Thesis for the Degree of Doctor of Philosophy

EARWD: AN EFFICIENT ACTIVITY RECOGNITION SYSTEM USING WEB ACTIVITY DATA

A. M. Jehad Sarkar

Department of Computer Engineering
Graduate School
Kyung Hee University
Seoul, Korea

August, 2010

EARWD: AN EFFICIENT ACTIVITY RECOGNITION SYSTEM USING WEB ACTIVITY DATA

A. M. Jehad Sarkar

Department of Computer Engineering
Graduate School
Kyung Hee University
Seoul, Korea

August, 2010

EARWD: an efficient activity recognition system using web activity data

by

A. M. Jehad Sarkar

Supervised by

Prof. Sungyoung Lee, Ph.D.

Prof. Young-Koo Lee, Ph.D.

Submitted to the Department of Computer Engineering and the Faculty of the Graduate School of Kyung Hee University in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Dissertation Committee:

Prof. Byeong-Soo Jeong, Ph.D. (Chairman)

Prof. Sungyoung Lee, Ph.D.

Prof. Young-Koo Lee, Ph.D.

Prof. Brian J. d'Auriol, Ph.D.

Prof. Jinsung Cho, Ph.D.

Abstract

Activity recognition systems (ARSs) have been a goal of researchers due to its strength in providing personalized support for many diverse applications such as medicine and health-care. A considerable amount of activity recognition systems have been proposed. The fundamental problem of such systems is that these are not general-purpose. To configure the system, extensive interaction with the experts is compulsory. Such a system would not be able to recognize the new activities of interest. An activity recognition system trained in an environment would only be applicable to that environment.

This dissertation presents an efficient activity recognition system that uses a set of simple and ubiquitous sensors. It is assumed that a set of sensors are embedded with appliances (or objects) like, door, cabinet, desk. Given a set of activities to monitor, object names with attached sensors and their corresponding locations, a method of mining activity information from World Wide Web (WWW), converting these into activity models is proposed. It is shown that it is possible to use such information for activity recognition in real time. The propose system is general-purpose for following reasons: First, it would be applicable to almost all environments. Second, it is configurable by the end-user with little expert knowledge. Third, it could learn its model parameters automatically from web as well as from the environment (e.g. home) to which it is deployed. Forth, it has the ability to handle growing amounts of activities and sensors in a graceful manner (effortlessly scalable). The novelty of the system, compared to the existing general-purpose systems, lies in: (1) it uses more robust activity models (2) it significantly reduces the time to mine web activity data. The system is tested with the activity data obtained from both web and real-world. The proposed mechanism yields significant improvement in comparison with the existing activity recognition systems proposed in the literature.

Acknowledgment

I am exceptionally fortunate to have two great advisors, Professor Sungyoung Lee and Professor Young-Koo Lee, during my Ph.D. study. I am greatly indebted to both of my advisors for teaching me how to do research, write papers, give presentations and write project proposals. I am especially grateful to Professor Young-Koo Lee for his constant guidance and long-lasting support. Without his help I would not be able to finish my study.

I would like to pay my highest level of gratitude to Professor Byeong-Soo Jeong, Professor Brian J. d'Auriol and Professor Jinsung Cho for their very useful comments and suggestions that helped me a lot to enrich the quality of the thesis. I would also like to express my deepest gratitude to Dr. Syed Khairuzzaman Tanbeer and anonymous reviewers whose valuable comments and suggestions helped me to consolidate and strengthen my research work. I have tried my best to incorporate their suggestions to demonstrate the effectiveness of the proposed system.

I am grateful to Dr. Abdur Razzaque, Dr. Mahbub Alam, Mohammed Shoyaib, Md. Shariful Islam, Kamrul Hassan and rest of the Bangladeshi students for their help, encouragement, suggestions and valuable ideas over the last four years. This thesis owes much of its contents to their ideas, guidance and wise advice. I am also grateful to KHU Bangladeshi community for their love throughout my four-year stay in Korea. Without them I would not be able to finish my Ph.D. smoothly.

I would like to thank Khondokar Tarik-ul Islam for helping me to get the opportunity for pursuing Ph.D. in Ubiquitous Computing Lab. Without his help it would not have been possible. I am thankful to Dr. Donghai Guan, Adil Mehmood Khan and Yong-Koo Han for being with me all the time and giving me constant support. I am also thankful to all my foreign lab members for giving me their constructive comments and suggestions that helped me a lot to improve the quality of my research. I am grateful to all my Korean lab mates for helping me to stay tuned in Korea during my Ph.D. study. Above all, I would like to thank all Kyung Hee University faculty members and stuffs to provide all the supports I need for my study.

