ARHMAM: an Activity Recognition System based on Hidden Markov minded Activity Model

A. M. Jehad Sarkar, Young-Koo Lee, Sungyoung Lee
Dept. of Computer Engineering
Kyung Hee University
Korea
jehad@oslab.khu.ac.kr, yklee@khu.ac.kr, sylee@oslab.khu.ac.kr

ABSTRACT

The fundamental problem of the existing Activity Recognition (AR) systems is that these require real-world activity data to train the underneath activity classifier. It significantly reduces the applicability and scalability of the system. An AR system trained in an environment would only be applicable to that environment and would not be able to recognize new activities of interest. To overcome such difficulties, in this paper we propose a simple and ubiquitous sensor based AR system that uses web activity data to train its classifier. It would work to almost any environment and would be scalable by its very design. Given a set of activities to monitor, object names with embedded sensors and their corresponding locations, the ARHMAM first mines activity data from web, and uses these to build a Hidden Markov Model (HMM). In comparison with the existing web data based AR systems, it has the following advantages: (1) it uses more strong activity model, (2) it reduces the mining time significantly. It is observed that the class accuracy of activity recognition of our system for a real-world dataset is more than 64%, which is 20% more in comparison with its counterpart. Additionally, the mining time complexity is far better than its counterpart.

Categories and Subject Descriptors

H.4 [**Ubiquitous Systems Applications**]: Miscellaneous

Keywords

Activity recognition, web, Activity model, and HMM

1. INTRODUCTION

The purpose of an Activity Recognition (AR) system is to observe, understand, and act on day-to-day physical activities (e.g. sleeping, and cooking) of one or more individuals [5, 6, 7, 11, 14, 15]. Such computing systems have profound conceptual and practical implications. These systems have long been a goal of researchers due to its strength in providing personalized support for many diverse applications such as medicine and health-care.

The fundamental problem of the existing AR systems is that these require real-world activity data to train the underneath activity classifier. The system that would work to a particular environment, needs to acquire data for a prespecified period of time with the help of one or more individuals (to label their own activities). It limits the acceptability and scalability of the system. Given a large number of activities, and the fact that individuals are lay persons, it would be difficult to obtain much labeled data [16]. Additionally, an AR system trained in an environment would not be applicable to other environment. Moreover, such system needs extensive interaction with the domain experts during the training phase.

To overcome the above limitations, we need an alternate source of activity data such that an AR system can use such source to train the classifier. Advancement of Internet and WWW encourages millions of users to promote billions of web pages of varieties of contents [3]. A fraction of these pages describe in details how to perform daily activities. They do not only state the activity but also depict where to perform this activity, what objects to use and in what sequence. These pages can be regarded as the textual representations of real-world activities. An AR system would be broadly applicable and scalable by its very design if it can use such pages to train the classifier.

A few efforts have been made to train the activity classifier from web data (we call it, *Train From web (TFW)*) rather than from the real-world data (we call it, *Train From Environment (TFE)*) [12, 16]. In [12], Perkowitz et al. proposed a TFW based system, which is packaged with thousands of activity models for different domains. It significantly limits the applicability of the system because they fail to capture the idiosyncrasies of the environment to which it will be deployed [16]. Although the proposed system in [16] has focused on a particular environment, the mining method is extremely time-consuming. It might take hours to mine a single activity data.

Additionally, the classification accuracy of the above approaches is not up to satisfactory level. This is mainly because their activity models are only *Object-usage Based Models (OBMs)*. The problem of using only OBM is that in any environment there could be tens of objects, many of these could be used for may activities. For example, refrigerator can be used for cooking or it could be used for bathing. It would be hard for an AR system to discriminate

ICUIMC '10, January 14-15, 2010, SKKU, Suwon, Korea

such activities using only OBMs.

To overcome the aforementioned limitations, in this paper, we propose a *Location-and-Object-usage Based observation Model (LOBM)* which can be used by a HMM to classify activities. We then develop a novel and a straightforward algorithm to mine activity data for that model. The algorithm uses a set of advance operators of a search engine to mine object-usage and location-usage knowledge for the activities. It not only reduces the mining time dramatically, but also makes the system *easy to use and configure* and *highly scalable*. We performed *three experiments* to validate our systems performance and we proved that our proposed mechanism achieved higher recognition accuracy in comparison with its counterpart.

