Mobeacon: An iBeacon-Assisted Smartphone-Based Real Time Activity Recognition Framework

Mohammad Arif Ul Alam, Nilavra Pathak, Nirmalya Roy
Department of Information Systems
University of Maryland Baltimore County
alam4@umbc.edu, nilavra1@umbc.edu, nroy@umbc.edu

ABSTRACT

Human activity recognition using multi-modal sensing technologies to automatically collect and classify daily activities has become an active field of research. Given the proliferation of smart and wearable devices and their greater acceptance in human lives, the need for developing real time lightweight activity recognition algorithms become a viable and urgent avenue. Although variants of online and offline lightweight activity recognition algorithms have been developed, realizing them on real time to recognize people's activities is still a challenging research problem due to the computational complexity of building, training, learning and storing activity models in resource constrained smart and wearable devices. To navigate the above challenges, we build Mobeacon: a mobile phone and iBeacon sensor-based smart home activity recognition system. We investigated the viability of extending Bagging Ensemble Learning (BEL) and Packaged Naive Bayes (PNB) classification algorithms for high-level activity recognition on smartphone. We incorporated the semantic knowledge of the testing environment and used that with the built-in adaptive learning models on smartphone to ease the ground truth data annotation. We demonstrated that Mobeacon outperforms existing lightweight activity recognition techniques in terms of accuracy (max. 94%) in a low resource setting and proves itself substantially efficient to reside on smartphones for recognizing ADLs in real time.

Categories and Subject Descriptors

C.3.2 [**Special-Purpose and Application-based Systems**]: Real-time and embedded systems —*Ubiquitous computing*; I.5.5 [**PATTERN RECOGNITION**]: Implementation—*Interactive systems*

General Terms

Algorithm, Experimentation, Human Factors, Performance

Keywords

activity recognition, adaptive lightweight classification, semantic knowledge, multi-modal sensing system

1. INTRODUCTION

Automatic Activity Recognition (AR) has becoming a fundamental research concern of the ubiquitous computing community [2] and plays an important role in the context-aware applications and interaction design for cognitive computation. With the rapid advancement of wireless networking and sensing technologies in recent years, recognizing Activities of Daily Living (ADLs) based on wearable sensors combined with ambient sensors have drawn much research interest. In this paradigm, wearable smart devices with sensing and wireless communication capabilities are synchronized with ambient sensors to capture different motion patterns of a user. Continuous sensor readings are collected and processed for extracting useful features, training an appropriate activity model, and recognizing a variety of activities.

When aiming at building ubiquitous applications that are ready to work in uncontrolled environments, it is often necessary to retrieve activity data from one or more wearable sensors i.e., multi-modal sensing. For daily living, even the most compact solution tends to be computationally expensive, intrusive and difficult to maintain. The ubiquitous usage of mobile devices makes these sensing platforms ideal in terms of user adoption to gather low-level activity information and further to get adherence to activity logging applications. For example, collecting locomotive data through ambient motion sensor requires an intermediate server phase to process the location signature. The extracted locomotive information needs to be sent to mobile devices in real time for online classification of activities. The entire process is computationally inefficient and prone to transmit unreliable and noisy data stream. Clearly, the use of the smartphones as activity data loggers, transmitters, receivers and processors has operational and functional limitations (such as low battery power). Although high-level activity recognition and classification through hierarchical learning algorithms (e,g., Dynamic Bayesian Network [23], Hidden Markov Model [4] etc.) offer better accuracy based on multi-modal environmental sensing, offline training and apriori ground truth data annotation, they can be hardly implemented on smart devices due to their heavyweight resource hungry computational need. On the other hand, activity classification based on ambient sensor and wearable smart devices (such as wristband, smart-phone etc.) is practically challenging due to the unavailability of adequate resources (CPU, memory, battery power etc.) to support real time recognition. Enumerating the tradeoffs between activity recognition accuracy and cost associated with the classifiers, and executing the pipeline of the activity recognition tasks either offline (server) or online (mobile) necessitate further investigation to benchmark the performance of state-of-the-art hardware and viability of implementing lightweight in-situ machine learning algorithms there.

Existing activity recognition frameworks suffer from many other practical problems. Many of them [9] are based on supervised learning where the training data requires a ground truth with accurate labeling of all activities. To ensure high accuracy, the classifiers need to be trained with long traces of data that may range from months to years. However, gathering and accurately labeling ground truth data for such a long period is difficult on smart devices and hinders the real time decision making on activity recognition. There are some existing unsupervised activity recognition algorithms that do not require ground truth ([11]) but they either require mining activity models from web definitions or they depend on detailed domain knowledge about activities and the environment.

1.1 Research Questions

Our research attempts to answer the following research questions:

• Can semantic knowledge about the environment and indoor localization help reduce the costs of ground truth labeling and bootstrap the classifier learning? If so, what kind of non-intrusive sensor devices and software applications are required to design lightweight activity recognition methodologies with reduced reliance on intermediate server and ease the process of semantic knowledge base creation?

• What kind of algorithmic models and their embodiments with the underlying learning parameters are required to support smart-phone based real time activity recognition that can take benefit from prior defined semantic information?

• How we can scale the activity recognition approaches across multi-modal sensing and indoor localization concerning accuracy and battery power consumption?

1.2 Key Contributions:

The main contributions of our work are summarized as below.

• We exploit indoor location contexts based on the iBeacon provided signal and correlated that with the user provided semantic definition of smart home and accelerometer-detected low-level activities as inferred by the smart-phone.

• We investigated Bagging Ensemble Learning (BEL) and Packaged Naive Bayes (PNB) classification algorithm for employing them on smart-phone for real time activity recognition.

• We propose an intelligent inference caching technique to opportunistically sense the semantic correlations of context tuples and add a lightweight adaptive semantic definition of smart home.

• We designed a *Mobeacon* smart home and evaluate our proposed algorithmic models using real time data traces collected from four individuals for a period of 2 weeks. The experimental study shows that our system provides an accuracy of up to 94% for high-level activity recognition. Our high-level activity prediction methodology reduces classification cost significantly compared to traditional high-level activity recognition frameworks.

2. RELATED WORKS

The idea of combining ambient sensors along with smart device sensors to recognize daily activities has been investigated in the past [6, 7, 10]. We briefly summarize several works which have investigated activity recognition models and algorithms.

Activity classification with multiple on-body sensor nodes (i.e., wearable sensor devices or smart phones) along with ambient sensor technologies have been proposed recently. SAMMPLE [3] classified high-level activities exploiting discriminatory power of activity structures along the dimension of statistical features by using a sequence of individual locomotive low-level activities. Roy et. al. [4] investigated a Coupled Hidden Markov Model (CHMM) model to infer spatial contexts and then considered spatial contexts as an additional parameter to classify high-level contexts activities. Other works [17] proposed multiple on-body sensor motes to detect user activities, body posture, or medical conditions with an offline analysis which hinders user mobility and real time analysis due to the periodic communication with a fixed base station. A model has been proposed in [12] to train with a short amount of initial data, but model updating was performed using a back-end server. Pering et. al. in [13] proposed the use of a sensor mote with an SD card attachment to interface with a mobile phone in order to meet the computational requirements of body-area sensor networks application. Lu et. al. [14] proposed a component-based approach to mobile phone-based classification for different sensors and different applications, but each sensor component required a separate classifier implementation. Activity recognition and energy expenditure was calculated by Albinali et. al. [8] using an accelerometer specific sensing model for on-body wireless nodes. Peebles et. al. [15] used AdaBoost for activity recognition with mobile phones, but focused mainly on ground truth labeling inconsistencies without delving a practical system for long term use. Wang et. al. [16] focused on duty cycling mobile phone sensors to save energy based on a rigid rule-based recognition model that must be defined before runtime but failed to provide any user-friendly interface to define the rules.

