Appl Intell (2011) 35:242–259
DOI 10.1007/s10489-010-0217-4

GPARS: a general-purpose activity recognition system

Jehad Sarkar · La The Vinh · Young-Koo Lee · Sungyoung Lee

Published online: 13 March 2010

Abstract The fundamental problem of the existing Activity Recognition (AR) systems is that these are not general-purpose. An AR system trained in an environment would only be applicable to that environment. Such a system would not be able to recognize the new activities of interest. In this paper we propose a General-Purpose Activity Recognition System (GPARS) using simple and ubiquitous sensors. It would be applicable to almost any environment and would have the ability to handle growing amounts of activities and sensors in a graceful manner (Scalable). Given a set of activities to monitor, object names (with embedded sensors) and their corresponding locations, the GPARS first mines activity knowledge from the web, and then uses them as the basis of AR. The novelty of our system, compared to the existing general-purpose systems, lies in: (1) it uses more robust activity models, (2) it significantly reduces the mining time. We have tested our system with three real world datasets. It is observed that the accuracy of activity recognition using our system is more than 80%. Our proposed mechanism yields significant improvement (more than 30%) in comparison with its counterpart.

J. Sarkar · L.T. Vinh · Y.-K. Lee (✉) · S. Lee
Dept. of Computer Engineering, Kyung Hee University,
1 Seocheon-dong, Giheung-gu, Yongin-si, Gyeonggi-do,
446-701, Korea
e-mail: yklee@khu.ac.kr

J. Sarkar
e-mail: jehad@oslab.khu.ac.kr

L.T. Vinh
e-mail: vinhlt@oslab.khu.ac.kr

S. Lee
e-mail: sylee@oslab.khu.ac.kr

Keywords Activity recognition · Simple and ubiquitous Sensors · Web mining using Google · Naive Bayes-based classifier · Hierarchical classifier · Smoothing

1 Introduction

The purpose of an Activity Recognition (AR) system is to recognize everyday activities (e.g. walking, sleeping, and cooking) of one or more individuals from a series of observations [1–6]. It is now one of the current focuses of the researchers due to its strength in providing personalized support for many diverse applications such as medicine and healthcare.

An AR system requires training. The dataset to train an AR system is usually obtained through annotation. Context-aware experience sampling tool (ESM) [6–8] is the typical annotation technique. When using ESM, the participants either carry a PDA (with an ESM software installed on it) or wear a headset to report the current persuasion of an activity to a server (with an ESM software installed on it). Despite the simplicity of ESM, it has several disadvantages [9]: it may appear burdensome to the participants, it could be intimidating if the participants are not familiar with the system or device. Another major problem of such techniques is that it is not always possible to annotate all the activities. In any environment there could be hundreds of activities. Given such a large number of activities, it is impossible to label all such activities for the end users without expert knowledge.

The above problems greatly reduce the applicability and easiness of an AR system. Such a system is not general purpose for three reasons. First, an AR system trained in an environment would not be applicable to another environment. Second, it would not be able to recognize new activities.

Third, it requires extensive interaction with the experts, especially in the training phase.

To overcome the aforementioned limitations, we need an alternate source of the training data and an efficient method to extract such data. Advancement of the Internet and the WWW encourages millions of users to promote billions of web pages with varieties of contents [10]. A fraction of these pages describe in detail how to perform daily activities. They do not only state the activity but also depict where to perform this activity and what objects to use. An AR system would be broadly applicable and scalable by its very design if it can use such web pages to be trained by itself.

A few efforts have been made to train an AR system from the web (or *Train From Web* (*TFW*)) rather than from the environment (or *Train From Environment* (*TFE*)) [11, 12]. The system in [11] is packaged with thousands of activity models for different domains. It significantly limits the applicability and the accuracy of the system because the system fails to capture the idiosyncrasies of the environment to which it will be deployed. Although the system in [12] can focus on a particular environment to increase the applicability of their system, the mining method is complex and extremely time-consuming. It might take hours to mine a single activity.

Additionally, the accuracy of activity recognition of the above approaches is not up to the mark. One of the main reasons is that their activity model is only the *Object-usage Based Model* (*OBM*). The problem of using only OBMs is that in any environment there could be hundreds of objects, many of these objects could be used for different activities. For example, door could be used for kitchen activities or it could be used for bathroom activities. It would be hard for an AR system to discriminate such activities using only OBMs. Therefore, only the OBM is not enough for an AR system to produce highly accurate recognition results.

To overcome these limitations, in this paper, we first propose two activity models, *Location-and-Object-usage Based Model* (*LOBM*), and *Object-usage Base Model* (*OBM*). We then introduce a novel two-layer activity recognition framework. The first layer narrows down the scope for the recognition task, and the second layer does the actual recognition. More specifically, the first layer uses the LOBM to recognize a group of (location specific) activities from a set of activities, and the second layer uses the OBM to recognize the actual activity from that group.

We develop a novel and a straightforward algorithm to mine each of the models parameter from the web. It uses the advance operators of a search engine (we use Google for our experiment) to mine object-usage (how frequently an object is used to do an activity) and location-usage probabilities from the web. It not only reduces the mining time dramatically, but also makes the system *easy to use-and-configure* and *highly scalable*.

We performed *four experiments* with three real-world activity datasets to validate our system's performance. We demonstrate that our proposed system achieved higher recognition accuracy and significantly reduces the mining time in comparison with its counterpart.

The rest of the paper is organized as follows. In Sect. 2, we present the reviews of previous works related to AR. In Sect. 3, we describe the proposed GPARS. In Sect. 4, we present our experimental results to support our claims. In Sect. 5, we discuss the key issues related to our system. In Sect. 6, we conclude our paper with a direction of future work.

2 Related work

Many research groups have been investigating how to construct smart living environments that target medical care to the individual. Intel Research group in Seattle and the University of Washington have built a prototype system that can infer a person's activities of daily livings (ADLs) [13]. University of Rochester is building the Smart Medical Home, which is a five-room house equipped with infrared sensors, computers, bio-sensors, and video cameras for use by research teams to work with research subjects as they test concepts and prototype products [14]. Georgia Institute of Technology builds an Aware Home as a prototype for an intelligent space [15]. Massachusetts Institute of Technology (MIT) and TIAX are working on the PlaceLab initiative, which is a part of the House_n project [16]. The mission of House_n is to conduct research by designing and building real living environments—"living labs"—that are used to study technology and design strategies in context. Many projects are building body networks for the collection of vital signs, such as AMON. All these systems demonstrate the excitement and need for activity recognition systems [17].

To the best our knowledge, Tapia et al. [5] first employed simple and ubiquitous sensors for activity recognition. The authors provided the ESM in a PDA to the user to annotate their daily activities. Naïve Bayes classifier was used to recognize activities. They have showed an excellent promise, even though their mechanism suffers from low recognition accuracy.

Kasteren et al. [6] used the similar settings, except their annotation technique was quite innovative. They employed predefined set of voice commands to start and end points of an activity through a bluetooth enabled headset combined with speech recognition software. The problem of this annotation technique is that, it can not be guaranteed that the start and end points of an activity will always be marked properly by the participants. They did not even alert the participants to label the start and end points.

Perkowitz et al. [11] introduced the notion of mining the generic activity models from the web. They have shown that

it is possible to convert natural-language recipes into activity models. And these models can be used in conjunction with *RFID* tags to detect activity. Their model consists of a sequence of states and is based on a particle filter implementation of Bayesian reasoning. Their model extractor works as follows:

- Select a set of websites like, http://www.ehow.com/, http://www.epicurious.com/ that describes activities, and understand the HTML structure of such websites,
- search for a page that describes an activity and extract the activity direction from this page,
- set the title of the direction as the label of the activity,
- parse and extract the object phrases from the direction,
- remove the phrases that do not have noun sense,
- calculate the object-usage probability using the Google Conditional Probability (GCP),

$$GCP(o_i) = \frac{hitcount(object\ activity)}{hitcount(activity)}$$

  where $hitcount(x\ y)$ is the number of pages Google returns if we search with x and y,
- finally filter the tagged object (object with embedded *RFID* tags) from the phrases.

They use a Sequential Monte Carlo (SMC) approximation to infer activities probabilistically. They borrowed the inference engine from [4]. Despite their good performance in classifying hand-segmented object-use data, they suffered from low accuracy and limited applicability. In addition to this, they used specific web sites whose formats were known before mining the activity models [12].

Wyatt et al. developed an Unsupervised Activity Recognition System (UARS) using mined model from the web [12]. They developed two algorithms to mine activity information from web. The document genre classifier that would identify the pages describing an activity. The object identification algorithm that would extract objects from a page and calculate the object's weights within the page.

Their algorithm of mining for an activity works as follows:

- It first queries the Google with the activity name along with "how to" as the discriminating phrase. The Google would return the number of pages it has indexed in its server for the query.
- The algorithm then retrieves P pages as the top z pages within the total pages returned by Google. In their paper they did not define the optimal value of z. The efficiency of mining is clearly related to z, the larger the value of z is the more efficient the mining would be.
- It then determines $\tilde{P}$, a subset of P, as the activity pages using the genre classifier.
- For each page p in $\tilde{P}$, it extracts the objects mentioned in the page and calculates their weights, $\hat{w}$ using object identification algorithm.
- Finally, the algorithm calculates the objects usage probabilities for that activity using following formula:

$$p(object|activity) = \frac{1}{|\tilde{P}|}\sum_{p} w_{object,p}$$

They assemble a Hidden Markov Model (HMM), M, from the mined information. It has the traditional 3 parameters: (1) prior probabilities for each state, π, were uniformly distributed, (2) the transition probability matrix, T (each of the elements of this matrix is set to a constant probability), and (3) the observation probability matrix B, where $B_{ji} = p(object_i|activity_j)$.

