Efficient Daily Activity Recognition using Data Mining and Adaboost

Md. Kamrul Hasan

Dissertation Submitted To

The Department of Computer Engineering
In Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy

Kyung Hee University
Seoul, Korea

August, 2010

Efficient Daily Activity Recognition using Data Mining and Adaboost

by

Md. Kamrul Hasan

Supervised by

Prof. Sungyoung Lee, PhD

Prof. Young-Koo Lee, PhD

Dissertation Committee

OkSam Chae

Prof. Oksam Chae (Chairman)

Prof. Tae-Choong Chung

Prof. Sungyoung Lee

Young Koo Lee

Prof. Young-Koo Lee

Prof. Dong Han Kim

Dedicated to my family

Acknowledgement

First of all, I thank and praise the almighty God, for His bounties and blessings that He granted on me throughout my life. I would like to convey my heartiest thanks to all those people who were with me during this journey of achieving the degree. Though it is not possible to acknowledge all of them here, I must mention few names.

I would like to acknowledge my sincere gratitude to Professor Sungyoung Lee for giving me the opportunity to work in Ubiquitous Computing Lab and giving me the freedom to research on the topic of my interest. I would like to thank Professor Young-Koo Lee for his insightful comments that he provided time to time. I would like to thank the knowledgeable Thesis Committee whose comments and suggestions greatly improved the quality of this dissertation.

I would like to thank all my lab members, especially to aCAMUS and SCOM Team members. I am grateful to Professor Kaykobad, Kim Anh, Lenin Mehedy for their great inspiration and active support.

I am thankful to all Kyung Hee University Bangladeshi (KHUBD) Members, who supported and taught me in different aspects of my life here. I can never forget the pleasant moments and memories that I passed with this community during my stay in Korea. I remember Dr. Julius Hossain who treated me as his own brother, a rare affection, usually not found in abroad.

I acknowledge my debt to my parents whose affection and scarifies cannot be repaid in my whole life. I wish I could be with them during their well and woes. I hope my achievements and deeds will bring happiness and long life to them.

Lastly, but not the least, I acknowledge the contribution of my wife. She has always been with me during the last three years leaving her parents. She always puts me on track when I am derailed from my purpose. Without her support, it would not have been possible to write this dissertation.

Md. Kamrul Hasan
August 2010

ABSTRACT

Today, activity is an import context that needs to be recognized for proactive service delivery. Recognition of Activities of Daily Living (ADLs) can help monitoring elderly people at home. To know what the user is doing, sensors need to be attached with the household devices and utensils for generating events and from a series of such events over a time window, user activity can be deduced.

Sequence of sensor activations are the main feature for the recognition of ADLs. However, given many sensors, the number of possible sensor sequences can be enormous and hence usual classifiers will not be able to handle such number of features. I devised a mechanism that reduces the number of such features and keeps only the important features contributing to the correct classification of the activities. I used data mining together with Multi-class Adaboost to reduce the dimension of the feature set. I propose four types of features namely Maximum Likelihood, Last Sensor, Sensor Sequence and Sensor Group. Maximum Likelihood is calculated by my devised algorithm, MLVC. Sensor Sequences and Sensor Groups are first mined from training data. Then the Maximum Likelihood values, Last Sensor, mined Sensor Sequences and Sensor Groups are put to Adaboost to further reduce the feature dimension.

Insertion of noise in the data can reduce the accuracy of the classifier. When multiple users are roaming around, a user data become the noise to other user. I propose each user carry an ID sensor that sends periodic beacon. The sensors capture the beacon and associate the events with the person carrying the ID sensor. I propose a sensor network system for such a scenario. The combination of ID sensor and my classification algorithm can provide a framework for real-time activity recognition system.

Key words: Activity recognition, Activity of Daily Living (ADL), Context Awareness, Sensor Sequence, Data Mining, Adaboost, Distributed, Sensor Network, State Sensors, Machine Learning, Hidden Markov Model, Ubiquitous, Proactive, Real-time.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recognizing Activity of Daily Living (ADL) [1] is of particular interest to researchers for its various application domain, especially in healthcare industry. Detecting and learning the daily activities of elderly person can save the caregiver's time and give the elderly more independence.

A human can easily understand what the persons around him are doing. This ability of activity recognition seems so natural and simple for ordinary people, but it actually requires complicated functions of sensing, learning, and inference. Imagine recognizing daily activities in the kitchen. Finding a person in the kitchen can provide some evidence that the person is cooking. But it is not that straightforward to recognize all the activities in the kitchen; other sensory evidences are needed. Such as gas burner turned on can indicate the person is cooking, sink faucet turned on can indicate the person is washing dishes. We learn from our past experiences. However, all these functions of sensing the environments, learning from past experience, and applying knowledge for inference are still great challenges for machines. The goal of our research is to enable computers to have similar capabilities as humans for recognizing people's activities.

The first step to achieve this goal is to augment the system with sensing capabilities. There are three basic methods for this: video based, wearable sensor based, environmental sensor based. Video Based activity recognition [2-4] though good at recognition accuracy, is usually not deployable for detecting ADLs because of privacy issue. Wearable sensors [5-7] can also detect a fundamental set of ADLs like walking, standing etc. Wearable sensors have limitations in detecting more advanced set of activities like cooking, hygiene. Moreover, users are usually reluctant to wear sensors in special positions and orientations in their body needed for the specific recognition algorithm. So, sensor deployed in the environment [8-12] is more desirable for detecting ADLs.

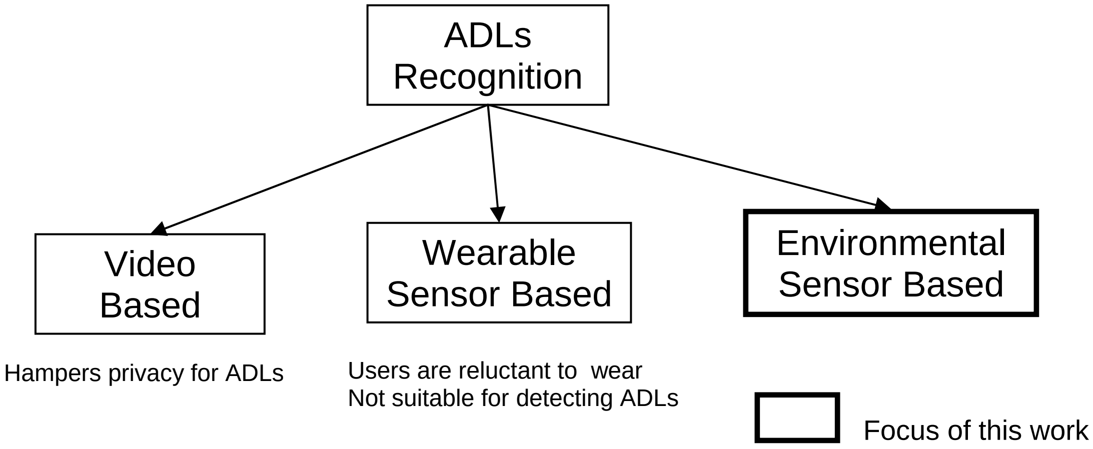
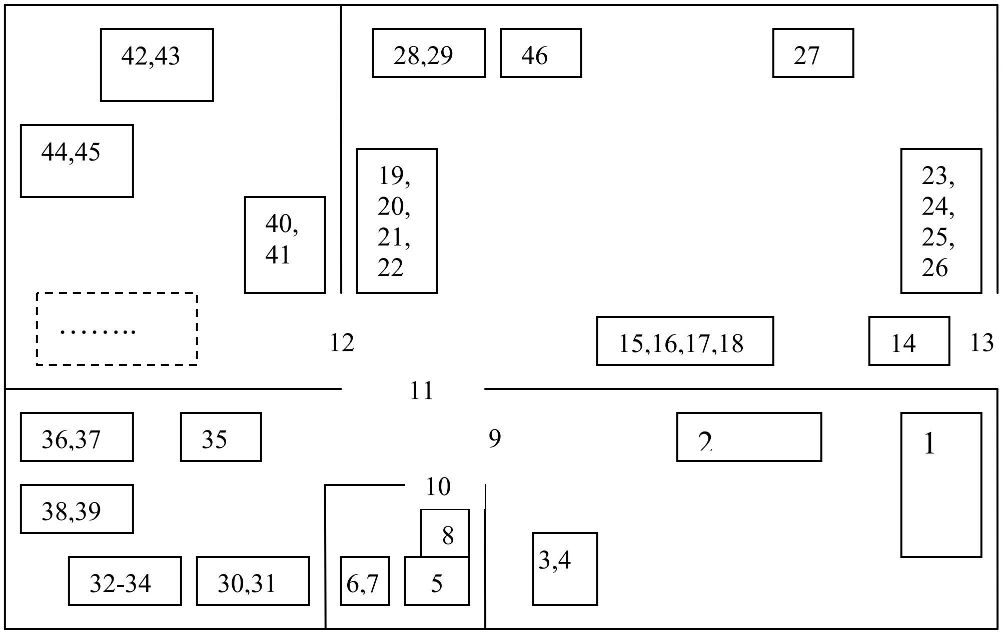


Figure 1.1 Three Approaches to ADL recognition

In a typical setup, various types of simple sensors, especially binary on-off state sensors, are deployed in the environment. The sensors are supposed to be deployed and forgot. So, the system needs to gather training data over a period of time. Sensor activation sequences are then fed to the system to train and test the classifier. Due to unavailability of any special features, the sensor states of the whole house are usually taken as the observation at a time instance. Location could be a key feature for separating out activity groups. However, location is not used here; users may do the same activity in different places. Again, if activities are detected independent of location and the names of the sensors are generalized to higher semantics; it might be possible to transfer the learning [48,49] to same or similar environment. So, the way of doing things need to be discovered.

Once the sensing capability in the system is achieved, the system needs to be trained. The trained classifier can infer the activities when fed with real-time sensor data. The training phase has three important sub-components: feature selection, feature dimension reduction and classification. Feature dimension reduction is optional if the number of features is less. Choosing the suitable classifier along with appropriate features for the problem domain is important. If the features to be calculated are huge, a feature dimension reduction technique is needed.



1: bed pressure sensor
2: pressure sensor in computer chair
3,4: almirah door sensors
9-13: door sensors
5: toilet flush
6,7: toilet cabinet sensors
8: toilet basin sensors
14: shoe box
15-18: dinning chairs
19-26: sofa pressure sensors

27: tv sensor
28,29: reading sensors
30,31: cooking sensors
32-34: kitchen cabinet sensors
35: microwave sensors
36,37: fridge sensors
38,39: sink sensors
40,41: almirah sensors
42,43: dressing table drawer sensors
44,45: medicine cabinet sensors

An observation at a time instance:

000010000000110000001100000000….(45 digits)

Figure 1.2 Example sensor deployment scenario in a home environment

The goal of the feature selection and training the classifier is to recognize the activities in real-time when deployed with higher or reasonable accuracy. For real timeliness the system needs to decide on sensor data over a small time window. The number of features to be calculated should also be small, so that they can be calculated quickly and fed to the trained classifier. Accuracy of the recognition is always the important factor to be useful in the practical applications. However, accuracy solely depends on the training data at hand. The more representative training data is available, the better accuracy can be achieved. I term a recognition algorithm that uses less number of features over a small time window and still provides high accuracy as efficient.

The classifier such trained is designed for a single user. But when multiple users are present or the user deviate from usual activity, noise is inserted into the sensor data which eventually may lead to misclassification of the activity. So, we need a mechanism to remove the noise from the data and correctly attribute the events to users.

Formally, I define two problem statements R&A (Real-time & Accuracy) and Multi-user (MU).

Problem Statement:

- R&A: Given *large number of sensors* (typically binary) deployed in the home environment, can we recognize the Activities of Daily Living (ADLs) for *a single user* without using location information?
    - Using small window and less number of features
    - Providing reasonable accuracy based on training data.
- MU: Given the setup as above, can we provide a mechanism to support multiple users so that:
    - Sensor data are correctly attributed to each user for whom an efficient activity recognition algorithm can be run?

1.1. Solution Approach and Motivation

Activities of Daily Living (ADLs) involve the household devices and utensils. A user manipulates and uses the devices and utensils in a certain fashion. Still there is so much variation in doing an activity that it is very difficult to learn all the models due to the shortage of enough training data [10].

In addition to this, there can be noise in the data as well, as user's movements are not always purposeful, and even if purposeful, the movement can be for performing a different activity. Figure 1.3 shows a deviation of the user from the default course of movement.

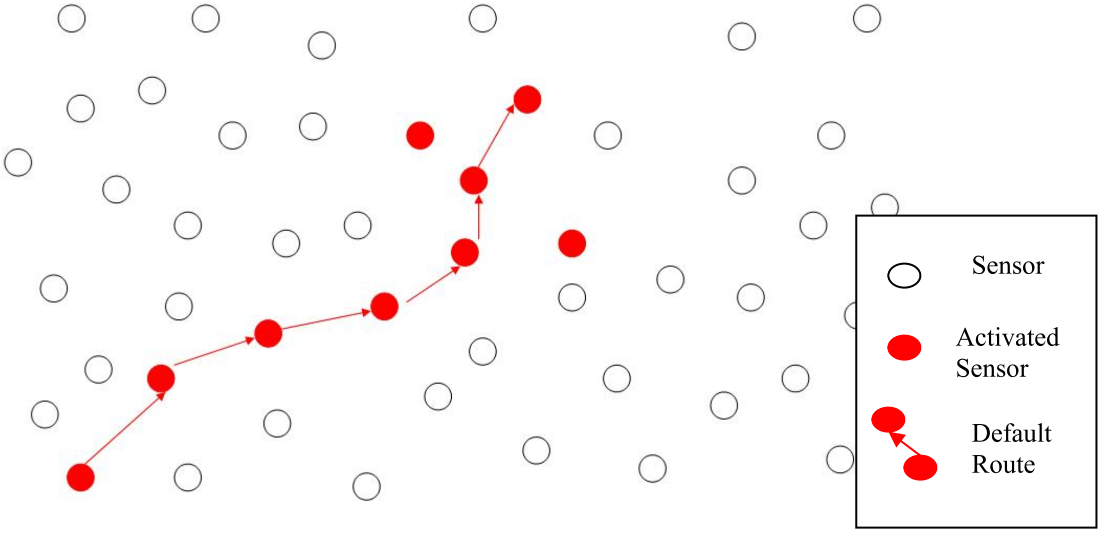


Figure 1.3 Example of sensor activations as user deviates from his default course

So, I move away from using a high level model defining the networks for ADLs involving all possible combination of daily object usage. I rather focus on using the temporal relationships among the sensors. As for example, a 'preparing meal' activity is indicated by 'going to the kitchen', 'turning on the gas burner', and so on. Naturally, those sensors attached to the cooking utensils will be activated while the user is preparing a meal. Some other sensors can also be activated during that period as the user deviates from meal preparation. Finding the most relevant sensors and ignoring other sensor events, can contribute to the detection of the user activity.

Moreover, in a practical system, the data window from which the features to be calculated should be small. Otherwise the system will be not deployable in real environment. What features can be calculated from a small sensor data window? The features can be some numeric values, some sensor value that is representative of the activity, a set of sensor sequences or group of sensors.

I take maximum Likelihood value for each activity because the values can vote for highest likely activity. I calculate the values by Maximum Likelihood Value Calculation (MLVC) method. Last Sensor Activated can be the representative feature. Sensor Sequences and Sensor Groups are intuitive features for activity recognition problem.

The usual approach to model temporal relationships is to use Hidden Markov Model (HMM) or variations like Conditional Random Field (CRF), Dynamic Bayesian Network (DBN) etc. The problems with such approaches are modeling the network, long temporal sequences needed, use of correlation of activities or intractable time for inference. Moreover, with many sensors deployed, the networks cannot handle the conditional probability tables. So, using Sensor Sequences and Sensor Groups features directly into a non temporal classifier makes the features unique.

But even in a small data window, the number of possible Sensor Sequences or Sensor Groups can be huge. If 100 sensors are deployed, and we want to calculate a sequence of length 5, then there are 100^5 possibilities. A sensor group of length 5 has ${}^{100}C_5$ combinations. Calculating such an enormous set of features and using them for training is not possible in a standard environment. The features are not also possible to be calculated in real-time in the deployed environment. However, not all the features are related. Some of the sequences and groups may not occur at all. Patterns commonly occurring during an activity period are actually representative features for the activity. Based on the idea, I used data mining technique to select important sequences and groups for each activity from the training data.

After selecting the features and reducing the feature dimension, we need to use a classifier. For recognizing ADLs the features I calculate are not key features rather are indicative of the activities being performed. Adaboost is suitable for situation when there is no key feature but the features together, may form a strong classifier. So, Adaboost is the best choice for recognizing ADLs. Adaboost has the additional quality of reducing feature dimension further by selecting a set of hypothesis (a subset of feature space). I needed the quality also because less number of features, the lightweight the classifier becomes. Adaboost was originally proposed as a binary classifier that was extended to multi-class by using one-versus-all strategy. Thanks to a recent work [77] that provides a clean theoretical background of Adaboost for multi-class problem (see section 2.4 for details).

Once the classifier is trained, it needs to be deployed in a real home where it is highly unlikely to have single resident. A users' activity event can become noise to other users' data and the classifier will eventually make wrong decision. Location based tracking is not enough for this problem. We need an object usage based tracker which needs huge domain knowledge and computing time and still prone to error due to change of user behavior. I propose each user carry an ID sensor that send periodic beacon. The sensors attached to daily objects, capture the beacon and associate the events with the person carrying the ID sensor. I describe a sensor network system for such a scenario. The combination of ID sensor and my classification algorithm can provide a framework for accurate real-time activity recognition.

So, the contribution of this dissertation can be stated as below:

1.2. Contribution

For activity recognition in a home environment

Feature Selection: four types of features that capture the effectively capture temporal relationship in small data window:

- Maximum Likelihood (ML)
- Last Sensor (LS)
- Sensor Sequence (SS)
- Sensor Group (SG)

– Maximum Likelihood and Sensor Sequence, Sensor Group features are unique to this environment.

Feature Reduction: Use data mining technique to reduce the feature dimension of *Sensor Sequence and Sensor Group*

Classification: Use a multi class Adaboost for classification.

Multi-user Support Mechanism: Provide a mechanism to use ID sensor and sensor network for multi user activity recognition.