Our proposed framework is closest to PBN [17]. The central difference between PBN and *Mobeacon* is that, prior one does not exploit semantic location information which helps train the classifiers in absence of ground truth data. Though, PBN partially relies on ground truth labeling and offers significant reduction of sensing cost, it incorporates multiple wearable sensor devices and intermediate server dependencies which hinders it's use in practice. Combining the indoor location knowledge with a semantic definition, we propose a lightweight high-level activity recognition framework with reduced reliance on ground truth labeling and any intermediate server. We propose lightweight classification algorithms based on ensemble learning and packaged naive Bayes and investigate the model building, training, storing and learning on the fly to recognize activity using smart-phone.

Table 1: Mobeacon Classification Groups

Context Semantic & Inference Category	Location/Postural/High-level Contexts
Indoor Environmental Contexts	Exercise Bike, Couch, Dining Table, Bed, Closet, Reading Table, Bookshelf, Bathroom, Kitchen, Porch
Low-level contexts	Standing, Walking, Cycling, Lying, Sitting
High-level contexts	Exercising, Prepare Food, Dining, Watching TV, Prepare Clothes, Studying, Sleeping, Bathrooming, Cooking, Past Times, Random

3. APPLICATION REQUIREMENTS

Our *Mobeacon* system design is motivated by the requirements of a real time application for activity recognition. Data from multi-modal sensor devices are captured and analyzed online on mobile devices. Thus the system must be able to accurately and efficiently classify typical daily activities, postural and environmental contexts (Table 1). Despite these categories being common for many individuals and previous work [17] has identified some of them, reliance on ground truth and intermediate servers are disregarded. From the Table 1, we break down our target classifications into three groups: indoor environmental contexts, low-level contexts and high-level contexts. With the environmental and low-level contexts, we can infer high-level activity providing insight into the physical states of users for personal health and physical wellness monitoring applications. Our goal is to detect high-level activities in which a user engages in different indoor area with particular low-level activity sets, for example, 'watching TV' in 'couch' area where 'sitting' represents a low-level context state or 'preparing food' in the 'dining table' area where low-level context is composed of set {'standing','walking'}. We define this indoor areas of interest in a smart home environment a-priori with the underlying low-level contexts to infer high-level activities. This piece of information is referred as Semantic Knowledge Base (SKB) about the environment and user contexts in our work and eventually aided in on-the-fly to accelerate the real time high-level activity recognition on resource constrained devices. The requirements to provide such a practical activity recognition system are as follows.

Application Interface: Real time activity recognition relies on different sensor modalities and knowledge base input. Appropriate and user-friendly application design requirement differs based on application goal and system architecture. Both supervised and semi-supervised activity classifiers need an user-friendly software application design which must provide an intuitive interface for adding, removing, and configuring different sensors, ambient objects and ground truth related knowledge base geared to detect the user's intended activities. Defining the correlations to build such a semantic knowledge base should also be a simple, adaptive and non-invasive effort, facilitated by an easy to use mobile phone interface.

Reduced Reliance on Ground Truth: Ground truth labeling is one of the major issue for supervised and semi-supervised learning algorithms. Efficient ground truth data annotation is challenging and generates some arduous problems such as time synchronization, labeling ambiguity, interleaved activity dilemma, constant observer etc. Therefore, reliance on ground truth should be reduced as much as possible for mobile deployed real time activity recognition system where time is critical. A minimal need for ground truth reduces the burden on the user to label training data and accelerate just-in-time processing and decision making for activity recognition.

Minimal Need of Intermediate Server: Relying on intermediate server and constant connectivity violate underpinning requirements of real time activity recognition. Involvement of intermediate server implies problems such as, data inconsistencies due to lossy channel, network overflow and communication failure, security and privacy breach of data etc. Thus, conducting efficient intermediate server based classification, it is necessary to employ several energy intensive fault tolerant encryption techniques. To design real time lightweight energy efficient activity recognition framework, the system should refrain from relying on intermediate server.

Adapting Fault Tolerant Classifiers: The system must be fault tolerant and adaptive in nature thus can accurately maintain high accuracy in case of noises in sensing, device position, data transmission etc., environmental ambient dynamics (changes in the semantic location of objects/furniture etc.) or spatiotemporal changes of user context behaviors.

Implementing Lightweight Classifiers: smart-phone and ambient sensing-based technologies are subject to severe processing, storage and energy constraints. Redundant classifications result huge computational overhead and unnecessary drainage of battery power. smart-phone-based activity recognition system must be capable of selecting relevant sensor for classifying activities while powering down the other sensors. Furthermore, the system must avoid extensive parameter tuning as well as relying on complex sensing models for activity recognition in real time.

4. OVERVIEW OF OUR SYSTEM

Our *Mobeacon* system assumes a multi-inhabitant smart home instrumented with ceiling mounted iBeacons in each room transmitting RSSI signals which are being captured in the occupant's smart-phone. Our system also assumes the presence of a smart-phone in the right-pocket of each inhabitant to collect the fine-grained postural-level activities. The basic steps are summarized as follows: i) 'Sensing planar' gathers sensor data from smart-phone and RSSI signals from iBeacons; ii) 'Feature processing' module is involved with feature extraction of sensor attributes, signal processing of RSSI signals and feature processing for small training sets collected for low-level activity recognition; iii) 'Semantic knowledge processing' engine is involved with building semantic knowledge base (SKB), expression tree (ET) and training classifiers based on the small training sets; iv) 'Classification' module is involved with sub-region classification and utilizing the context information and low-level activity training sets, for classifying high-level activities employing two lightweight classifiers, packaged naive Bayes (PNB) and bagging ensemble learning (BEL). v) 'Intelligent inference cache' opportunistically updates semantic knowledge based on computationally lightweight rule mining technique. We evaluate the practicability of our proposed system consisting of the above processing engines by quantifying the reduction in computational complexity and improvement in high-level activity recognition accuracy we perceive as a result of combining iBeacon based indoor localization and

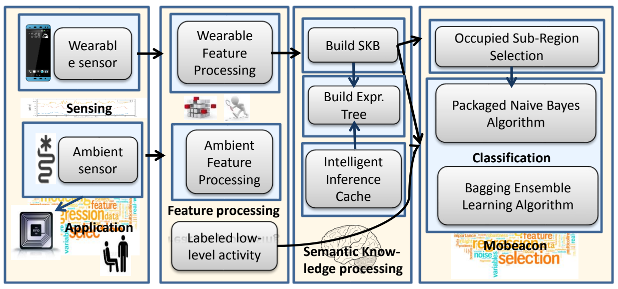
Wearable sensor
Sensing
Ambient sensor
Application
Wearable Feature Processing
Ambient Feature Processing
Feature processing
Labeled low-level activity
Build SKB
Build Expr. Tree
Intelligent Inference Cache
Semantic Know-ledge processing
Occupied Sub-Region Selection
Packaged Naive Bayes Algorithm
Classification
Bagging Ensemble Learning Algorithm
Mobeacon selection

Figure 1: Overview of our System

smartphones provided low-level activities with lightweight classifiers based naive Bayes and ensemble learning. The schematic representation of all the components, and underlying logical steps are shown in Fig 1.

