Supervised with manually created features

A Data Mining Framework for Activity Recognition In Smart Environments

Chao Chen
School of Electrical Engineering and Computer Science
Washington State University
cchen@eecs.wsu.edu

Barnan Das
School of Electrical Engineering and Computer Science
Washington State University
barnandas@wsu.edu

Diane J. Cook
School of Electrical Engineering and Computer Science
Washington State University
cook@eecs.wsu.edu

***Abstract*—Recent years have witnessed the emergence of Smart Environments technology for assisting people with their daily routines and for remote health monitoring. A lot of work has been done in the past few years on Activity Recognition and the technology is not just at the stage of experimentation in the labs, but is ready to be deployed on a larger scale. In this paper, we design a data-mining framework to extract the useful features from sensor data collected in the smart home environment and select the most important features based on two different feature selection criterions, then utilize several machine learning techniques to recognize the activities. To validate these algorithms, we use real sensor data collected from volunteers living in our smart apartment test bed. We compare the performance between alternative learning algorithms and analyze the prediction results of two different group experiments performed in the smart home.**

Keywords-Activity Recognition; Smart Environments; Machine Learning.

I. INTRODUCTION

The recent emergence of the popularity of Smart Environments is the consequence of a convergence of technologies in machine learning, data mining and pervasive computing. In the smart home environment research, most attention has been directed toward the area of health monitoring and activity recognition. Over the past few years, there has been an upsurge of innovative sensor technologies and recognition algorithms for this area from different research groups. Georgia Tech Aware Home [1] identifies people based on the pressure sensors embedded into the smart floor in strategic locations. This sensor system can be used for tracking inhabitant and identifying user's location. The smart hospital project [2] develops a robust approach for recognizing user's activities and estimating hospital-staff activities using a hidden Markov model with contextual information in the smart hospital environment. MIT researchers [3] recognize user' s activities by using a set of small and simple state-change sensors, which are easy and quick to install in the home environment. Unlike one resident system, this system is employed in multiple inhabitant environments and can be used to recognize Activities of Daily Living (ADL). CASAS Smart Home Project [4] builds probabilistic models of activities and used them to recognize activities in complex situations where multiple residents are performing activities in parallel in the same environment.

In this paper we design a data mining framework to recognize activities based on raw data collected from CASAS smart home environment. The framework synthesizes the sensor information and extracts the useful features as many as possible. To improve the performance of the recognition, we considered two information-theoretic feature selection methods to find the most optimal subspace of the features. Then, we make use of several popular machine learning methods to compare the performance of activities recognition.

This paper is organized as follows: Section 2 introduces our data-mining system architecture as found in the CASAS smart environment project and describes the functions of different modules in the system. Section 3 describes four machine learning methods and their comparative advantages for the activity recognition problem. Section 4 presents the results of our experiments and compares the performance between different feature selection methods and machine learning approaches based on two different group experiments.

II. CASAS DATA MINING SYSTEM ARCHITECTURE

The smart home environment testbed that we are using to recognize activity is a three bedroom apartment located at the Washington State University campus. Figure 1 describes the data mining architecture used by our CASAS smart home project. There are four main parts in this system architecture. Data collection is responsible for collecting sensor data in the smart home environment. In data annotation, the researchers annotate the activity labels into the database for recognizing activities in future. The function of feature extraction is to extract as many as features as possible from the sensor system in our smart home environment and use feature selection methods to select the optimal feature subset . In the final part, we apply several popular machine learning methods to recognize activity given these selected optimal features.

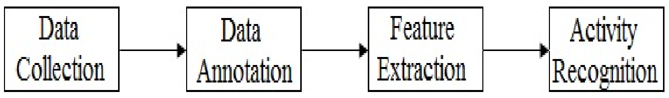

Figure 1. Data mining system architecture for a smart home.

A. Data Collection

As shown in Figure 2, the smart apartment testbed located on the WSU campus has three bedrooms, one bathroom, a kitchen and a living/dining room. In order to track the mobility of the inhabitants, the testbed is equipped with motion sensors placed on the ceiling. The circles in the figure indicate the positions of motion sensors. They allow tracking of the people moving across the apartment. In addition, the testbed also has installed temperature sensors along with custom-built analog sensors to provide temperature readings and usage of hot water, cold water and stove burner. A power meter records the amount of instantaneous and total power usage. We use contact switch sensors to monitor usage of the phone book, a cooking pot, and the medicine container. Sensor data is captured using a customized sensor network and is stored in a SQL database.