I am thankful to my wife, Khodeza Israt Jahan for her sacrifice and constant support during my Ph.D. study. I am also thankful to my little children, Ivan Jehad and Jinan Israt who have endured very much while I was busy in study. I am greatly indebted to my parents-in-low, Professor Muhammed Khoda Rakha and Professor Eliza Akhter Begum, brother-in-low, Captain Monirul Hassan Ahemed for their constant supports and blessings. Above all, I am grateful to my parents, Shahana Begum and M. A. Jalil Sarkar for the sacrifices they have made throughout their life for raising me. They have always tried their best to support my dreams, aspirations, and shown me the proper way of success. Everything I am, I owe to my parents. I would like to dedicate this thesis to my family.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Understanding human activities or behavior has long been a goal of humans from the beginning of mankind. We are curious about the daily activities (e.g. taking medication, playing games, watching TV) of our parents, children or even relatives. Companies may be interested in monitoring the activities of one or more workers. By recognizing the activities of daily livings (ADLs), it is possible reduce the risk of potentially life-threatening abnormalities. By recognizing activities during production, it is possible reduce the incidence of making harmful products. However, employing a person to monitor (or recognize) ADLs is not realistic because it needs constant monitoring. Therefore, researchers are trying to automate the recognition process. This can be termed as the computerized activity recognition.

Even though it sounds a simple task, it is a quite complex for a computer because it involves many complicated tasks like, sensing, learning and inference. For example, if a system to recognize an activity of a person is developed, the system needs to know the location in which the person is pursuing the activity, time of the persuasion, the set of objects he/she is interacting with, previous history of the person, etc. The main intention of this research is to make a reliable computerized system that would have similar capabilities as human to recognize ADLs.

Activity recognition is the key element for many ubiquitous computing applications ranging from office worker tracking to home healthcare. In general, activity recognition systems observe the behavior of people, and, when necessary, take actions in response [1–8]. For example, in a home environment, it can remind users to perform missed activities or complete actions (e.g. taking medicine), help them recall information, or encourage them to act more safely [9]. In a hospital environment, it can remind a doctor or nurse to perform certain tests before operating. Or in a production environment, it can ensure the quality of the product by monitoring the set of actions. In addition to these, patterns mined from home activities can be used to support a wide range of applications. For example, recommended healthy lifestyle of a Young Parkinson disease (YOPD) patient.

If an automated activity recognition system is considered, the first thing we need to consider is how to determine the state of the physical world. There are several ways to do this using different sensor systems [10]:

1. remote observation of the subject (user) using audio and video sensors and analyze the generated signals,

2. track and identify primitive human actions by attaching sensors to the body of the subject, and

3. embed sensors to appliances (or object) and track their usage.

Audio and video sensors based activity recognition [11–19] is complex to implement because it requires processing of highly multidimensional data. Additionally, both types of sensors information may violate user's privacy. Although body attached sensors is promising to identify primitive sequences of movements [15, 20–24] such as walking and running, it is difficult to identify goal-oriented activities (or activities of daily living) such as cooking and bathing. Many recent papers [5, 6, 10, 25–28] have shown that it is better to use embedded sensors to identify goal-oriented activities, since such activities require interaction with objects. In this thesis, an activity recognition system is developed, which recognize goal-oriented activities using embedded sensors. This is seemingly the best choice since it is possible to make such sensors nearly invisible and therefore should not impact on daily life activities; it would be inexpensive and would be used to recognize the essential activities.

The mechanism of learning and inference would be the next thing to consider. The theme of this thesis is to provide an efficient mechanism to do such tasks. Activity recognition based on sensors is a challenging task due to the inherent noisy nature of the input. Thus, the temporal probabilistic reasoning and machine learning techniques are powerful tools for this task. However, building a comprehensive activity recognition system for real-world environments is extremely challenging even with state-of-the-art tools. In this thesis, a novel learning and inference techniques are developed to make an automated activity recognition system to recognize a variety of real-world human activities from real sensor data.