Our work is closely related to the above two experiments. We developed an AR system using simple and ubiquitous sensors that would be broadly applicable, and easy-to-use. Our system also mines activity knowledge from the web to be trained by itself. Despite these similarities, we have several differences which are summarized below:

- Activity models: The GPARS integrated the *Location-usage Based Model* (*LBM*) with the *object-usage* based model. The location (e.g. kitchen) of persuading an activity is an important factor to determine the activity. We usually use different locations to do different activities, for example, kitchen is for cooking, bathroom is for bathing. Therefore, utilizing both the *location-usage* and the *object-usage* in the same model would increase the accuracy of activity classification.
- Mining the Model parameters: Our proposed method to mine the model parameters is *completely unsupervised* in comparison with [11] and [12]. It does not need any supervision to identify the pages describing activity performances. It uses the advance operators of a search engine to identify such pages. It also uses advance operators to determine the object-usage and the location-usage likelihoods. The benefits of using such technique are,
  - It not only dramatically reduces the mining time, but also makes the system easy to use and configure. To configure the system, it only requires the object names, their corresponding locations and the activities to monitor. Once the above parameters are provided, it will automatically mine the model parameters and make the system workable within a very short period of time.
  - It makes the system highly scalable. To add a new activity or a new object, the system only takes the new activity name or the object name and re-configures itself.
- Type of sensors: Our system uses a set of small and simple state-change (or switch) sensors (but not limited to). Using such sensors has several advantages over RFID tags such as,

– error free sensor readings,
– no need to wear RFID reader.

In summary, the GPARS uses a more sophisticated activity model to improve the accuracy of activity classification. It uses a straightforward algorithm to mine activity knowledge from the web and dramatically reduces the mining time.

3 Activity recognition system

In this section, we first give an overview of the components of our system, and then give detailed descriptions of the major components of the system.

3.1 Overview

We developed an activity recognition system that is scalable, easy-to-use and can recognize a large number of activities concerning different environments. Given a set of activities to monitor, object names and their corresponding locations, our proposed GPARS mines the activity knowledge from the web and stores this knowledge into a repository. From the mined data, it then estimates the *location-usage* and *object-usage* likelihoods. Once we have all the likelihoods, the system is thus ready to recognize activities in real-time. The overview of our proposed system is shown in Fig. 1.

The GPARS consists of five components:

1. The *environment* (e.g. home) to which a number of simple and ubiquitous sensors (or state-change sensors) are embedded into the appliances like door, cabinet, and desk.
2. The *Activity Classifier* (*AC*), which make's a classification decision based on the likelihoods learned from the web and the stream of data captured from the environment through sensors. The GPARS uses Naïve Bayes-based (NB) two-layer classifier to recognize an activity.
3. The *Activity Mining Engine* (*AME*), which mines activity knowledge from the web.
4. The *Parameter Estimator* estimates the model parameters using the activity knowledge mined by the AME.
5. The *Visualizing tool* provides a Graphical User Interface (GUI) to monitor the day-to-day activities. This is a web-based tool that shows in detail (e.g. Activity label, Object used, date/time) of an activity such that an authenticated person can access a secure website where he/she can scan a check-list.

3.2 Activity Classifier (AC)

In this subsection, we first justify the need for adopting two activity models and two-layer classifier. This is followed by describing the activity models and the classification methods in each layer.

Most of the ACs [6, 11, 12, 18] utilize only the OBM to classify activities. The downside of such an approach is that as the number of activities to monitor grows, the number of distinguishing objects between activities decreases. Such systems would produce more confusion between activities. Therefore, only the OBMs would not be enough for a highly accurate AR system.

Location of a person provides important context information for activity recognition and thus could be very helpful to make the classification decision [19, 20]. It is common to use a specific location to do an activity. For example, the kitchen is for cooking and the bathroom is for bathing. The group of activities are limited for a given location (an example is shown in Fig. 2).

In this paper, we propose a two-layer classifier, in which, the first layer classifies the group of activities (e.g., kitchen activities) using the LOBM, and the second layer classifies the individual activity (e.g., doing laundry) within the activity group using the OBM. We use object with location in the first layer to resolve any location-confusion. A subject (or user) may switch among locations while perform-

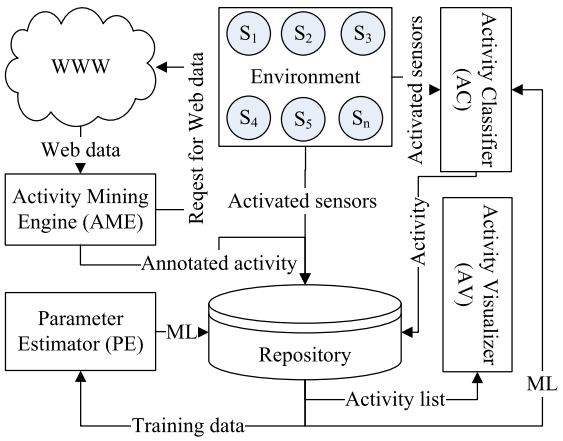


Fig. 1 Overview of the General-Purpose Activity Recognition System

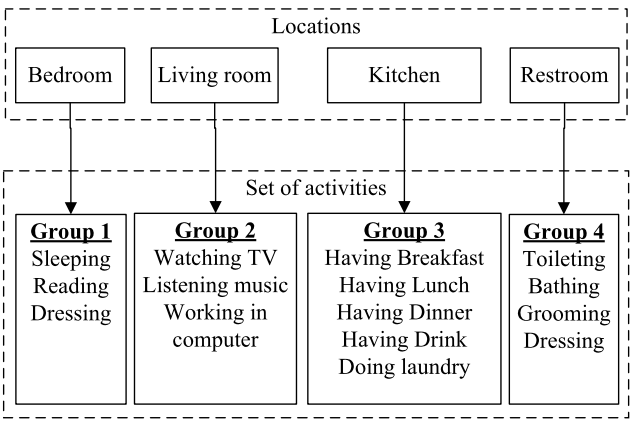


Fig. 2 An example: Location specific activities

ing an activity (e.g., moving back and forth between living-room and kitchen while cooking). Such a situation could be named as location-confusion. It restricts the system's ability to recognize the true activity group. Therefore, the GPARS employs the object information at the first layer along with the location information to resolve any resulting location-confusion. For example, the use of stove as the object information would increase the probability of the activity group to which the cooking belongs to.

It would be possible to design a one-layer classifier that uses object information along with the location information to classify an activity. However, if we design such a classifier to discriminate all activities, some activities with no specific location may not be well classified. For example, the activity, doing laundry, is usually performed with a washing machine (object) which could be located in locations like, kitchen, foyer and bathroom, may not be well classified using a one-layer classifier. The reason is, as we are mining activity information from the web, the probability of a location given an activity with no specific location would be relatively low compared to other activities specified for that location. For example, the probability of using kitchen for doing laundry will be low with respect to other kitchen activities (e.g., dinner). For such activities, both of the object and location would not be appropriate because using location with object in a model reduces the influence of object to that model. An activity has its own key object(s), for example, washing machine is the key object for doing laundry. If we reduce the influence of such object(s) for an activity, it will reduce the probability of classifying that activity.

In order to overcome such situation, we use a two-layer classifier, in which the first layer uses the object information along with the location information to classify a group of (location specific) activities, and the second layer uses the object information to recognize the actual activity from that group.

In our current version of the GPARS the activity groups are constructed manually based on the external input. The location at which an activity is performed is highly dependent on an individual and an environment to which the system is applied. Therefore, we choose to construct the groups manually using user's preference of an activity/location and the environment.

3.2.1 The goal of the classifier

Let $A = \{a_1, a_2, \ldots, a_m\}$ be the set of activities, $O = \{o_1, o_2, \ldots, o_t\}$ be the set of objects and $L = \{l_1, l_2, \ldots, l_q\}$ be the set of locations in the environment. Where, m, t, and q are the total number of activities, objects, and locations respectively. Let $\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\} \in O$ be the set of object-usage (activated object) sequence at any given time, and $l_{\theta_1}, l_{\theta_2}, \ldots, l_{\theta_n} \in L$ be the corresponding locations. Where,

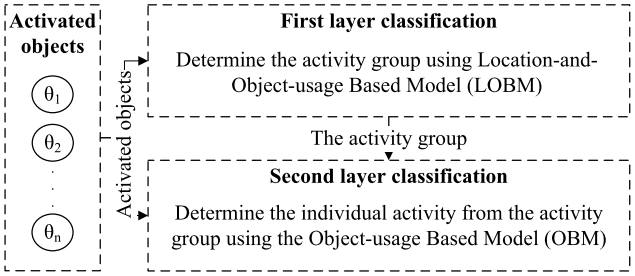


Fig. 3 An overview of activity classification

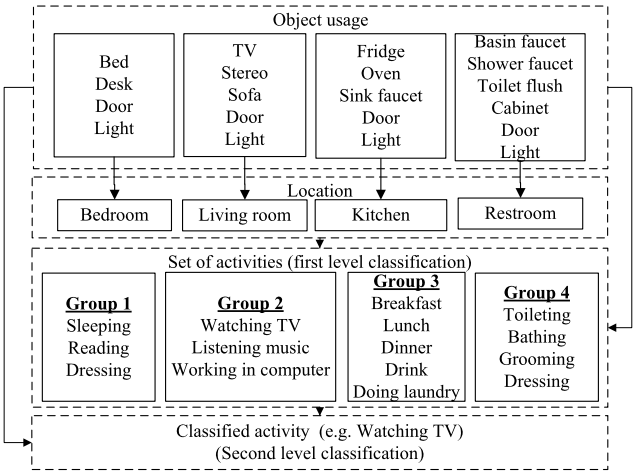


Fig. 4 An example: Two-layer activity classification

n is the total number of object-usage. The goal is to map the observation sequence (i.e. object-usage sequence, Θ) into predefined activity labels.

In the first layer classification, the GPARS classifies a group of activities, $A_j \in A$, using the LOBM. The individual activity, $a_i \in A_j$, is classified in the second layer using only on the OBM.

Figure 3 shows an overview of two-layer classification and an example is shown in Fig. 4.

3.2.2 Activity models

The activity models are Naïve Bayes-based probabilistic models.