5. DESCRIPTION OF OUR FRAMEWORK

Our activity recognition framework consists of the following components. 1) Semantic knowledge base; 2) Application; 3) Ambient Sensors Positioning; 4) Semantic Correlations Definition; 5) Building Expression Tree; 6) Occupied Sub-Region Detection; 7) Bagging Ensemble Learning classifier; 8) Packaged Naive Bayes classifier, and 9) Intelligent Inference Cache. Next we discuss each of these components in detail.

5.1 Semantic knowledge base

Our proposed framework depends on a simplified domain knowledge consists of the spatio-contextual correlations. Thus we define two modules: a semantic knowledge base and a semantic correlation rule sets. Prior to setting up *Mobeacon* in smart home environment, user needs to define a set of correlations to build semantic knowledge base. User also needs to define the sub-regions prior to *Mobeacon* smart home installation as follows: 1) First, presume the high-level activity sets to be recognized. In *Mobeacon* smart home, we selected 11 high-level activities as shown in Table 1 which are relevant to functional health of older adults, 2) Identify and define the critical sub-regions which are involved with pre-selected high-level activities, for example, sub-region 'SR1' is involved with high-level activity 'Exercising' as shown later in Fig 9, We selected 11 sub-regions that are involved with single or multiple high-level activities; 3) Locate the ambient sensors and demark the area of each sub-region; 4) Set an approximate minimum time threshold for any high-level activity recognition (we consider 2 minutes). Our adaptive heuristic changes the minimum threshold over time based on the activity monitoring; 5) Finally, define the correlations among high-level activities, low-level activities and sub-regions. Table 2 shows our *Mobeacon* smart home semantic knowledge base correlations.

5.2 Application

We design an Android (version 5.0 Lollipop) based user-friendly mobile application that simplifies the procedures of defining, controlling, and monitoring the semantic knowledge and context recognition as shown in Fig 2(a). We combine Estimote iBeacon SDKs with Android smart-phone sensor manager and provide intuitive application interfaces to confirm different sensors inclusiveness and their usability.

Table 2: Semantic Knowledge Base Correlations

High-level activities	Sub-regions	Low activities
Exercising	SR1	Cycling, Sitting
Prepare Food	SR4	Standing, Walking
Dining	SR4	Sitting
Watching TV	SR2, SR3	Sitting
Prepare Clothes	SR8	Standing, Walking
Studying	SR7	Sitting
Sleeping	SR5	Lying
Bathrooming	SR9	Standing, Walking, Sitting, Lying
Cooking	SR10	Standing, Walking
Past Times	SR11	Standing, Walking, Sitting
Random	SR1-SR11	Standing, Walking, Lying, Sitting

5.3 Ambient Sensors Positioning

For this task, user needs to localize the ambient sensor. This localization process addresses two important scenarios. 1) Single ambient sensor rooms; 2) Three ambient sensor rooms. For single ambient sensor rooms (for example, kitchen, bathroom etc.), user needs to select 'single ambient sensor room' from the drop-down list of our develop app, bring the smart-phone as close as possible to the ceiling mounted iBeacon and then press the button 'sensor localize' to scan. This action records the particular sensor UUID, major-minor values and the distance measure (in meters) and stores the information as origin (x=0, y=0) coordinate of the room. For three sensors room, user needs to do the above procedure once for both the sensors 1 and 2 (illustrated in Fig 2(c)). Our application automatically calculates X *and* Y values based on the average of l, m *and* n using following equations:

$$X = l; Y = \sqrt{n^2 + \frac{l}{2}^2} \tag{1}$$

5.4 Semantic Correlations Definition

In order to define the semantic correlations though our user-friendly application interface, user needs to bring the smart-phone at the center of each of the pre-defined sub-regions, select/add new sub-region, select correlated low- and high-level activities, type approximate length of the region considering it as a square shaped one and click 'Set' button to confirm definition (Fig 2(a)). The above procedure calculates the four corners' coordinates using the iBeacons provided distance values (Fig 2(a)), and saves all values in the smart-phone memory.

5.5 Building Expression Tree

An expression tree for a tuple $a = v$ is a tree representation of the Boolean expression that implies the value v for the attribute a. It is generated by combining various rules that imply $a = v$ directly, or indirectly via transitive relationships with other rules. The expression tree for a given high-level activity 'cooking' is constructed as follows. We start with 'cooking' high-level activity and continue expanding the rules stated in the semantic knowledge base ('cooking'⇒'walking' AND 'kitchen', 'cooking'⇒'standing' AND 'kitchen') until we reach the leaf nodes which can't be expanded any further(illustrated in Fig 3(a)).

5.6 Occupied Sub-Region Detection

This application takes the semantic knowledge base (SKB) information where sub-region set is defined as $SR = \{sr1, sr2$

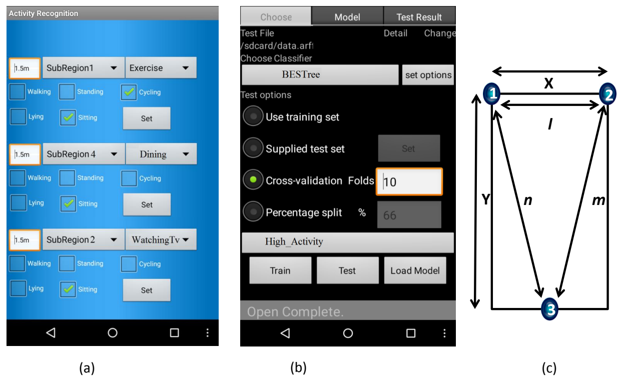

Figure 2: (a) Semantic knowledge base creation app (b) Classification app (c) iBeacon position coordination

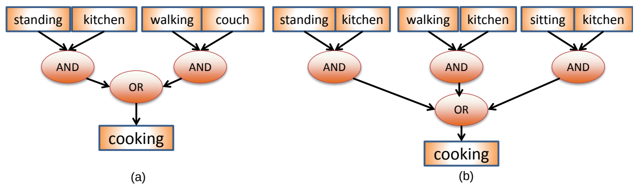

Figure 3: 'Cooking' high-level activity tuple expression tree (a) at the beginning of inference (b) after 1 hour of running rule mining

$\ldots, sr11\}$. Consider $sr = \{\langle x1, y1\rangle, \langle x2, y2\rangle, \langle x3, y3\rangle, \langle x4, y4\rangle\}$ where x and y values define the four corner coordinates of sub-region, sr. Our application continuously senses the iBeacon features (id and distance measure) and calculates the coordinates of the smart-phone device inside the smart home using trilateration [19]. It checks whether the Trilateration provided values (x and y coordinates) are inside any of the pre-defined critical sub-region space not using simple scalar dot products of vector method.