The rest of the paper is organized as follows. In Section 2, we presented the reviews of previous works related to AR. In Section 3, we described the proposed activity recognition system. In Section 4, we presented our experimental results to support our claims. In Section 5, we concluded our paper with a direction of future work.

2. RELATED WORK

A variety of simple and ubiquitous sensors based AR systems have been proposed. For example, Tapia et al. [14] first employed such 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. [15] 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. [12] introduced the notion of mining the generic activity models from web. They have shown that it is possible to convert the 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 understands the HTML structure of such website,
- 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 [11]. 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 [16].

Wyatt et al. [16] proposed an Unsupervised Activity Recognition System (UARS) using mined model from web. They first developed two algorithms: First one is the document genre classifier that would identify the pages describing an activity. Second one is the object identification algorithm that would extract objects from a page and calculate the object's weights within the page.
Their proposed 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 an 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, which is set to a constant probabilities, 3) and the observation probability matrix B, where $B_{ji} = p(object_i|activity_j)$.

Our work is closely related to above two works. We developed an AR system using simple and ubiquitous sensors that would be broadly applicable, and easy-to-use. Our system also mine activity data from web to train its classifier. Despite these similarities, we have several differences which are summarized below:

- We use both object and location in the same model.
- The proposed mining algorithm not only dramatically reduces the mining time, but also makes the system easy to use and configure.

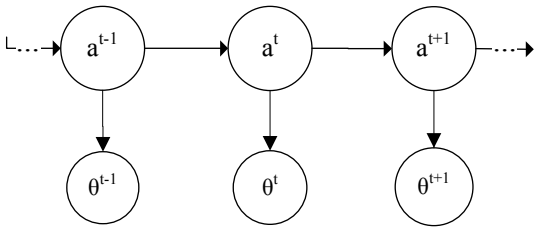

Figure 1: The Graphical model of a Hidden Markov Model.

- Our proposed system is highly scalable. We can add new activity or new object by simply giving activity name or the object (with embedded sensor) name.

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

3. ACTIVITY RECOGNITION SYSTEM

Given a set of activities to monitor, object names (with embedded sensors) and their corresponding locations, the goal of the AR system is to recognize subject's activity from a series of activated objects (as the subject interacts with the objects) at any given time.

Let $A = \{a_1, a_2, ..., a_m\}$ be the set of activities to monitor, $O = \{o_1, o_2 ..., o_t\}$ be the set of objects and $L = \{l_1, l_2, ..., l_q\}$ be the set of locations in the environment. Where, m, t, and q are the total number of activities, objects, and locations respectively. Let $\Theta = \{\theta_1, \theta_2, ..., \theta_n\} \in O$ be the activated objects (or set of object-usage sequence) at any given time and $l_{\theta_1}, l_{\theta_2}, ..., l_{\theta_n} \in L$ be the corresponding locations. Where, n is the total number of activated objects. The goal is to map the observation sequence (i.e. activated objects, Θ) into predefined activity labels.

We utilize the Hidden Markov Model (HMM) to capture this mapping, which requires us to do the followings:
-learn the HMM parameters from web,
-infer the activity labels for the observation sequence.

3.1 The Hidden Markov Model for AR

The Hidden Markov Model (HMM) is a sequential probabilistic model in which the system being modeled is assumed to be a Markov process with hidden states. The transitions among the hidden states are governed by a set of probabilities called transition probabilities. In a particular state an observation can be generated, according to the associated probability distribution. The graphical model of an HMM is shown in Figure 1. It consists of a Hidden state (e.g. activity) a^t at time t and the observation (e.g. activated objects) Θ^t on each state. Hidden state at time t depends on the previous state at time $t-1$. And the observed variable at time t depends on the state at time t. The goal is to find the joint probability distribution,

$$P(a, \Theta) = \prod_{t=1}^{T} P(a^t|a^{t-1})P(\Theta^t|a^t) \quad (1)$$

Where, $P(a^{t-1}|a^t)$ is the the transition probability from state a^{t-1} to a^t and $P(\Theta^t|a^t)$ is the probability of observing Θ^t in state a^t.