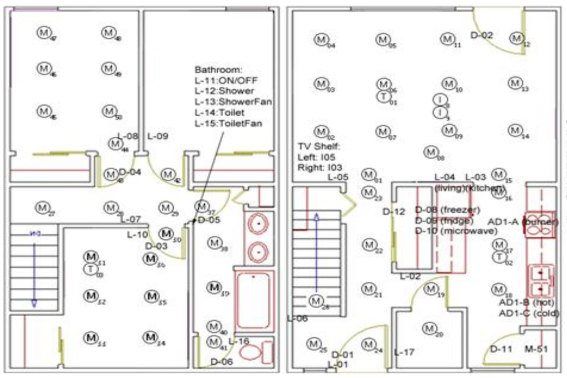

Figure 2. Three-bedroom smart apartment used for our data collection (motion (M), temperature (T), water (W), burner (B), telephone (P),and item (I))

The gathered sensor data has a format shown in Table 1. These four fields (Date, Time, Sensor, ID and Message) are generated by the CASAS data collection system automatically.

TABLE I. RAW DATA FROM SENSORS

Date	Time	Sensor ID	Message
2009-02-06	17:17:36	M45	ON
2009-02-06	17:17:40	M45	OFF
2009-02-06	11:13:26	T004	21.5
2009-02-05	11:18:37	P001	747W
2009-02-09	21:15:28	P001	1.929kWh

To provide real training data, we have collected data while a student in good health was living in the smart apartment. A large amount of sensor events were produced as training data during several months. All of our experimental data are produced by these the student's day to day life, which guarantee that the results of this analysis are real and useful.

B. Data Annotation

After collecting data from smart home apartment, sensor data is annotated for activity recognition based on people's activities. Because the annotated data is used to train the machine learning algorithms, the quality of the annotated data is very important for the performance of the learning algorithms. A large number of sensor data events are generated in smart home environments. Without a visualizer tool, it is difficult for researchers and users to interpret raw data as the residents' activities [5]. To improve the quality of the annotated data, we build an open source Python Visualizer, called PyViz, to visualize the sensor events, which can display events in real-time or in playback mode from a captured file of sensor event readings.

With the help of PyViz, activity labels are optionally added at the end of each sensor event, marking the status of the activity. For our experiment, we selected six activities that the volunteer participant regularly performs in the smart home environment to predict and classify activities. These activities are: Cook; Watch TV; Computer; Groom; Sleep; Bed to toilet.

C. Feature Extraction

Before making use of machine learning algorithms, another important step is to extract useful features or attributes from the raw annotated data. We have considered some features that would be helpful in energy prediction. These features have been generated from the raw sensor data by our feature extraction module. The following is a listing of the resulting features that we used in our energy prediction experiments.

- Activity length in time (in seconds)
- Time of day (morning, noon, afternoon, evening, night, late night)
- Day of week
- Whether the day is weekend or not
- Previous/Next activity
- Number of kinds of motion sensors involved
- Total Number of times of motion sensor events triggered
- Energy consumption for an activity (in Watt)
- Motion sensor M1…M51 (On/Off)

D. Feature Selection

After feature extraction, a large number of features are generated to describe a particular activity. However, some of these features are redundant and irrelevant and thus suffer from the problem of drastic raise in computational complexity and classification errors. We choose two popular feature selection approaches based on two different criterions and describe them in the following sections.

1) Minimal Redundancy and Maximal Relevance.

One of the most popular feature selection approaches is the Max-Relevance method, which selects the features x_i individually for the largest mutual information $I(x_i; c)$ with the target class c, reflecting the largest dependency on the target class. However, the results of Cover [6] show that the combinations of individually good features do not necessarily lead to good classification performance. To solve this problem, Peng et al. [7] proposed a heuristic minimum-Redundancy-Maximum-Relevance (mRMR) selection framework, which selects features mutually far away from each other, while still maintaining high relevance to the classification target. In mRMR, Max-Relevance is to search features with the mean value of all mutual information values between an individual feature x_i and class c:

$$\max D(S,c), D = \frac{1}{|S|}\sum_{x_i \in S} I(x_i; c)$$

Because the Max-Relevance features may have a very high possibility of redundancy, a minimal redundancy condition can be added to select mutually exclusive features:

$$\min R(S), R = \frac{1}{|S|^2}\sum_{x_i, x_j \in S} I(x_i; x_j)$$

Thus, the operator $\Phi(D,R)$ is the mRMR criterion combining the above two constraints:

$$\max\Phi(D,R), \Phi = D - R$$

2) Decision Tree Based on Information Gain.