$$\begin{aligned} &M \textit{ of coordinates } (x, y) \textit{ is inside the} \\ &\textit{rectangle } ABCD \text{ iff } (0 < AM \cdot AB < AB \cdot AB) \\ &\wedge (0 < AM \cdot AD < AD \cdot AD) \end{aligned} \tag{2}$$

If it does not satisfy any of the pre-defined sub-region occupancy, it returns 'Other' sub-region which is defined only for 'Random' type of high-level activity. The detailed algorithm is stated in Fig 4

```
Procedure Occupied_Sub-Region_Detection
(input:  Sub-region definition $SR$, iBeacon $IB$;
output:  Occupied Sub-region ID $S$)
1.  if $IB.size = 1$//one iBeacon is discovered
2.    for each $sr$ in $SR$
3.      if $sr.size = 1$ and $sr.UUID = IB.UUID$
4.    Return $sr$
5.  else if $IB.size = 3$// 3 iBeacons are discovered
6.    {x, y} $\Rightarrow$ Trilateration ($SR$,$IB$)
7.  For each $sr$ in $SR$
8.    if {x, y} inside $sr$// Equation 2
9.      Return $sr$
10.   else
11.     Return 'Other'
```

Figure 4: Subregion Detection Algorithm

```
Procedure BESTree_Training
(input:  Training set $S$; Ensemble
 classifier $E$; Integer T (number of bootstrap
  samples
 output:  Trained Ensemble Classifier $E$)
1.  for $i = 1$ to $T$
2.    $S_b$ = bootstrap sample from $S$ // sample with replacement
3.    $S_{oob}$ =out of bag sample
4.    train base classifiers in $E$ on $S_b$
5.    $E_i$ =BESTree_Selection($M$,$S$)
6.  Return $E$
```

Figure 5: BESTree Training Algorithm

5.7 Lightweight Bagging Ensemble Learning

We propose an extension of bagging ensemble learning algorithm adding an expression tree based inference engine to help guide the activity recognition process in real time. Our proposed algorithm is described in two folds, training and classification.

We use BESTree selection algorithm [20] to train our lightweight activity recognition model and Decision Tree (DT) as a base classifier. We consider smart-phone provided accelerometer values (x, y and z) and Subregion Detection algorithm provided spatial context (SR), in other words the locomotive micro activity contexts and location context.. We have also incorporated the minimum duration values of activities as obtained from the semantic knowledge base.

Let x be an instance and m_i; $i = 1...k$, be a set of base classifiers associated with probability distributions $m_i(x, c_j)$ for each class label c_j, $j = 1..n$. The output of the final classifier ensemble $y(x)$ for instance x can be expressed as:

$$y(x) = \arg\max_{c_j} \sum_{i=1}^{k} \omega_i m_i(x, c_j), \tag{3}$$

where ω_i is the weight of base classifier m_i. We employ, ensemble learning strategies as underlying methods for calculating optimal weights for each base classifier given the hierarchical approach consisting of micro activity recognition and combining this with SKB and location context for higher-level activity recognition.

Bagging: Bagging is based on the instability of base classifiers, which is in our approach exploited to improve the predictive performance of any unstable base classifiers. The basic idea is that, given a training set T of size n and a classifier A, bagging generates m new training sets, T_i , each of size $n' \leq n$. Bagging then applies classifier A with each training set T_i to build m models and use simple voting mechanism to finalize the output.

Lightweight Design: We extend BESTree classification algorithm incorporating sub-region detection, expression tree based high-level activity composition and activity duration. We bring in lightweight expression tree based inference engine after recognizing the micro-activities based on each bag of BESTree base classifier. Expression tree based inference engine extracts the expression trees saved in the inference cache, augments the sub-region information obtained from the Sub-Region Detection algorithm, applies inference on m

```
Procedure BESTree_Classification
(input:  Test set $T$; Trained Ensemble Selection classifier $E$;
iBeacon $IB$; smart-phone accelerometer values (x,y,z); Semantic
Knowledge Base $SKB$; output:  High-level activity $H$)
1.   $F$ = Feature_Extraction(x,y,z)
2.   $C$ = BESTree_Ensemble_Classifier($F$,$E$)
3.   $LA(t)$ = Max_Vote($H_{set}$)// low-level
     // activity $LA$
4.   $HA_t$ = Expression_Tree_Classification($L$,$ET$)
     // high-level activity $HA$
5.   if $HA(t) \neq HA(t-1) \& HA(t) = HA(t-2)$
6.      $\& HA(t-1) < MinDuration(HA(t-1))$
7.      $HA(t-1) \Leftarrow HA(t-2)$
8.      Duration($HA(t)$)=Duration($HA(t-1)$)+
9.      Duration($HA(t-2)$)
10.  Return $HA$
```

Figure 6: Bagging Ensemble Learning Classification

model predicted low-level activities along with sub-region information and finally votes on each expression tree output to predict high-level activities. In addition, we design a duration model based heuristic algorithm to confirm more fine-grained prediction of high-level activity. Whenever, we find an end of a high-level activity, we check the last ended high-level activity duration. If it does not meet the minimum duration, it is merged with its previous one. The entire algorithm is stated in 6.

5.8 Lightweight Packaged Naive Bayes Algorithm

We investigate the viability of Packaged Naive Bayes (PNB) for implementing real time activity recognition on smartphones. We consider PNB over hidden naive Bayes or naive Bayes [1] due its advantage in terms of reduced test time in the recognition phase and higher accuracy over multi-modal sensor datasets [5].

Assume that $A_1, A_2, \ldots, A_n$ are n attributes corresponding to attribute nodes in a Bayesian Network. An example instance E is represented by a vector $< a_1, a_2, \ldots, a_n >$, where a_i is the value of A_i. Let C represent the class variable corresponding to the class node in a BN. We use c to represent the value that C takes and $c(E)$ to denote the class of E. Assume that all attributes are independent given the class, the resulting classifier Naive Bayes is represented by,

$$c(E) = \arg\max_{c \in C} P(c) \prod (a_i|c) \tag{4}$$

If we denote hidden parent by a_ϕ, HNB can be defined as follows.