Location-and-Object-usage Based Model (LOBM): We now formally define the LOBM.

Definition 1 The LOBM is a mixture model which involves a linear interpolation of the location and the object, using a Influential Coefficient (IC), $0 < \alpha < 1$, to control the influence of each.

$$P_{LOBM}(A_j|\Theta) \propto \prod_{k=1}^{|\Theta|} (\alpha P(l_{\theta_k}|A_j) + (1-\alpha) P(\theta_k|A_j)) \quad (1)$$

where, l_{θ_k} is the location of θ_k, $P(l_{\theta_k}|A_j)$ and $P(\theta_k|A_j)$ are the probabilities of using a location and an object respectively for a given activity group.

In Sect. 3.4, we describe how we estimated the probabilities for this model.

The LOBM produces a movement of probability mass from the object to location. A large value of α means more emphasis on location and a small value of α means more emphasis on object. The IC can be set to a value that maximizes the average performance of the classifier or to a value that can represent the importance of the locations in a dataset. In Sect. 3.4, we describe how we can estimate the IC.

Object-usage Based Model (OBM): The OBM is based on the model we proposed in [21].

In a Naïve Bayes-based classifier for activity recognition, the model parameters are usually approximated using the relative frequencies of the object-usage in a training set. This is called likelihood estimation of the probabilities. If a given activity and the object-usage value never occur (unseen object) together in the training set then the estimated likelihood will be zero. This is problematic since it will wipe out all information in the other object-usage probabilities when they are multiplied. To prevent such estimation problem, in [21], we proposed a smoothing technique which is based on the Jelinek-Mercer (JM) [22] (also referred as the linear interpolation language model) smoothing technique used in Information Retrieval.

Before defining the OBM, we need to define two terms.

Definition 2 An Activity Model $(AM) = \{v_1, v_2, \ldots, v_n\}$ is an observation vector of n number of objects for an activity. Where, v_i, being the observed frequency of ith object for an activity.

Definition 3 A Collective Model $(CM) = \{AM_1, AM_2, \ldots, AM_m\}$ is a collection of observation vectors of m number of activities. Where, AM_i, being the activity model for ith activity.

Let us consider an activity dataset shown in Table 1. Each cell of the table represents the number of times tth object, o_t, triggers for an activity. Each row of this table represents an AM, M_{a_i} for ith activity, a_i. Entire table represents the *Collective Model* (CM), M_c, for the activity collection, c.

Definition 4 The OBM is also a mixture model which involves a linear interpolation of the AM and with the CM, using a *Smoothing Coefficient* (SC), $0 < \lambda < 1$, to control the influence of each:

$$P_{OBM}(a_i|\Theta) \propto P(a_i) \prod_{k=1}^{|\Theta|} (\lambda P(\theta_k|M_{a_i}) + (1-\lambda) P(\theta_k|M_c)) \quad (2)$$

Table 1 An example dataset: Each cell represents how many times an object triggers for an activity

Activity	Objects						
	o_1	o_2	o_3	o_4	o_5	...	o_t
Going out	2	17	21	1	5	...	2
Toileting	1	40	195	0	0	...	16
Bathing	0	68	1	0	0	...	0
Sleeping	0	15	13	0	0	...	44
Breakfast	7	1	0	2	38	...	0
Dinner	5	0	4	4	23	...	0
Drink	0	0	0	17	30	...	1

where $P(a_i)$ is the *Prior Probability* (PP) of an activity, $P(\theta_k|M_{a_i})$ is the probability of using an object given an AM and $P(\theta_k|M_c)$ is the probability of using an object given a CM.

In Sect. 3.4, we describe how we estimated the probabilities for this model.

In (2), $0 < \lambda < 1$ is the smoothing parameter. Smaller values of λ means more smoothing. The smoothing method produces a movement of probability mass from seen objects to unseen objects. λ can be set to a value that maximizes the average performance of the classifier or to the average number of zero-frequencies in a dataset. In Sect. 3.4, we describe how we can determine λ.

3.2.3 *Classification of activity group*

In the first-layer classification, the GPARS uses the NB based classifier to classify a group of activities from a set activities based on the LOBM (shown in (1)). To classify the activity group of Θ, $P_{LOBM}(A_i|\Theta)$ is evaluated for each activity group $A_i \in A$. The classifier predicts that the activity group of the vector Θ is the activity group A_i if and only if

$$P_{LOBM}(A_i|\Theta) > P_{LOBM}(A_j|\Theta) \quad \text{for } 1 \le j \le q,\ j \ne i$$

where, q is the total number of activity groups.

3.2.4 *Classification of individual activity*

Similarly, in the second layer classification, the GPARS uses the NB based classifier to classify the individual activity from the activity group, A_j (from the first-layer classifier), using only on the OBM (shown in (2)). To classify the activity label of Θ, $P_{OBM}(a_i|\Theta)$ is evaluated for each activity $a_i \in A_j$. The classifier predicts that the activity label of vector Θ is the activity a_i if and only if

$$P(a_i) P_{OBM}(a_i|\Theta) > P(a_j) P_{OBM}(a_j|\Theta)$$
$$\text{for } 1 \le j \le g,\ j \ne i$$

where, g is the total number of activities in the activity group, A_j.

3.3 Activity Mining Engine (AME)

In this subsection we first define the goals of the AME and then we describe the algorithm utilizes by the AME to accomplish these goals.

Before defining the goals of the AME, we need to define activity knowledge.

Definition 5 Activity Knowledge is the information associated with the object-usage and location-usage which are specific to an activity.

3.3.1 Goals of the AME

We can see from (1) and (2) that during training we need to estimate the following likelihoods: $P(l_{\theta_k}|A_j)$, $P(\theta_k|A_j)$, $P(a_i)$, $P(\theta_k|M_{a_i})$ and $P(\theta_k|M_c)$. The goal of the AME is to provide sufficient activity knowledge such that these can be used by the PE to estimate the likelihoods.

To facilitate the understanding of how the AME mines activity knowledge we need to know the structure of an *Activity Page* (*AP*). An AP in general has a title that portrays what the activity is, and has a detailed description section that describes how to do such activity. In description section, an AP contains object and locations names. For example, as shown in Fig. 5, the web page with title "How to Prevent Bathing Injuries Among the Elderly" describes the sequence of steps require to prevent bathing injuries among the elderly. In detail description section, it mentioned the terms like, "Bathroom", "Tub faucet", and "Water heater".

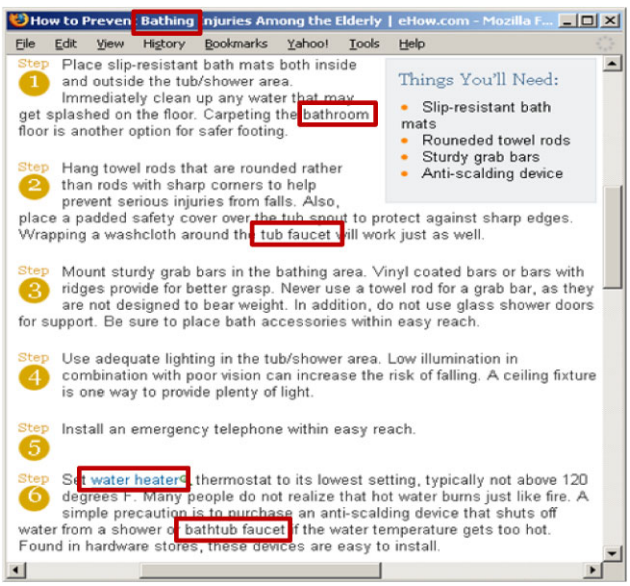


Fig. 5 A sample web page that describes an activity

The AME finds such APs and determine how many of these pages (i.e. Page Count (PC)) contain the given objects and how many of these pages contain the given locations, and generate a table like Table 2.

3.3.2 Mining of activity knowledge

The AME uses Google to search such pages and determines the PCs. Searching on Google is simple, choosing the appropriate search terms is the key to find the required information [23]. Google supports a bunch of advanced operators, which are query words and have special meaning for Google. We can modify our search in some way, or even instruct Google for a different search [24]. For instance, "intitle:" is a special operator, and the query [intitle:Bathing] does not do a normal search, instead finds all the web pages that have Bathing in their title. Table 3 shows the modifiers and operators we use to mine likelihoods.

Figure 6 shows how the AME mines activity data from the web using Google. The corresponding algorithm is shown in Algorithm 1. For each $a_i \in A$, the AME would first search the number of potential pages that describe a_i, using the query *intitle* :"a_i". Let the set of *activity pages indexed* (*API*) by Google be Ω for the given query. The cardinality of Ω is denoted by $|n| = |\Omega|$. Next step is to determine the number of pages indexed by Google for a location (*or location pages indexed* (*LPI*)) $l_j \in L$ within the activity pages. The AME uses the query, *intitle* :"a_i"+"l_j"

Table 2 An example of PCs for Activity = Bathing and number of APs for Bathing = 694,000

Location	PC for location	Object	PC for object
Kitchen	20,400	Cups	11,100
Bathroom	27,500	Door	14,500

Table 3 Google Query Modifiers and Operators

Name	Description
""	The quotes forces Google to search for the exact phrase. For example, the query ["Preparing dinner"] would find the pages containing the exact phrase "Preparing dinner"
intitle	If we include [intitle:] in our query, Google would return all the web pages containing the word in the title of the web pages. For instance, the query [intitle:"Preparing dinner"] would find all the web pages that have "Preparing dinner" in their title
+	By attaching a + immediately before a word, we can instruct Google to match that word precisely (without including synonyms). For instance, the query [intitle:"Preparing dinner" + "Butler pantry" would find all the pages containing the phrase "Preparing dinner" in their title and containing the exact phrase "Butler pantry" in their text

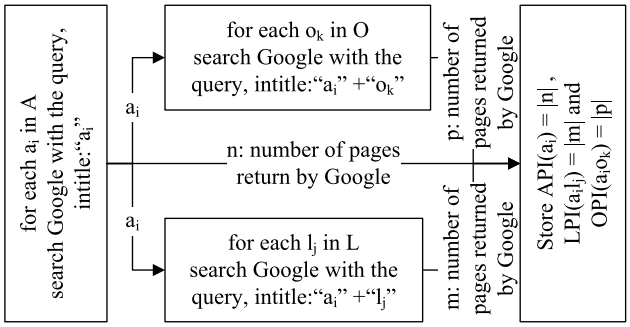


Fig. 6 Activity Mining

Algorithm 1: AME(A, O, L). The activity mining engine to mine activity knowledge from the web.