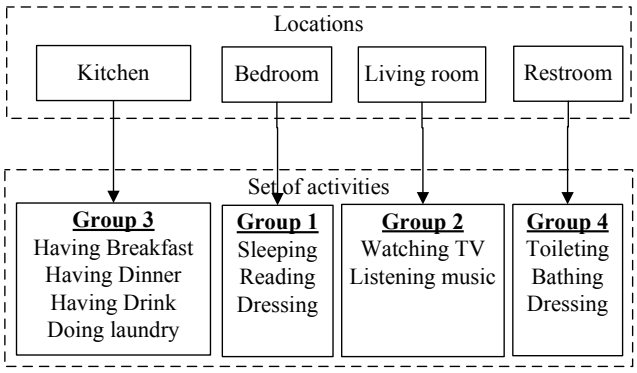

Figure 2: An example: Location specific activities.

If we train an AR system from environment, we count the number of occurrences of transitions, observations and states to find the probabilities that maximize the joint probability [15, 13]. However, as we consider web data to train the classifier, there is no way we can count the transitions because the these are highly subject dependent. Therefore, the transition probability matrix T is set to a constant probabilities (similar to [16]). We can only count the number of occurrences of observations.

We have n number of distinct observation symbols per state (i.e. the activated objects Θ^t). The observation symbols correspond to the physical output of the system being modeled. For our case we consider activated objects and their corresponding locations as the observations symbols per state. We describe more in details in the following sub subsection.

During inference, the Viterbi algorithm is used to find the the most likely labels for the new observation sequences [15]. This algorithm has been successfully applied with HMM to solve many activity recognition problems.

3.1.1 Location and Object together as the observations

Most of the AR systems [15, 12, 16, 8] utilize the object as the only observation. The downside of such systems is that the number of distinguishing objects between activities decreases if the number of activities to monitor grows. Such systems would produce more confusion between activities because there will be many overlapping objects.

Location of a person's provides important context information which is extremely helpful to make the classification decision [4, 10]. It is common to observe a specific location for an activity. For example, the kitchen can be observed for cooking and the bathroom for bathing. The group of activities are limited for a given location as shown in Figure 2. Combining both location and object as observations would increase the separation between activities, in comparison with the object only observation model.

3.1.2 Observation probability distribution in a state

Recall that in our case we consider activated object (object-usage) and their corresponding location (location-usage) as the observation symbols per state. We now formally define the distribution model.

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

$$P(\Theta^t|a^t) = \prod_{k=1}^{|\Theta|}(\lambda P(l^t_{\theta_k}|a^t) + (1-\lambda)P(\theta^t_k|a^t)) \qquad (2)$$

where, $l^t_{\theta_k}$ is the location of the object θ^t_k, $P(l^t_{\theta_k}|a^t)$ and $P(\theta^t_k|a^t)$ are the probabilities of location and an object usage respectively for a given an activity, a^t, at time t.

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.

3.2 Activity mining

In this subsection we first define the goals of activity mining and then we describe the mining algorithms to accomplish these goals.

We can see from Equation (2) that during training we need to estimate the following probability distribution:
$P(l_{\theta_i}|a^t)$, $P(\theta_i|a^t)$
The goal of mining is therefore to provide sufficient *activity data* such that these can be used to estimate the distribution. By the term activity data we mean the knowledge associated with the object-usage and location-usage for an activity.

To ease the understanding of the mining algorithm, we need to know the types of activity pages that are available in web.

3.2.1 Types of activity pages in web

There are two types of pages in WWW which are related to human activities: The *Explicit Activity Catalog Page (EACP)* and the *Implicit Activity Catalog Page (IACP)*.

Definition 2. Explicit Activity Catalog Page (EACP): A web page is an Explicit Activity Catalog Page (EACP) if it provides instructions in detail, how to perform an activity. Such a page has a title, which in most cases contains the activity name. It also has a body, which provides detail descriptions of how to perform the activity and may also specify the object-usage and location-usage for that activity.

For example, the web page[1] shown in Figure 3 is an EACP that contains the activity name (i.e. "bathing") in its title. In description section, it describes how to perform that activity and what object(s) (e.g. door, lights) to use and their usage sequence.

Definition 3. Implicit Activity Catalog Page (IACP): A web page is an Implicit Activity Catalog Page (IACP) if it does not directly defines how to perform the activity but instead provides the instructions that would influence the activity. It has the similar features (e.g. object-usage) as an EACP.