$$c(E) = \arg\max_{c \in C} P(c) P(a_1, a_2, a_3, ..a_n|a_\phi, c) \tag{5}$$

In Packaged Naive Bayes (PNB) each attribute either has a bag of HNB as its hidden parent node or a bag of NB. During the classification phase, the attributes which have bag HNB use HNB algorithm, and other attributes in the bag NB correspond to NB algorithm. At training phase, each attribute selects the attributes whose dependence with it are greater than the threshold for being considered in corresponding bag HNB_i (HNB_i is hidden parent). However, if none of the attributes are selected, the attribute is being considered into the bag NB (similar to bagging). When the dataset dimension is low, the threshold is set to a smaller value, and that there are more attributes being chosen with

```
Procedure Training_PNB
(input:  $n^{C_L}$, $\mathbf{D}$;//$\mathbf{D}$ probability distribution
output:  PNB Model $\mathcal{M}$)
1.   for each $c_{Li} \in C_L$// low-level activity state
2.     Compute $P(c_{Li})$ from $D$// observation probability
3.     for each pair of attributes $A_i$ and $A_j$
4.       for each $a_i$,$a_j$, and $c_{Li}$ to $A_i$,$A_j$, and $C_L$
5.         Compute $P(a_i|a_j, c_{Li})$ from $D$
          // conditional probability between two attributes
6.     for each pair of attributes $A_i$ and $A_j$
7.       Compute $I_p(A_i;A_j|C_L)$ and $W_{ij}$ from $D$
         // Conditional Mutual Information between two attributes
8.       for each $A_i$
          for each $A_i \neq A_j$ in $D$
            If $I_p(A_i;A_j|C)$ $\geq$threshold $HNB_i \rightarrow A_j$
              // conditionally dependent attributes
9.           else NB$\rightarrow$ $A_i$// conditionally
              // independence attributes
10.  $\mathcal{M} \Rightarrow$ HNB($I_p$,$A$,$P$,$C$,$W$)
11.  Return $\mathcal{M}$
```

Figure 7: Packaged Naive Bayes Training Algorithm

its corresponding HNB bag with a higher accuracy. Though the classification process in this case compared to ensemble learning takes a little longer, it it performs better because the dimension of datasets remain low. When the threshold is small enough (such as a negative value), this model evolves into HNB. So if the threshold is set an optimal value, the model would be a perfect combination of HNB and NB, which would have a high accuracy and short classification time. The idea is if just the most correlative attributes are concerned, it is not only reducing the classification time but also improving the accuracy. The classifier corresponding to PNB on an instance E is defined as follows:

$$c(E) = \arg\max_{c \in C} P(c) \prod_{i=1}^{number\_of\_HNB\_bags} P(a_i|a_\phi, c) \prod_{j=1}^{number\_of\_attributes\_in\_NB} P(a_j|c) \tag{6}$$

The conditional probability of the attribute and its hidden parent as well as W_{ij} are defined below.

$$W_{ij} = \frac{I_P(A_i;A_j|C)}{\sum_{j=1, j\neq i}^{\#attributes\_HNB_i} I_P(A_i;A_j|C)} \tag{7}$$

$$I_P(A_i;A_j|C) = \sum_{a_i,a_j,c} P(a_i,a_j,c) \log \frac{P(a_i,a_j|c)}{P(a_i|c)P(a_j|c)} \tag{8}$$

$$P(a_i|a_\phi, c) = \sum_{j=1, j\neq i}^{\#attributes\_HNB_i} W_{ij} \times P(a_i|a_j, c) \tag{9}$$

We extend the PNB by adding an expression tree based classifier at the end of PNB based low-level activity classification and applying our heuristic model to confirm a fine-grained high-level activity classification. Fig 7 and Fig 8 show our modified PNB algorithm training and inference algorithms respectively.

5.9 Intelligent Inference Cache

The relationships among various context attributes (i.e., low-level, location and high-level context attributes) cannot be static within an application. It is also tedious and

```
Procedure Inference_PNB
(input:  an instance $E$; a PNB model $\mathcal{M}$
output:  classified instance $HA$)
1.  for each value $c_{Li}$ of $C_L$// each low-level
2.    for each $a_i \in E$// each attribute
3.      If $HNB_i \neq \emptyset$, Calculate $P(a_i|a_\phi, c_L)$
          //$a_\phi$ hidden parent attribute
4.      else NaiveBayes(NB,E,$C_L$)
5.  Compute $K \rightarrow c_L(E)$
6.  $HA_t =$ Expression_Tree_Classification($K$,$ET$)
      //high-level activity $HA$ at time $t$
7.  if $HA(t) \neq HA(t-1) \& HA(t) = HA(t-2)$
      $\& HA(t-1) < MinDuration(HA(t-1))$
      $HA(t-1) \Leftarrow HA(t-2)$
8.   Duration($HA(t)$) $=$ Duration($HA(t-1)$)+
9.   Duration($HA(t-2)$)
10.Return $HA$
```

Figure 8: Packaged Naive Bayes Inference Algorithm

error-prone for a single person to define all possible context relationships within a smart-home environment. Behaviors of the person are also volatile and they change over time through a specific pattern, regularity or correlation remains consistent. We introduce an intelligent inference cache module to solve these problems employing an adaptive update of the expression tree using rule mining techniques. Our intelligent inference cache learns additional rules that are not so-obvious to semantic knowledge definers and dynamically adapts over time. Meanwhile, applying rule mining in every time segment may drastically consume battery power of smart-phone device. Therefore, we propose to aid in rule mining technique on temporal segments when there is confusion within the classifier to recognize two activities. Thus we propose to merge one activity with prior activity and consider that new activity as an occurrence of behavioral pattern change. We update this pattern change in the expression tree configuration to make our inference process more robust. These techniques are described below.

Rule Mining: Through the rule mining, we aim to generate rules that have the following general form: $\langle C_1, C_2, ..., C_n \Rightarrow \mathcal{R} \rangle$ which implies that $\mathcal{R}$ holds whenever all the tuples $\langle C_1, C_2, ..., C_n \rangle$ are true. For example, the rule $\langle$*walking=True; kitchen=True ⇒cooking_together=True*$\rangle$ implies that if a user is in the *location* 'kitchen' with low-level context state 'walking' then he is in high-level context state 'cooking'. We investigated Apriori algorithm to operate on naive activity state space model to pinpoint the underpinning spatial contexts [18]. A threshold *thresh* is defined, that is used by Apriori algorithm to identify the sets of low-level context states which are subsets of at least T high-level activity. We formulated the Association rules with a support and a confidence. For example, if a state space model has 1000 context states, out of which 200 include both context states A and B and 80 of these include context state C, the association rule $A, B \rightarrow C$ has a support of 8% (= 80/1000) and a confidence of 40% (= 80/200). The algorithm takes two input parameters: *minSup* and *minConf* to generate all the rules with *support* $\geq$ *minSup* and *confidence* $\geq$ *minConf*. We assume *minSup = 90%* and *minConf = 10%* which help strike good balance between tolerating occasional inconsistencies and highlighting the viable rules for our hierarchical context state space model. Our heuristic model based high-level activity merging event activates rule mining, calculates *minSup* and *minConf* over merged segments, generates rules (if meets the threshold) and helps update the existing expression trees. Note that Inference Cache can sometimes return results that are *i. incorrect*, if user's behavior deviates from context rules, or *ii. stale*, if results are inferred from stale values due to idle nature of iBeacon and signal strength distraction by human obstruction. The severity of incorrectness is reduced by the rule miner conservatively choosing rules that potentially hold in the activity merging event. Similarly, to reduce the effect of staling, we choose a small cache expiry time of 2 minutes.