Data: List of activities A, List of objects O, List of locations L
Result: Activity Pages Indexed (API), Location Pages Indexed (LPI) and Object Pages Indexed (OPI)

```
for i ← 1 to length(A) do
    API_i = this ← SG("intitle :"a_i""); /* SG (Search Google) would return the number of pages indexed by Google for the given query */;
    for j ← 1 to length(L) do
        LPI_ij = this ← SG("intitle :"a_i" +"l_j""");
    end
    for k ← 1 to length(O) do
        OPI_ik = this ← SG("intitle :"a_i" +"o_k""");
    end
end
```

to return the number of pages containing the l_j in their text for a given activity pages. Let Google return $m \subseteq \Omega$ pages that contain an occurrence of l_j. Similarly, let $p \subseteq \Omega$ be the pages returned by Google for an object (*or object pages indexed* (*OPI*), if the AME searches with the query, *intitle* :"a_i"+"o_k". The AME finally saves $API(a_i) = |n|$, $LPI(l_j|a_i) = |m|$ and $OPI(o_k|a_i) = |p|$ into repository such that the PE can estimate the model parameters.

3.3.3 Number of queries required for mining

Given a set of activities A, objects O and their corresponding locations L, the total number of queries, r, required by the AME to mine activity knowledge from the web is:

$$r = m + m(q + t) \tag{3}$$

Where, m, t, and q are the total number of activities, objects, and locations respectively. As we can see in Algorithm 1, for m activities, the AME required m queries to mine *APIs*, for q locations and m activities, the AME required mq queries to mine *LPIs*, and for t objects and m activities, the AME required mt queries to mine *OPIs*.

For example, if we consider an environment where 20 objects are embedded with sensors in 5 different locations and there are 10 activities to monitor. To mine the model parameters, the AME would need 260 queries in total.

3.4 Parameter Estimator (PE)

The main task of the PE is to transform the activity knowledge mined by the AME into likelihoods such that the AC could use these to classify activities. Additionally, it estimates the coefficients associated with the models.

3.4.1 Estimation of likelihoods

The PE uses the following formulas to calculate the likelihoods:

$$P(o_j|A_i) = \frac{\sum_{a_k \in A_i} OPI(o_j|a_k)}{\sum_{a_k \in A_i, o_c \in O} OPI(o_c|a_k)}$$

$$P(l_j|A_i) = \frac{\sum_{a_k \in A_i} LPI(l_j|a_k)}{\sum_{a_k \in A_i, l_c \in L} LPI(l_c|a_k)}$$

$$P(o_j|M_{a_i}) = \frac{OPI(o_j|a_i)}{\sum_{o_c \in O} OPI(o_c|a_i)}$$

$$P(o_j|M_c) = \frac{\sum_{a_i \in A} OPI(o_j|M_{a_i})}{\sum_{a_k \in A, o_c \in O} OPI(o_c|M_{a_k})}$$

It is to be noted here that to estimate the conditional probability we use $\sum_{o_c \in O} OPI(o_c|a_i)$ and $\sum_{a_k \in A,\, o_c \in O} OPI(o_c|M_{a_k})$ instead of $API(a_i)$ and $\sum_{a_k \in A_i} API(a_k)$ respectively, as the denominators.

For example, let "Preparing Breakfast", "Preparing dinner" be two activities, and "Fridge", "Oven" be two objects. After mining the activity knowledge, we have,

$API(Preparing\ breakfast) = 53$

$API(Preparing\ dinner) = 119$

$OPI(Fridge|Preparing\ breakfast) = 3$

$OPI(Oven|Preparing\ breakfast) = 4$

$OPI(Fridge|Preparing\ dinner) = 3$ and

$OPI(Oven|Preparing\ dinner) = 5$

The PE estimates,

$$P(Oven|M_{Preparing\ breakfast}) = 4/(4+3) = 0.571$$

instead of $4/53 = 0.075$.

Similarly, it estimates,

$$P(Oven|M_c) = 4/((4+3)+(3+5)) = 0.26$$

instead of $4/(53+119) = 0.023$.

It means that the PE considers only the activity pages containing the specified objects or locations. It reduces the noise associated with the number of APs returned by Google (APIs). As we can see in Algorithm 1, we use the query, *intitle* :"a_i", to mine the number of pages describing an activity. It does not always guarantee that the Google would return only the pages that describe the activity, a_i. Therefore, to reduce such noise, the PE uses only the pages containing given objects or locations.

3.4.2 Estimation of the Prior Probability (PP)

In our current approach, the PP, $P(a_i)$, of an activity, a_i, is estimated based on the external input. The frequency of pursuing an activity per day(e.g. two showers/day) is the input, which is then converted into per minute to get the PP. It is to be noted that we were not able to find a suitable way to mine and estimate PPs from the web because it is *highly subject dependent*. If we consider, $P(a_i) = \frac{API(a_i)}{\sum_{a_j \in A} API(a_j)}$, which would not be an accurate measurement, because it would be biased to the number of pages indexed by Google. For example, Google would return $n = 694{,}000$ for "Bathing" and $n = 1220$ for "Toileting", using the above formula, the probability of "Toileting" would be $\ll$ then "Bathing". In real life, frequency of "Toileting" is much higher than "Bathing".

On the other hand, if we use some existing datasets to compute the PPs, it would limit the applicability and scalability of the system.

3.4.3 Estimation of coefficients

In (1), the coefficient $0 < \alpha < 1$ is used to control the influence of location. We need to estimate how much influence would be optimal (or nearly optimal) for a given dataset. In other words, we need to calculate the importance of the locations for all the activity groups. If we calculate the sum of average number of times the locations appeared in the activity documents, it would give us the importance. Therefore, we can calculate α as,

$$\alpha = \frac{\sum_{i=1}^{q} \frac{\sum_{a_k \in A_i, l_c \in L} LPI(l_c|a_k)}{\sum_{a_k \in A_i} API(a_k)}}{q}$$

Where, q is the number of activity groups.

In (2), we use the coefficient $0 < \lambda < 1$ to control the smoothing. The smoothing is clearly related to the number of zero-frequencies in a dataset. In other words, the smoothing is proportional to the number of zero-frequencies. The more zero-frequencies we have in a dataset, the more smoothing is required. Therefore, if we calculate the λ as the average of the average number of objects with zero-frequencies in each activity, it will give us the optimal (or nearly optimal) λ. That is,

$$\lambda = \frac{\sum_{a_i \in A} \frac{\sum_{o_c \in O} \delta(OPI(o_c|a_i))}{t}}{m}, \quad \delta = \begin{cases} 1 & \text{if } OPI(o_c|a_i) == 0 \\ 0 & \text{otherwise} \end{cases}$$

Where, m and t are the number of activities and objects respectively.

4 Evaluation

4.1 Objectives

Our objective is to validate the performance of the GPARS. We performed four experiments to test our system: First, we verify the efficiency of mining method by checking the likelihoods estimated by PE with the help of the AME. Second, we evaluate the classifier's performance in classifying activities of three datasets (described in next subsection). Third, we analyze the impact of the coefficients (α and λ) in activity classification, and evaluate the proposed methods of estimating these coefficients. And finally, we compare different classifiers in terms of their classification accuracy and mining time.

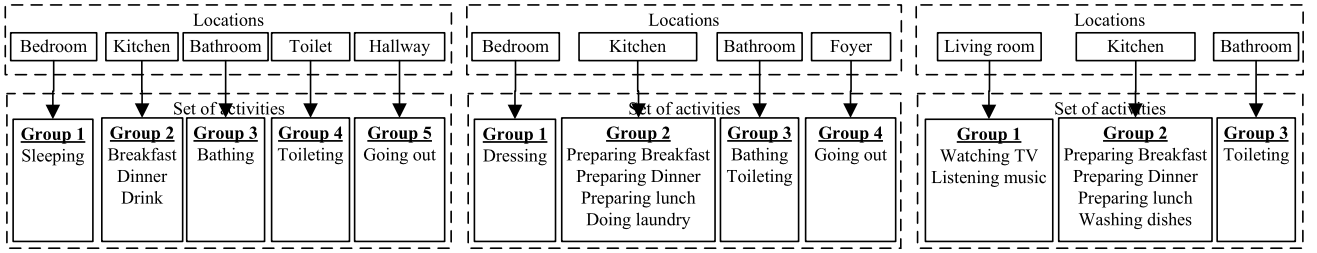
4.2 Experimental setup

4.2.1 Setup for mining

The AME uses the site, http://ajax.googleapis.com/ (developed by Google for applications to retrieve data from the Google server asynchronously), instead of the original site, http://www.google.com/, to mine the activity knowledge. For example, to mine the API for "Cooking", the AME would send a query as, http://ajax.googleapis.com/ajax/services/search/web?v=1.0&q=Cooking. In response, Google would return a page that would contain the formatted results like, the estimatedResultCount (i.e. API), the links of few (usually 4) result pages, the link for more results etc. Searching with Ajax would retrieve a bit old data with respect to the original site. It is to be noted here that Google would not allow automated search using their original site.