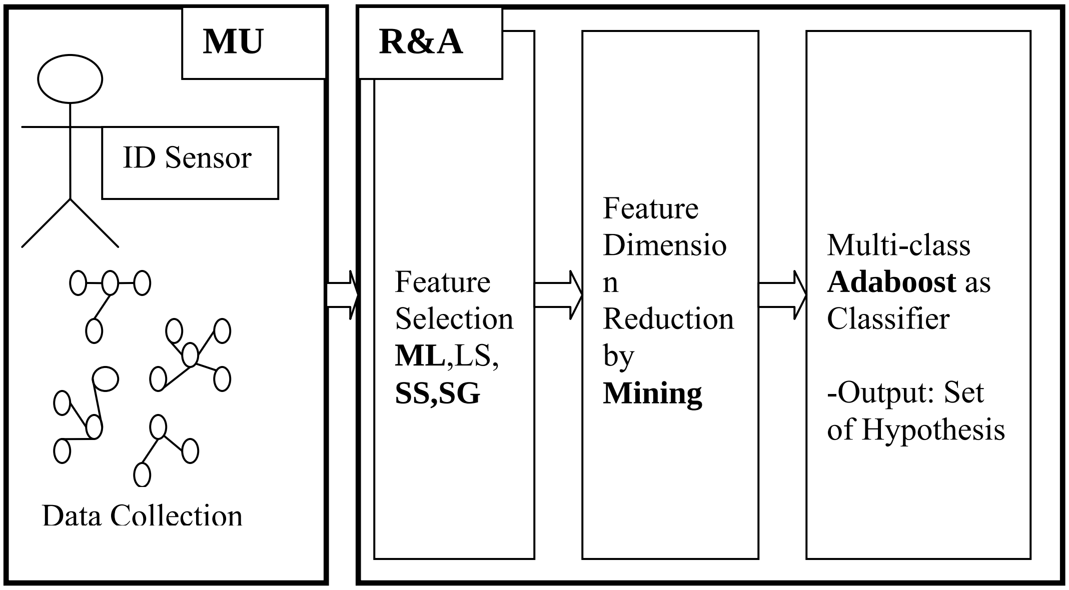
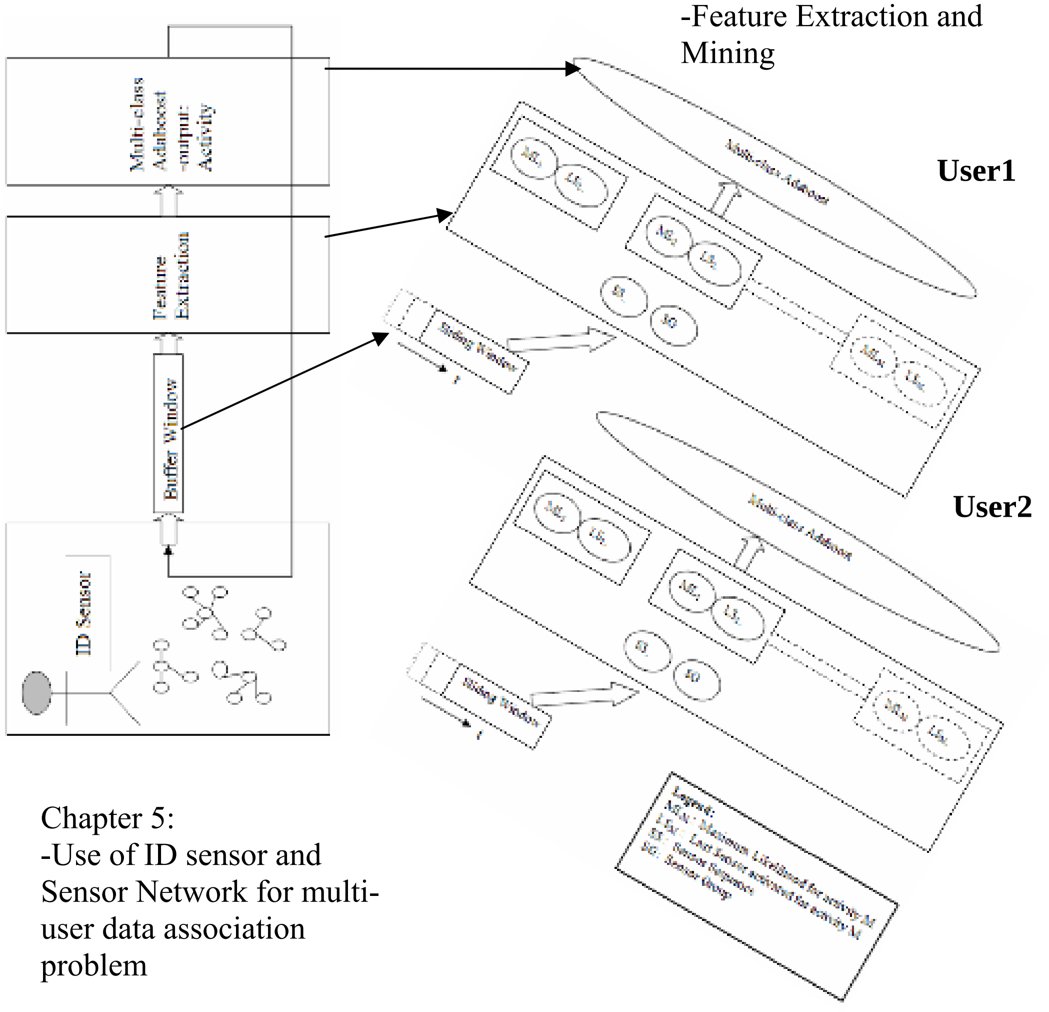

MU
ID Sensor
Data Collection
R&A
Feature Selection **ML**,LS, **SS,SG**
Feature Dimensio n Reduction by **Mining**
Multi-class **Adaboost** as Classifier
-Output: Set of Hypothesis

Figure 1.4 Contributions

1.3. Organization of the Dissertation

The rest of the dissertation is organized as follows. In chapter 2, I discuss some preliminaries needed to understand the later sections. Chapter 3 is the related research works. In chapter 4, I discuss feature selection, dimension reduction and classification method along with experimental results. Chapter 5 describes the ID sensor based sensor network system for noise removal and its simulation and implementation results. I conclude in chapter 6.



Chapter 4:
-Training and Testing the classifier for single user
-Feature Extraction and Mining

Chapter 5:
-Use of ID sensor and Sensor Network for multi-user data association problem

Figure 1.5 Organization of the dissertation

Chapter 2

Preliminaries

Before writing the related works and main chapters, I discuss on some important methods and terminologies referred or used in this thesis.

2.1 Hidden Markov Model (HMM)

Hidden Markov Model (HMM) is a statistical model consisting of a set of states(S), a set of state transition probabilities (A), a set of observations (O) and the output probabilities (B).

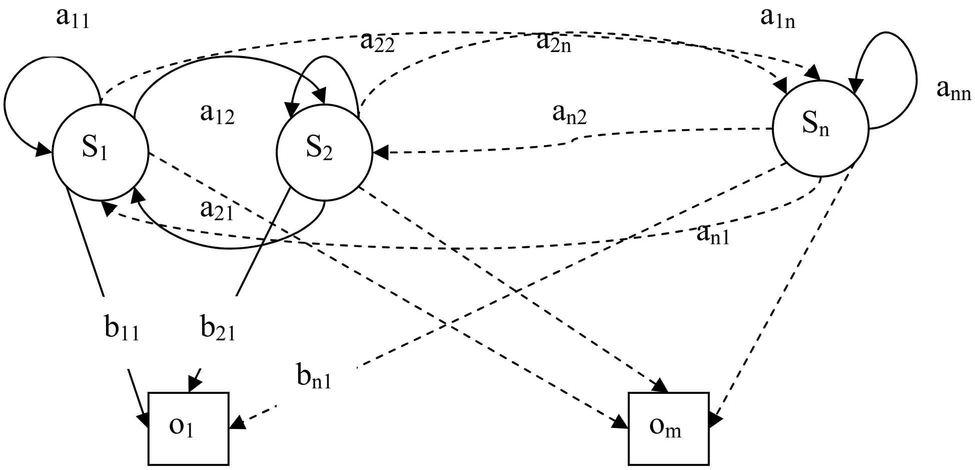


Figure 2.1 A sample HMM

HMM, $\lambda = \{S, A, O, B\}$

$$S = \{S_1, S_2, ..., S_n\}$$

$A = \{a_{11}, a_{12}, ..., a_{1n}, a_{21}, ..., a_{2n}, ..., a_{n1}, ..., a_{nn}\}$, where $a_{ij} = P(S_j \mid S_i)$ $, i, j = 1, ..., n$

$O = \{o_1, o_2, ..., o_m\}$

$B = \{b_{11}, ..., b_{1m}, ..., b_{n1}, ..., b_{nm}\}$, where $b_{ij} = P(O_j \mid S_i)$, $i = 1, ..., n$, $j = 1, ..., m$

HMM assumes Markovian property, which means a state transition, depends only on previous state, not by others. HMM is the simplest form of Dynamic Bayesian Network (DBN). HMM assume all the outputs are independent which may not be true, especially for recognizing Activities of Daily Living (ADLs).

HMM has three basic problem models that applications using HMM map into.

Problem 1: Given the observation sequence $O = o_1 o_2 .... o_L$ and the HMM model λ, how do we efficiently compute $P(O \mid \lambda)$, the probability of the observation sequence given the model? The problem can be solved by forward algorithm.

Problem 2: Given the observation sequence $O = o_1 o_2 .... o_L$ and the HMM model λ, how do we find out the optimal state sequence A=S_1S_2…..S_L, that is optimal in some meaningful sense (i.e. best explains the observation sequence). The problem can be solved by Viterbi algorithm.

Problem 3: How do we adjust the model parameters that maximizes $P(O \mid \lambda)$? The problem can be solved by Baum-Welch algorithm which is actually an Expectation Maximization algorithm. Finding the globally optimum value can be computationally intractable. The algorithm finds local maximum using the training data.

Viterbi algorithm has been widely used in speech recognition. Baum-Welch algorithm could be useful but the wide success of HMM in speech recognition has been much dependent on the hand tuning of the parameters. A detailed tutorial can be found in [13,74, 84].

In speech recognition [84], features extracted from acoustic signal are observations. An HMM is constructed for each Phoneme (smallest pronounceable portion of a word) which in tern are incorporated in a word's network. A phoneme can be uttered in differently in different context. Those are termed as Allophones. Each of the Allophones are modeled in a separate HMM. The decision as to how many allophones should be considered for a given phoneme may depend on many factors, e.g., the availability of enough training data to infer the model parameters. The probabilities for phonemes are generated from Acoustic Model where as, the probabilities for word sequences are generated from Language Model. Viterbi algorithm is used to find the most suitable word sequence from the very big network containing the networks for phonemes, allophones and as a whole the word sequences. The composite model grows with the vocabulary, and can lead to large search spaces. Probability distribution variation in different paths can help. Beam search use the technique. However, for very big composite network, better techniques are needed, such as rescoring. The word hypotheses in the lattice are scored with a more accurate language model, and sometimes with more detailed acoustic models. Lattice rescoring may require new calculations of HMM probabilities [85], may proceed on the basis of precomputed probabilities only [86,87], or even exploit acoustic models which are not HMMs [88].

For ADLs automatically learning the structure and parameters of HMM suffers the same problem as speech recognition. The problem is actually worse, because, human behavior change time to time. And still now it is not found how many steps an activity consists of or what the dependencies among the activities are. Forward algorithm can be used for finding the likelihoods if each activity has got its own HMM (one state or multiple states). Viterbi algorithm can be used if only one HMM is constructed assuming each activity as a state of the HMM. Currently researchers are constructing HMM by calculating the state transition probabilities and observation probabilities from the training data or the web. The parameters leant are not universal of all users and all times.

2.2 Dynamic Bayesian Network (DBN)

DBN is constructed by using several time slices of the same network. DBN also assumes Markovian property which means variable of a time slice will depend on previous time slice only. What is different from HMM, is that DBN can also have dependency of variables in the same time slice. HMM represents state of the world using single random variable (S_t), whereas DBN represents state of the world by a group of random variables $\left\{X_t^1, X_t^2, ..., X_t^D\right\}$. DBN can represent domain knowledge in terms of dependencies among random variables but inference is expensive due to loop introduced into the network. A good tutorial on DBN can be found in [75].

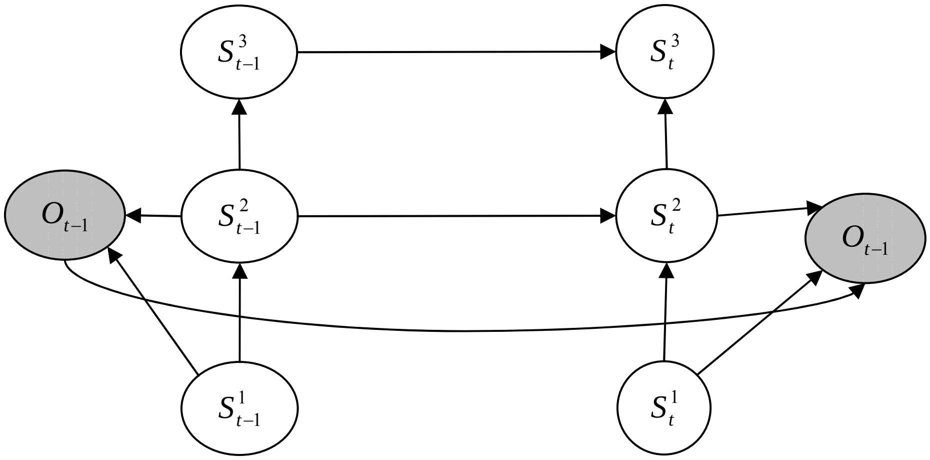


Figure 2.2 A sample DBN (Shaded nodes denote observed variables).

2.3 Adaboost

Adaboost makes a strong classifier based on weak hypotheses set each of which work just better than random. Adaboost is better understood by understanding boosting. Let us take an example of boosting[52,53].

> Consider creating three component classifiers for a two-category problem through boosting. Randomly select *n1* < *n* samples from *D* (total sample space) without replacement to obtain *D1*. Train weak learner *C1*. Select *n2* < *n* samples from *D* with half of the samples misclassified by *C1* to obtain *D2.* Train weak learner *C2.* Select all remaining samples from *D* that *C1* and *C2* disagree on. Train weak learner *C3*. Final classifier is vote of weak learners.

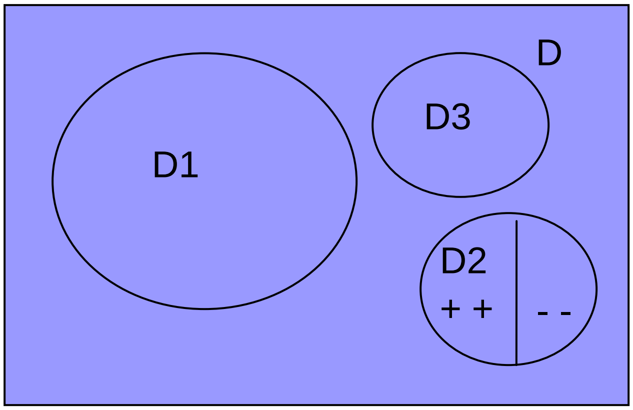


Figure 2.3 Boosting

Adaboost [27, 54], instead of re-sampling, uses training set re-weighting. Each training sample uses a weight to determine the probability of being selected for a training set. AdaBoost is an algorithm for constructing a "strong" classifier as linear combination of "simple" "weak" classifier. Final classification based on weighted vote of weak classifiers. Below I rewrite the adaboost algorithm:

```
Given: $(x_1,y_1),\ldots\ldots.,(x_m,y_m)$ where $x_i \in X$, $y_i \in Y=\{-1,+1\}$
Initialize $D_1(i)=1/m$.
For $t=1,\ldots,T$:
    • Train weak learning using distribution $D_t$
    • Get weak hypothesis $h_t$: $X \rightarrow \{-1,+1\}$ with error
          $\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$
    • Choose $\alpha_t = \log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$
    • Update:
          $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha t} & if \quad h_t(x_i) = y_i \\ e^{\alpha t} & if \quad h_t(x_i) \neq y_i \end{cases}$
          $= \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
          where $Z_t$ is a normalization factor (choosen so that $D_{t+1}$
          becomes a distribution)
```

```
Output the final Hypothesis:
        $H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$
```

Figure 2.4 Adaboost learning and recognition algorithm [54]

The main disadvantage of adaboost is that it is a binary classifier. The problem has been addressed to extend it to multi-class (actually multi-label) problem by putting multiple labels to a simple example. So, for multi-class problem, an example will retain the true label and in addition, will assume negative labels for all other classes (one-versus-all). Details can be found in [76] and an implementation has been provided named Adaboost.MH. However, a new algorithm, SAMME [77] has been proposed that looks almost same as original adaboost algorithm and does not need to do the cumbersome process of Adaboost.MH. Work [77] has shown that a simple addition of a term *log(K-1)* with the update equation can convert it to multi-class. So, the output, instead of taking the sign, the class having the maximum value is chosen (Figure 2.5). An implementation has been provided by the name ICSIBoost.

```
Given: a set of training example $(x_1,c_1),\ldots\ldots,(x_m,c_m)$
Initialize observation weights $w_i=1/m$, i=1,2,…,m
For $t=1,\ldots,T$:
    • Train weak learning $L^t(x)$ using weights $w_i$
    • Get weak hypothesis $h_t$ with error
            $\varepsilon_t = \sum_{i=1}^{n} w_i \mathrm{I}(c_i \neq L^t(x_i)) / \sum_{i=1}^{n} w_i$
    • Choose $\alpha_t = \log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$+log(K-1)
    • Update:
            $w_i \leftarrow w_i.\exp(\alpha_t.\mathrm{I}(c_i \neq L^t(x_i)))$ for i=1,…,n
            renormalize $w_i$

Output :
```

$$C(x) = \arg\max_{k} \left(\sum_{t=1}^{T} \alpha_t \mathrm{I}(L^t(x) = k \right)$$

Figure 2.5 Multi-class Adaboost

2.4 Frequent Itemset Mining

Mining is a set of techniques to find interesting patterns from data. Frequent Itemset mining is a technique to find association rules from large database. Aggarwal [78] popularized the concept of mining association rules from transactions in a supermarket to find out which items are sold together. He proposed Apriori algorithm [79] . It uses a breadth-first search strategy to counting the support of itemsets and uses a candidate generation function which exploits the downward closure property of support. Candidate generation and testing used by Apriori is costly, because it needs many database scans.

FP-growth (frequent pattern growth) [55] uses an extended prefix-tree (FP-tree) structure to store the database in a compressed form. FP-growth adopts a divide-and-conquer approach to decompose both the mining tasks and the databases. It uses a pattern fragment growth method to avoid the costly process of candidate generation and testing used by Apriori.

FP-tree algorithm takes two database scan. Since frequency of items play an important role, first the items are counted and item orders is decided. In the second scan, transaction are scanned, ordered and inserted into a tree starting from root node. So, if two transactions have a prefix in common, they share a common sub-tree and then divide at some point. The sub-tree generated from same prefix of multiple transactions sums up the frequency of the items. A quick lookup table is maintained for each item instances inside the tree.

Table 2.1 A transactional database

100	*f, a, c, d, g, i,m,p*
200	*a, b, c, f, l,m,o*
300	*b, f, h, j,o*
400	*b, c, k, s,p*
500	*a, f, c, e, l, p,m,n*

If we want to mine all the frequent patterns with support 3, then we get a FP-tree shown in figure 2.6.