For example, the web page[2] shown in Figure 4 is an IACP that describes the list of steps required to make bathing safe

[1] `http://www.healthguidance.org/entry/7400/1/Bathing-in-Style---The-Art-of-Bathing-Well.html`
[2] `http://ezinearticles.com/?id=2148355`

Bathing in Style - The Art of Bathing Well

...
The perfect length of a bath is 10-15 minutes. After that your skin starts to wrinkle and your water gets cold. It is always good to have a bottle of water available since a warm or hot bath can be dehydrating. Be sure to sip water if you feel the need. Some people prefer a glass of red wine, champagne or port to help relax. Or you can have a cup of green tea or even chamomile tea to help you relax. Lock your ***door***, turn the ***lights*** off and light as many candles as you can (candles are essential as they affect your mood). Support your head with a bath pillow or a folded up towel. Pick a nice relaxing ***CD***, close your eyes and enjoy your peace and quiet.
...

Figure 3: An explicit activity catalog page that provides information related to object usage for an activity.

Table 1: Google Query Modifiers and Operators.

Name	Description
""	We use the quotes to force 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 their title. For instance, the query [intitle:"Preparing dinner"] would find all web pages that have "Preparing dinner" in their title.
+	We can force Google to include the word(s) in search result. 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 phrase "Butler pantry" in their text.

for independent seniors. In detail description section of this page, it mentioned the terms like, "bathroom", "bathtub", and "doorway". It does not directly reflect the steps required for bathing but it provides information that is related to bathing.

3.2.2 Mining of activity data from web

The mining algorithm uses Google to search the number of pages that describe an activity. Searching on Google is simple, choosing the appropriate search terms is the key to find the required information [1]. 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 [2]. 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 1 shows the modifiers and operators we used to mine activity data.

Figure 5(a) shows the schematic diagram for mining. The corresponding algorithm is shown in Algorithm 1. For each

How to Make Bathing Safe For Independent Seniors

...

The simplest and most inexpensive remedy is to have a ***bath seat*** installed. This way, the user can have a seat in the ***bathtub*** for stability. The seat would be too high to take a bath, so the best way to bathe in this manner would be to use a hand-held shower head. Keep in mind the drawbacks for this method: The user still must step over the side of the tub to get in and out, and they will have to manually clean themselves with the shower head. If the user's mobility is not terribly restricted, just a little slow or unsteady, this method could easily work.

A ***bath lift*** would work better than a bath seat for someone with limited mobility. These mobility aids allow a user to sit comfortably before lowering them down into the bath. Once they are ready, the device lifts them back up to a sitting position. Often, they will feature a transfer bench so that the user can "slide" over the edge of the tub to get in or out. Bath lifts are more expensive than bath seats, but can restore privacy and independence even for seniors with moderately severe mobility restrictions.

Installing a tall walk-in bathtub in a separate area of the ***bathroom*** is probably the best way to guarantee safety for a senior with limited mobility. Walk-in ***bathtubs*** feature a ***doorway*** so that the user doesn't have to step over the side. While some walk-in bathtubs are meant to replace an ordinary ***bathtub*** as a permanent installation, I don't recommend those because they don't offer an easy way for the user to sit/stand and they will bring down the resale value of a house. Instead, opt for a tall walk-in bathtub with a bench. This type of walk-in tub can be removed from the ***bathroom*** when it's no longer needed. This way, the user can take a bath while sitting upright, similar to sitting in a ***hot tub***. There's no need to lower the body to ground level, and getting in and out is easy. Walk-in ***bathtubs*** are gaining in popularity for residential use as more and more people decide that they are worth the cost to maintain their privacy and independence.

...

Figure 4: An implicit activity catalog page that provides information related to location and object usage for an activity.

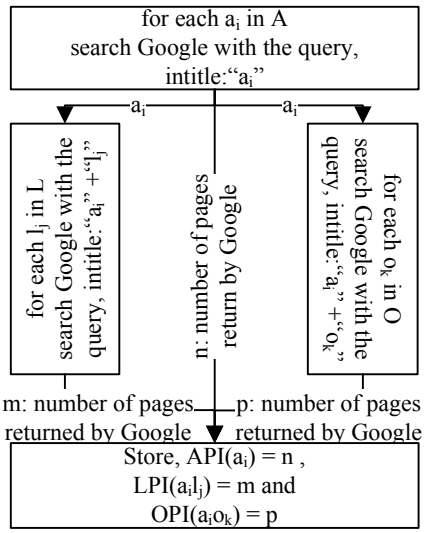

Figure 5: Activity Mining.