4.2.2 Setup for evaluating system's performance

We use the data gathered by Tapia et al. [5] at MIT PlaceLab (we call it PlaceLab dataset), and by Kasteren et al. [6] at Intelligent Systems Lab Amsterdam (ISLA) (we call it



a. ISLA activities and their grouping b. PlaceLab activities (Subject one) and their grouping c. PlaceLab activities (Subject two) and their grouping

Fig. 7 The activities and their grouping

ISLA dataset), to evaluate the performance of GPARS. Tapia et al. utilized 77 and 84 sensory data collection boards equipped with reed switch sensors, deployed these in two single-person's (i.e. Subject one, Subject two) apartments, and collected data for two weeks. The sensors were installed in everyday objects such as drawers, refrigerators, containers to record activation/deactivation events (opening/closing events) as the subject carried out everyday activities. Their data were collected by a base station and labeled using an ESM. Kasteren et al. deployed 14 digital sensors in a house of a 26-year-old man, attached these sensors to doors, cupboards, a refrigerator, and a toilet flush, and they collected data for 28 days. Their activities were chosen from Katz ADL index [25]. Figure 7 shows the ISLA and the PlaceLab activities and their grouping we use to validate the system's performance.

As the activity instances are imbalanced between classes, two types of measurements are used to evaluate the performance of our system, similar to [6]. The time slice accuracy is measured by,

$$\frac{\sum_{i=1}^{N} detected_i == true}{N}$$

and Class Accuracy is measured by,

$$\frac{1}{C}\sum_{c=1}^{C}\left\{\frac{\sum_{i=1}^{N_c} detected_i == true}{N_c}\right\}$$

where, N is the total number of activity instances, C is the number of classes and N_c is the total number of instances for class c.

Even though the time-slice accuracy is a typical way of evaluating classifier's accuracy [6], it is not always true for AR classifiers because the dataset would contain dominant classes that appear a lot frequently than others. For example, let us consider the ISLA dataset, in which total number of instances of "Toileting" is 114 and that of "Dinner" is 10. If a classifier correctly classify 110 instances of "Toileting" (accuracy = 96.491%) and 4 instances of "Dinner" (accuracy = 40%) then the time-slice accuracy would be $\approx 92\%$, whereas the class accuracy would be $\approx 68\%$. Therefore, the class accuracy should be the primary way to evaluate the activity classifiers performance. However, in this paper we report both the time-slice and the class accuracy.

4.3 Experiment 1: effectiveness of activity mining engine

The purpose of this experiment is to evaluate the effectiveness of the AME in mining activity knowledge from the web. We say that the mining would be effective, if the likelihoods estimated from the mined data are realistic. Table 4 shows the calculated object-usage likelihoods for the ISLA activities. The likelihoods are estimated by the PE in conjunction with the AME. Most of these likelihoods are highly expected as shown in these tables. For example, in Table 4, the likelihoods of using a "Microwave" to "Prepare breakfast" or "Prepare dinner" are considerably high with respect to other activities and the likelihood of using a "Toilet flush" for "Toileting" is reasonably higher than other activities. Therefore, we can say that the mining method is effective.

4.4 Experiment 2: activity recognition accuracy

The purpose of this experiment is to see how accurate our proposed method is to classify the activities. Additionally, we compare the performance of the two-layer classifier with the one-layer classifier.

In two-layer classifier, the first layer classifies a *group of activities*, $A_j \in A$, based on the LOBM (shown in (1)). The second layer classifies the individual activity, $a_i \in A_j$, based only on the OBM (shown in (2)).

But in one-layer classifier, the classifier classifies the individual activity, $a_i \in A$, based on a slightly different (with respect to two-layer classifier) version of the LOBM (we named it, $LOBM_{ol}$),

$$\begin{aligned} &P_{LOBM_{ol}}(a_i|\Theta) \\ &\propto P(a_i)\prod_{k=1}^{|\Theta|}(\alpha P(l_{\theta_k}|a_i) \\ &\quad + (1-\alpha)(\lambda P(\theta_k|M_{a_i}) + (1-\lambda)P(\theta_k|M_c))) \end{aligned} \tag{4}$$

Table 4 The likelihoods of object-usage generated by the PE in conjunction with the AME for ISLA activities

Objects	Activities						
	Going out	Toileting	Bathing	Sleeping	Breakfast	Dinner	Drink
Microwave	0.0088164	0.0025294	0.0132640	0.0153710	0.0909570	0.0578950	0.0323460
Door	0.2135000	0.1782600	0.1725600	0.2063300	0.1591700	0.1030200	0.1305400
Cups	0.0214200	0.0707970	0.1789800	0.0295520	0.0549320	0.0984660	0.1345800
Fridge	0.0137750	0.0012652	0.0082012	0.0206340	0.0861020	0.0377920	0.0606490
Plate	0.0477970	0.0303420	0.0434270	0.0403930	0.0480340	0.2671300	0.1207200
Dishwasher	0.0037889	0.0037937	0.0093421	0.0114370	0.0141810	0.0341450	0.0135750
Flush	0.0117090	0.0657410	0.0199670	0.0118910	0.0043699	0.0036114	0.0206210
Freezer	0.0085409	0.0012652	0.0047072	0.0100550	0.0188310	0.0388860	0.0391620
Pans	0.0098495	0.1074600	0.0146190	0.0131850	0.0098119	0.0264860	0.0174440
Washing machine	0.0038577	0.0037937	0.0111960	0.0154060	0.0286160	0.0022119	0.0026637
Groceries	0.0164610	0.0000010	0.0060620	0.0067505	0.0074869	0.0212900	0.0361010

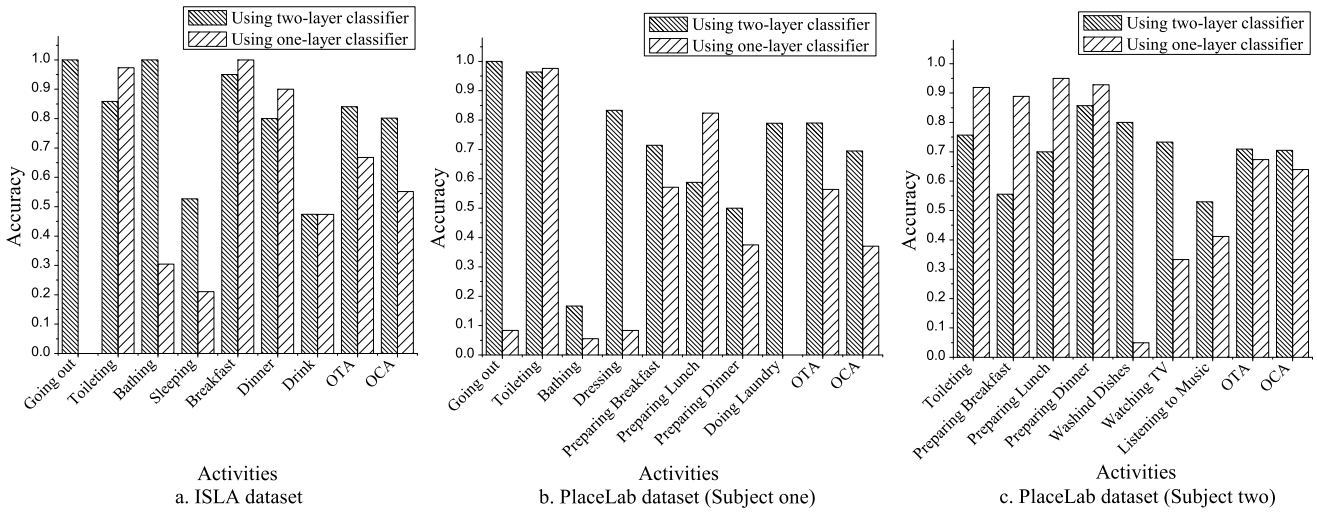


Fig. 8 The accuracies per class for three datasets, two-layer classifier (*left*), one-layer classifier (*right*). The rightmost two pairs of bars compare the overall timeslice accuracy (OTA) and the overall class accuracy (OCA)

where $P(a_i)$ is the *Prior Probability* (*PP*) of an activity, a_i, and $P(l_{\theta_k}|a_i)$, $P(\theta_k|a_i)$ are the probabilities of using a location and an object given an activity respectively. The $P(\theta_k|M_{a_i})$ is the probability of using an object given an AM and $P(\theta_k|M_c)$ is the probability of using an object given an CM. The IC, α, is calculated as,

$$\alpha = \frac{\sum_{i=1}^{m} \frac{\sum_{l_c \in L} LPI(l_c|a_i)}{API(a_i)}}{m}$$

Where, m is the total number of activities. The SC, λ, is calculated as it is done for two layer classifier.

Similar to [5], we measured the probability that an activity is correctly classified for the duration of labeled activity.

Figures 8(a), (b) and (c) summarize the accuracies per class for three datasets. Each pair of bars shows the overall accuracy for both the two-layer (left bar) and the one-layer (right bar) classifier. The rightmost two pairs of bars in Figs. 8(a), (b) and (c), compare the overall timeslice accuracy (OTA) and the overall class accuracy (OCA). Table 5 shows the summary of the accuracies (of both classifiers) for three datasets. The estimated coefficients (we discussed more about the effect of α and λ in the next subsection) are shown in Table 6.

The GPARS achieves an overall class accuracy of 80.12% (timeslice accuracy is 84.03%) for the ISLA dataset using the two-layer classifier. However, using the one-layer classifier, the GPARS achieves an overall class accuracy of 55.17% (timeslice accuracy is 66.81%). The GPARS achieves an overall class accuracy of 69.97% (timeslice accuracy is 78.97%) for the PlaceLab dataset (Subject one) using the two-layer classifier. On the otherhand, using one-layer classifier, the GPARS performs worse in classifying

the PlaceLab dataset (Subject one). It achieves an overall class accuracy of 37.10% (timeslice accuracy is 56.41%). The overall classification accuracy of the GPARS for the PlaceLab dataset (Subject two) using the two-layer classifier is not that significant. It achieves an overall class accuracy of 70.46% (timeslice accuracy is 70.92%). Whereas it achieves an overall class accuracy of 64.02% (timeslice accuracy is 67.38%) using the one-layer classifier.