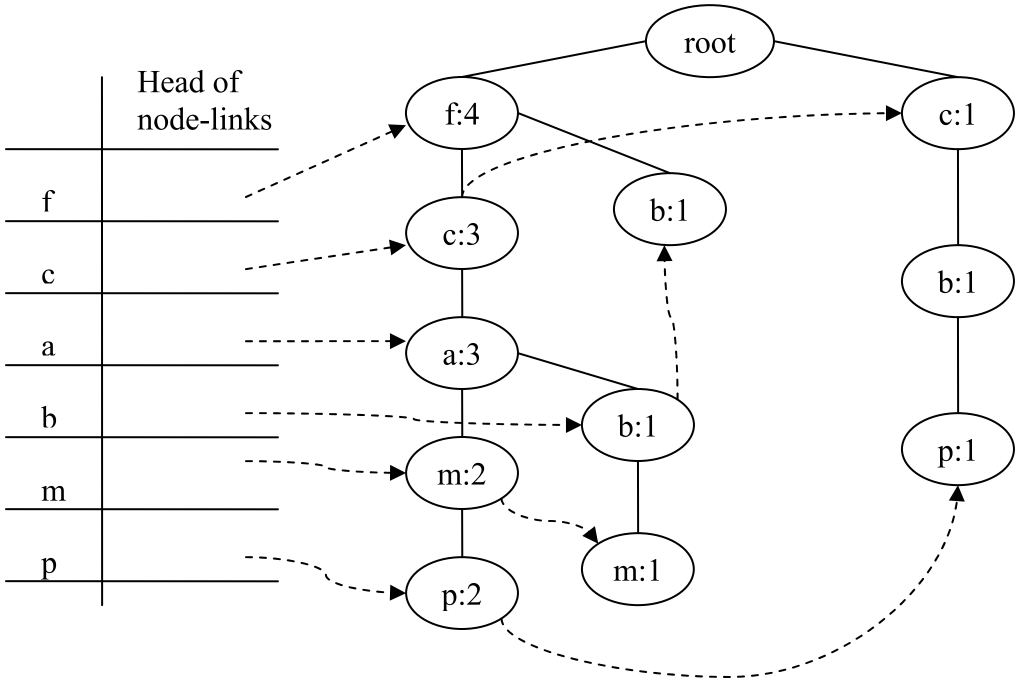
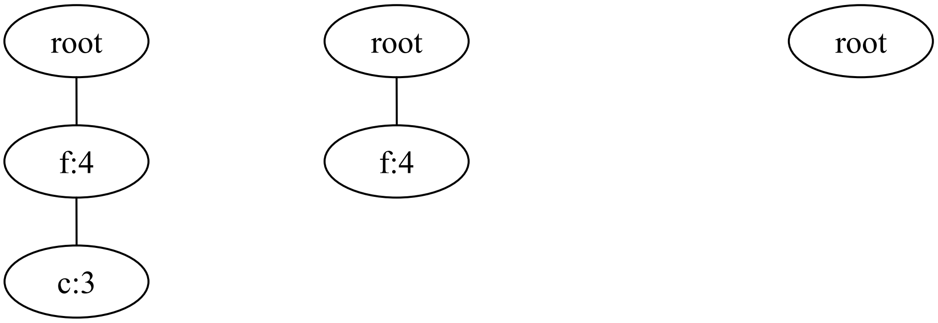


Figure 2.6 FP-tree constructed for table 2.1 and support 3.

FP growth technique uses the FP-tree in a recursive way by projecting the database. For any item, the projected database consists of a sub-tree with all the transactions in the FP-tree starting from the item to upwards (excluding itself) and removing all infrequent items. Figure 2.7 depicts the process.

I use FP-growth algorithm for mining sensor activation association. But my goal here is not to find simple association, rather to find temporal association. I use FP growth with a simple trick to mine interesting temporal sensor sequences.



Conditional sub-tree for 'a' Conditional sub-tree for 'c' Conditional sub-tree for 'f'

So, the frequent patterns are {a, ac, c, f, cf, acf ,af }

Figure 2.7 FP-growth technique

2.5 Interleaving of Activities

It is the nature of human to do multiple activities simultaneously or pause one activity, start another and then restart the previous one. For example, it is common for an individual to watch TV and answer the telephone at the same time concurrently. As an example of interleaving goals, an individual may be having his breakfast when the water boils, so he has to pause the activity of having breakfast and attend to the water boiling activity. Then he may return to the previous goal of having his breakfast.

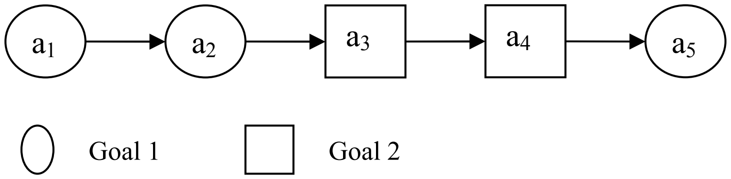


Figure 2.8 An example of interleaving goal

The higher level goals are assumed the more interleaving and concurrency is found. For example, there is a good level of concurrency between dishwashing and meal preparation, and interleaving between dishwashing and cleaning indoor. A more discussion about concurrent and interleaving goals can be found in [50,51].

2.6. Feature Selection and Dimension Reduction

One of the problems with high dimensional data is that not all the variables are important to understand the underlying phenomena. Dimension reduction is a technique or techniques to reduce the number of variables and still retaining the original content according to some criteria [89]. Mathematically given a p dimensional random variable $x=(x_1,x_2, \ldots, x_p)^T$ find a lower dimension representation of it, $s=(s_1,s_2,\ldots,s_k)^T$,$k<p$ and still capture the content of the original data. A comprehensive survey can be found in [89]. In this section I discuss some of the techniques in short in respect to their applicability in my problem domain.

PCA: Principal Component Analysis (PCA) seeks to reduce the dimension of the data by finding a few orthogonal linear combinations (the PCs) of the original variables with the largest variance. This is a second order statistical method. The first PC, s1, is the linear combination with the largest variance. We have $s1=x^T w_1$, where p dimensional coefficient vector $w_1=(w_{1,1},\ldots,w_{1,p})^T$ which is solved by:

$$w_1 = \arg\max_{\|w=1\|} Var\{x^T w\} \qquad \ldots.. (2.1)$$

The second PC is the linear combination with the second largest variance and orthogonal to the first PC, and so on. There are as many PCs as the number of the original variables. For many datasets, the first several PCs explain most of the variance, so that the rest can be disregarded with minimal loss of information. PCA does not take class labels into account.

In case of binary sensors, the variability of the sensor values is not meaningful. If sequence values are taken as variables, there are huge set of variables that the matrix multiplication cannot be done in lifetime.

LDA: Linear Discriminate Analysis (LDA) [110] takes into account class labels to find inter-class correlations in the dataset. LDA projects features that maximally account for this inter-class difference.

In my observation LDA has the same drawback as PCA for binary sensor data.

Information Gain: Information gain selects and test feature values based on mutual information among the features. In brut force method, all the possible combinations are tried which is impractical for high dimensional data. Usually, for large data, heuristic methods are applied to choose feature values to test.

In binary sensor data, reducing the number of sensors by using information gain will not help in activity recognition. In case of sensor sequence selection, heuristics is needed to select best values. Frequent pattern mining is a good heuristics. However, data mining does not need the help of other techniques.

Chapter 3

Related Research Works

First I discuss some research works in three areas: video based activity recognition, activity recognition using environmental sensors and wearable sensors. Then I discuss some distributed activity recognition efforts. Lastly, I discuss on mining and tracking for activity recognition.

3.1 Video Based Activity Recognition

Given a sequence of images with one or more persons performing an activity, can a system be designed that can automatically recognize what activity is being or was performed [2]? Video based activity recognition systems, try to solve the problem stated above. The major steps involved are: Extraction of concise low-level features, Mid-level action descriptions from low-level features, High-level semantic interpretations from primitive actions.

Videos consist of massive amounts of raw information in the form of spatio-temporal pixel intensity variations. But most of this information is not directly relevant to the task of understanding and identifying the activity occurring in the video. So, low level features are collected first. Few approach for creating features are optical flow [2], point trajectories [2], background subtracted blobs and shapes [2], filter responses [2].

Approaches for modeling actions can be categorized into three major classes – non-parametric [2], volumetric [2] and parametric time-series [2, 56-61] approaches. Non-parametric approaches typically extract a set of features from each frame of the video. The features are then matched to a stored template. Volumetric approaches on the other hand do not extract features on a frame-by-frame basis. Instead, they consider a video as a 3D volume of pixel intensities and extend standard image features such as scale-space extrema, spatial filter responses etc to the 3D case. Parametric time-series approaches specifically impose a model on the temporal dynamics of the motion. The particular parameters for a class of actions is then estimated from training data. Examples of parametric approaches include Hidden Markov Models (HMMs), Linear Dynamical Systems (LDSs) etc.

From actions of individual persons and complex activity is made. In this level different graphical models such as Bayesian Network [62], Dynamic Bayesian Network (DBN) [63-65], Petri Nets[66,67] or Knowledge Based rules [68-70] and Ontology [71-73] are used.

Here it is notable that, Bayesian Networks or DBN are used at the top of the inference hierarchy, to reduce the number of input and dependencies to model. Otherwise, the network may become intractable. Also note that temporal sequence is needed to discover actions a person is doing. Learning has also been proved difficult in these networks as either they need exponential dataset, or need expert to hand tune the probabilities.

3.2 Activity Recognition using Sensors Deployed in the Environment

Given a set of sensors deployed in the environment and with house hold utensils, can we develop a system that can recognize with the user/users is/are doing? This is the ideal setup for any activity recognition system, as it does not hamper privacy and also users are not burdened with carrying special devices. The work that popularized the concept was [8] in MIT. They deployed 77 binary on/off state sensors in a house and used a PDA to label the activities the person (subject 1) was doing. In another house they deployed 84 sensors for subject 2. However, the data collected from subject 1 has better quality in terms of labeling and noise. In our work , we used the data collected from subject 1 only.

In work [8] different feature window was taken for each activity, each window being the average of the activity. The feature window was shifted 3 minutes forward every time. Within the window, sensor activated (exists, E) and Before Time (BT) feature was calculated. The features were then fed to Multiclass Naïve Bayes classifier. The window size being very large, Before Time feature did not work better. Again as the number of features to be calculated for higher order sequences is huge, they did not go for that. The system with the 3 minutes sliding window, is not real-time.

The experience of the system deployment and pros-corns was shared in paper [10]. The system was further upgraded to incorporate RFID sensors, Tap sensors, Current sensors, Accelerometer sensors, motion sensors to find out which work better and the result was reported in [11]. It was found that motion sensor performs better. They also reported that RFID sensors are difficult to deploy with daily utensils and tags are not always read. They also suffered from labeled data, as users were asked to live as naturally as possible. From video labeling 104 hours of data were collected.

The idea of deploying sensors in the environment was picked in [12]. They deployed 14 on/off sensors in the doors and recognized 7 ADLs only. They reported that change events are the most useful features. The event continues to be valid until another sensor event is fired. They used HMM and CRF for recognition. Change event is useful but reduces accuracy for activities, especially idle activity. HMM and CRF used conditional probabilities for activities, *P(Activity1|Activity2)* that is subject to change from user to user and time to time. Moreover with higher number of sensors, the classifiers fail to load the probability tables in memory which they failed to observe. They also reported that the classifiers provide highest accuracy when the data of the whole day is fed. That means the algorithm is not real-time.

3.2.1 Activity Recognition using RFID

RFID based activity recognition has been studied first in [9,26] and then in [11]. Even though RFID based systems promise better results, they are actually impractical for ADLs as studied in [11] in naturalistic settings.

3.3. Activity Recognition using Wearable Sensors

Activity recognition using wearable sensors, especially using accelerometers has been studied for a long time. Though the accuracy is promising for basic activities like walking, running, standing, sitting etc, they are not suitable for ADLs, such as cooking, toileting. However, researchers are putting their efforts into it. Work [22] recognizes some dietary activities using accelerometer. Work [23] provides a mobile platform for recognizing basic activities but they do not indicate anything about orientation and position freeness. Work [6] also shows promising results for few basic activities, but it also suffers from position and orientation problem.

Still much effort has been made to recognize physical activities from accelerometer data over the last few years. I discuss some those for the wide range of classifiers used such as dynamic and static Bayesian networks (e.g. [28, 29, 31, 32]), nearest neighbors, decision trees, and support vector machines (e.g. [28, 30]). Among the most popular classifiers applied to the problem are dynamic Bayesian networks (DBNs). A DBN is formed when a static Bayesian network is unrolled over several times slices (time intervals) and graphical dependencies are specified among them. These dependencies among consecutive time slices allow DBNs to capture first order temporal information of the problem at hand. DBNs also allow common sense knowledge of the problem to be encoded in the internal structure of the network by manually specifying the nodes and links of the network. For example, the work by Raj et al. [31,32], handcrafted the internal structure of a DBNs to simultaneously recognize human activities from accelerometer data and location from GPS traces. This approach is not scalable since it depends on experts coding common sense knowledge in the network structure that is likely to change depending on the activities to recognize. Due to the high computational complexity necessary to perform exact inference in DBNs, approximate inference algorithms such as particle filters [33] are often utilized [31,32]. The number of particles used by this algorithm to perform inference can be adjusted depending on the processing power available on a particular device (e.g. in handheld devices); Nevertheless, its performance degrades as the number of particles is reduced.

Simpler DBNs such as hidden Markov models (HMMs) that have shown excellent performance in speech recognition applications have also been widely applied to classify activities from accelerometer data (e.g. [29, 34-36]). The most common approach used is to train one HMM per activity to recognize using the Baum-Welch algorithm [37]. Once the models are trained, the classification is performed by choosing the model (HMM) that results in the highest log-likelihood over the observation sequence (sequence of feature vectors) as computed using the forward-backwards algorithm [37]. Even though this approach successfully incorporates intra-activity sequential information, its main disadvantage is its high computational requirements because one HMM per activity to recognize is required. Moreover, the forward-backwards algorithm has to be run as many times as there are activities (i.e., HMM models) to recognize so it is computationally expensive. Another disadvantage of this approach is that the number of internal hidden states needs to be specified a priori either using expert knowledge of the activities structure or learning it from training data using a cross-validation procedure. Another approach to recognizing activities using HMMs is to use a single HMM where each internal state represents one of the activities to recognize [38]. In this approach, sequences of activities performed can be inferred online using the Viterbi algorithm [16, 37] or particle filtering [33]. The main advantage of this approach is the incorporation of information about the transitions between activities (inter-activity sequential information) via the transition matrix of the system and the reduction of computational complexity (with respect to the use of one HMM per activity). One obvious disadvantage of this approach is that since only one hidden state is used to represent each activity, some internal temporal structure for each activity is lost (intra-activity sequential information).

Other less computationally expensive Bayesian networks such as the naïve Bayesian classifier [39] have also been applied to detect activities from wearable accelerometers [28,30,40,41]. In the naïve Bayesian classifier, the class (or activity) node is the parent to all attribute nodes (or features) and thus, its main assumption is that all the attribute variables are conditionally independent given the class. It calculates the most probable class given the data (attributes) using Bayes rule. Despite its simplicity, this classifier has obtained excellent results with respect to more complex classifiers in realistic datasets [42-44], One disadvantage of simple Bayesian networks is that they do not capture temporal information automatically unless it is encoded in the features extracted. The main advantage is that naïve Bayesian classifiers are fast to train and also perform fast classifications in comparison with DBNs. Decision tree (DT) classifiers such as the C4.5 algorithm [45] are among the most used to recognize activities from wearable accelerometers [28,30,40,46,47]. This is because decision tree rules are easier to understand. A disadvantage of decision trees is that they tend to over fit the data if they are trained on small datasets and may not combine probabilistic evidence as well as other methods. Furthermore, decision trees are static classifiers that do not incorporate any temporal transition information of the modeled activities unless it is encoded in the features used.

The work [30] explored the performance of different classifiers in recognizing activities such as decision tables, decision trees, support vector machines, nearest-neighbor, and naïve Bayes classifiers individually and in different meta-classifier configurations such as boosting, bagging, stacking, and plurality voting. The results presented were evaluated over data collected from two participants performing eight activities times while wearing a single triaxial accelerometer at the pelvic region. In general, it was found that meta-classifiers outperformed base-level or single classifiers as expected.

3.4. Distributed Activity Recognition

Distributed activity recognition is quite a new concept. The first approach of this kind is found in [15]. They use the concept of hierarchical feature extraction from tracking data, such as, ‘the cluster head detects in which direction the user is moving’ and from that calculates few office activities, such as, chatting, meeting. They compute only 17 motion based features, which are not enough for recognizing Activities of Daily Life (ADLs). Work [19] has used fuzzy logic for inference where features from low level signals are processes within the sensors. The used the data of a car assembly scenario with only 12 sensors. Context Zone [20] proposes a theoretical framework that sensors from collaborative groups of which a more capable sensor takes the role of processing node.

3.5 Mining for Activity Recognition

Data mining has been proposed to find out periodic patterns for behavior learning in [21]. The patterns are then used match similar pattern. Periodic activity patterns are used for activation of services in home. If the user changes the habit, the system uses some adaptation algorithm to learn that.

In my work, I am using mining to find patterns (may not be periodic) to recognize ADL, which is different than finding period patterns re-occurring. I am not using day and time rather the patterns to classify the patterns into activity labels.

3.6 Event Attribution to User

The most desirable setup for Activity of Daily Living (ADL) at home environment is to deploy simple ubiquitous sensors. MIT has been working to make a Living Laboratory [10], [11] from where naturalistic data can be collected. Work [12] has used sensors deployed on the doors to detect few ADLs. However, it has been pointed out in [11], if multiple users are present, the second user inserts noise in the data collected which significantly reduces the accuracy of the training of the classifier and recognition of ADL. So, ID sensor can necessarily solve this problem.

Passive RFID deployed with household utensils and users carrying the reader [9] could be a good solution for reducing noise from other users. However, RFID tags cannot be attached with all the utensils (such as mug that is put inside the micro woven)

[11]. Moreover, RFID reader needs to be in close contact with the tags that discourages tag deployment in doors and furniture. So, a mixed deployment of sensors and RFID tags are more likely where users carrying ID sensors are very much helpful. Without loss of generality, RFID tag readers can be augmented with a beacon sensor. Researchers are also experimenting with body wear sensors for detecting few ADLs like dietary activity [22] due to the fact that body wear sensors provide higher accuracy for fine grained activities as such. A prototype has also made to detect activities by embedding sensors on mobile phone [23]. So, it is not much demanding a beacon sensor to be carried by the user. Interestingly, all the above approaches are essentially trying to solve the problem of single user activity. Multi object recognition problem can be approached with tracking and filtering.

Tracking can detect key location based activity detection. STAR [24] proposes a location based tracking and activity detection. But the idea does not scale to the environment with many simple sensors deployed. Because, the filtering algorithms used for tracking needs input of domain or common sense knowledge otherwise the reasoning may take huge computation and memory and can in fact become intractable. For deployed sensors there are enormous ways of interacting with them that restricting the search space for the filtering algorithm is difficult and hence the system may make wrong decision about users' activities. ID sensor can generate proximity event along with user's identification. So, the system is not left with guessing about users' movements. The failure of tracking system for multi user activity detection in home environment is the main motivation for us to propose ID sensors.

Attributing events to individuals in multi-inhabitant environment has also been addressed in [80]. They proposed using history of individual's habit to be used for attributing an event. The idea suffers from similar problem of a tracking system and a lot of false attribution may occur.

Chapter 4

Activity Recognition using Mining and Multi-class Adaboost

In introduction, I emphasized and in related works it is seen that capturing temporal relationship is the way to best recognize activities. However, making sequences are expensive. Capturing and calculating sequence information as features is huge and very computation intensive. Suppose, we want to calculated a sequence length of *L* and there are *n* sensors with *s* possible states, then there are $(ns)^L$ number of possible features that can be constructed. Given 100 sensors, and 5 sequence length, the number of possible activation sequence will be 100^5. Calculating such a huge number of features real-time and feeding to classifier is impossible. An alternative to calculating sensor sequences is to put the observation sequences to Hidden Markov Model (HMM). However, the key to the success of HMM is to provide longer sequence of observations which is unlikely to get in real-time activity recognition system. HMM uses conditional Probability of Activities, $P(Activity_1|Activiyt_2)$ and much of their results are dependent on the exact measurement of these probabilities. But user behavior usually varies person to person and even change time to time. If a person goes to toilet every time he wakes up from bed, P(toileting |wake up) will be almost equal to 1 and P(other activity | wake up) will be almost equal to 0. This will lead to misclassification of the activity because most of the emphasize is given to activity transition.