The second feature selection approach we use is the J48 Decision Tree classifier [8], which uses information gain to create a classification model, a statistical property that measures how well a given attribute separate the training examples according to their target classification. Information gain is a measure based on entropy, a parameter used in information theory to characterize the purity of an arbitrary collection of examples. By utilizing the J48 decision tree algorithm, we can learn which feature plays a more important role towards the target classification and select several important features as the main features for other machine learning methods.

E. Prediction Models

We make use of four popular machine learning methods [9] to represent and recognize the activities based on the optimal features we selected: Bayes Belief Networks, Artificial Neural, Network, Sequential Minimal Optimization, and LogitBoost method as follows.

1) Bayes Belief Networks.

Bayes belief networks [10] represent a set of conditional independence assumptions by a directed acyclic graph, whose nodes represent random variables and edges represent direct dependence among the variables and are drawn by arrows by the variable name. Bayesian belief networks apply conditional independence assumptions only to subsets of the variables. It can be suitable for small and incomplete data sets and they incorporate knowledge from different sources. After the model is built, they can also provide fast responses to queries.

2) Artificial Neural Networks.

Artificial neural networks (ANNs) [11] are composed of interconnecting artificial neurons, which provide a general and robust method to learn a target function from input examples. Although there is no guarantee that an ANN will find the global minimum, ANNs can be applied to problems where the relationships are dynamic or non-linear and capture many kinds of relationships that may be difficult to model by other machine learning methods. In our experiment, we choose Multilayer-Perceptron learning method to recognize activities.

3) Sequential Minimal Optimization.

For a normal Support Vector Machine (SVM) [12], the quadratic programming problem involves a matrix, whose elements are equal to the number of training examples. If the training set is large, the SVM algorithm will use a lot of memory. To solve this problem, Sequential Minimal Optimization (SMO) [13] decomposes the overall quadratic programming problem into a series of smaller problems. During the training process, SMO picks a pair of Lagrange multipliers for each iteration to solve this small quadratic programming problem and repeats the same process until it converges on a solution.

4) LogitBoost Ensemble.

Boosting [14] is a general supervised learning method for improving the accuracy of any given learning algorithm. The idea of boosting is to combine many "weak" classifiers to create a "strong" classifier. Therefore, we use LogitBoost [15] for our experimentation, which minimizes the expectation of the loss function by using Newton-like steps to fit an additive logistic regression model to directly optimize a binomial log-likelihood. The property of LogitBoost turns out to change linearly with the output error and be less sensitive to the noisy data.

III. Experimental Results

We performed two series of experiments: sensor data collected during two summer months and three winter months of data in the testbed. The testing tool we choose, called Weka [16], provides an implementation of the learning algorithms that we can easily apply to our own dataset. Using Weka, we assessed the classification performance of our four selected machine learning algorithms using 10-fold cross validation. In the mRMR experiment, we have used the subset size of twenty features. However, decision tree feature selection doesn't have a fixed-size subset feature space. We use these notations in our following experiments. (BN: Bayes Belief Networks; MP: Multilayer-Perceptron; SMO: Sequential Minimal Optimization; LB: LogitBoost)

A. Comparison of the average accuracy

As shown in Figures 3, most of the algorithms have a good performance in recognizing activities. LogitBoost yields the highest accuracy around about 90% for both datasets, while SMO yields the lowest accuracy at around 70% based on J48 feature selection for both datasets. These results support the claim that our data mining architecture is quite efficient in recognizing activities. By testing the different machine learning algorithms, we can get the best algorithm for our activity recognition task.

In comparison of the performance between the different algorithms, LogitBoost proves to be the best for both the datasets. That is because LogitBoost can generate a powerful classifier by combing the performance of many “weak” classifiers. For three other algorithms, it is very difficult to judge which one is better than others on the accuracy performance.

In the winter dataset, the Bayes Net algorithm based on two feature selection methods have a better performance than the same algorithm without employing a feature selection method. LogitBoost has almost the same performance with and without feature selection. For Multilayer-Perceptron and SMO, the dataset without feature selection performs better than its subset after two different feature selections. In summer dataset, the performance of all the algorithms is better with mRMR feature selection. Except SMO, three other algorithms also have improved in performance after using J48 feature selection. As shown in Figures 3, the classification accuracy based on J48 and mRMR is almost the same or better than the performance without feature selection in the most of cases.

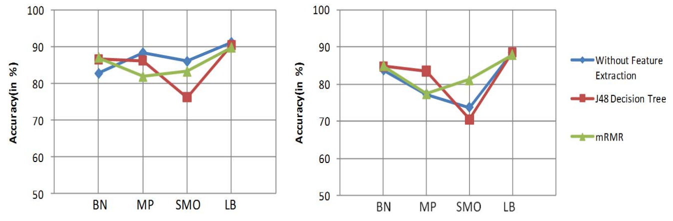

Figure 3. Comparison of the average accuracy (left: winter dataset, right: summer dataset).