Table 5 The summary of the accuracies for all the datasets

Datasets	Two-layer classifier		One-layer classifier	
	Timeslice	Class	Timeslice	Class
ISLA	84.034	80.120	66.807	55.172
PlaceLab (Subject one)	78.974	69.448	56.410	37.101
PlaceLab (Subject two)	70.922	70.460	67.376	64.021

Table 6 The estimated α and λ for three datasets

Datasets	α		λ
	Two-layer	One-Layer	
ISLA	0.3343	0.5663	0.0051
PlaceLab (Subject one)	0.1529	0.5116	0.1475
PlaceLab (Subject two)	0.3642	0.4775	0.1224

In Tables 7, 8, 9, we show the corresponding $n \times n$ confusion matrices. The ith row, and the jth column represents the number of times an activity, a_i, is recognized as activity a_j. The numbers with parenthesis and without parenthesis are the counts for the one-layer classifier and the two-layer classifier respectively.

For the ISLA dataset, the two-layer classifier performs better in classifying "Going out", "Bathing", and "Sleeping" with respect to the one-layer classifier (shown in Table 7). On the other hand, the one-layer classifier gives better accuracy in classifying "Toileting", "Breakfast" and "Dinner". Both of these classifiers exhibit same performance in classifying "Drink".

For the PlaceLab dataset (Subject one), the two-layer classifier performs better in classifying all the activities except "Bathing" and "Preparing lunch" in comparison with the one-layer classifier (shown in Table 8). For the PlaceLab dataset (Subject two), the one-layer classifier performs better in classifying "Toileting", "Preparing Breakfast", "Preparing lunch" and "Preparing dinner" (shown in Table 9).

Although, the one-layer classifier exhibits slightly better performance in classifying some of the activities (e.g. Toileting), it performs worse in some cases. For example, it performs poorly in classifying "Going out", for the ISLA and the PlaceLab dataset (subject one). The overall performance

Table 7 The Confusion matrix for the ISLA dataset: two-layer classifier (one-layer classifier)

	Going out	Toileting	Bathing	Sleeping	Breakfast	Dinner	Drink
Going out	34(0)	0(32)	0(1)	0(0)	0(0)	0(0)	0(1)
Toileting	0(0)	97(110)	16(3)	0(0)	0(0)	0(0)	0(0)
Bathing	0(0)	0(16)	23(7)	0(0)	0(0)	0(0)	0(0)
Sleeping	0(0)	4(11)	5(1)	10(4)	0(2)	0(1)	0(0)
Breakfast	1(0)	0(0)	0(0)	0(0)	19(20)	0(0)	0(0)
Dinner	1(0)	0(0)	0(0)	0(0)	0(0)	8(9)	1(1)
Drink	0(1)	0(0)	1(0)	0(0)	3(3)	6(6)	9(9)

Table 8 The Confusion matrix for the PlaceLab dataset (Subject one): two-layer classifier (one-layer classifier)

	Going out	Toileting	Bathing	Dressing	Preparing breakfast	Preparing lunch	Preparing dinner	Doing laundry
Going out	12(1)	0(11)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)
Toileting	0(0)	80(81)	0(0)	0(1)	0(0)	0(0)	1(1)	2(0)
Bathing	0(0)	15(17)	3(1)	0(0)	0(0)	0(0)	0(0)	0(0)
Dressing	0(0)	3(22)	0(0)	20(2)	0(0)	0(0)	0(0)	1(0)
Preparing breakfast	1(1)	1(1)	0(0)	1(3)	10(8)	1(1)	0(0)	0(0)
Preparing lunch	0(1)	0(0)	0(0)	1(1)	0(0)	10(14)	1(1)	5(0)
Preparing dinner	1(2)	0(0)	0(0)	1(2)	0(0)	0(0)	4(3)	2(1)
Doing laundry	1(2)	2(3)	0(0)	1(1)	0(3)	0(8)	0(2)	15(0)

Table 9 The Confusion matrix for the PlaceLab dataset (Subject two): two-layer classifier (one-layer classifier)

	Toileting	Preparing breakfast	Preparing lunch	Preparing dinner	Washing dishes	Watching TV	Listening music
Toileting	28(34)	2(0)	1(1)	1(0)	2(1)	2(0)	1(1)
Preparing breakfast	0(0)	10(16)	1(1)	0(0)	7(1)	0(0)	0(0)
Preparing lunch	0(0)	0(0)	14(19)	0(0)	6(1)	0(0)	0(0)
Preparing dinner	0(0)	0(0)	0(0)	12(13)	2(1)	0(0)	0(0)
Washing dishes	0(1)	0(4)	1(10)	2(4)	16(1)	1(0)	0(0)
Watching TV	1(5)	1(0)	0(0)	1(1)	1(2)	11(5)	0(2)
Listening music	0(2)	0(0)	3(6)	0(1)	5(1)	0(0)	9(7)

of the one-layer classifier is worse in comparison with the two-layer classifier. The two-layer classifier yields significant improvement (in some cases it is more than 30%) in comparison with the one-layer classifier.

4.4.1 Discussion

The two-layer classifier makes more confusion between the activities which were performed in a same location using similar objects. This is expected because, the objects within that location are equally likely to be used for these activities. For example, as we can see in Table 8, the classifier made more confusion between "Toileting" and "Bathing" because, most of the instances of these activities were performed in "Toilet", the number of distinguishing objects are low and the prior probability of "Toileting" is high.

It is to be noted here that the prior probabilities are estimated based on the external input (frequency of persuasion). The frequency of "Toileting" is set to 3/day, 6/day and 4/day for the ISLA dataset, PlaceLab dataset for Subject one and two respectively. For the rest of the activities, the frequency of persuasion is set to 1/day.

The one-layer classifier tends to classify better for an activity with high prior probability. For example, the activity, "Toileting" is better classified, because the prior probability of "Toileting" is high. This is expected because, the LOBM can only provide the high-level view of an activity. The differences between the activities would be marginal between similar activities and the prior probability would be the decision parameter.

The one-layer classifier performs poorly in classifying the activities with no obvious locations. For example, it performs worse in classifying the "Going out" and the "Doing laundry" (referring Figs. 8(a) and (b)). The reason is that, generally, either of these two activities does not have an obvious location. As we trained our system from the web, the location usage probability of such activities are low with respect to other activities. On the other hand, the two-layer classifier performs well in classifying these activities. The "doing laundry" is well classified because the two-layer classifier uses the OBM in the second layer to discriminate the actual activity from a group of activities. The "Going out" is properly classified because of the grouping of activities (referring Figs. 7(a) and (b)).

4.4.2 Observations

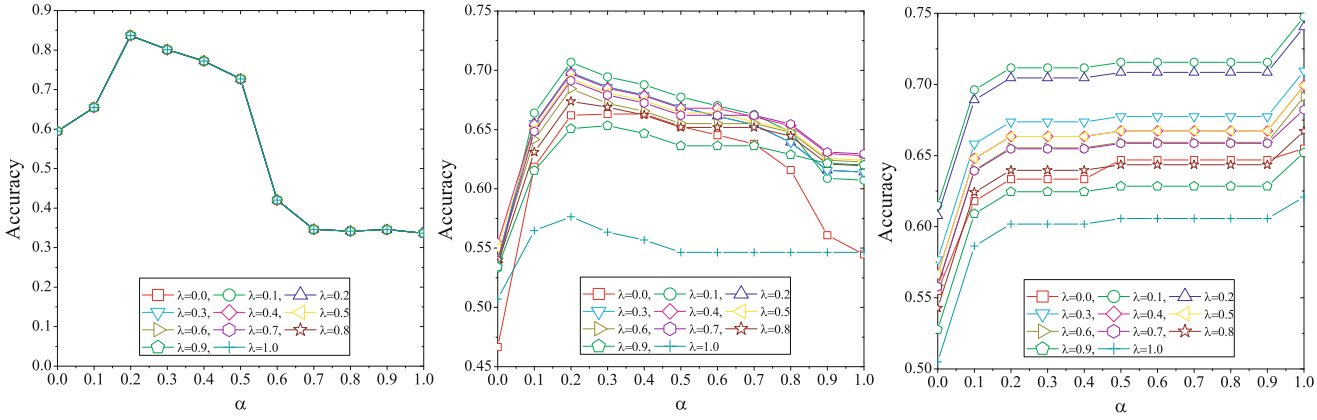
It is observed that the two-layer classifier performance is proportional to the number of objects utilized in an environment. The more objects we utilize, the more likely it is to create confusion. For example, the classifier performs best in classifying the ISLA activities, with a classification accuracy of 80.12%. This is because, in their experiment they only utilized 14 objects, and the number of objects per location was limited. Nevertheless, for the other two datasets, PlaceLab Subject one and PlaceLab Subject two, the accuracy of classifications are 69.448% and 70.460% respectively. They have utilized 84 objects for Subject one and 77 objects for Subject two.

It is observed that a group of similar activities is more distinguishable (by the two-layer classifier) if performed in different locations. For example, "Toileting" and "Bathing" (or "Showering") in the ISLA dataset are more distinguishable (as shown in Table 7) than in the PlaceLab dataset (as shown in Table 8) because "Toilet" and "Bathroom" are two different locations in the ISLA setup.

4.5 Experiment 3: varying the model coefficients

The goal of this experiment is two-folds: Analyze the impact of the coefficients (α and λ) in accuracy of activity classification and to see whether the proposed methods of estimating the coefficients can determine the nearly optimal values or not. We ran the test with α and λ values: 0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 and 1.0. The results are shown in Figs. 9(a), (b) and (c) for the ISLA, PlaceLab (Subject One) and PlaceLab (Subject Two) datasets respectively.

As expected, all the three datasets are sensitive to the α values. For example, as we can see in Figs. 9(a)–(c), for $\alpha = 0.0$, the accuracies of activity classification are relatively low with respect to $\alpha = 0.1$. It indicates that incorporating the LOBM significantly improve the activity clas-



a. ISLA dataset b. PlaceLab dataset (Subject one) c. PlaceLab dataset (Subject two)

Fig. 9 Activity recognition accuracy with different α and λ settings

sification accuracy. But only the LBM is not always sufficient for first-layer classification. For example, as shown in Figs. 9(a)–(b), the accuracies are relatively low when α is set to 1.0.