Rebuild Expression Tree: Intelligent Inference Cache is activated when a high-level activity merging happens and generates a set of rules using Apriori algorithm based rule mining. These set of rules are used to generate new expression tree for merged high-level activities as follows. Suppose a new rule for 'cooking' high-level context is generated by the inference cache: 'cooking'⇒{'sitting' AND 'kitchen'}. The rebuilding of expression tree for 'cooking' context starts with expanding 'cooking' node with all associated base rules including the new rule (i.e., 'cooking'⇒{'standing' AND 'kitchen' }, 'cooking'⇒{'walking' AND 'kitchen'} and 'cooking'⇒{ 'sitting' AND 'kitchen'}). Finally a new expression tree is formed and cached in our intelligent inference cache for 'cooking' activity classification as shown earlier in Fig 3(b).

6. EXPERIMENTAL EVALUATION

In this section, we introduce our, *Mobeacon* smart home system setup, configuration and data collection. We perform preliminary experiments to show the performance of our proposed lightweight algorithms for real time activity recognition using bare minimal sensing, processing, computing, and storing resources.

6.1 Hardware Interfacing

We developed an integrated hardware software toolkit focusing on having an intuitive application interface and reduced reliance on ground truth and intermediate server. We first describe two Bluetooth Low Energy (BLE) technology based devices as part of our *Mobeacon* smart home system setup.

Estimote iBeacon: Estimote Beacons [22] are small wireless sensors that can be attached to any location or object. They broadcast tiny radio signals (RSSI) which smartphones can receive and interpret, thus unlocking micro-location and contextual awareness. With the Estimote Android SDK, applications on Android based smart-phone are able to approximate their proximity to nearby locations and objects, recognize their types, ownerships and measure temperature and motion. We integrate Estimote Android SDK with our *Mobeacon* Android application that gives distance measure (in meter) between smart-phone and the Estimote Beacon along with the UUID, major and minor values. A single area covered by multiple Estimote Beacons generate same UUID but different major and minor values. Thus we can separate individual sub-room level occupancy using the unique ID (UUID).

6.2 Testbed setup and Data Collection

We develop a real testbed for *Mobeacon* smart home system in a family apartment with a bedroom, living room, kitchen, long corridor, a porch and a bathroom (Fig 9(a)). We installed 8 iBeacons and one SensorTag Beacon on the

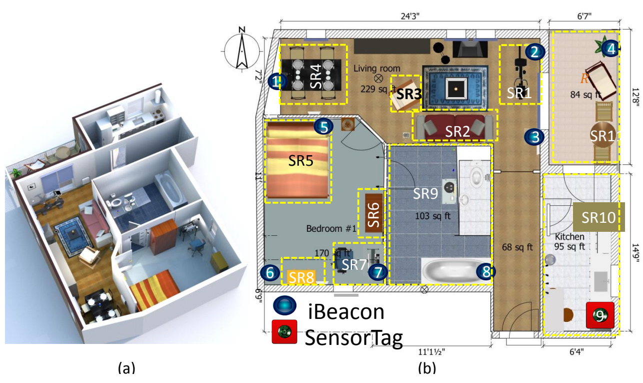

Figure 9: Mobeacon Smart Home Set Up

ceiling as displayed in Fig 9(b). The rooms which are involved with multiple high-level activities and complex subregion correlations (such as, bedroom is involved with studying, sleeping, pressing clothes etc.) are equipped with three iBeacons. The rooms with less activities are equipped with single iBeacon (such as porch area). We also set up three IP cameras in three appropriate places thus we can exclusively collect the ground truth of all the performed activities.

We recruited one married couple to live in this apartment for two weeks. They were trained of using our *Mobeacon* Android application and showed them how to build their own semantic knowledge base of daily livings (took 20 minutes on average to complete semantic knowledge base inputs). They were asked to perform our pre-defined 5 low-level activities keeping the smart-phone in their right pocket for two minutes each. During this phase we gathered a small amount of sensor readings and created training sets for our study. Finally, they were left alone to perform their natural daily activities. Besides the two inhabitants we have recruited another two participants in the test apartment to perform all of the scripted ADLs several times without any prior instructions. They have been asked to keep the *Mobeacon* Android application installed smart-phone on their right pocket. After gathering entire testbed data, we recruited two graduate students to label sub-regions, low-level activities and high-level activities based on the recorded videos and validate each others' annotations. This generated 12 hours of fine-grained labeled ADLs data with 35 chunks of continuous data sets. These chunks are defined individually for (couple), (couple + 1) or (couple + 2) inhabitants smart home activity logs. The entire dataset finally reflects the, natural ADLs of 4 individuals with semantic knowledge base, smart-phone accelerometer values (x, y and z axis), iBeacon values (UUID, major, minor, distance), sub-region IDs and time.

6.3 Cost Accuracy Tradeoffs

In this section we present the results comparing the cost-accuracy tradeoffs of our models with other traditional models. We implemented our proposed BEL and PNB algorithms and investigated their performance over existing algorithms. We chose threshold = 0.138 for PBN and a set of parameters $P = \langle num\_bag = 4, num\_execution\_slots = 4, num\_features = 2, num\_trees\_per\_bag = 7\rangle$ for BEL which were found optimal through experiment for getting reduced overhead and improved accuracy.

6.3.1 Accuracy Tradeoffs

We have taken different approaches in different accuracy

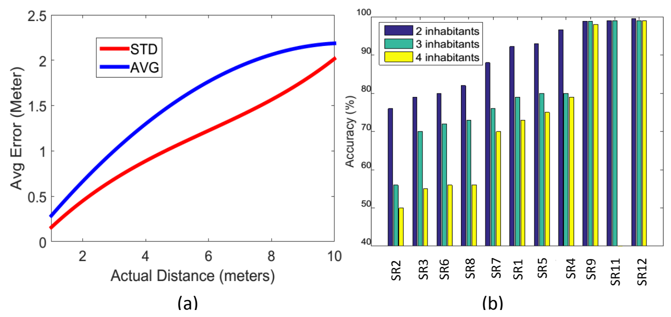

Figure 10: (a) iBeacon distance accuracy (b) Trilateration based localization accuracy for each sub-region (sorted by area)

tradeoffs in our experiment. In each approach, we choose the following equation to measure accuracy:

$$accuracy = \frac{\#TP + \#TN}{\#TP + \#FP + \#FN + \#TN} \tag{10}$$

where TP, TN, FP and FN represent True Positives, True Negatives, False Positives and False Negatives respectively. To measure accuracy of activity classification, we used datasets chunk by chunk to maintain its continuity.

iBeacon Sensing Accuracy: To test the accuracy of distance measures as obtained from the iBeacon, we calculated errors between the actual and iBeacon SDK provided distance measures in a no obstacle environment. We tested the distances ranging from 0 to 10 meters for 40 times each and plotted the error measures as shown in Fig 10(a). We observed that the error measures are not exactly linear but proportional to the physical distances between the physical phone and the iBeacon. The error within a range of 1 meter was almost ≈ 0 but it became ≈ 2 meter at the range of 10 meters.

Indoor Localization Accuracy: We implemented trilateration based sub-region detection algorithm using Java and evaluated its accuracy through our *Mobeacon* dataset. Fig 10(b) shows the regions with one iBeacon ('SR9', 'SR10' and 'SR11') and depicts $\approx 100\%$ accuracy for any number of inhabitants. Other sub-regions that were involved with multiple iBeacons, accuracies were proportional to their area measure. Furthermore, we noted that the sub-region detection accuracy has been decreased as the number of inhabitants are increased (87%, 71% and 65% average accuracies are noted in case of 2, 3 and 4 inhabitants respectively).