Algorithm 1: Mining(A, O, L). Activity information mining.

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)

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

activity $a_i \in A$, the algorithm 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 the location *(or location pages indexed (LPI))* $l_j \in L$ within the activity pages. The algorithm uses the query, *intitle* :"a_i" + "l_j" to return the number of pages containing the l_j in their text for a given activity pages. Let Google returns $m \subseteq \Omega$ pages that contain an occurrence of l_j. Similarly, let $p \subseteq \Omega$ be the pages return by Google *(OPI)*, if it searches with the query, *intitle* :"a_i" + "o_k". The algorithm finally saves $API(a_i) = |n|$, $LPI(l_j|a_i) = |m|$ and $OPI(o_k|a_i) = |p|$ into repository such that the observation probabilities can be estimated.

3.2.3 Number of queries require for mining

Given a set of activities A, objects O and their corresponding locations L, the total number of queries, r, required by the mining algorithm to mine activity data from WWW 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, it requires m queries to mine $APIs$, for q locations and m activities, it requires mq queries to mine $LPIs$, and for t objects and m activities, it requires 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 mining algorithm would need 260 queries in total.

3.3 Estimation of the observation symbol probability distribution

In this sub section we describe how the system transforms the mined activity data into probability distribution. It uses the following formulas (Equation (4), (5)) to calculate the

distribution:

$$P(l_{\theta_i}|a^t) = \frac{LPI(l_{\theta_i}|a^t)}{\sum_{l_j \in L} LPI(l_j|a^t)} \quad (4)$$

$$P(\theta_i|a^t) = \frac{OPI(\theta_i|a^t)}{\sum_{o_c \in O} OPI(o_c|a^t)} \quad (5)$$

It is to be noted that to estimate the conditional probability, the system uses $\sum_{l_j \in L} LPI(l_j|a^t)$ or $\sum_{o_c \in O} OPI(o_c|a^t)$ instead of $API(a^t)$ as the denominators. For example, let "Preparing Breakfast", "Preparing dinner" be two activities, and "Fridge", "Oven" be two objects. After mining the activity data for this scenario, we have,
$API(Preparingbreakfast) = 53$,
$API(Preparingdinner) = 119$,
$OPI(Fridge|Preparing\,breakfast) = 3$,
$OPI(Oven|Preparing\,breakfast) = 4$,
$OPI(Fridge|Preparing\,dinner) = 3$ and
$OPI(Oven|Preparing\,dinner) = 5$.
The system estimates the distribution as,
$P(Oven|Preparing\,breakfast = 4/(4+3) = 0.571$,
instead of $4/53 = 0.075$.
It means that we only consider the activity pages containing the specified objects or locations to reduce the *mining noise*. The mining noise is the noise associated with the number of activity pages returned by Google (APIs). As we can see in Algorithm 1, we use the query, $intitle$:"a_i" to mine the number of pages described 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, PE uses only the pages containing given objects or locations.

4. EVALUATION

4.1 Objectives

Our objective is to validate the performance of the AR system. We performed three experiments to test our system: First, we evaluate the classifier's performance in classifying activities. Second, we analyze the impact of the coefficients α in classification. 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 ARHMAM uses the site "`http://ajax.googleapis.com/`" (developed by Google for applications to retrieve data from the Google server asynchronously) to mine activity data instead of the site "`http://www.google.com/`". For example, to search the API for "Cooking", the mining algorithm sends an HTTP request, "`http://ajax.googleapis.com/ajax/services/search/web?v=1.0&q=Cooking`". In response, Google would return the formatted results like, estimatedResultCount (i.e. API), URLs of few the result pages (usually 4), link of more results, etc. Searching with the Ajax's site would retrieve a bit old data with respect to the original "`http://www.google.com`". It is to be noted here that it is not possible to search Google using "`http://www.google.com`" because it would not allow automated search.