HMM suffers the dimension problem for storing the probability tables. With *n* sensors each with *s* states, there are multiple of s^n observation probabilities, if all the sensors together represent the state of the environment. With 100 binary sensors we have multiple of 2^{100} ($\approx 10^{30}$) observation probabilities, which cannot be loaded on any available disk size. So, it is not lightweight.

The goal of this part of thesis to use a small data window from which sequence information is to be calculated. The dimension of the features needs to be reduced. We also need a classifier that can handle or make use of the features. Based on this goal I used four types of features:

1) Maximum Likelihood (ML)
2) Last Sensor (LS)
3) Sensor Sequence (SS)
4) Sensor Group (SG)

I use Multi-class Adaboost as the classifier based on the four types of features. The motivation behind using Adaboost is that it can use weak features and produce a strong classifier based on them. Adaboost has the added advantage of reducing feature dimension which will facilitate real-time activity recognition by reducing the feature calculation time. During the training of Adaboost, number of sensor sequence and sensor group features proved to be huge and needed to be reduced. I used frequent item set mining technique to perform the feature dimension reduction. The features thus selected are then fed to Multi-class Adaboost for testing.

4.1. Training and Testing of the Classifier

Like every supervised learning algorithm, Adaboost needs to be trained before it can recognize the activities. I use part of the labeled data to train the classifier and part of the data for testing the accuracy. The selection of training and testing data is varied several times and then results are averaged. The process is known as cross validation.

4.1.1. Training Steps

-Decide the sliding window:

A sliding window is a time window that slide by a time offset. The sensor data that falls into the window can be used to make features. For real-time activity recognition, the window should be small. I used 20sec window and 5 sec slide.

-Prepare Training data:

Sample sensor data and activity label each 5 sec.

-Mine the Sensor Sequence and Sensor Group:

Slide the window over the data, process the sensor events according to mining algorithm and store in file(s). The file can be used by the mining algorithm to find patterns.

-Prepare the data for Adaboost Training:

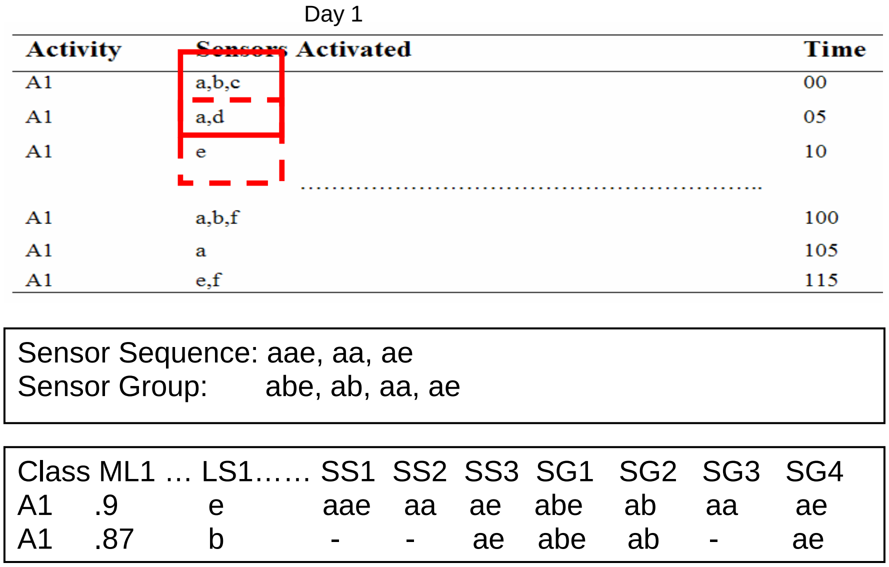
Take a window of data, calculate feature vector. Decide the activity label. As there might be activity transition over the same time period, this is important for the data preparation. I take the activity label which has the most occurrences within the window. If two labels have equal number of occurrence, I take last one. The prepared data is something as shown below:

Class ML1 ML2 …ML19 LS1…LS19 SB1…SBi…GB1…GBi….

The training data is a matrix. Each row of the matrix has the class label, the maximum likelihood (ML_1,…,ML_i,…) and last sensor value (LS_1,…,LS_i,…) for each of the activities, sensor sequence beans (SB_1,…,SB_i,…) and sensor group beans(GB_1,…,GB_i,…). Many of the sensor sequences and sensor groups may occur within the same window and the matrix does not allow multiple values to co-exist in the same position.

-Train the Adaboost.

Decide a weak learning algorithm that will find out hypotheses. Decide the number of hypothesis. If the emphasis is on accuracy, then leave the task on the Adaboost which will know when to stop.



Day 1

Activity	Sensors Activated	Time
A1	a,b,c	00
A1	a,d	05
A1	e	10
	…………………………………………………………..	
A1	a,b,f	100
A1	a	105
A1	e,f	115

Sensor Sequence: aae, aa, ae
Sensor Group: abe, ab, aa, ae

Class	ML1 …	LS1……	SS1	SS2	SS3	SG1	SG2	SG3	SG4
A1	.9	e	aae	aa	ae	abe	ab	aa	ae
A1	.87	b	-	-	ae	abe	ab	-	ae

Figure 4.1 Training Steps

4.1.2. Testing Steps

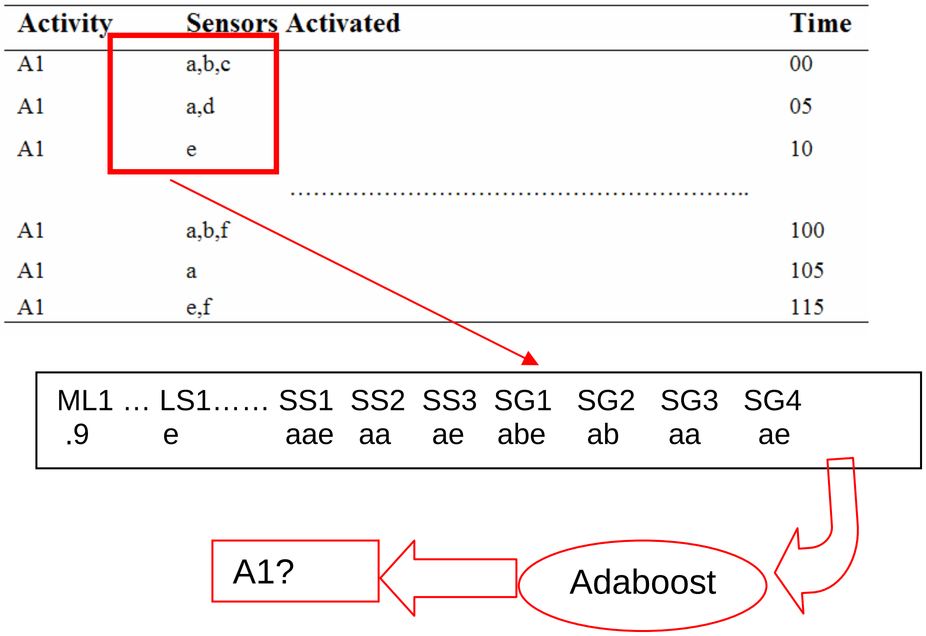
- *Slide the window over the testing data sampled in each 5 sec.*

-*Decide the actual activity label:* Majority, and if equal, select last one

-*Make the feature vector*

-*Feed the feature vector to Adabboost*

-*Measure the correctness of the output by comparing with the actual.*



Activity	Sensors Activated	Time
A1	a,b,c	00
A1	a,d	05
A1	e	10
	……………………………………………………….	
A1	a,b,f	100
A1	a	105
A1	e,f	115

ML1 … LS1…… SS1 SS2 SS3 SG1 SG2 SG3 SG4
.9 e aae aa ae abe ab aa ae
A1?
Adaboost

Figure 4.2 Testing Steps

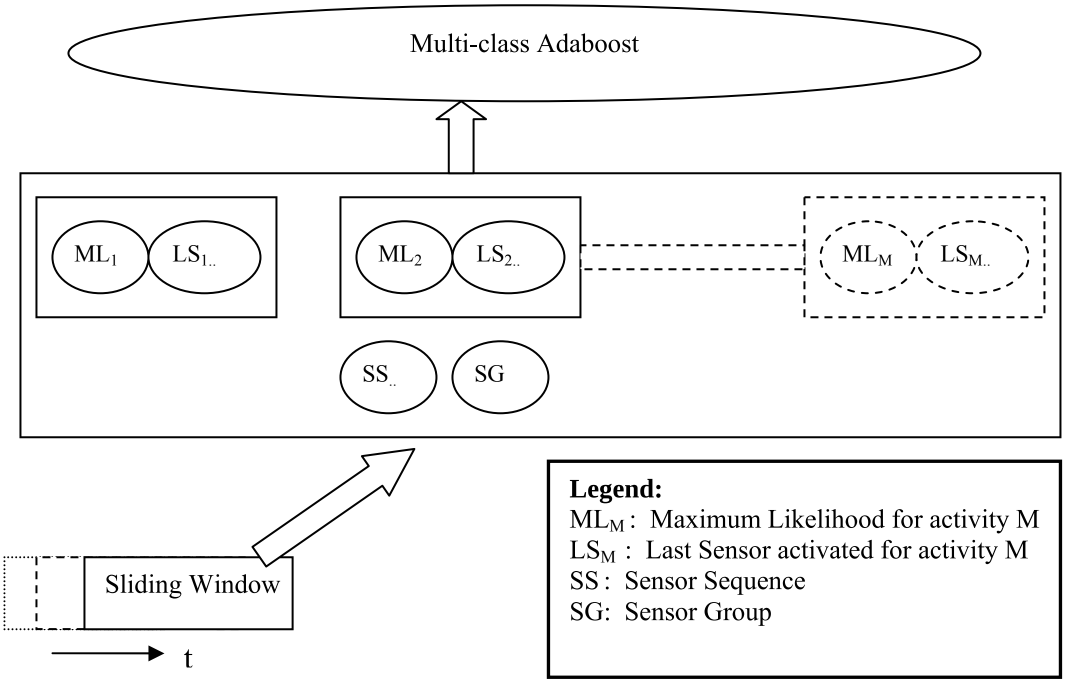


Figure 4.3 classifier

4.2. Feature Selection

4.2.1 Maximum Likelihood (ML)

Sensor sequence can help recognizing what the user is doing. First I obtained numeric value from the sequence. If we use a probabilistic approach then, the maximum probability value for an activity will be obtained from the most likely activated sensor sequence. With this in mind, I take sensor activation samples after a fixed time interval and place them in layers; the most recent activated sensors on the top (see Figure 4.4). If there is no activated sensor in any time slice, an idle sensor is assumed which favors all the activities equally. The number of layers to be constructed depends on the maximum sequence length we are taking.

Each activated sensor is assumed to be a state in that layer or time slice. Connection is assumed from all the states of previous layer to the next layer. As no concurrency is modeled, no connection is assumed among the states in the same layer or time slice. Probabilities are assumed for each state and each connection in the structure. The probability for each state is actually the prior probability for the sensor for the activity. The probability for each connection is the conditional probability of a sensor in a layer given the sensor in the previous layer.

4.2.1.1 Maximum Likelihood Value Calculation (MLVC)

Let us assume we are to calculate the maximum likelihood value for a sequence length of T. Before introducing the calculation algorithm, let us introduce the terminologies:

$P_a(S_i)$ =Prior probability of sensor i for activity ‘a’

$P_a(S_j \mid S_i)$ = conditional probability of S_j given S_i for activity ‘a’

$P_a(O_{oi})$ =Probability of output O_o from sensor i for activity ‘a’

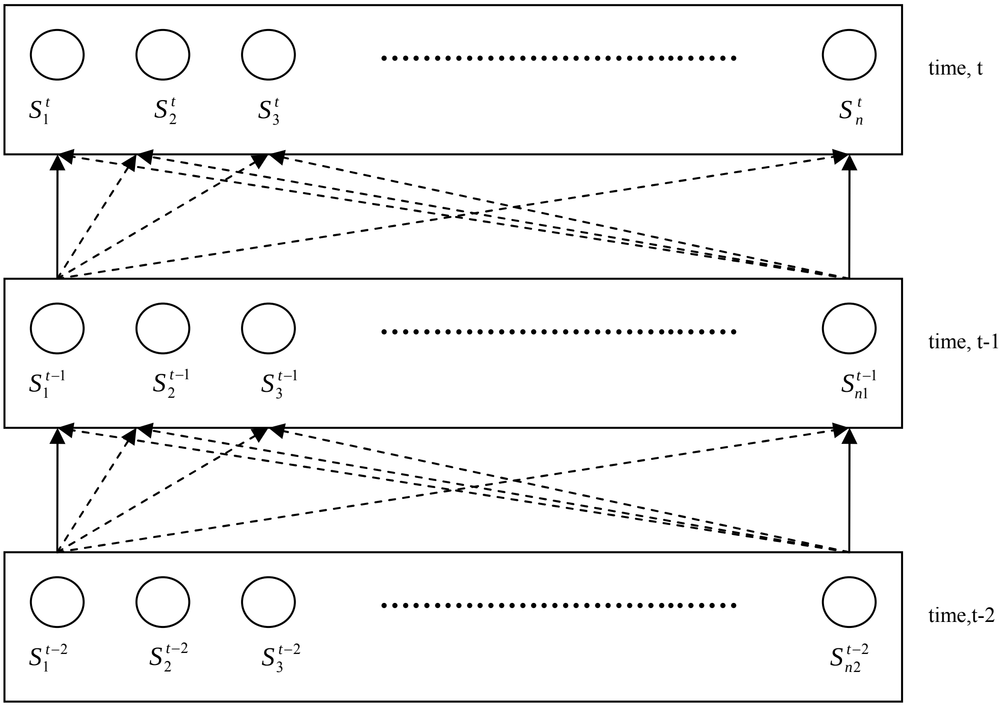


Figure 4.4 layer constructed with the sensors in the last three time sequences for any particular activity. By box and arrows I indicate that all states in time (l-1) are connected with all the sensors in time l.

S_i^t= State constructed with sensor i at time t

$P_a(S_i^t)$ =Calculated probability (usually maximum likelihood) of S_i^t for activity 'a'

$P_a(S_j^t \mid S_i^{t-1})$ =State transition probability of S_j^t given S_i^{t-1} $=P_a(S_j \mid S_i)$

So, once the structure is constructed with sensors activated in T time slices, I calculate the maximum likelihood value using the algorithm described by equation 4.1. This is a standard message passing algorithm [17,18] adopted to my network. Each state calculates the maximum likelihood using the formula below:

$P_a\,(S_i^l) = P_a(S_i)P_a(O_{oi})\,,$ l=last sequence

$P_a\,(S_i^l) = P_a(O_{oi})\max\limits_{c}[P_a(S_i^l \mid S_c^{l-1})P_a\,(S_c^{l-1})]\,,$ otherwise ………. (4.1)

It is important to understand how the MLVC calculation is done. Each state actually calculates the likelihood based on the values calculated in the previous states only. The state gets the values from previous states, multiplies them with corresponding conditional probability $P(S_{i+1} \mid S_i)$ and takes the maximum of those multiplication results. Then accepted multiplication result is further multiplied by the output probability. The first state does not have any previous states and so takes the prior probability multiplied by output probability as the likelihood at that point.

As the term ‘maximum’ is distributive in nature, it is possible to calculate the global maximum likelihood from local maximum as shown in equation 4.2 and Figure 4.5. From equation 4.2, each sensor gets the maximum from the previous sensors activated, multiply that with a local term $P(S_{i+1} \mid S_i)P(O_{i+1})$ and then take the maximum of those multiplication results.

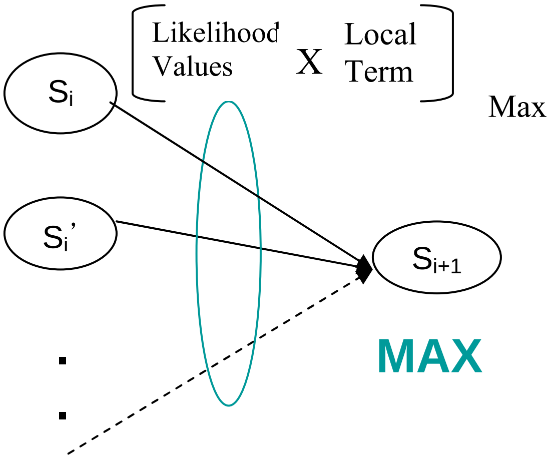


Figure 4.5 Nature of maximum likelihood calculation

$$P(O_iO_{i+1}.....O_{i+N}) = \max\begin{bmatrix} ....\max\begin{bmatrix} \max\left[P(S_i)P(O_i)\right]\times \\ P(S_{i+1} \mid S_i)P(O_{i+1}) \end{bmatrix}\times \\ P(S_{i+N} \mid S_{i+N-1})P(O_N) \end{bmatrix} \qquad \ldots\ldots\ldots\ldots\ldots\; (4.2)$$

So, from a data window of T time slices, maximum likelihood for each activity can be calculated. In work [25], I used maximum of those values as classifier (i.e., the global maximum). The classifier was performing average. Here, instead of using the global maximum I choose to use maximum likelihood values as features. The values are quantized into several ranges. When quantized values are used as hypothesis, each of them does not have much recognition power (may be better than random). The range values work as hypotheses in Adaboost.

4.2.2. Last Sensor (LS)

Last sensor activated can provide a key event for some activities. For example, if the burner is on, it indicates cooking or if the shower is on, it indicates bathing. However, key sensors are not always activated. If sensors are placed in strategic places (knowing the key positions), it is possible to design a rule based activity recognition system using key sensor activation events. But in a randomly deployed sensor environment, it is not known whether there are any key sensors and whether they are on as long as the activities are being performed. That is why last sensor activated becomes a hypothesis in Adaboost, rather than becoming a key feature.

During the calculation of Maximum Likelihood, the sensor state (at T time instance) that provides the maximum value is the last sensor activated. The last sensor activated can be different for different activities.

4.2.3. Sensor Sequence (SS)

Besides calculating quantitative value such as likelihood, we also need qualitative features. The sequences sometimes can be indicative of activity performed. For example, going to kitchen, turning on the burner can indicate preparing meal. So, sensor sequences need to be fed as hypotheses in Adaboost.

However, the possible number of sequence of sensors is exponential and is very difficult to handle, though not impossible. With 100 sensors there are 100^5, 100^4, 100^3 and 100^2 possible sequences for sequence length 5, 4, 3 and 2 respectively. Not all the sequences are important or discriminating. Adaboost has the capability of finding the important features but handling such number of features is impossible. I use frequent item set mining technique: FP-tree and FP-growth on the training data to find out the important sequences first.

I use the intuition that a sequence commonly occurring within an activity period indicates that the pattern is important for the activity. A sequence occurring in an activity period and not found in another activity will become a discriminating feature. The task of finding such sequences is left to Adaboost. I use mining to find important sequences for all activities. Patterns are mined for each activity which are then merged together for the training of the Adaboost. Suppose we have a training dataset in Table 4.1 for activities A1 involving sensors a,b,c,d,e,f.

Table 4.1 Sample training data for activity A1

Activity	Sensors Activated	Time
A1	a,b,c	00
A1	a,d	05
A1	e	10
	…………………………………………………..	
A1	a,b,f	100
A1	a	105
A1	e,f	115

FP- growth works on transactional database. So, I will have to make transactions for each activity, including the sensors only. As I am interested in sequences several records from the training data need to merged in one transaction. The sensors are post fixed with a number indicating the appearance of it in the sequence. So, if we want to make transactional database for activity A1, the database looks something as shown in Table 4.2. Assume that we want to construct a sequence of maximum length 3.