Low-level Activity Recognition Accuracy: To classify the low-level activities, the 3-axis accelerometer data streams collected from the smart-phone were broken up into successive frames. We extracted a number of statistical features (i.e., mean, variance, standard deviation, maximum and minimum, magnitudes, energy etc.) for accelerometer to create a 20-dimensional feature vector set. We chose 20 samples per second and 1.5 seconds windowing for enumerating the feature sets for low-level activity recognition. The low-level ground-truth annotated training set was then fed into the classifiers. We identified 6 best optimized feature sets (avg. X, max. Y, avg. magnitude, std. magnitude, XY correlations and energy) through Correlation Feature Selection (CFS) algorithm [24]. To employ BEL and PNB for low-level activity recognition, we excluded the expression tree based inference. We also fed the entire training dataset into Weka toolkit [21] and trained 4 classifiers: Naive Bayes,

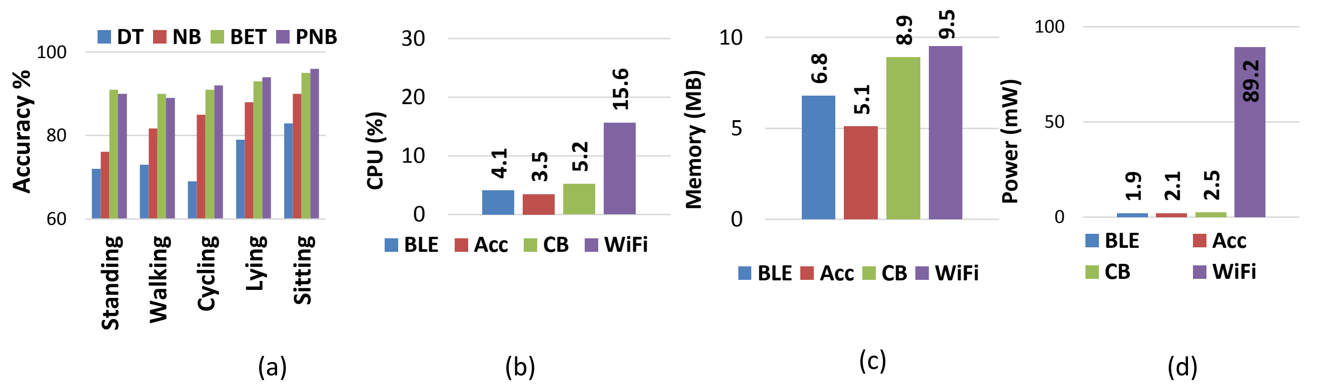

Figure 11: (a) Low-level activity recognition accuracy and (b) CPU utilization (%) (c) Memory Usage (MegaBytes) and (d) Power Consumption (miliWatt) measure for Bluetooth Low Energy (BLE), Accelerometer (ACC), Classical Bluetooth (CB) and WiFi

High Activities	T-BEL	T-PNB	LB	RF	HMM	L-BEL	L-PNB
Exercising	100	100	100	100	100	100	100
Prepare Food	76.9	90.6	76.9	75.5	75.9	87	91
Dining	94.1	100	94.1	93.2	92.5	99	100
Watching TV	100	72.3	100	97.5	96.2	100	75
Pressing Clothes	93.6	96.3	93.6	93	95.2	96	97
Studying	95.8	99.3	94.3	95	91.5	99	99.3
Sleeping	100	96.9	100	100	100	100	96.9
Bathrooming	97.5	93.7	96.5	97	95.5	99	95.3
Cooking	99.6	94.2	98.3	99	94.2	99.6	94.8
Past Times	75.4	97.9	72.2	75.5	72.6	85	98.2
Random	75.2	84.3	75.1	75.2	81.2	87	85.2
Overall	92	92.2	91.1	91	93.7	94.5	94.3

Table 3: High-level activity recognition accuracy using traditional BEL, traditional PNB, LogitBoost, RandomForest, HMM, lightweight BEL and lightweight PNB

Decision Tree (DT), BEL and PNB. Fig 11(a) illustrates the accuracy measures for each low-level activities. We note that BEL and PNB outperform two most popular low-level activity classification algorithms, DT and NB, achieving ≈ 87% and ≈ 88% accuracy respectively.
High-level Activity Recognition Accuracy: We implemented several other popular activity classification algorithms such as Hidden Markov Model (HMM), LogitBoost (LB) and RandomForest (RF) using java APIs in Weka [21] and compared the accuracy with lightweight BEL and PNB implementation. From Table 3, we see that lightweight model helps improve high-level activity recognition accuracy by 2%. Although, BEL and HMM show similar accuracy, (lightweight BEL 94.5%, lightweight PNB 93.9%, HMM 93.7%), classification accuracy in case of complex activities like 'cooking', 'prepare food' etc. tend to be more accurate in lightweight BEL model than the other ones.

6.3.2 *Cost Tradeoffs*

We define cost tradeoffs for classification in three different perspectives: 1) CPU usage; 2) memory usage, and 3) power consumption. We discuss the details cost tradeoffs below.

iBeacon Sensing Cost: Bluetooth Low Energy (BLE) has been used as a signal transmission protocol of iBeacon. To compare iBeacon sensing power consumption with other sensing technologies, we implemented four Android applications in Google Nexus 4 smartphone. Each application was involved with accelerometer, iBeacon, Classical Bluetooth (CB), and WiFi sensing technologies. Each application started with activating their relevant sensors, recording sensing values and executing their individual tasks. We transmitted data at 100 bytes per second rate over WiFi, Bluetooth and BLE technologies to measure their respective performance. We ran the 4 applications separately for 10 minutes and measured average CPU, memory and power consumption for each application using PowerTutor Android application. We note that WiFi based data transmission uses as much as 42 times of the power than Classical Bluetooth communication protocol. On the other hand, iBeacon technology based data transmission consumes 1.3 times less power than CB for data transmission (Fig 11)(d). iBeacon and smartphone's accelerometer sensing based approaches also occupy much lesser CPU and memory usage that other sensing technologies (Fig 11(b) and Fig 11(c)).
Classification Cost: We implemented WEKA in Android platform (MobileWeka) to evaluate the performance of our two proposed algorithms, lightweight BEL and lightweight PNB. The advantage of using MobileWeka is that, it supports WEKA toolkit in Android platform that helps to implement LogitBoost and RandomForest classifiers on smartphone too. We ran each algorithm separately for 10 minutes and measured average CPU, average memory and total power consumption for each application using PowerTutor. We illustrated these results in Fig 12 where we see that, lightweight BEL performs better than other lightweight classifiers occupying ≈ 4% less CPU and 30MB less memory and consuming 24% less power.

6.4 Intelligent Inference Cache Performance

We incorporated an intelligent inference cache that periodically senses the contexts (low-, high- and location) and updates the expression tree over time. At the initial phase, inference cache frequently updates the expression trees incurring higher cost with lower accuracy. When expression tree becomes stable it helps improve the accuracy with a lower cost as shown in Fig 12(d).