4.2.2 Setup for evaluating system's performance

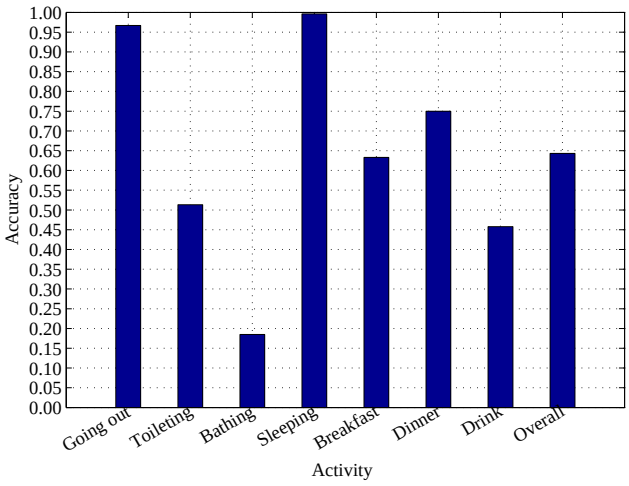

Figure 6: The accuracies per activity ($\alpha = 0.5$).

To evaluate the performance of the system we used the data gathered by Kasteren et al. [15]. They deployed 14 digital sensors in a house of a 26-year-old man, attached these to doors, cupboards, a refrigerator, and a toilet flush, and they collected data for 28 days. Their activities were chosen from Katz ADL index [9].

In our experiment, the sensor readings are divided into units of 60 seconds as it was done in [15]. This time is long enough for high recognition accuracy [15].

To test the TFE based learning technique, we used leave-one-day-out cross validation. In this strategy, one day is used for testing and remaining days are used for training.

As the activity instances were imbalanced between classes, two types of measurements were used to evaluate the performance of our system, similar to [15]. The time slice accuracy was measured by, $\frac{\sum_{i=1}^{N} detected_i == true}{N}$, and Class Accuracy was measured by, $\frac{1}{C}\frac{\sum_{i=1}^{N_c} detected_i == true}{N_c}$. Where, N is the total number of activity instances, C is the number of classes and N_c the total number of instances for class c.

Even though the time-slice accuracy is a typical way to evaluate classifier's performance [15], it is not always true for AR classifiers because dataset would contain dominant classes that appear a lot frequently than others. For example, the number of instances of "Toileting" is 114 and that of "Dinner" is 10. If a classifier correctly classifies 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, class accuracy should be the primary way to evaluate classifiers performance. However, in this article we report both the time-slice and the class accuracy.

4.3 Experiment 1: Activity Recognition Accuracy

The purpose of this experiment is to test the classifiers performance to classify the activities using mined data.

Figure 6 summarizes the results. The rightmost bar shows the overall accuracy. We achieve an overall class accuracy of 64.31% (timeslice accuracy was 96.17%). Table 2 shows the corresponding 7×7 confusion matrices. The entry in row i, column j represents the percentage of times an activity

Table 2: The Confusion matrix.

	Going out	Toileting	Showering	Sleeping	Breakfast	Dinner	Drink
Going out	**96.68**	00.89	00.03	01.67	00.00	00.08	00.65
Toileting	08.16	**50.00**	05.79	28.68	01.58	02.37	03.42
Showering	04.15	78.11	**14.34**	00.00	03.40	00.00	00.00
Sleeping	00.00	00.30	00.01	**99.68**	00.00	00.00	00.01
Breakfast	00.00	16.51	08.26	11.93	**63.30**	00.00	00.00
Dinner	02.30	03.74	00.00	00.00	00.00	**75.00**	18.97
Drink	03.39	03.39	00.00	01.69	06.78	38.98	**45.76**

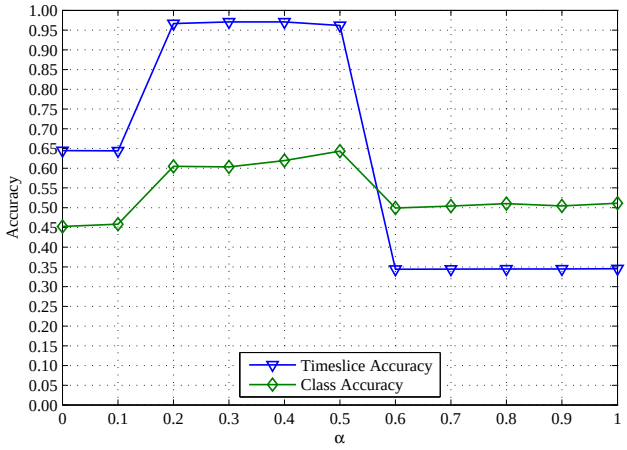

Figure 7: Activity recognition accuracy using different α settings.

with ground truth label i was recognized as activity j. The classifier performs worse in recognizing "Showering". Most of the time "Showering" is classified as "Toileting". This is because these two activities are closely related, same sort of objects are used to perform these activities. Similar things happened for "Drink" and "Dinner".