Table 4.2 Transactional database for sensor sequence mining for activity A1

Transaction#	Items
1	a1,b1,c1,a2,d2,e3
2	a1,d1,e2
3	e1
4	a1,b1,f1,a2,e3,f3
5	a1,e2,f2
6	e1,f1

With the transactional database constructed, an FP tree is built from which frequent patterns are mined. In a transaction, a sensor might occur more than one time but they have different item names. Now if I mine the transactional database, I will get a FP tree. In FP-growth technique, I restrict mining operation such that no single item pattern is mined; no patterns are mined with same postfix (to restrict concurrent) and no final pattern not having postfix numbers in sequence. So, if I mine the database for support 2, I will get only the patterns {a1a2e3, a1a2,a1e2} which is equivalent to {aae, aa, ae}. The FP-tree constructed is shown in Figure 4.6.

The patterns derived from same sub-tree can occur concurrently in the real environment. So, categorizations of those patterns are needed so that they fall into different categories. I term those categories as bucket. During the customized FP-Growth mining (with the mining constraints applied) it is also possible to sort the patterns into different buckets. Each of the buckets then becomes a feature column for the training of the Adaboost.

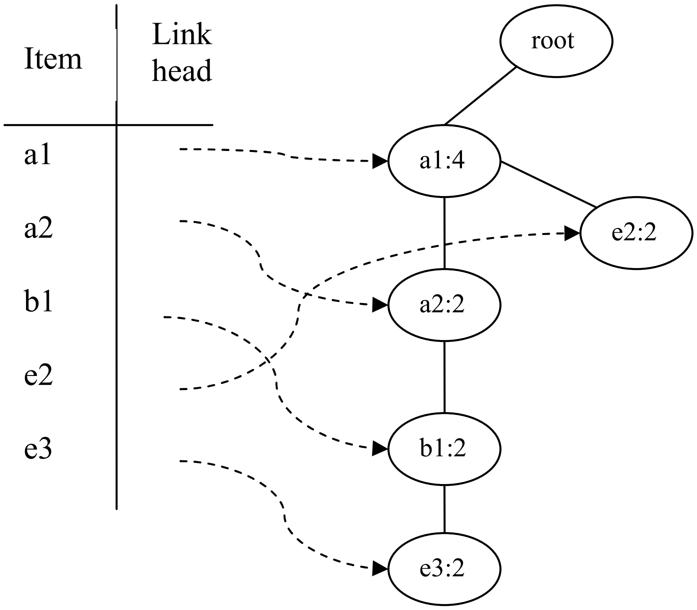


Figure 4.6 FP-Tree constructed for the transactional database in Table 4.2

4.2.4. Sensor Group (SG)

Sensors activated during a period of time are indicative of activity being done. For example, refrigerator door open, kitchen cabinet open may indicate some meal preparation activity being performed. That is why; group of sensors activated is taken as hypotheses in Adaboost.

There are ${}^{100}C_5$+${}^{100}C_4$ +${}^{100}C_3$ + ${}^{100}C_2$ (approx. 100^4 or 10^8) possibilities of grouping of sensors given 100 sensors and allowing a maximum of 5 sensors in a group. Like sensor sequence it is also huge. So, I needed to mine them also. The mining process is same as the sensor sequence mining. But no postfix is added to the sensors. Duplicate sensors are merged together as sensor group is considered, not sequence. During FP-Growth, only restriction is that no single item pattern is mined. Bucket is also constructed for sensor group. Given the training data in table 4.1, the transactional database for activity A1 is shown in Table 4.3 and FP-tree for support 2 is shown in Figure 4.7.

Table 4.3 Transactional database for sensor group mining for activity A1

Transaction#	Items
1	a,b,c,d,e
2	a,d,e
3	e
4	a,b,f,e
5	a,e,f
6	e,f

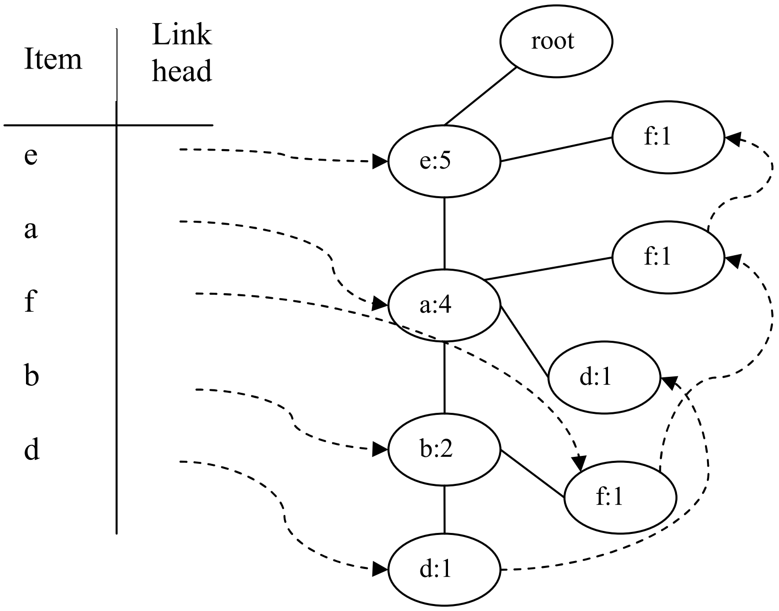


Figure 4.7 FP-Tree constructed for the transactional database in Table 4.3

4.3. Experimental Results

4.3.1. About the Data and Results

I am using the open data provided by MIT Place Lab [8,14]. They used 77 on/off state sensors in a one person apartment. The person lived there for two weeks and was given a PDA which provided periodic beeps to get input about what the user was doing. The method of labeling was named Experience Sampling Method (ESM) [8]. When the user used to do something, events were fired and the sensors sent the event to central server.

The server kept the time stamped sensor event along with the labeling in a log. Often the user forgot to label the activities. Those periods were later presented to the user to label from their memory. Often the user forgot to turn off the sensors and those were added as noise to the data (a sample was shown in figure 5.1). Sometimes the user used to interleave between activities but only one label was taken. There were 19 activity instances found in the data, namely, bathing, toileting, going out to work, preparing lunch, preparing dinner, preparing breakfast, dressing, grooming, preparing a snack, preparing a beverage, washing dishes, doing laundry, cleaning, putting away dishes, washing hands, putting away groceries, watching tv, going out to entertainment. The number of example instances for each activity is small and contain noises.

The result was calculated for each activity on per frame basis. Frame is the data window on which the recognition algorithm is run.

Activity Accuracy = (Number of frames correctly classifier for the activity X 100)/
(Total number of frames for the activity)
Average Accuracy = (sum of Activity Accuracies) / (Number of Activities)

Work [14] reported a maximum of 27% time frame accuracy for 8 activities for the MIT data set. They used multi-class Naïve Bayes classifier and a window size of 7min30sec and 3 min sliding window. They used Last Sensor and a sequence of length 2. Due to cost of feature calculation they did not use longer sequence. Temporal relationship of sensors in 3 minute interval actually did not help increasing the accuracy.

There is another work [12] that used only 14 state sensors placed strategically (4 door sensors) to recognize 7 ADLs. The proposed a feature that last sensors state change event should be carried until another sensor event occurs. The feature was useful to capture door sensor change events and detect room level activity like sleeping very well. But the feature includes idle states into activity. The feature also does not allow parallel sensors activations. Taking just sensor activation information as events, they got relatively lower accuracy even using a temporal classifier. 14 sensors are much less for a environment I am considering. With such small number of sensors, there is no simultaneous activation of sensors which is a basic assumption of my algorithm.

Table 4.4 Results for two related works

Work	Real time?	Window size	Feature	Classifier	No of sensors	Accuracy (time slice)
MIT [8]	No	7min30sec	Sensor activated, Sequence of 2 length	Naïve Bayes	77	27% for 8 activities
AMSTERDAM [12]	No	Whole day	Last change event continued	HMM,CRF	14, Strategically placed	HMM:79.4% CRF:70.8% For 7 activities
			Sensor activated			HMM:49.2% CRF:44.6%

I used MIT data to measure the accuracy of my algorithm. My work uses small window size and a moderate number of features and provides better accuracy. For accuracy comparison purpose, HMM was not possible to be implemented because of prohibitive probability table size. The drawbacks of using HMM for ADLs have been discussed before. Still for the sake of comparison, I report another work [83], where HMM and CRF is used with each sensor assumed independent. In implementation each activity is taken as a hidden state of HMM and the sensor values of the whole environment at a time instance is taken as the observation. But literally storing the observation probability table in intractable in space. So, instead of storing P(Sensor$_1$Sensor$_2$......|Activity) the algorithm stores P(Sensor$_i$|Activity) ,i=1,…, number of sensors. The probability for an observation {$Sensor_1$=on, $Sensor_2$=off}is calculated as P($Sensor_1$=on|Activity)X P($Sensor_1$=off|Activity). They reported accuracy for 8 selected activities taking a sequence length of whole day. They use activity transition probability. Making the activity transition probabilities equal and using small window will drastically reduce the accuracy. The observation probability and activity transition probability, i.e. P($Activity_2$|$Activity_1$) was learnt from the training data by counting.

Table 4.5 Comparison of results

	Window Size	Feature	No of features	No of classes	Accuracy
MIT result [8]	7min30sec	LS, SS (2-len)	77+77X77=6006	8	27%
My result [81]	20sec	ML,LS,SS (5-len max),SG (5-len max)	20,000	19	73%
Work [83] Modified HMM, CRF	Whole day	Senosor state	77, internally 2^{77}	8	HMM:66.18% CRF:54.4%

I experimented on the same data using HMM. To deal with probability table intractability, I used the same scheme as discussed in the above paragraph and modified the Viterbi algorithm to incorporate such observation probability calculation. Viterbi algorithm provides the best plausible sequence of activities given the observation sequence. However, in a real time system HMM needs to output an activity label in each time frame. To adapt, I provided an observation sequence to the HMM, ran Viterbi algorithm on that and took only the last activity label instead of taking the whole output sequence. The observation sequence was shifted by 5 second, and again the algorithm was run to find the most probable activity sequences from which only the last label was taken. I used the data for all 19 activities.

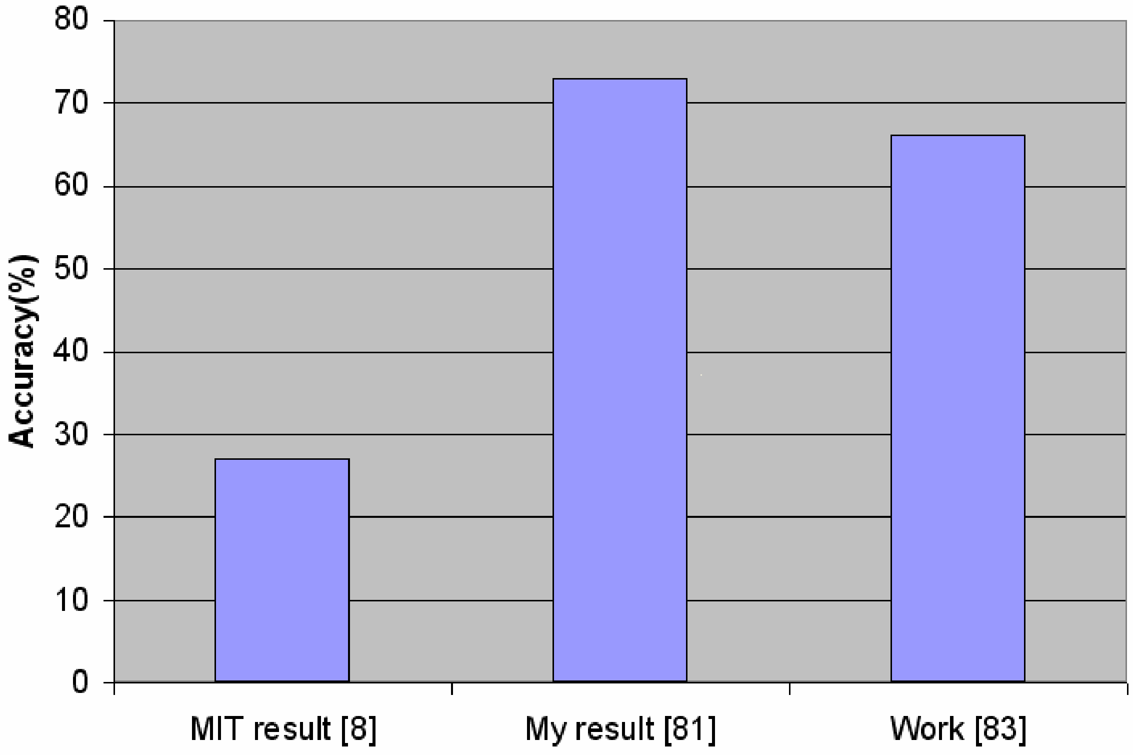


Figure 4.8: Comparison of results

I also studied the effect of activity transition probability, P(Activity$_2$|Activity$_1$). I made the transition probabilities equal for all, ran the HMM on sliding window of observation sequences. The results are shown in Table 4.6.

It is interesting to notice that the accuracy increases steadily with the increase of observation sequence length. This is because Viterbi algorithm optimized result based on the whole observation sequence. Even changing a part of the observation sequence will lead to a different result. Viterbi algorithm solves the number 2 canonical problem of HMM. Here Viterbi algorithm makes the calculation faster only by using a dynamic programming approach (reusing partial calculation results).

Table 4.6 HMM accuracy for different observation sequence length

Window Size	20sec	60sec	10min	60min	120min	Whole day
HMM	31.4%	31.8%	34%	41%	46.2%	58.3%
HMM with equal activity transition probabilities	30.8%	31.5%	33.5%	38.5%	44.3%	49.8%

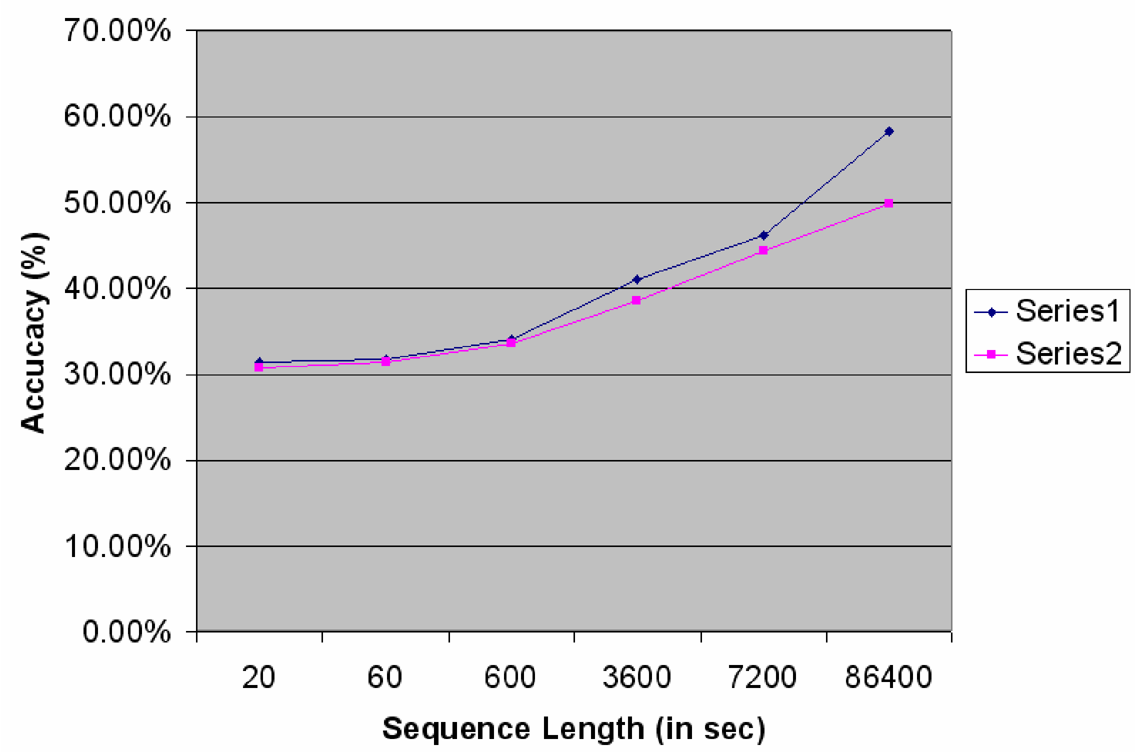


Figure 4.9: Results for HMM. Series1, Series2 present results for unequal and equal activity transitions respectively.

The observation probability P(observation|activity) is calculated by multiplying the individual sensor observation probabilities ∏ P(sensor$_i$|activity). Doing this, we essentially loose the relationship among sensors (as each sensor is assumed independent of sensor activation). Again, this does not take the contribution of the individual sensor activation sequences into account (as each observation is a combination of all the sensors).

4.3.2 Results for MLVC used as classifier

In my initial work [25], I used the maximum likelihood values to decide among activities. The global maximum among the values were taken and the activity having that value was selected as the activity being performed.

The purpose of using MIT data in my work is to show the effectiveness of my model in a classification problem, where temporal relationship among the features is necessary.

I use the text data from [14], take samples of sensor values at every 5 seconds and feed through my simulator. My program actually emulates the user activities recorded in the data file and provides a continuous sensor activation sequence needed for the algorithm. The program runs in two phases. First, it learns the prior and conditional probabilities for the sensors based on this data by frequency counting. In the second phase, it emulates the ADLs, reproduce the sensor sequences and recognize the activities using algorithm given by equation 4.1. As the sensors are just on/off sensors, a sensor is considered activated only when it is in on state. So, the only output in active state is on state and the probability of on state is equal to the prior probability of the sensor node, i.e. $P_a(O_{oi}) = P_a(ON_i) = P_a(S_i)$. So, $P_a(O_{oi})$ is replaced by $P_a(S_i)$ in equation 4.1.

I have found that, increasing the sequence length increases the recognition accuracy up to a certain limit. Using maximum likelihood belief propagation algorithm as given in equation 4.1, I have found average recognition accuracy as given in table 4.7, where 5 is the optimal sequence length. I also have tested our algorithm on the sensor data sampled in 1 second and 30 second intervals where I got the highest average accuracy for sequence length 5 (accuracy 47%) and 4 (53%) respectively.

Table 4.7 Average recognition accuracy for ADL data from MIT [14], using MLVC

Sequence Length	2	3	4	**5**	6	7	8
Accuracy	40	48	52	***54***	51	48	45

Table 4.8 presents the average accuracy results for all the activities given in [14] with a sequence length 5 and sensor values sampled at an interval of 5 seconds.