The activity recognition accuracy is sensitive to λ values for all the datasets except the ISLA dataset. This is because they used only 14 sensors and the AME was able to mine the activity knowledge efficiently (e.g. number of unseen objects are almost zero).

The estimated coefficient, α, for the ISLA dataset and for the PlaceLab dataset (Subject One) as shown in Table 6, are near their optimal values. For example, as we can see in Fig. 9(a) that the maximum performance for the ISLA dataset is observed for $\alpha = 0.2$ and the estimated α for this dataset is 0.3343, which is near optimal. However, for the MIT (Subject Two) dataset, the estimated α is not near to the optimal value. As we can see in Fig. 9(c), the best performance is observed for $\alpha = 1$. There are two reasons behind this. The number of activity groups (as shown in Fig. 7(c)) are low and the switching between locations (by the subject) while doing an activity was relatively less.

The estimated coefficient, λ, as shown in Table 6, are near their optimal values. For example, in Fig. 9(c), we can see that the maximum performance for the PlaceLab dataset (Subject two) is observed when the $\lambda = 0.1$ and the estimated λ for this dataset is 0.1224, which is nearly optimal.

4.6 Experiment 4: comparison with the other methods

The goal of this experiment is two-fold:

1. Compare the performance of the GPARS in classifying the activities with the both TFW and TFE based methods.
2. Compare the time complexity of our proposed mining technique with the mining technique proposed in [12].

4.6.1 Performance comparison of the classifiers

Activity recognition based on sensors is a challenging task due to the inherent noisy nature of the input. Thus, the temporal probabilistic models are the state-of-the-art to solve this task. A set of probabilistic models have been proposed for activity recognition, for example, the Naïve Bayesian (NB) in [5, 21, 26], the Hidden Markov Model (HMM) in [6, 12, 27, 28] and the Conditional Random Field (CRF) in [2, 3, 6, 29, 30].

For many years NB and HMM have been used in several field of artificial intelligence including activity recognition. In recent years, the CRF becomes a very popular choice. The CRF was originally proposed for natural language processing. It has recently been used in a wide varieties of applications like, image processing, motion tracking and activity recognition.

We compare our classifier's performance with a TFW based classifier, proposed by Wyatt et al. in [12], which uses HMM as the classifier. We also compare the performance of the classifier with the TFE based classifiers proposed by Tapia et al. [5] and by Kasteren et al. [6] for activity recognition. In [5], Tapia et al. utilized a NB based classifier and in [6], Kasteren et al. utilized a HMM and a CRF for activity recognition.

We separate the training and testing data using "leave one day out" strategy to train and test the TFE based systems. In this strategy, one day is used for testing and remaining days are used for training. The comparison results are shown in Table 10.

It is observed that the GPARS achieved superior performance for all the datasets other methods except the Naïve Bayes (NB) based classifier for ISLA dataset. This is expected because Kasteren et al. only used 14 sensors, focused on 7 activities and the number of unseen objects are almost zero to improve by smoothing.

4.6.2 Mining time comparison

To the best of our knowledge, only two systems [11, 12], have been proposed to train an AR system from the web. We compare the proposed mining technique with the technique used in [12] (we call their system UARS). It is not feasible to perform a direct comparison to the technique used in [11]. Their models were mined from a single web site, they had to manually map their models to the activities found in their data as well as map the tagged object (object with embedded sensors) names with the mined object names [12]. Our method has a strict one-to-one match between activities. We do not need to map any object names since we only mine the information associated with the tagged object.

The Fig. 10, shows the time required for mining by both the GPARS and the UARS for three datasets. As expected, the GPARS outperforms for all the datasets.

We analyze the total time, t, the GPARS and the UARS would take to mine an activity knowledge. For this purpose, let us consider an environment to which there are 20 objects in 5 different locations, and we are trying to monitor 1 activity (e.g. "Going out").

The GPARS would take $t = 1 + 1(5 + 20) = 26$ (using (3)) seconds to mine activity information regarding "Going out", assuming that Google would take 1 second to provide the search result for each query.

We calculate total time, t, UARS would take to the mine activity knowledge, using following steps (in Sect. 2, we describe the mining method):

1. The UARS would first search Google with the query "How to" "Going out". Google would return $\hat{P}$ pages. Let us assume that $|\hat{P}| = 10{,}700{,}000$ and we set $t = 1$ (assuming that Google would take 1 second for each query).
2. It then retrieves $P \subset \hat{P}$ pages. Let $|P| = 10{,}700$ (0.1% of $|\hat{P}|$).
3. It then determines $\tilde{P} \subset P$, as the activity pages. Let $\tilde{P} = 107$ (1% of $|P|$). To determine $\tilde{P}$, the UARS needs to load and check all the pages in P and it would take 2 seconds in average for each page. Therefore, we set $t = 1 + 10700 * 2 = 21401$.
4. For each page $p \in \tilde{P}$, it extracts the objects mentioned in the page and calculate their weights. Let us assume that UARS would take 2 seconds (on average) per page to extract and calculate objects weights. So, we set $t = 21615$.

Therefore, the UARS would take 21615 seconds (or around 6 hours) to mine a single activity knowledge, whereas the GPARS would only take 26 seconds.

Table 10 Comparison with other methods (class accuracies are used to compare)

Dataset	Accuracies of the classifiers (%)				
	TFW		TFE		
	GPARS	UARS	NB	HMM	CRF
ISLA	80.12	63.26	82.80	71.75	67.57
PlaceLab (Subject one)	69.45	50.86	31.27	66.18	54.40
PlaceLab (Subject two)	70.46	54.56	32.47	57.35	64.44

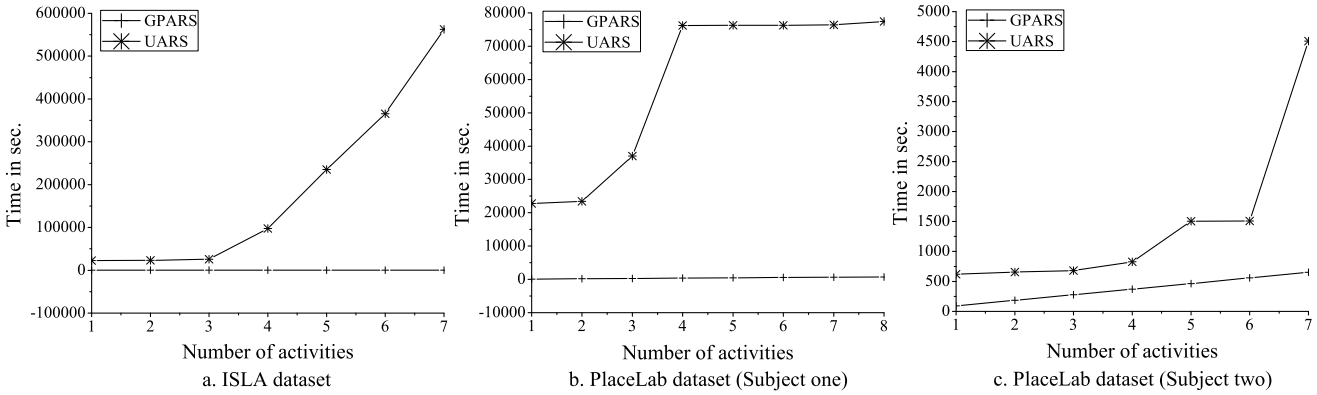


Fig. 10 Mining time comparison between the GPARS and the UARS, (**a**) for the ISLA dataset (Mining is performed in the sequence as shown in the first column of Table 7, i.e. 1. Going out, 2. Going out + Toileting, 3. Going out + Toileting + Bathing, and so on), (**b**) for the PlaceLab dataset Subject one (Mining is performed in the sequence as shown in the first column of Table 8, i.e. 1. Going out, 2. Going out + Toileting, 3. Going out + Toileting + Bathing, and so on), (**c**). for the PlaceLab dataset for Subject two (Mining is performed in the sequence as shown in the first column of Table 9, i.e. 1. Toileting, 2. Toileting + Preparing breakfast, 3. Toileting + Preparing breakfast + Preparing lunch, and so on)

5 Discussion

5.1 Providing activity name, object name and location name

One of the most important components of the GPARS is to choose the names of the activities, objects and locations because the efficiency of the AME depends on choosing appropriate names. For the current version of the GPARS we have chosen the names manually. For example, for the PlaceLab datasets we used the exact activity names as they used in their paper. But for the ISLA dataset we changed the activity "Leaving" to "Going out" to make it more sensible and consistent to other datasets. For object name we tried to keep the name in one word however for objects like "Washing machine", "Shower faucet" we kept the entire name to make it meaningful.

5.2 Choosing the right object vs accuracy

Choosing the right object to embed a sensor is an important factor for accuracy of activity classification. For example, embedding a sensor in the "shower faucet" would increase the classification accuracy of "Bathing", because it is highly likely that "shower faucet" would be used while "Bathing". In Kasteren et al.'s setup, they did not place any sensor to "shower faucet". Replacing the "bathroom door" with the "shower faucet" would improve the classification accuracy of "Bathing". Additionally, if we use both of these sensors, the accuracy could be improved further.

6 Conclusion and future work

In this paper we present an Activity Recognition System which is broadly applicable and easy-to-use. We consider a set of simple and ubiquitous sensors is deployed in an environment, embedded with the daily life objects.

We first address the problems of acquiring activity data from the environment to train an AR system and discuss how we can use WWW as the alternate source of such data. We then address the problems associated with the current state of the art techniques to train an AR system from WWW. Finally, we propose a novel way to mine human activity knowledge from WWW. We have shown that it is possible to train an AC using such knowledge. One of the major advantages of such technique is that it eliminates the amount of human effort in labeling the activities while still achieving high recognition accuracy. Another advantage of this technique is that it is possible to label thousands of activities within a very short period of time.