4.4 Experiment 2: Varying the model coefficients

The goal of this experiment is to analyze the impact of the coefficient, α,in accuracy of activity classification. We perform the test with α 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 (for both timeslice and class accuracy) are shown in Figure 7.

As expected, the accuracy of activity recognition are sensitive to the α values. For example, as we can see in Figures 7, for $\alpha = 0.0$, the accuracies of activity classification are relatively low with respect to $\alpha = 0.2$. It indicates that incorporating the location-usage based model significantly improve the recognition accuracy. But only the location-usage based model is not sufficient for high recognition. For example, as shown in Figures 7, the accuracies were relatively low when α is set to 1.0. The best performance is seen when α is set to 0.5. It means that both location and objects are equally useful to recognize activities.

Table 3: Comparison with other methods (class accuracies are used to compare).

Accuracies of the classifiers (%)		
TFW		TFE
ARHMAM	UARS	AARS
64.31	47.21	71.75

4.5 Experiment 3: Comparison with the other methods

The goal of this experiment is two-fold:

1. Compare the performance of the system in classifying activities with both TFW and TFE based methods.
2. Compare the time complexity of our proposed mining technique with the mining technique proposed in [16].

4.5.1 Performance comparison of classifiers

We compare the classifier's performance with both TFW and TFE based system. We compare with the TFE based classifier proposed in [15] by Kasteren et al. (we call their system as, Accurate Activity Recognition System (AARS)). We also compare the classifier's performance with a TFW based classifier, proposed in [16]. The comparison results are shown in Table 3.

In [15], Kasteren et al. proposed two probabilistic methods for activity classification, Hidden Markov Model (HMM) and Conditional Random Field (CRF). We compare our system's performance with HMM, because in their experiment class accuracy of HMM was better with respect to CRF.

It is observed that our methods achieved superior results in comparison with the TFW based methods. With the proposed mining technique, we achieve an accuracy of 64.31%, whereas with the proposed mining technique of UARS, we achieve an accuracy of 47.21%. With the AARS, accuracy goes up to 71.75%.

4.5.2 Mining time comparison

We now compare the mining time of the ARHMAM with the UARS [16]. The Figure 8, shows the required mining times for ARHMAM and UARS. As expected, the proposed system significantly reduces the mining time.

We analyze the total time, t, the ARHMAM and the UARS would take to mine an activity data. For this purpose, let us consider an environment to which there are 20 objects in 5 different locations, and we are trying to monitor

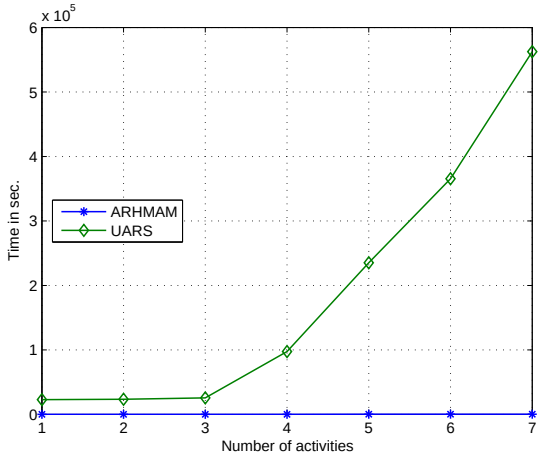

Figure 8: Mining time comparison between ARHMAM and UARS, Mining is performed in the sequence as shown in the first column of Table 2, i.e. 1. Going out, 2. Going out + Toileting, 3. Going out + Toileting + Showering, and so on).

1 activity (e.g. "Going out").

The ARHMAM would take $t = 1 + 1(5 + 20) = 26$ (using the Equation (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 mine the activity data, using following steps (in section 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 URAS 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 data, whereas the ARHMAM would only take 26 seconds.

5. CONCLUSION AND FUTURE WORK

In this paper we present an activity recognition system based on simple and ubiquitous sensors, which is broadly applicable and easy-to-use. We consider an environment to which a set of sensors are embedded with daily life objects.