Table 4.8 Activity wise recognition accuracy of MLVC (5 sec interval, sequence length 5)

Activity	Accuracy (%)
Bathing	69.95
Toileting	56.03
Going_out_to_work	100.0
Preparing_lunch	35.63
Preparing_dinner	31.63
Preparing_breakfast	44.96
Dressing	69.76
Grooming	58.07
Preparing_a_snack	58.73
Preparing_a_beverage	70.0
Washing_dishes	76.66
Doing_laundry	61.33
Cleaning	18.57
Putting_away_dishes	81.25
Washing_hands	100.0
Putting_away_groceries	74.19
Watching TV	48.57
Going_out_for_entertainment	80
Lawnwork	97.43
Average Accuracy	54.02%

I believe that the medium accuracy (54%) of our result is because of the small training set and noise, which is also indicated in [8]. I looked at the confusion matrix of the ADLs and based on that merged a number of activities to prove our claim. I found that meal preparation activities contributed more to the confusion and merging them further improved the accuracy as shown in Table 4.9:

Table 4.9 Activity wise recognition accuracy , meal preparation activities treated as one

Activity	Accuracy (%)
Bathing	71.04
Toileting	56.47
Going_out_to_work	25.0
Meal preparation (Preparing_lunch, Preparing_breakfast, Preparing_dinner, Cleaning, Washing_dishes, Preparing_a_snack, Putting_away_dishes, Putting_away_groceries)	65.21
Dressing	72.44
Grooming	59.02
Preparing_a_beverage	75.0
Doing_laundry	62.0
Washing_hands	100
Watching TV	60.86
Going_out_for_entertainment	80.0
Lawnwork	97.43
Average Accuracy	65.16%

I also compared the result with the result of MIT [8]. Our algorithm gives 73.47% of accuracy compared their highest accuracy, 27%. Table 4.10 shows the activity wise performance analysis. The further improvement of our results on a restricted set of activities, having many samples, and clearer boundary, proves the effectiveness the algorithm.

Table 4.10 Comparison of result with MIT on the same set of activities

Activity	MIT Result [8] (in %)	Accuracy of my Algorithm
Preparing lunch	25	74.52

Toileting	27	62.77
Preparing breakfast	8	67.87
Bathing	25	63.20
Dressing	7	85.18
Grooming	26	77.63
Preparing a beverage	7	87.14
Doing Laundry	9	84.04

4.3.3 Results for Multi-class Adaboost

I extend the MLVC work to incorporate other features such as Last Sensor and Sensor Sequence and Sensor Group. Senor Sequence and Sensor Group are mined by a customized FP-Growth algorithm. The reduced set of Sensor Sequences and Sensor Groups along with their buckets is used in the simulator to calculate which sequences or groups are on within the data window. The simulator uses 5 time slices each sampled in 5 second interval. So, a sensor data window of 20 second is used.

Quantization of Maximum Likelihood: I found that maximum likelihood values for each activity have some patterns and sometimes near to some values. So, I took only 200 value ranges for each activity. So, the total number of hypotheses possible is 19 X 200. The quantization is needed because other features are discrete. The quantization enables Adaboost to weak classifier that uses discrete values.

Frequent Itemset mining: The sensor sequence and group of sensors activated for each activity were mined from the whole dataset and then were merged together. The approach will mine frequent patterns for each activity, some of which may be important sequences. Moreover, the approach reduces number of items to be mined. Mining less supported items increase the hypotheses set but does not increase the recognition accuracy. As Adaboost selects the best hypotheses set to be used, less useful hypotheses are dropped automatically.

The set of sequences that are possible, mutually exclusive feature groups are formed for sequences that may occur within the same window (concurrently). I term the groups are buckets. The same thing is done for sensor group. Suppose sequence ‘abc’ never occurred together with ‘cfg’. So, they are put in the same bucket as the feature to the adaboost. If the sequences ‘abc’ and ‘abe’ occurred in the same time window, then they must be put in different buckets. The input for the adaboost becomes something as shown below:

Class ML_1 ML_2 …ML_{19} LS_1…LS_{19} SB_1…SB_i…GB_1…GB_i….

Where class is the class label, MLi is the Maximum Likelihood value for class i, LSi is the last sensor for class i, SBi is the i-th bean for sensor sequence. GBj is the j-th bean for group of sensors of length j.

I mined items with support 10 and around 1,40,000 items were mined. From that more than 60,000 beans were constructed. Then the input data was constructed to be fed to Adaboost.

Classfication: I used SAMME [77] algorithm for multi-class Adaboost. Weak Threshold classifier was used to find out the hypotheses. As the features are discrete, threshold classifier chooses the values of the features as hypotheses. For example, it may choose sequence ‘abe’ as a hypothesis or possibly the sensor group ‘efg’ may be chosen as another hypothesis. The accuracy of Adaboost increases with increasing number of hypothesis but the progress stops after choosing a certain number of such hypotheses, depending on the training set. For the data under consideration, the accuracy stopped increasing for iterations 20,000. That means 20,000 hypotheses need to be computed each time a decision is to be made by Adaboost.

Table 4.11 Accuracy for different number of iterations in Adaboost

Iterations	5000	10000	15000	20000
Average	31.28	52.47	66.58	73.05

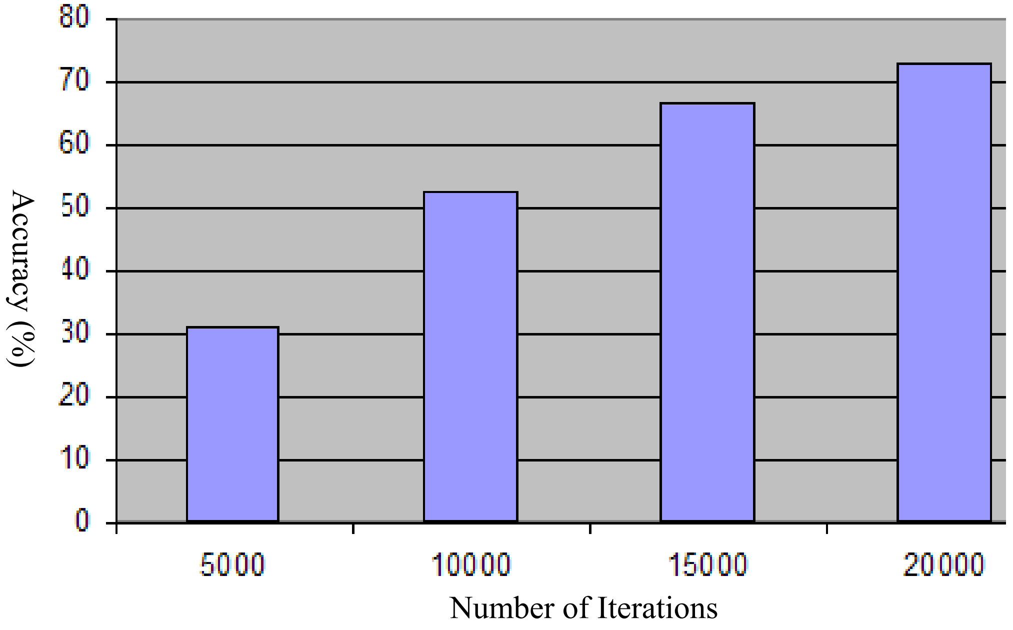


Figure 4.10 Accuracy of Adaboost for different number of iterations

Table 4.12 Highest accuracy for iterations 20,000

Activity	Accuracy (%)
Bathing	78.43
Toileting	80.82
Going_out_to_work	62.18
Preparing_lunch	70.29
Preparing_dinner	48.80
Preparing_breakfast	69.44
Dressing	70.20
Grooming	84.45
Preparing_a_snack	79.27
Preparing_a_beverage	78.80
Washing_dishes	74.63
Doing_laundry	87.76
Cleaning	26.48
Putting_away_dishes	85.24

Washing_hands	73.84
Putting_away_groceries	80.87
Watching TV	57.52
Going_out_for_entertainment	84.36
Lawnwork	94.59
Average	73.05

4.4. Discussion

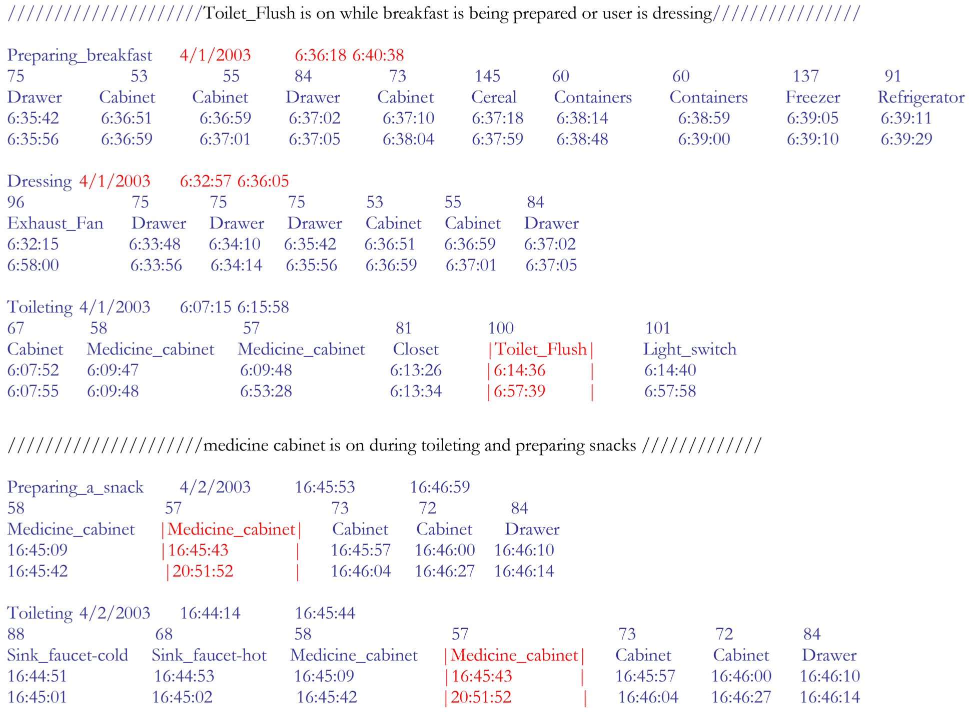
The strength of my algorithm is exploiting sensor activation patterns from small dataset. Given big dataset, HMM or other temporal classifiers may perform better. However, ADL environment, collecting a big dataset is difficult and no such dataset is available. In my proposed system, MLVC algorithm finds probability for the default patterns of ADLs, data mining selects the representative patterns for ADLs and Adaboost further chooses most fitting patterns as hypothesis set.

The patterns are actually strings. So, no distance based methods can be suitably applied on them. Distance based methods are usually appropriate for features with continuous numeric values. An effort could be made to present the patterns as numbers by representing each sensor by a numeric value and then replacing sensor names by their corresponding number. The issue here is in which order the sensors should be numbered. If the sensor numbering and consequently numbering of patterns are optimized for one activity or activity set, that may lead to discontinuous numbering of patterns for other activity or activity set. If a suitable scheme could be found, the classification system could be generalized for other systems.

Chapter 5

ID Sensor for Removing Noise from Data

Recognizing Activities of Daily Living (ADLs) by deploying sensors in the environment is very challenging. One of the challenges is to remove the unwanted noise. Figure 5.1 shows noise inserted in real data. The task becomes more challenging when multiple inhabitants co-exist due to the noise introduced by the other users [11].



```
/////////////////////Toilet_Flush is on while breakfast is being prepared or user is dressing////////////////

Preparing_breakfast   4/1/2003     6:36:18 6:40:38
75          53        55       84       73       145      60             60          137       91
Drawer      Cabinet   Cabinet  Drawer   Cabinet  Cereal   Containers     Containers  Freezer   Refrigerator
6:35:42     6:36:51   6:36:59  6:37:02  6:37:10  6:37:18  6:38:14        6:38:59     6:39:05   6:39:11
6:35:56     6:36:59   6:37:01  6:37:05  6:38:04  6:37:59  6:38:48        6:39:00     6:39:10   6:39:29

Dressing 4/1/2003     6:32:57 6:36:05
96             75       75       75       53       55       84
Exhaust_Fan    Drawer   Drawer   Drawer   Cabinet  Cabinet  Drawer
6:32:15        6:33:48  6:34:10  6:35:42  6:36:51  6:36:59  6:37:02
6:58:00        6:33:56  6:34:14  6:35:56  6:36:59  6:37:01  6:37:05

Toileting 4/1/2003     6:07:15 6:15:58
67        58               57               81        100            101
Cabinet   Medicine_cabinet Medicine_cabinet Closet    |Toilet_Flush| Light_switch
6:07:52   6:09:47          6:09:48          6:13:26   |6:14:36     | 6:14:40
6:07:55   6:09:48          6:53:28          6:13:34   |6:57:39     | 6:57:58

/////////////////////medicine cabinet is on during toileting and preparing snacks /////////////

Preparing_a_snack     4/2/2003     16:45:53     16:46:59
58                 57                   73        72        84
Medicine_cabinet   |Medicine_cabinet|   Cabinet   Cabinet   Drawer
16:45:09           |16:45:43        |   16:45:57  16:46:00  16:46:10
16:45:42           |20:51:52        |   16:46:04  16:46:27  16:46:14

Toileting 4/2/2003     16:44:14     16:45:44
88                68               58                57                  73        72        84
Sink_faucet-cold  Sink_faucet-hot  Medicine_cabinet  |Medicine_cabinet|  Cabinet   Cabinet   Drawer
16:44:51          16:44:53         16:45:09          |16:45:43        |  16:45:57  16:46:00  16:46:10
16:45:01          16:45:02         16:45:42          |20:51:52        |  16:46:04  16:46:27  16:46:14
```

Figure 5.1 Noise introduced during Activity of a single user in [14]

The noise needs to be removed, before feeding the events to the classifier. Otherwise, the results derived may not be accurate. If one user is cooking and another user is picking up the phone, the combination will not mean cooking for the first user, if training data did not contain this specific case. That motivates me to the problem.

A usual suggestion can be using a tracking system to separate out the sensors activated by a particular user. However, tracking fails for ADLs. Let us assume a scenario.

In the morning Alice and Bob wakes up from sleep. Alice goes to kitchen to make breakfast and Bob to toilet. Suddenly the telephone rings. Usually Bob picks up the phone but this time Alice goes to pick up. While Alice is going to pick up the phone, Bob comes out of the toilet and goes to kitchen to open the fridge. Bob opens the fridge before Alice picks up the phone.

Both Alice and Bob usually go to toilet after waking up. So, the tracking system will be confused to detect who is making the breakfast and who is in toilet. Suppose this time the tracking identified the user right knowing that Alice usually makes the breakfast. Then the system will make a mistake in the second case. It will find Alice still making breakfast and Bob picking up the phone.

Let us assume that the tracking system does not have any knowledge about user activity pattern; it can only detect separate users doing activity using its domain knowledge. Then the tracker will take just an arbitrary decision about who picked the phone up, because from both kitchen and toilet it is possible to go to pick up the phone. The tracker even fails when the phone is cordless. In real activity recognition problem, the system should detect Alice picking up the phone and her breakfast making is paused temporarily. Figure 5.2 pictorially depicts the scenario.

To associate sensor states with users, each sensor should know who activated it. I propose each user carries an ID sensor with him which broadcasts a short range beacon periodically. Ideally, the sensors activated, should associate its state with the user sending the beacon. But engaging each sensor in the task is communication intensive. I provide a simple clustering scheme that elects a leader for the association task. This does not hamper the ‘deploy and forget’ principle for environmental sensors. Attributing events to individuals in multi-inhabitant environment has also been addressed in [80].

They proposed users past behavior is used to for the event association. But it is not pragmatic, what if the users change their behavior as in the case of Alice and Bob.

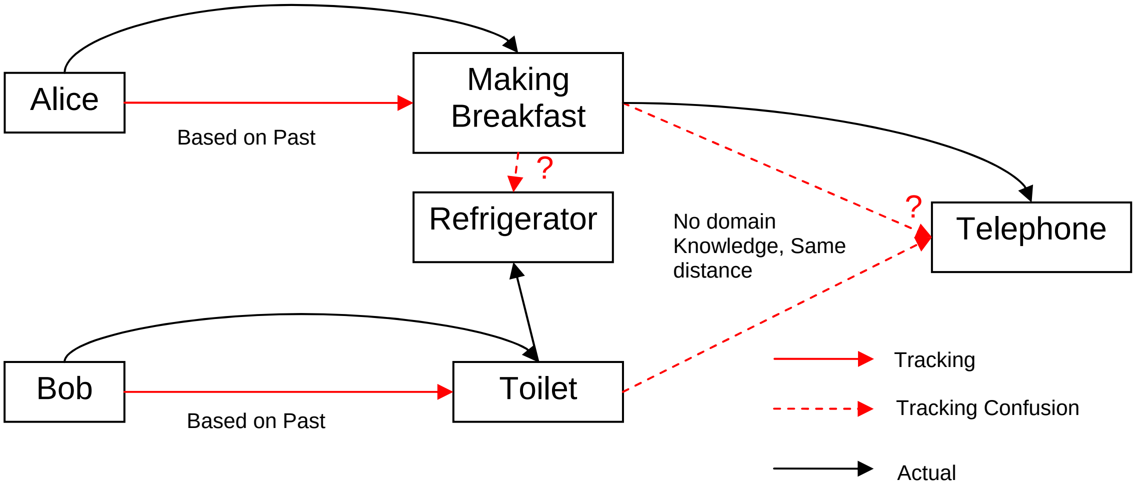


Figure 5.2 Confusion in tracking system

5.1. Sensor Network System with Users Carrying ID Sensors

I assume a home where simple ubiquitous sensors are deployed for everyday activity detection. The users are carrying ID sensors with them, so that sensor activation can be associated with them. It is possible that passive sensors like RFID may also be installed within the system. RFID are usually tagged with moving objects and in such a case users are supposed to be carrying RFID readers with them [9]. The reader can easily act as an ID sensor. The functionality of the ID sensor is just to transmit beacon periodically in a short range of about 1~1.5 meter. The transmission range of 1~1.5 meter is based on the fact that human being walks at a speed of 4-5 km/hour in an open space which is equivalently 1-1.5m/sec. Hence a beacon signal should remain valid for 1 sec. If a beacon signal is not received after 1 sec, the user is supposed not to be present there.

The beacon signals can be received by the sensors and if this sensor is activated, it is associated with the user carrying the ID sensor. But, it is communication intensive for the sensors to send the beacons received, to the sink node. Even if sensor keep the beacon received with itself, until it is activated, the sensor has to stay awake. This is also costly. So, I propose a maximum neighbor based clustering algorithm for the problem. Maximum neighbor based algorithm is because sensors are usually deployed in small clusters in the household utensils and the algorithm will usually select only one cluster head for a group of sensors. The cluster heads will be receiving and associating the beacons with the activated sensors. Otherwise cluster heads and sensors activated can send their information directly to the sink node where the association task can be performed. I propose selecting cluster heads because, only those sensors are entrusted with high communication responsibilities and need unlimited power source. The proposal works in two phases. In the first phase, the clustering algorithm is run to select the cluster heads. The heads show signs to the user, so that they are provided unlimited power supply. Then the sensors enter into second phase when usual sensors wait for events, cluster heads receive the beacon signals and ID sensors broadcast periodically. There is a possibility that a sensor is activated but the cluster head has not got any beacon signal. In such a case, the beacon signal of previous second can be used. Otherwise, the event is associated with anonymous user. The two phase algorithm is shown as below:

Phase I: Clustering Algorithm

```
//ID Sensors are excluded from this stage
1. Run stage1: set transmission range;
2. Run stage2: neighbor count;
3. Run stage3:neighbor count broadcast;
4. Run stage4: cluster heads selection;
5. Run stage5: nodes joining to clusters;
```

Stage 1: Set Transmission Range

```
for each sensor do
    set transmission range to 1 meter;
end
```

Stage 2: Neighbor Count

```
for each sensor do
    while !timer (TIME_OUT) do
        broadcast  sensor id;      |  save received ids;
    end
    count the number of neighbors (including itself);
end
```

Stage 3: Neighbor Count Broadcast

```
for each sensor do
    while !timer (TIME_OUT) do
        broadcast  the number of neighbors;
    end
end
```

Stage 4: Cluster Heads Selection

```
for each sensor do
    if  has maximum neighbors and lowest id do
        decide itself as cluster head;
    end
end
```

Stage 5: Nodes joining to clusters

```
1.  for each sensor do
2.      if  cluster head do
3.          while !timer (TIME_OUT) do
4.              broadcast the claim ; | process joining msg;
5.          end
6.      else
7.          wait (cluster head claim, timer(TIME_OUT));
8.          if claim comes before timer expires do
9.              join the cluster;
10.         else
11.             decide itself as cluster head;
12.             do 3~5;
13.         end
14.     end
15. end
```

Phase II: Event Attribution Algorithm

```
1. Set Transmission range of ID sensors and non cluster heads to 1m;
2. Set Transmission range of cluster heads such that they can transmit
   to sink node;
3. ID_Process;   |   Sensor_Process;   | ClusterHead_Process;
```

ID_Process

```
1. for each ID sensor repeat infinite
2.     broadcast the ID;
3.     delay(1 sec);
4. end
```

Sensor_Process

```
for each non-cluster head sensor repeat infinite
    wait(state change);
    send the event to cluster head;
end
```

ClusterHead_Process

```
for each cluster head sensor do
    repeat infinite
        1. wait (beacon);
        2. store beacon in list;

        1. delay (1sec);
        2. invalidate 2sec  old beacon;

        1. wait (state change event);
        2. Associate event with
           beacons in list;
        3. send the event to sink node;
    end
end
```

5.2. Simulation and Implementation Results

I designed a simulation program to judge the practicability of the idea. It simulates simple user movement patterns in a 10X10 square meter space with 100 sensors. The space represents a typical small apartment. I also augmented our work with an implementation prototype.