We also present a two-layer Naïve Bayes-based activity classifier with a smoothing technique to improve the accuracy of activity classification. We perform four experiments to ensure the validity of our proposed technique. We demonstrate that our method can classify the activities with high accuracy. We compared our proposed technique with both "Train From Web" and "Train from Environment" based methods. It is observed that our proposed mechanism yields *significant improvement* on activity classification in comparison to its existing counterparts.

The performance of our system is sensitive to the names of activities, objects and locations. In our current version we do not have any technique that would automatically check for the best names. We would incorporate such a technique in our next version of the GPARS.

Acknowledgements This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Advancement)″ (IITA-2010-(C1090-1002-0003)), also was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2009-1515).

References

1. Hodges MR, Pollack ME (2007) An 'object-use fingerprint': The use of electronic sensors for human identification. In: Proc UbiComp, pp 289–303
2. Hu DH, Pan SJ, Zheng VW, Liu NN, Yang Q (2008) Real world activity recognition with multiple goals. In: Proc UbiComp. ACM, New York, pp 30–39. doi:10.1145/1409635.1409640
3. Hu DH, Yang Q (2008) Cigar: Concurrent and interleaving goal and activity recognition. In: Fox D, Gomes CP (eds) Proc AAAI. AAAI Press, Menlo Park, pp 1363–1368
4. Patterson DJ, Liao L, Fox D, Kautz HA (2003) Inferring high-level behavior from low-level sensors. In: Dey AK et al (ed) UbiComp 2003: Ubiquitous computing, 5th international conference, Seattle, WA, USA, October 12–15, 2003, Proceedings. Lecture notes in computer science, vol 2864. Springer, Berlin, pp 73–89
5. Tapia EM, Intille SS, Larson K (2004) Activity recognition in the home using simple and ubiquitous sensors. In: Ferscha A, Mattern F (eds) Pervasive. Lecture notes in computer science, vol 3001. Springer, Berlin, pp 158–175
6. van Kasteren T, Noulas A, Englebienne G, Kröse B (2008) Accurate activity recognition in a home setting. In: Proc UbiComp. ACM, New York, pp 1–9. doi:10.1145/1409635.1409637
7. Intille SS, Rondoni J, Kukla C, Ancona I, Bao L (2003) A context-aware experience sampling tool. In: CHI '03: CHI '03 extended abstracts on Human factors in computing systems. ACM, New York, pp 972–973. doi:10.1145/765891.766101
8. Intille SS, Tapia EM, Rondoni J, Beaudin J, Kukla C, Agarwal S, Bao L, Larson K (2003) Tools for studying behavior and technology in natural settings. In: Dey AK et al (ed) UbiComp 2003: ubiquitous computing, 5th international conference, Seattle, WA, USA, October 12–15, 2003, Proceedings. Lecture notes in computer science, vol 2864. Springer, Berlin, pp 157–174
9. Consolvo S, Walker M (2003) Using the experience sampling method to evaluate UbiComp applications. IEEE Pervasive Comput 2(2):24–31. doi:10.1109/MPRV.2003.1203750
10. Cilibrasi R, Vitányi PMB (2007) The Google similarity distance. IEEE Trans Knowl Data Eng 19(3):370–383
11. Perkowitz M, Philipose M, Fishkin K, Patterson DJ (2004) Mining models of human activities from the web. In: WWW '04: Proceedings of the 13th international conference on World Wide Web. ACM, New York, pp 573–582. doi:10.1145/988672.988750

12. Wyatt D, Philipose M, Choudhury T (2005) Unsupervised activity recognition using automatically mined common sense. In: Veloso MM, Kambhampati S (eds) Proc AAAI. AAAI Press/The MIT Press, Menlo Park, pp 21–27. http://www.informatik.uni-trier.de/~ley/db/conf/aaai/aaai2005.html#WyattPC05
13. Exploratory research projects (accessed: 2009, July 22). Available: http://techresearch.intel.com/articles/Exploratory/1435.htm
14. Smart medical home research laboratory (accessed: 2009, July 22). Available: http://www.futurehealth.rochester.edu/smart_home/
15. The aware home research initiative (accessed: 2009, July 22). Available: http://awarehome.imtc.gatech.edu/
16. Mit house_n (accessed: 2009, July 22). Available: http://architecture.mit.edu/house_n/
17. Smart houses info (accessed: 2009, July 22). Available: http://gero-tech.net/smart-homes.html
18. Intille SS, Larson K, Tapia EM, Beaudin J, Kaushik P, Nawyn J, Rockinson R (2006) Using a live-in laboratory for ubiquitous computing research. In: Fishkin KP, Schiele B, Nixon P, Quigley AJ (eds) Pervasive. Lecture notes in computer science, vol 3968. Springer, Berlin, pp 349–365
19. Fox D (2007) Location-based activity recognition. In: KI '07: Proceedings of the 30th annual German conference on Advances in Artificial Intelligence. Springer, Berlin, Heidelberg, pp 51–51. doi:10.1007/978-3-540-74565-5_6
20. Liao L, Fox D, Kautz HA (2005) Location-based activity recognition using relational Markov networks. In: Kaelbling LP, Saffiotti A (eds) IJCAI. Professional Book Center, Denver, pp 773–778. http://www.informatik.uni-trier.de/ley/db/conf/ijcai/ijcai2005.html#LiaoFK05
21. Sarkar AMJ, Lee YK, Lee S (2010) A smoothed Naïve Bayes based classifier for activity recognition. IETE Tech Rev 27(2):107–119. doi:10.4103/0256-4602.60164
22. Jelinek F, Mercer RL (1980) Interpolated estimation of Markov source parameters from sparse data. In: Gelsema ES, Kanal LN (eds) Proceedings, workshop on pattern recognition in practice. North Holland, Amsterdam, pp 381–397
23. The essentials of Google search (accessed: 2009, July 22). Available: http://www.google.com/support/bin/static.py?page=searchguides.html&ctx=basics
24. Google search basics: More search help—web search help (accessed: 2009, July 22). Available: http://www.google.com/support/websearch/bin/answer.py?hl=en&answer=136861
25. Katz S, Down T, Cash H et al (1970) Progress in the development of the index of adl. Gerontologist 10:20–30
26. Korpipää P, Koskinen M, Peltola J, Mäkelä SM, Seppänen T (2003) Bayesian approach to sensor-based context awareness. Pers Ubiquitous Comput 7(2):113–124. doi:10.1007/s00779-003-0237-8
27. Hu DH, Zhang XX, Yin J, Zheng VW, Yang Q (2009) Abnormal activity recognition based on hdp-hmm models. http://www.aaai.org/ocs/index.php/IJCAI/IJCAI-09/paper/view/521
28. Buettner M, Prasad R, Philipose M, Wetherall D (2009) Recognizing daily activities with rfid-based sensors. In: UbiComp '09: Proceedings of the 11th international conference on Ubiquitous computing. ACM, New York, pp 51–60. doi:10.1145/1620545.1620553
29. Lian CC, Hsu JYJ (2008) Chatting activity recognition in social occasions using factorial conditional random fields with iterative classification. In: AAAI'08: Proceedings of the 23rd national conference on Artificial intelligence. AAAI Press, Menlo Park, pp 1814–1815
30. Vail DL, Veloso MM, Lafferty JD (2007) Conditional random fields for activity recognition. In: AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems. ACM, New York, pp 1–8. doi:10.1145/1329125.1329409

Jehad Sarkar received B.S. and M.S. degree in Computer Science from National University, Bangladesh, in 1999 and 2000, respectively. Since September 2006, he is pursuing Ph.D. degree in Computer Engineering at Kyung Hee University, Korea. From May, 2000 to August, 2006, he served as a Principal Software Engineer in ReliSource Technologies Ltd., Bangladesh. From April, 2000 to April, 2005, he served as a Senior Software Engineer in TigerIT Ltd., Bangladesh. His research interests are Activity Recognition, Web mining, Data mining and Software Engineering. Email: jehad@oslab.khu.ac.kr.

La The Vinh received his B.S. and M.S. from Ha Noi University of TechnologyVietnam, in 2004 and 2007 respectively. Since September 2008, he has been working on his Ph.D. degree at the Department of Computer Engineering at Kyung Hee University, Korea. His research interests include digital signal processing, pattern recognition and artificial intelligence. Email: vinhlt@oslab.khu.ac.kr.

Young-Koo Lee received his B.S., M.S., and Ph.D. in Computer Science from Korea Advanced Institute of Science and Technology (KAIST), Korea in 1988, 1994 and 2002, respectively. Since 2004, he has been an assistant professor at the Dept. of Computer Engineering, College of Electronics and Information, Kyung Hee University, Korea. From 2002 to 2004, he was a Post Doctoral Fellow Advanced Information Technology Research Center (AITrc), KAIST, Korea, and a Postdoctoral Research Associate at Dept. of Computer Science, University of Illinois at Urbana-Champaign, USA. His research interests are Ubiquitous Data Management, Data Mining, Activity Recognition, Bioinformatics, Online Analytical Processing, DataWarehousing, Database Systems, Spatial Databases, and Access Methods. Email: yklee@khu.ac.kr.

Sungyoung Lee received his B.S. from Korea University, Seoul, South Korea. He got his M.S. and Ph.D. degrees in Computer Science from Illinois Institute of Technology (IIT), Chicago, Illinois, USA in 1987 and 1991 respectively. He has been a professor in the Department of Computer Engineering, Kyung Hee University, South Korea since 1993. He is a founding director of the Ubiquitous Computing Laboratory, and has been affiliated with a director of Neo Medical ubiquitous-Life Care Information Technology Research Center, Kyung Hee University since 2006. Before joining Kyung Hee University, he was an assistant professor in the Department of Computer Science, Governors State University, Illinois, USA from 1992 to 1993. His current research focuses on Ubiquitous Computing and applications, Context-aware Middleware, Sensor Operating Systems, Real-Time Systems, and Embedded Systems. He is a member of the ACM and IEEE. Email: sylee@oslab.khu.ac.kr.