We first address the problems of using real-world activity data to train an AR classifier and describe how to use WWW as an alternate source of activity data. We then address the problems associated with the models and the mining techniques of the existing web based systems. Finally, we propose a new model and a novel way to mine the model parameters from web. We have shown that it is possible to use the proposed models along with a HMM to recognize activities. We performed *three experiments* to validate our systems performance and we proved that our proposed mechanism achieved higher recognition accuracy in comparison with its counterpart.

In this paper we have treated the sequence of object-usage as independent and identically distributed (i.i.d.). Such a approach, however, would fail to exploit the sequential pattern of object-usage. To express such effect we need to relax the i.i.d. assumptions. Therefore, in future, we would propose a new observation model that would capture the sequential pattern, and a new mining method that would mine the object-usage sequence for an activity.

6. ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government (MEST-2009-0083121).

7. REFERENCES

[1] The essentials of google search (accessed: 2009, july 22). Available: `http://www.google.com/support/bin/static.py?page=searchguides.html&ctx=basics`.

[2] 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`.

[3] R. Cilibrasi and P. M. B. Vitányi. The google similarity distance. *IEEE Trans. Knowl. Data Eng.*, 19(3):370–383, 2007.

[4] D. Fox. Location-based activity recognition. In *KI '07: Proceedings of the 30th annual German conference on Advances in Artificial Intelligence*, pages 51–51, Berlin, Heidelberg, 2007. Springer-Verlag.

[5] M. R. Hodges and M. E. Pollack. An 'object-use fingerprint': The use of electronic sensors for human identification. In J. Krumm, G. D. Abowd, A. Seneviratne, and T. Strang, editors, *Proc. Ubicomp*, volume 4717 of *Lecture Notes in Computer Science*, pages 289–303. Springer, 2007.

[6] D. H. Hu, S. J. Pan, V. W. Zheng, N. N. Liu, and Q. Yang. Real world activity recognition with multiple goals. In *Proc. UbiComp.*, pages 30–39, New York, NY, USA, 2008. ACM.

[7] D. H. Hu and Q. Yang. Cigar: Concurrent and interleaving goal and activity recognition. In D. Fox and C. P. Gomes, editors, *AAAI*, pages 1363–1368. AAAI Press, 2008.

[8] S. S. Intille, K. Larson, E. M. Tapia, J. Beaudin, P. Kaushik, J. Nawyn, and R. Rockinson. Using a live-in laboratory for ubiquitous computing research. In K. P. Fishkin, B. Schiele, P. Nixon, and A. J. Quigley, editors, *Pervasive*, volume 3968 of *Lecture*

Notes in Computer Science, pages 349–365. Springer, 2006.

[9] S. Katz, T. Down, H. Cash, and et al. Progress in the development of the index of adl. *Gerontologist*, 10:20–30, 1970.

[10] L. Liao, D. Fox, and H. A. Kautz. Location-based activity recognition using relational markov networks. In L. P. Kaelbling and A. Saffiotti, editors, *IJCAI*, pages 773–778. Professional Book Center, 2005.

[11] D. J. Patterson, L. Liao, D. Fox, and H. A. Kautz. Inferring high-level behavior from low-level sensors. In A. K. Dey, A. Schmidt, and J. F. McCarthy, editors, *Proc. Ubicomp*, volume 2864 of *Lecture Notes in Computer Science*, pages 73–89. Springer, 2003.

[12] M. Perkowitz, M. Philipose, K. Fishkin, and D. J. Patterson. Mining models of human activities from the web. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 573–582, New York, NY, USA, 2004. ACM.

[13] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. pages 267–296, 1990.

[14] E. M. Tapia, S. S. Intille, and K. Larson. Activity recognition in the home using simple and ubiquitous sensors. In A. Ferscha and F. Mattern, editors, *Pervasive*, volume 3001 of *Lecture Notes in Computer Science*, pages 158–175. Springer, 2004.

[15] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kröse. Accurate activity recognition in a home setting. In *Proc. UbiComp*, pages 1–9, New York, NY, USA, 2008. ACM.

[16] D. Wyatt, M. Philipose, and T. Choudhury. Unsupervised activity recognition using automatically mined common sense. In M. M. Veloso and S. Kambhampati, editors, *AAAI*, pages 21–27. AAAI Press / The MIT Press, 2005.