makes use of the structure of the output space for classification purposes. It is generally used for classification where the goal is to predict a sequence as compared to a single label in SVM. The loss function is represented by

$$y^* = \arg\max_{y \in Y} g(x, y) \tag{7}$$

where x is the input, Y is the set of all possible output and g is the loss function given by,

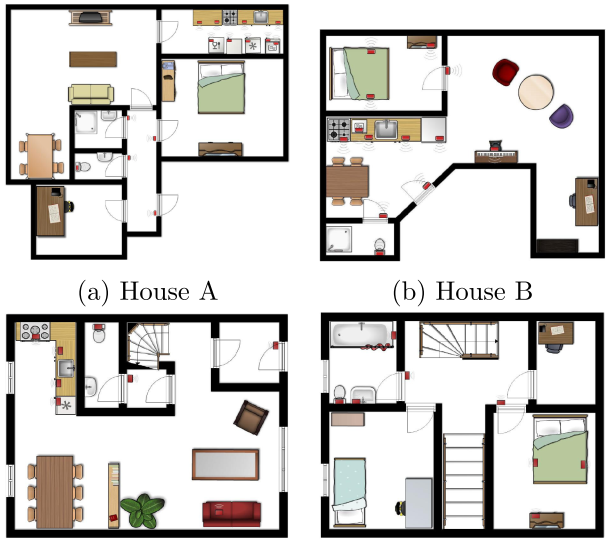
$$g(x, y) = w^T f(x, y) \tag{8}$$

Here, the f is feature function, and we use the linear chain CRF model to represent the feature function i.e. the linear combination of the feature potential(nodes) and the transition potential (edges). The parameters of $g(x, y)$ is learnt by minimizing a loss. We use the libraries provided in *pystruct* for classifying the input variables.

5 Experimental Evaluation

5.1 Datasets

Table 1, shows the summary of the dataset we use for our evaluation. We evaluate our models on three datasets provided by [6], each home having sensors deployed. Figure 1 displays the floor plans and sensor locations of the three residences. Each home is a singe occupant home, so there is no concurrent conflicting activities. The activities are documented manually using bluetooth/diary by the occupant, and is used as 'ground truth' data. The data is collected for a period of almost 24 days with at least 2000 sensor events and 245 activity instances.



(a) House A (b) House B

(c) House C first floor (d) House C second floor

Figure 1: Home floor plans and sensor locations.

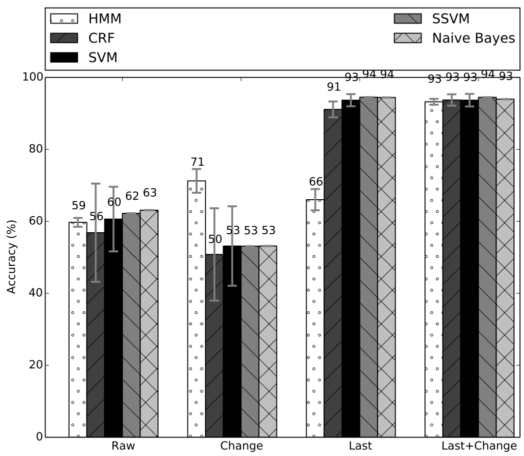
Type	*House A*	*House B*	*House C*
Age	26	28	27
Gender	M	M	M
Setting	Apartment	Apartment	House
Room	3	2	6
Duration(days)	25	14	19
Sensors	14	23	21
Activities	10	13	16
Annotation	Bluetooth	Diary	Bluetooh

Table 1: Dataset description

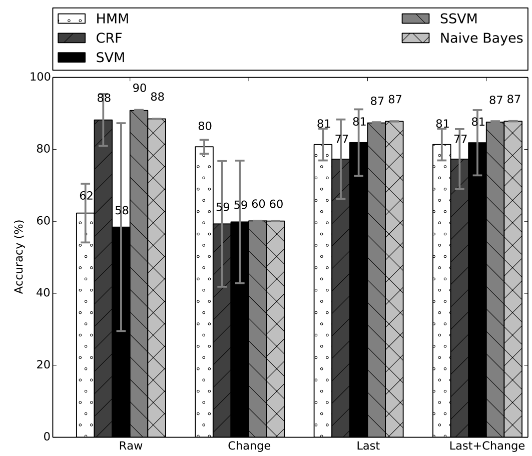
5.2 Discussion

We use 5-fold cross validation to evaluate different methods. We use the timeslice duration of 1 sec for all our evaluation. Figure 2, compares the accuracy of the different models. We observe that the *last* and *last+change* feature representation has the highest accuracy compared to *raw* and *change* feature representation in House A and House C. We note that combining *last* and *change* feature set has similar or better accuracy compared to other feature representations. However, in House B the *raw* representations performs better using Naive Bayes. We believe that the amount of training data available for House B affected the learning of the parameters.