The sensors' deployed coordinates are assumed randomly and then cluster head selection process is run. After that ID sensor's movement pattern is performed as straight lines in different directions, circles of different radius and zigzag pattern with a maximum deviation from the main axis.

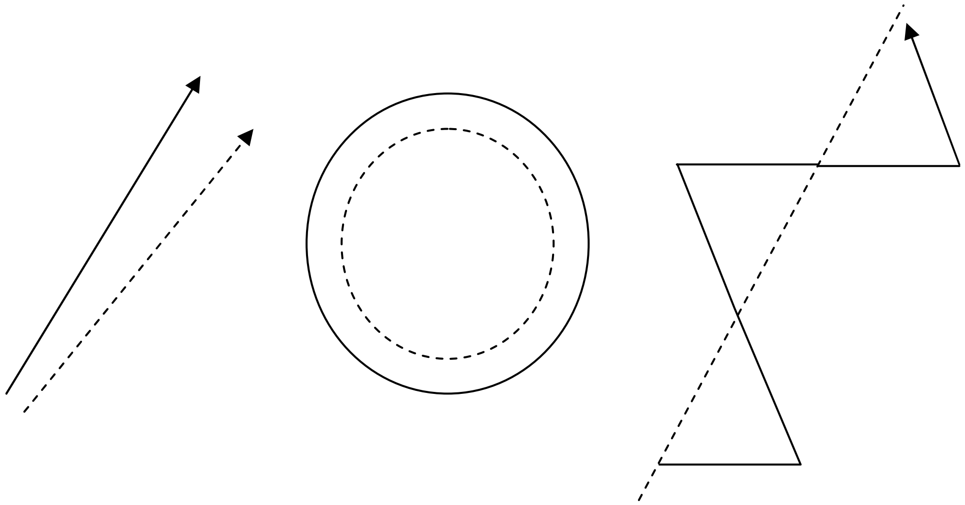


Figure 5.3 Simulated movement patterns

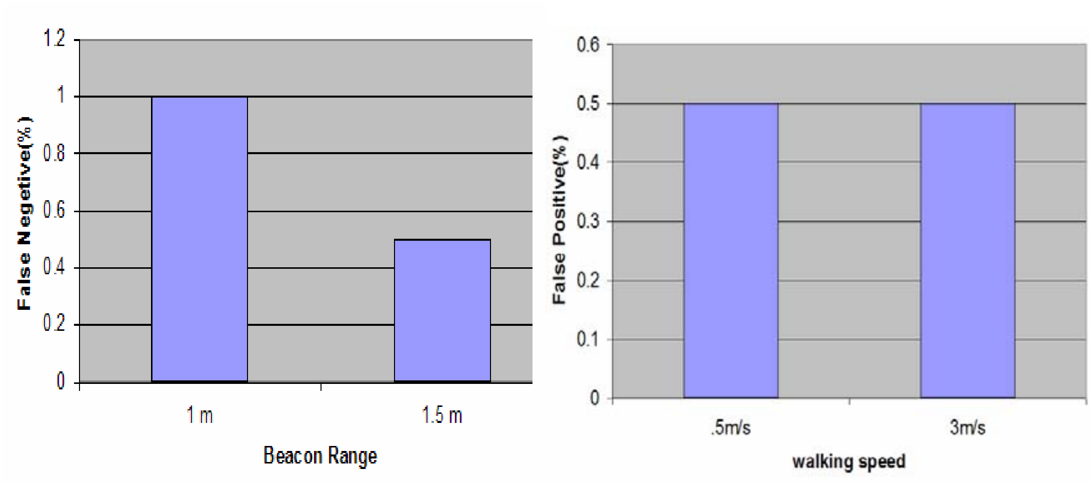
Table 5.1 Simulation Parameters

Parameter	**Value**
P(Beacon Loss)	1/10
P(Not Activate)	1/20
Walking speed	0.5~3 meter/sec
Beacon Valid period/ Broadcast Interval	1 sec
Beacon range	1meter and 1.5 meter
Simulation time unit	0.5sec
Stopping criteria	1000 sensor event
Number of simulation runs	200

After every simulation time unit, the ID sensor's position is calculated with different walking speed of 0.5 meter (walking slowly) to 3meter/second (running fast). Simulation time unit is chosen to be 0.5 sec. Then ID sensor's beacon is broadcasted (after every alternative simulation unit). Cluster heads keep the ID, with a probability of [1-P(Beacon Loss)], if The ID sensor is within the range selected for the particular simulation (1 and 1.5 meter). Whether not to activate any sensor is determined by P(Not Activate) and the rest of the probability is distributed uniformly among the sensors within the reach. After a sensor is decided to be activated, its cluster head associate the event with the beacon(s)/ID(s) the head has in its list according to our algorithm. The randomness for beacon loss and not activating the sensor is added to simulate real environmental scenario where both the events are possible.

The simulation was evaluated using two parameters: False Negative and False Positive event. False Negative occurs when the user activates a sensor but the event is not attributed to him. False Positive means the opposite, i.e. the user did not trigger an event but the event is attributed to him.

With a single user roaming around, the simulation results shows that there is no False Negative except 1% and 0.5% for walking speed of 3meter/sec and beacon range of 1 and 1.5 meter respectively. However, with slow walking speed of 0.5 meter/sec or fast walking speed of 3 meter/sec, there are False Positives (0.5%) when two users are present. Though 3 meter/sec is a very unlikely speed for a person in home, 0.5 meter/sec is very common. However, the slow walking speed means the users are socializing closely and no algorithm can reduce this error margin.



Single User

Multiple User

Figure 5.4 Accuracy for single and multiple user environments in simulation

I implemented a simple setup with 9 sensors from Crossbow's WSN Kit. I used MIB520 as interfacing board and MPR2400CA as zigbee sensors. I created two clusters each consisting of 3 sensors. Of the 3 sensors, one is the cluster head. The cluster heads

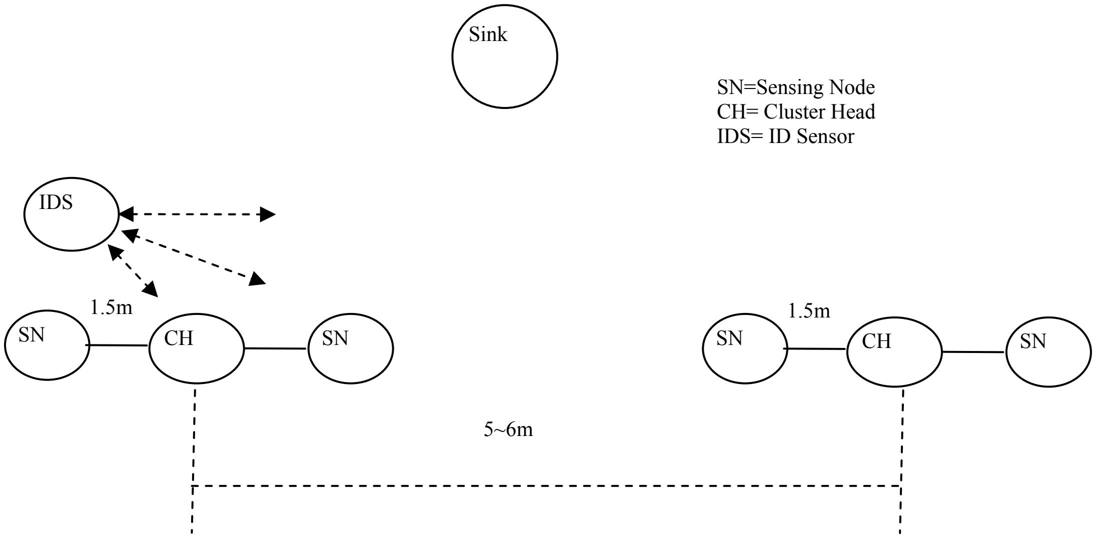


Figure 5.5 Implementation scenario

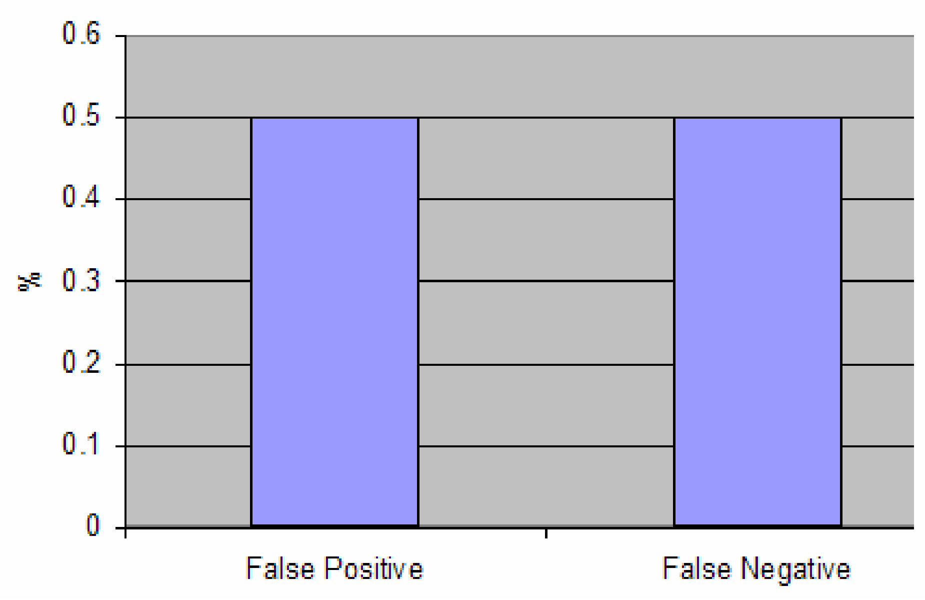


Figure 5.6 Accuracy for implementation

do not have any sensing functionality except receiving the beacon signals and sending that to the sink node. Sensing nodes send events directly to the sink node. I took two ID sensors to simulate two moving users. Sensing nodes implement user proximity by short range infrared light sensors. When a user with ID sensor walks around any sensing node, sensing node sends the proximity event and the cluster head sends the received beacon signal to the sink node.

Sensing nodes are placed at a distance of around 1~ 1.5 meters from the cluster head and opposite to each other. Two clusters are placed around 5~6 meter apart. ID sensors have a transmission range of 1~1.5 meter. The range was set experimentally by using the function CC2420Control.setRFPower(unit8_t) and passing para-meter 3. The function can receive parameters from 3~31. 3 means -25dBm and 31 equals to max power (0dBm).

In the real environment, the users were asked to move freely. On an average, we got False Positives and False Negatives of 0.5% each.

5.3. Discussion

The low false positive and false negative rate in simulation and real implementation scenario motivates the applicability of the idea in ADL environment. In future miniature sensors will be produced with can be build into the objects the user always carry.

Chapter 6

Conclusions

Overall, I propose a method to recognize Activities of Daily Living (ADLs). I provide a method to calculate temporal features from many binary sensors, a technique of using data mining for feature dimension reduction and Adaboost for producing a strong classifier from the feature set with an added advantage of further reducing the feature dimension. The result is a real-time activity recognition framework.

The algorithm extensively deals with patterns or strings which are the key to the good accuracy found. The algorithm performs well in a small training dataset. This is one of the advantages of this method, because it is very difficult to collect annotated training data from daily living environment. Other methods such as HMM might be appealing but, in reality, HMM needs lot of tuning for better accuracy.

I also propose a sensor network scenario where ID sensor is used to remove noise. The idea can also be helpful to recognize multiple user activity.

The framework and the classifier need to be deployed and tested in a real environment, which I leave a future work. In addition, incremental learning using Adaboost needs to be investigated as in home environment; quick and automated retraining is the key to the commercial success of activity recognition system. The algorithm can be investigated for ADLs scenario. Moreover, it is very important to transfer the learning from one environment to another similar environment. It will be interesting if it is possible to transfer the hypotheses set automatically.

References

[1] Activity of Daily Living: http://en.wikipedia.org/wiki/Activities_of_daily_living.

[2] P. Turaga, R. Chellappa, V.S. Subrahmanian, O. Udrea. Machine Recognition of Human Activities: A Survey. IEEE Transactions on Circuits and Systems for Video Technology, Volume 18, Issue 11, Nov 2008, pages: 1473-1488

[3] S. Mitra, T. Acharya. Gesture Recognition: A Survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Volume 37, Issue 3, May 2009, pages: 311-324.

[4] M. Zia Uddin, J. J. Lee, and T.-S. Kim. Independent shape component-based human activity recognition via Hidden Markov Model , Journal of Applied Intelligence, Springer Netherlands, January 2009, DOI 10.1007/s10489-008-0159-2

[5] Ermes M, Pärkka J, Mantyjarvi J, Korhonen I. Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions. IEEE Trans Inf Technol Biomed. 2008 Jan; 12(1): 20-6.

[6] Adil Mehmood Khan, Tae-Seong Kim. Accelerometer Signal-Based Human Activity Recognition Using Augmented Autoregressive Model Coefficients and Artificial Neural Nets. 30th Annual International IEEE EMBS Conference, August 20-24, 2008, Vancouver, British Columbia, Canada.

[7] Ermes M, Parkka J, Cluitmans L. Advancing from offline to online activity recognition with wearable sensors. Conf Proc IEEE Eng Med Biol Soc. 2008; 2008: 4451-4

[8] E. Tapia, S. Intille, and K. Larson. Activity Recognition in the home setting using simple and ubiquitous sensors. Proceedings of PERVASIVE 2004, vol. LNCS 3001, A. Ferscha and F. Mattern, Eds. Berlin Heidelberg: Springer-Verlag, 2004, pp. 158-175.

[9] Donald J. Patterson, Dieter Fox, Henry Kautz, Matthai Philipose. Fine-Grained Activity Recognition by Aggregating Abstract Object Usage. Proceedings of the 2005 Ninth IEEE International Symposium on Wearable Computers (ISWC'05).

[10] S. S. Intille, K. Larson, E. Munguia Tapia, J. Beaudin, P. Kaushik, J. Nawyn, and R. Rockinson. Using a live-in laboratory for ubiquitous computing research. In Proceedings of PERVASIVE 2006 , vol. LNCS 3968, K. P. Fishkin, B. Schiele, P. Nixon, and A. Quigley, Eds. Berlin Heidelberg: Springer-Verlag, 2006, pp. 349-365.

[11] Beth Logan, et.al. "A Long-Term Evaluation of Sensing Modalities for Activity Recognition". In the Proceedings of UbiComp 2007.

[12] Tim Van Kasteren, Athanasios Noulas, Gwenn Englebienne, Ben Krose. "Accurate Activity Recognition in a Home Setting". 10th International Conference on Ubiquitous Computing 2008, page(s): 1-9.

[13] Rabiner L. R., Juang B. H. An introduction to hidden Markov models. IEEE ASSP Magazine: 4-15, Jan 1986

[14] MIT Activity Recognition data: http://courses.media.mit.edu/2004fall/mas622j/04.projects/home/

[15] Christopher R. Wren and Emmanuel Munguia Tapia. Toward Scalable Activity Recognition for Sensor Networks. In Proceedings of The Second International Workshop in Location and Context-Awareness (LoCA '06), pp. 168-185, Springer vol. 3987 / 2006

[16] Jeff Bilmes. What HMM Can Do. Technical Report of University of Washington, Department of Electrical Engineering, Number UWEETR-2002-0003. http://www.ssli.ee.washington.edu/people/bilmes/mypapers/whathmmscannotdo04.pdf

[17] Alexander T. Ihler. Inference in Sensor Networks: Graphical Models and Particle Models. Ph. D Thesis, MIT, 2005.

[18] Christopher M. Bishop. Chapter 8: Graphical Models, Pattern Recognition and Machine Learning, Springer 2006.

[19] Mihai Marin-Perianu, Clemens Lombriser, Oliver Amft, Paul Havinga, and Gerhard Tröster. Distributed Activity Recognition with Fuzzy Enabled Wireless Sensor Networks , Distributed Computing in Sensor Systems (DCOSS), Lecture Notes in Computer Science, Springer Berlin / Heidelberg, Volume 5067 ,2008, pages: 296-313.

[20] Venet Osmani, Sasitharan Balasubramaniam, Dmitri Botvich. Self-Organising Object Networks using Context Zones for distributed activity recognition. In the Proceedings of the ICST 2nd international conference on Body area networks, Florence, Italy, Article No:14, 2007.

[21] Parisa Rashidi, Diane J. Cook. Keeping the Resident in the Loop: Adapting the Smart Home to the User. IEEE Transactions on Systems Man & Cybernetics, Part A., 2009

[22] Oliver Amft, Martin Kusserow and Gerhard Tröster. ”Probabilistic Parsing of Dietary Activity Events”. 4th International Workshop on Wearable and Implantable Body Sensor Networks, Mar 26 – 28, 2007, Germany, pp: 242-247.

[23] Tanzeem Choudhury, et.al. “The Mobile Sensing Platform: An Embedded Activity Recognition System”. Appears in IEEE Pervasive April 2008, page(s): 32-41.

[24] Daniel Wilson & Chris Atkeson. “Simultaneous Tracking & Activity Recognition (STAR) Using Many Anonymous, Binary Sensors”. In the proceedings of PERVASIVE 2005.

[25] Md. Kamrul Hasan, Husne Ara Rubaiyeat, Young-Koo Lee, Sungyoung Lee. “Mapping of Activity Recognition as a Distributed Inference Problem in Sensor Network”. IC-AI 2008, Las Vegas, USA, pp: 280-285.

[26] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson, D. Hahnel, D. Fox, and H. Kautz. Inferring Activities from Interactions with Objects. IEEE Pervasive Computing: Mobile and Ubiquitous Systems, 3(4):50–57, 2004.

[27] Yoav Freund, Robert E. Schapire. A Short Introduction to Boosting. Journal of Japanese Society for Artificial Intelligence, 14(5):771-780, September, 1999. (In Japanese, translation by Naoki Abe.)