B. Comparison of the time performance

Table 2 gives the details of running time of all the algorithms. Figure 4 makes use of logarithmic time as a scale to compare the different performances. There has been a remarkable improvement in the running time performance of the algorithms after using feature selection. However, there are some exceptions to our expectation in Table 2. That is because some other programs in our experimental systems might use some CPU resources, which leads to a slight reduction in performance time. As seen from the above section, the learning algorithms will not reduce the accuracy after J48 and mRMR feature selection in the most cases. Overall, feature selection methods can be helpful to improve the running time performance without reducing the accuracy based on the original data set.

TABLE II. COMPARISON OF RUNNING TIME PERFORMANCE

	Running Time(s)					
	Summer Dataset			Winter Dataset		
	Without	J48	mRMR	Without	J48	mRMR
BN	0.23	0.24	0.07	0.05	0.03	0.23
MP	59.93	5.25	15.15	562.41	50.29	81.2
SMO	0.35	0.53	0.31	1.08	0.51	2.24
LB	0.42	0.35	0.1	2.14	0.72	0.5

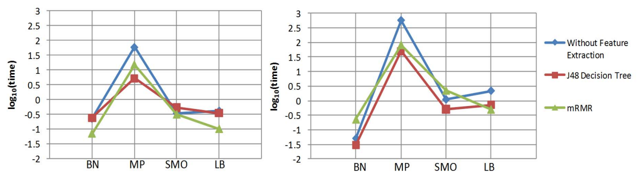

Figure 4. Comparison of time performance(left : winter dataset, right: summer dataset).

IV. CONCLUSIONS

In the current work we introduced an activity recognition data mining framework using an integrated system for collecting sensor data and applying machine learning techniques in a smart home environment. We extracted features from real sensor data in a smart home environment and selected the most important features based on information gain and mRMR. To assess the performance of the four machine learning methods, we performed two group experiments during two different periods, analyzed the results of the experiments and provided the explanation of those results.

In our ongoing work, we plan to further investigate new and pertinent features to recognize activities more accurately. To improve the accuracy of recognition activities, we intend to install more sensitive sensors to capture more useful information in the smart home environment. We are also planning to apply more feature selection methods and machine learning techniques to different environments in which different residents perform similar activities. This will allow us to analyze whether the same pattern exists across residents and environments.

REFERENCES

[1] R. J. Orr and G. D. Abowd, "The smart floor: A mechanism for natural user identification and tracking," in *Conference on Human Factors in Computing Systems*, ed, 2000, pp. 275–276.

[2] D. Sánchez*, et al.*, "Activity recognition for the smart hospital," *IEEE Intelligent Systems,* vol. 23, pp. 50–57, 2008.

[3] E. M. Tapia*, et al.*, "Activity recognition in the home using simple and ubiquitous sensors," *Pervasive Computing,* pp. 158–175, 2004.

[4] G. Singla*, et al.*, "Recognizing independent and joint activities among multiple residents in smart environments," *Journal of Ambient Intelligence and Humanized Computing,* pp. 1–7, 2010.

[5] S. Szewcyzk*, et al.*, "Annotating smart environment sensor data for activity learning," *Technology and Health Care,* vol. 17, pp. 161–169, 2009.

[6] T. M. Cover, "The best two independent measurements are not the two best," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 4, pp. 116–117, 1974.

[7] H. Peng*, et al.*, "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on pattern analysis and machine intelligence,* vol. 27, 2005.

[8] J. R. Quinlan, "Induction of decision trees," *Machine learning,* vol. 1, pp. 81–106, 1986.

[9] T. M. Mitchell, "Machine learning," *Burr Ridge, IL: McGraw Hill,* 1997.

[10] I. Rish, "An empirical study of the naive Bayes classifier," in *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*, ed, 2001, pp. 41–46.

[11] S. F. Zornetzer, *An introduction to neural and electronic networks*: Morgan Kaufmann, 1995.

[12] B. E. Boser*, et al.*, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, ed, 1992, pp. 144–152.

[13] J. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines," *Advances in Kernel Methods-Support Vector Learning,* vol. 208, 1999.

[14] Y. Freund and R. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *Computational Learning Theory*, ed, 1995, pp. 23–37.

[15] J. F. T. Hastie and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *The Annals of Statistics,* vol. 28, pp. 337–374, 2000.

[16] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*: Morgan Kaufmann Pub, 2005.