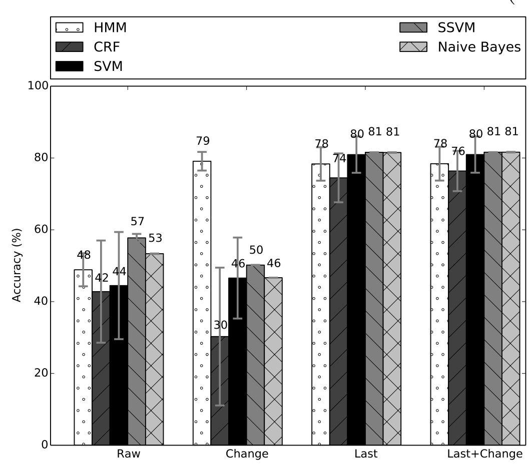
Even though, Naive Bayes is a simple approach that assumes independence between



(a) House A



(b) House B



(c) House C

Figure 2: Method comparison for the houses.

the features, we note that it performs better than among methods. This may be due to the high frequency of 1 sec or the property of the underlying dataset, where in the time-slice is independent of one another. We also observe that the Structured SVM performs similar or slightly than Linear SVM. This is because the structured SVM takes into account the underlying structure of the output space compared to the simplistic approach taken in Linear SVM. While we expected HMM and CRF to perform better than other models, as it explicitly captures the temporal relations, our results show that simple approaches can also work better.

Table 2 shows the performance metrics of House A. Since, there is class imbalance, precision and recalls provides a better insight into the quality of the classification. Again, we note that the simple Naive Bayes approach provides the highest precision and recall score.This means that the Naive Bayes approach is not only correctly classifying the frequent activities but also correctly classifying the infrequent ones. Due to space constraints, we do not show the results for other houses.

Table 2: House A: Performance metrics

Model	Feature	Precision	Recall	F-Measure	Accuracy
HMM	Change	36.9	35.2	31.8	71.2
	Raw	30.8	13.8	12.9	59.7
	Last	21.8	19.3	15.2	66.0
	Last+Change	27.1	27.8	27.3	93.2
SVM	Change	34.7	11.4	9.5	53.1
	Raw	33.2	17.0	17.4	60.6
	Last	26.7	27.9	27.3	93.7
	Last+Change	48.8	29.0	29.3	93.7
CRF	Change	9.1	10.5	9.4	50.8
	Raw	13.3	13.6	13.4	56.9
	Last	25.8	25.7	25.6	91.1
	Last+Change	26.7	28.0	27.3	93.7
SSVM	Change	13.9	10.3	7.5	53.1
	Raw	23.0	15.2	15.3	61.3
	Last	30.3	29.2	28.9	93.7
	Last+Change	36.1	28.2	27.6	94.5
NB	Change	38.1	11.6	9.9	53.2
	Raw	52.7	17.3	17.0	63.1
	Last	28.3	28.2	27.7	94.4
	Last+Change	49.4	30.1	30.7	93.9

Table 2 shows the confusion matrix for House A for the Naive Bayes Model. We note that Naive Bayes correctly classifies majority of the of the dominant classes correctly. Most of the confusion takes place in activities such as Breakfast or Dinner, where the activity period may be short or infrequent. However, the dominant classes such as 'idle' or 'leaving house' or 'go to bed' activities are labeled correctly.

	idle	leave house	use toilet	take shower	brush teeth	go to bed	prepare Breakfast	prepare Dinner	get snack	get drink
idle	86.1	6.7	0.2	0.0	0.0	7.0	0.0	0.0	0.0	0.0
leave house	0.1	99.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
use toilet	77.8	0.6	0.7	0.0	0.0	20.9	0.0	0.0	0.0	0.0
take shower	99.9	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
brush teeth	85.0	5.9	0.0	0.0	0.0	9.1	0.0	0.0	0.0	0.0
go to bed	4.3	0.0	0.0	0.0	0.0	95.7	0.0	0.0	0.0	0.0
prepare Breakfast	98.3	0.0	0.0	0.0	0.0	1.2	0.0	0.5	0.0	0.0
prepare Dinner	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
get snack	98.1	1.3	0.0	0.0	0.0	0.6	0.0	0.0	0.0	0.0
get drink	96.9	3.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 3: House A: Model NB Feature Last+Change

6 Conclusions and Lessons Learned

References

[1] T. Duong, D. Phung, H. Bui, and S. Venkatesh. Efficient duration and hierarchical modeling for human activity recognition. *Artificial Intelligence*, 173(7):830–856, 2009.

[2] S. Lühr, H. H. Bui, S. Venkatesh, and G. A. West. Recognition of human activity through hierarchical stochastic learning. In *null*, page 416. IEEE, 2003.

[3] N. Oliver, A. Garg, and E. Horvitz. Layered representations for learning and inferring office activity from multiple sensory channels. *Computer Vision and Image Understanding*, 96(2):163–180, 2004.

[4] D. J. Patterson, D. Fox, H. Kautz, and M. Philipose. Fine-grained activity recognition by aggregating abstract object usage. In *Wearable Computers, 2005. Proceedings. Ninth IEEE International Symposium on*, pages 44–51. IEEE, 2005.

[5] E. M. Tapia, S. S. Intille, and K. Larson. *Activity recognition in the home using simple and ubiquitous sensors*. Springer, 2004.

[6] T. Van Kasteren, G. Englebienne, and B. J. Kröse. An activity monitoring system for elderly care using generative and discriminative models. *Personal and ubiquitous computing*, 14(6):489–498, 2010.

[7] T. Van Kasteren, A. Noulas, G. Englebienne, and B. Kröse. Accurate activity recognition in a home setting. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 1–9. ACM, 2008.