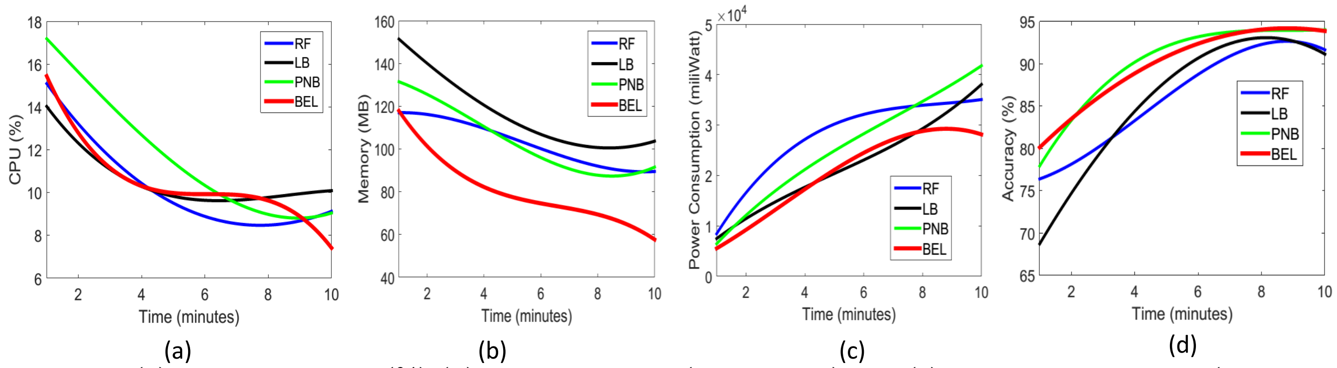

Figure 12: (a) CPU utilization (%) (b) Memory Usage (MegaBytes) and (c) Power Consumption (mili-Watt) (d) Accuracy measure for Bagging Ensemble Learning, RandomForest, LogitBoost and Packaged Naive Bayes classifiers

7. CONCLUSION

Our iBeacon sensor assisted smartphone-based activity recognition system holds promises to recognize activity in real time with higher accuracy and lesser CPU, memory, cache and power usage. Our contribution is mainly divided into two parts: first, we set up a *Mobeacon* smart home with cheap available energy efficient hardware and smart-phone software system. Second, we extend two types of lightweight robust classification algorithms separately: 1) bagging ensemble learning classifier; and 2) packaged Naive Bayes classification algorithms. We combine iBeacon with smart-phone; and exploit an indoor localization based energy efficient activity recognition framework which help reduce the reliance on ground truth data collection and intermediate server processing. We also design an effective smart-phone application interface for defining and creating an initial semantic knowledge base about the smart home environment. We effectively use our semantic knowledge base, expression tree based activity construction, and inference cache to accelerate the activity recognition process of our lightweight bagging ensemble learning (BEL) based approach. Our proposed BEL attests the promise of our methodologies and outperforms several existing lightweight classifiers in terms of CPU, memory, and cache occupancy and power consumption performance. We also show that our efficient duration model based high-level activity merging provides higher accuracy in high-level activity recognition even with less accurate low-level activity detection.

Acknowledgement

This work is supported partially by the NSF Award #1344990, UMB-UMBC Research and Innovation Partnership Grant, and Constellation E^2: Energy to Educate Grant.

8. REFERENCES

[1] L. Jiang, H. Zhang, Z. Cai, A Novel Bayes Model: Hidden Naive Bayes, IEEE Trans Knowl Data Eng (2009).
[2] L. Atallah, G. Yang, The use of pervasive sensing for behaviour profiling - a survey, PMC (2009).
[3] Z. Yan, V. Subbaraju, D. Chakraborty, A. Misra, K. Aberer, Energy-Efficient Continuous Activity Recognition on Mobile Phones, An Activity-Adaptive Approach, ISWC (2012).
[4] N. Roy, A. Misra, D. Cook, Infrastructure-assisted smart-phone-based ADL recognition in multi-inhabitant smart environments, PerCom (2013).
[5] Y. Ji , S. Yu , Yafeng ZhangA, Novel Naive Bayes model: Packaged Hidden Naive Bayes, ITAIC (2011).
[6] K. Lin, A. Kansal, Lymberopoulos, D., Zhao, F. Energy-accuracy trade-off for continuous mobile device location, MobiSys (2010).
[7] E. Hoque, J. Stankovic, AALO: Activity recognition in smart homes using Active Learning in the presence of Overlapped activities, PervasiveHealth (2012).
[8] F. Albinali, S. Intille, W. Haskell, M. Rosenberger. Using Wearable Activity Type Detection to Improve Physical Activity Energy Expenditure Estimation, UbiComp (2010).
[9] M. Buettner, R. Prasad, M. Philipose, D. Wetherall, Recognizing daily activities with rfid-based sensors, UbiComp (2009).
[10] M. Alam, N. Roy, GeSmart: A gestural activity recognition model for predicting behavioral health, Smartcomp (2014).
[11] T. Dimitrov, J. Pauli, E. Naroska, Unsupervised recognition of ADLs, SETN (2010).
[12] E. Miluzzo, C. Cornelius, A. Ramaswamy, T. Choudhury, Z. Liu, A. Campbell, Darwin Phones: The Evolution of Sensing and Inference on Mobile Phones, MobiSys (2010).
[13] T. Pering, P. Zhang, R. Chaudhri, Y. Anokwa, R. Want, The PSI Board: Realizing a Phone-Centric Body Sensor Network, BSN (2007).
[14] H. Lu, J. Yang, Z. Liu, N. Lane, T. Choudhury, A. Campbell, The Jigsaw Continuous Sensing Engine for Mobile Phone Applications, SenSys (2010).
[15] D. Peebles, T. Choudhury, H. Lu, N. Lane, A. Campbell, Community-Guided Learning: Exploiting Mobile Sensor User to Model Human Behavior, AAAI (2010).
[16] Y. Wang, J. Lin, M. Annavaram, Q. Jacobson, J. Hong, B. Krishnamachari, N. Sadeh, A Framework of Energy Efficient Mobile Sensing for Automatic User State Recognition, MobiSys (2009).
[17] M. Keally, G. Zhou, G. Xing, J. Wu, Andrew J. Pyles, PBN: towards practical activity recognition using smart-phone-based body sensor networks, SenSys (2011).
[18] Suman Nath, ACE: Exploiting Correlation for Energy-Efficient and Continuous Context Sensing, MobiSys (2012).
[19] R. Misra, S. Shukla, V. Chandel, Lightweight Localization Using Trilateration for Sensor Networks, IJWIN (2014).
[20] Q. Sun, B. Pfahringer, Bagging Ensemble Selection for Regression, ACALCI (2012).
[21] I. Witten, E. Frank, Data Mining: Practicial Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann (1999).
[22] Estimote iBeacon, http://estimote.com/
[23] A. Parate, M. Chiu, D. Ganesan, B. M. Marlin, Leveraging graphical models to improve accuracy and reduce privacy risks of mobile sensing, MobiSys (2013).
[24] Mark A. Hall, Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning, ICML (2000).