[28] L. Bao and S. S. Intille. Activity Recognition from User-Annotated Acceleration Data. In the Proceedings of the Second International Conference in Pervasive Computing (PERVASIVE '04), Vienna, Austria, 2004, pp. 1-17.

[29] D. Olguin Olguin and A. Pentland. Human Activity Recognition: Accuracy Across Common Locations for Wearable Sensors. In the Proceedings of The 10th IEEE International Symposium on Wearable Computing (Student Colloquium), Montreaux, Switzerland, 2006.

[30] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman. Activity Recognition from Accelerometer Data. In the Proceedings of the Seventeenth Innovative Applications of Artificial Intelligence Conference. Pittsburgh, Pennsylvania, 2005.

[31] A. Raj, A. Subramanya, D. Fox, and J. Bilmes. Rao-Blackwellized Particle Filters for Recognizing Activities and Spatial Context from Wearable Sensors. In the Proceedings of the 10th International Symposium on Experimental Robotics (ISER '06). Rio de Janeiro, Brazil: Springer-Verlag, 2006.

[32] A. Raj, J. Bilmes, and D. Fox. Hierarchical Models for Activity Recognition. In Proceedings of the IEEE Conference on Multimedia Signal Processing (MMSP '06). Victoria, BC, CA, 2006, pp. 233-237.

[33] B. Ristic, S. Arulampalam, and N. Gordon. Chapter 3: A Tutorial on Particle Filters," in Beyond the Kalman Filter: Particle Filters for Tracking Applications Artech House Radar Library, 2004.

[34] P. Lukowicz, J. A. Ward, H. Junker, M. Stager, G. Troster, A. Atrash, and T. Starner. Recognizing Workshop Activity Using Body Worn Microphone and Accelerometers. In the Proceedings of the International Conference in Pervasive Computing (PERVASIVE '04). Vienna, Austria: Springer Berlin/Heidelberg, 2004, pp. 18-32.

[35] D. Minnen, T. Starner, J. A. Ward, P. Lukowicz, and G. Troster. Recognizing and Discovering Human Actions from On-Body Sensor Data. In the Proceedings of the IEEE International Conference on Multimedia and Expo (ICME '05). Amsterdam, NL, 2005, pp. 1545-1548.

[36] J. A. Ward, P. Lukowicz, and G. Troster. Gesture Spotting Using Wrist Worn Microphone and 3-Axis Accelerometer," in Proceedings of the Conference on Smart Objects and Ambient Intelligence, vol. 121. Grenoble, France, 2005, pp. 99-104.

[37] L. Rabiner and B. Juang. Fundamentals Of Speech Recognition. Upper Saddle River, NJ, USA Prentice Hall, 1993.

[38] J. Lester, T. Choudhury, N. Kern, G. Borriello, and B. Hannaford. A Hybrid Discriminative/Generative Approach for Modeling Human Activities. In the Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI '05), 2005, pp. 766–772.

[39] R. Duda, P. Hart, and D. Stork. Pattern Classification, 2nd ed: John Wiley & Sons, 2000.

[40] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher. Activity Recognition and Monitoring Using Multiple Sensors on Different Body Positions. In the Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks (BSN'06): IEEE Computer Society 2006, pp. 113-116.

[41] H. L. Chieu, W. S. Lee, and L. P. Kaelbling. Activity Recognition from Physiological Data Using Conditional Random Fields. In the Proceedings of the Singapore-MIT Alliance (SMA) Computer Science Program. Cambridge, MA, 2006.

[42] P. Domingos and M. Pazzani. Beyond Independence: Conditions for the Optimality of a Simple Bayesian Classifier. In the Proceedings of the Thirteenth International Conference on Machine Learning, L. Saitta, Ed.: Morgan Kauffman, 1996, pp. 194-202.

[43] P. Clark and T. Niblett. The CN2 Induction Algorithm, Machine Learning, vol. 3, 4, pp. 261-283, 1989.

[44] G. H. John and P. Langley. Estimating Continuous Distributions In Bayesian Classifiers. In the Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence. San Mateo, 1995, pp. 338-345.

[45] J. R. Quinlan. C4.5: Programs for Machine Learning. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc, 1993.

[46] J. Parkka, M. Ermes, P. Korpipaa, J. Mantyjarvi, J. Peltola, and I. Korhonen. Activity Classification Using Realistic Data From Wearable Sensors. IEEE Transactions on Information Technology in Biomedicine, vol. 10, 1, pp. 119-128, 2006.

[47] A. Y. Benbasat and J. A. Paradiso. A Framework for the Automated Generation of Power-Efficient Classifiers for Embedded Sensor Nodes. In the Proceedings of the 5th ACM Conference on Embedded Networked Sensor Systems (SenSys '07). Sydney, Australia, 2007, pp. 219-232.

[48] T. Van Kasteren, G. Englebienne, and B. Kröse. Recognizing activities in multiple contexts using transfer learning. In AAAI Fall 2008 Symposium: AI in Eldercare, Washington DC, USA, 2008.

[49] Vincent Wenchen Zheng, Derek Hao Hu, Qiang Yang. Cross-domain activity recognition. In the Proceedings of the 11th international conference on Ubiquitous computing, Orlando, Florida, USA, Pages 61-70.

[50] D.H. Hu, S.J. Pan, V.W. Zheng, N.N. Liu, and Q. Yang. Real world activity recognition with multiple goals. In Proceedings of the Tenth International Conference on Ubiquitous Computing (UbiComp 2008), pages 30--39, 2008.

[51] D.H. Hu and Q. Yang. Cigar: Concurrent and interleaving goal and activity recognition. In Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI), pages 1363--1368, 2008.

[52] Rob Schapire. Strength of Weak Learnability. Journal of Machine Learning Vol. 5, pages 197-227. 1990

[53] Yoav Freund. Boosting a weak learning algorithm by majority. Proceedings of the Third Annual Workshop on Computational Learning Theory. 1990

[54] Yoav Freund and Robert E. Schapire A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55(1):119--139, 1997

[55] Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining frequent patterns without candidate generation. Data Mining and Knowledge Discovery 8:53-87, 2004

[56] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in timesequential images using hidden markov model," Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 379–385, 1992.

[57] J. Schlenzig, E. Hunter, and R. Jain, "Recursive identification of gesture inputs using hidden markov models," Proceedings of the Second IEEE Workshop on Applications of Computer Vision, pp. 187–194, 1994.

[58] A. D. Wilson and A. F. Bobick, "Learning visual behavior for gesture analysis," Proceedings of the International Symposium on Computer Vision, pp. 229–234, 1995.

[59] T. Starner, J. Weaver, and A. Pentland, "Real-time american sign language recognition using desk and wearable computer based video," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 12, pp. 1371–1375, 1998.

[60] N. Vaswani, A. K. Roy-Chowdhury, and R. Chellappa, ""Shape Activity": a continuous-state HMM for moving/deforming shapes with application to abnormal activity detection," IEEE Transactions on Image Processing, vol. 14, no. 10, pp. 1603–1616, 2005.

[61] N. P. Cuntoor and R. Chellappa, "Epitomic representation of human activities," Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2007.

[62] H. Buxton and S. Gong, "Visual surveillance in a dynamic and uncertain world," Artificial Intelligence, vol. 78, no. 1-2, pp. 431–459,1995.

[63] T. Huang, D. Koller, J. Malik, G. H. Ogasawara, B. Rao, S. J. Russell, and J. Weber, "Automatic symbolic traffic scene analysis using belief networks," National Conference on Artificial Intelligence, pp. 966–972,1994.

[64] P. Remagnino, T. Tan, and K. Baker, "Agent orientated annotation in model based visual surveillance," Proceedings of IEEE International Conference on Computer Vision, pp. 857–862, 1998.

[65] S. Park and J. K. Aggarwal, "Recognition of two-person interactions using a hierarchical bayesian network," ACM Journal of Multimedia Systems, Special Issue on Video Surveillance, vol. 10, no. 2, pp. 164–179, 2004.

[66] C. Castel, L. Chaudron, and C. Tessier, "What is going on? A High-Level Interpretation of a Sequence of Images," ECCV Workshop on Conceptual Descriptions from Images, 1996.

[67] N. Ghanem, D. DeMenthon, D. Doermann, and L. Davis, "Representation and Recognition of Events in Surveillance Video Using Petri Nets," Second IEEE Workshop on Event Mining 2004, CVPR2004, 2004.

[68] G. Medioni, I. Cohen, F. Br´emond, S. Hongeng, and R. Nevatia, "Event detection and analysis from video streams," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 8, pp. 873–889, 2001.

[69] S. Hongeng, R. Nevatia, and F. Bremond, "Video-based event recognition: activity representation and probabilistic recognition methods," Computer Vision and Image Understanding, vol. 96, no. 2, pp. 129–162, 2004.

[70] V. D. Shet, D. Harwood, and L. S. Davis, "Vidmap: video monitoring of activity with prolog," Proceedings of IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 224–229, 2005.

[71] B. Georis, M. Maziere, F. Bremond, and M. Thonnat, "A Video Interpretation Platform Applied to Bank Agency Monitoring," 2nd Workshop on Intelligent Distributed Surveillance Systems (IDSS), 2004.

[72] D. Chen, J. Yang, and H. D. Wactlar, "Towards Automatic Analysis of Social Interaction Patterns in a Nursing Home Environment from Video," MIR 04: Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval, pp. 283–290, 2004.

[73] A. Hakeem and M. Shah, "Ontology and Taxonomy Collaborated Framework for Meeting Classification." International Conference on Pattern Recognition, pp. 219–222, 2004.

[74] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," Proceedings of the IEEE, vol. 77, no. 2, pp. 257–286, 1989.

[75] *people.cs.ubc.ca/~murphyk/Bayes/bayes_tutorial.pdf*

[76] Robert E. Schapire, Yoram Singer. Improved Boosting Algorithms Using Confidence-rated Predictions. Machine Learning, 37, 297–336 (1999), Kluwer Academic Publishers, Netherlands.

[77] Ji Zhu, Hui Zou, Saharon Rosset, and Trevor Hastie. Multi-class AdaBoost. Statistics and Its Interface Volume 2 (2009) 349–360

[78] R. Agrawal, T. Imielinski, A. Swami. Mining Association Rules Between Sets of Items in Large Databases. SIGMOD Conference 1993: 207-216

[79] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, pages 487-499, Santiago, Chile, September 1994.

[80] Crandall, A.S., and Cook, D., "Attributing Events to Individuals in Multi-Inhabitant Environments", IET 4th International Conference on Intelligent Environments, 21-22 July 2008, Seattle, Washington, pages: 1-8.

[81] Md. Kamrul Hasan, Sungyoung Lee, Young-Koo Lee. Using Sensor Sequences for Activity Recognition by Mining and Multi-Class Adaboost. Accepted in ICAI'10 - The 2010 International Conference on Artificial Intelligence, Las Vegas, USA.

[82] Md. Kamrul Hasan, Manhyung Han, Sungyoung Lee, Young-Koo Lee. A Sensor Network System for Event Attribution in Multi-user Home Environment. Accepted in ICAI'10 - The 2010 International Conference on Artificial Intelligence, Las Vegas, USA.

[83] Jehad Sarkar, La The Vinh, Young-Koo Lee, Sungyoung Lee. GPARS: a General-Purpose Activity Recognition System. Applied Intelligence, Springer (in press)

[84] http://cslu.cse.ogi.edu/HLTsurvey/ch1node7.html

[85] H. Murveit, J. Butzberger, V. Digilakis, and M. Weintraub. Large-vocabulary dictation using SRI's DECIPHER speech recognition system: Progressive search techniques. In ICASSP [ICA93], pages 319--322.

[86] X. Aubert, C. Dugast, H. Ney, and V. Steinbiss. Large vocabulary continuous speech recognition of wall street journal data. In ICASSP [ICA94], pages 129--132.

[87] F. Alleva, X. Huang, and M. Y. Hwang. An improved search algorithm using incremental knowledge for continuous speech recognition. In ICASSP [ICA93], pages 307--310.

[88] F. Kubala, A. Anastasakos, J. Makhoul, L. Nguyen, R. Schwartz, and G. Zavaliagkos. Comparative experiments on large vocabulary speech recognition. In ICASSP [ICA94], pages 561--564.

[89] Fodor I. K., "A survey of dimension reduction techniques," LLNL technical report, June 2002, UCRL-ID-148494.

[90] George Lee, Carlos Rodriguez and Anant Madabhushi, “An Empirical Comparison of Dimensionality Reduction Methods for Classifying Gene and Protein Expression Datasets”, Lecture Notes in Computer Science, Springer Berlin, Volume 4463/2007, pages: 170-181

Appendix

List of Abbreviations

Abbreviation	Acronym
ADLs	Activities of Daily Living
HMM	Hidden Markov Model
DBN	Dynamic Bayesian Network
CRF	Conditional Random Field
RFID	Radio Frequency Identifier
SAMME	Stagewise Additive Modeling using a Multi-class Exponential loss function
MLVC	Maximum Likelihood Value Calculation
ML	Maximum Likelihood
LS	Last Sensor
SS	Sensor Sequence
SG	Sensor Group
FP	Frequent Pattern
PCA	Principal Component Analysis
LDA	Linear Discriminate Analysis
STAR	Simultaneous Activity Recognition and Trackin

List of Publications

Patents:

P-1. Sungyoung Lee, Young-koo Lee, **Md. Kamrul Hasan**, Lenin Mehedy, Salahuddin Mohammad Salim Zabir. Apparatus for hot swapping in software defined radio and method for operating the same. Registered Korean Patent Number: 10-2008-0062981

Journals:

J-1. **Md. Kamrul Hasan**, Kim Anh Pham Ngoc, Young-Koo Lee, Sungyoung Lee. Preference Learning on an OSGi Based Home Gateway. IEEE Transactions on Consumer Electronics, Volume 55, Issue 3 (August 2009), pp. 1322-1329, ISSN: 0098-3063. (SCI)

J-2. **Md. Kamrul Hasan**, Lenin Mehedy, Muhammad Salim Zabir, Sungyoung Lee, Young-Koo Lee. A Middleware Based Network Hot Swapping Solution for SCA Compliant Radio. IEEE Transactions on Consumer Electronics, Volume 55, Issue 3 (August 2009), pp. 1315-1321, ISSN: 0098-3063. (SCI)

J-3. **Md. Kamrul Hasan**, Mohammad Kaykobad, Young-Koo Lee and Sungyoung Lee. A comprehensive Analysis of Degree Based Condition for Hamiltonian Cycles. Theoretical Computer Science, Elsevier Publications, vol. 411, No.1, pp.285-287 (January, 2010), ISSN: 0304-3975. (SCI)

J-4. Lenin Mehedy, **Md. Kamrul Hasan**, Mohammad Kaykobad. An Improved Degree Based Condition for Hamiltonian Cycles. Published in Information Processing Letters (IPL), Elsevier Publications, Volume 102 , Issue 2-3 (April 2007), Pages: 108-112, ISSN:0020-0190 (SCI-E) .

International Conferences:

C-1. **Md. Kamrul Hasan**, Sungyoung Lee, Young-Koo Lee. Using Sensor Sequences for Activity Recognition by Mining and Multi-Class Adaboost. Accepted in ICAI'10 - The 2010 International Conference on Artificial Intelligence, Las Vegas, USA.

C-2. **Md. Kamrul Hasan**, Manhyung Han, Sungyoung Lee, Young-Koo Lee. A Sensor Network System for Event Attribution in Multi-user Home Environment. Accepted in ICAI'10 - The 2010 International Conference on Artificial Intelligence, Las Vegas, USA.

C-3. Jehad Sarkar, **Kamrul Hasan**, Young-Koo Lee, Sungyoung Lee and Salauddin Zabir. Distributed Activity Recognition using Key Sensors. 11th International Conference on Advanced Communication Technology (ICACT), 2009, Phoenix Park, Korea. pages: 2245-2250

C-4. **Md. Kamrul Hasan**, Husne Ara Rubaiyeat, Young-Koo Lee, Sungyoung Lee. Mapping of Activity Recognition as a Distributed Inference Problem in Sensor Network. In the proceedings of ICAI'08 - The 2008 International Conference on Artificial Intelligence, Las Vegas, USA, july 14-17, 2008, pages; 280-285

C-5. **Md. Kamrul Hasan**, Husne Ara Rubaiyeat, Yong-Koo Lee, Sungyoung Lee. A Reconfigurable HMM for Activity Recognition. In the proceedings of 10th International Conference on Advanced Communication Technology (ICACT), February 17-20, 2008, Phoenix Park, Korea, pages: 843-846

C-6. **Md. Kamrul Hasan**, Mohammad Kaykobad, Young-Koo Lee, Sungyoung Lee. A comprehensive Analysis of Degree Based Condition for Hamiltonian Cycles. In the proceedings of 11th International Conference on Computer and Information Technology (ICCIT 2008), Bangladesh, pages: 465-469

C-7. **Md. Kamrul Hasan**, Lenin Mehedy, Salahuddin Muhammad Salim Zabir, Sungyoung Lee, Sang-chul Oh. A Middleware Based Network Hot Swapping Solution for SCA Compliant Radio. In the proceedings of 7th IEEE Symposium on Signal Processing and Information Technology (ISSPIT’07),Cairo, Egypt, Dec 15-18, 2007,pages: 99-104

C-8. **Md. Kamrul Hasan**, Lenin Mehedy, Sungyoung Lee, Young-Koo Lee: Semantic Service Discovery in a Middleware based Ubiquitous Environment. In the proceedings of 2006 IEEE Conference on Systems, Man and Cybernatics, Taipei, Taiwan, Volume 5, 8-11 Oct. 2006 Page(s):4391 – 4395

C-9. **Md. Kamrul Hasan**, Kim Anh, Lenin Mehedy, Young-Koo Lee, Sungyoung Lee: Conflict Resolution and Preference Learning in Ubiquitous Environment. In the proceedings of 2006 International Conference on Intelligent Computing (ICIC), Kunming Yunnan Province, China, August 16–19, 2006, LNAI 4114 – 0355, ISBN: 978-3-540-37274-5

C-10. Kh. Tarik-Ul Islam, Jehad Sarkar, **Kamrul Hasan**, M Rezwanul Huq, Andrey V. Gavrilov, Young-Koo Lee, Sungyoung Lee. A Framework for Smart Object and Its Collaboration in Smart Environment. In the proceedings of 10th International Conference on Advanced Communication Technology (ICACT), February 17-20, 2008, Phoenix Park, Korea, pages: 852-855

C-11. Khandoker Tarik-Ul Islam, **Md. Kamrul Hasan**, Young-Koo Lee, Sungyoung Lee. Enhanced 1-NN Time Series Classification Using Badness of Records. In the proceedings of Second International Conference on Ubiquitous Information Management and Communication (ICUIMC), January 31 - February 1 2008, SKKU, Suwon, Korea, pages: 125-131

C-12. Lenin Mehedy, **Md. Kamrul Hasan**, Young Koo Lee, Sungyoung Lee, Sang Man Han. Hybrid Dissemination Based Scalable and Adaptive Context Delivery for Ubiquitous Computing. In the 2006 IFIP International Conference on Embedded And Ubiquitous Computing (EUC 2006), Seoul, Korea, August 01-04, 2006, LNCS 4096 – 0987, ISBN: 978-3-540-36679-9

Domestic Conferences:

D-1. **Md. Kamrul Hasan**, Husne Ara Rubaiyeat, Sungyoung Lee, Young-Koo Lee. Service Discovery Using Broadcasting Data Channel. In the proceedings of KIPS 33 Spring Conference, Gangwon, April 23-24, Korea.