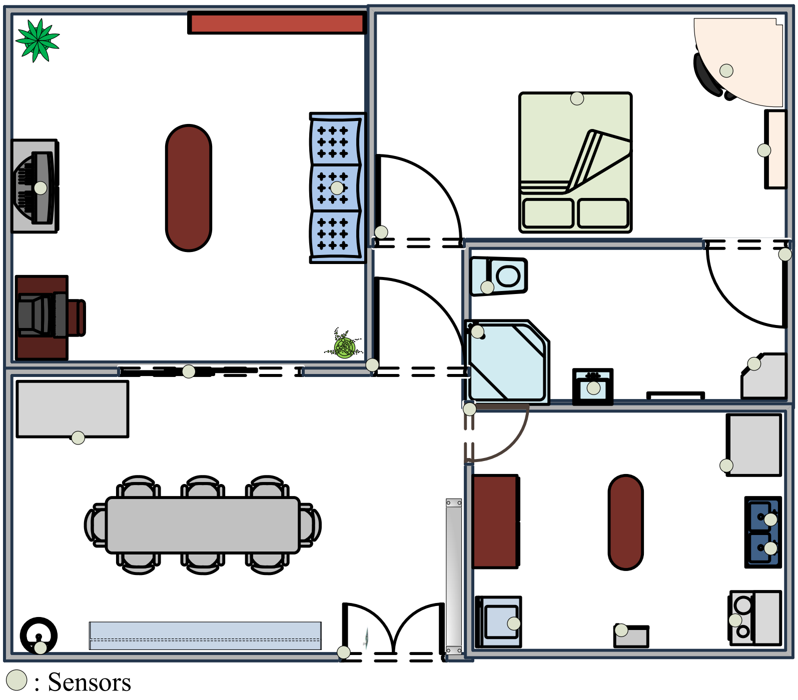


Figure 1.1: An example environment for activity recognition.

In our daily life we usually perform an activity by interacting with a series of objects. For example, for bathing we may interact with a door, light, exhaust fan, shower faucet, etc. The strategy is to embed sensors into these objects such that it is possible to determine the state of that object should a person interact with it. An example of an environment in which a set of objects are embedded with sensors is shown in Figure 1.1. The objective is to build an activity recognition system that can recognize activities based on a set of object-usage (interaction with an object) at a given time. An example of activity recognition based on a set of object-usage is shown in Figure 1.2.

1.1 Motivation

The sensor-based activity recognition system faces several challenges due to the noisy nature of the input and because of the large number of activities to be recognized [10]. There are several

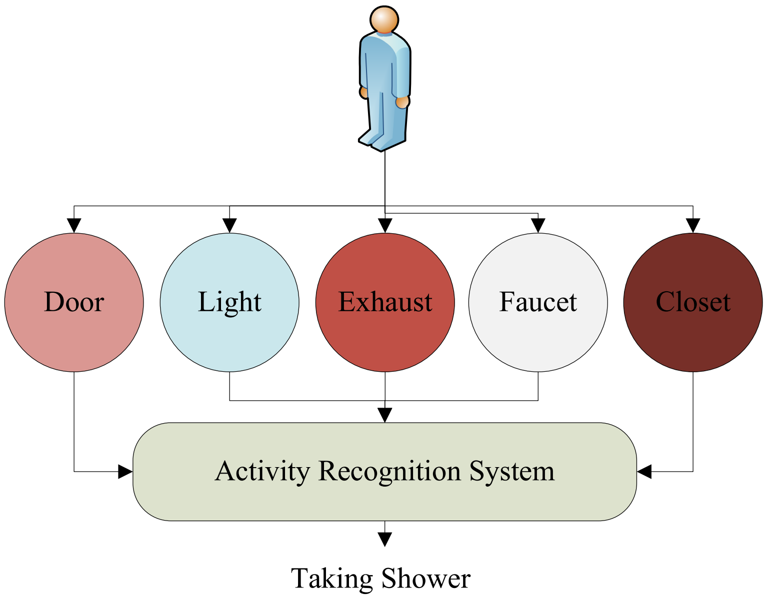


Figure 1.2: An example of activity recognition based on a set of object-usage.

ways of pursuing an activity. For example, one can use either bathtub or shower faucet or both to take a shower. The biggest challenge is to build a robust model that can represent the mapping between low-level sensor data to high-level activity.

The straightforward way would be to learn the models and train the system using the real-world activity data acquired from an environment. In order to do this, a sequence of steps is required as shown in Figure 1.3 (a),

1. Select the environment (e.g. home) in which the system will be deployed.

2. Choose and embed sensors to a set of objects (one sensor/object).

3. List the set of activities that will be recognized by the system.

4. Assign a participant (or volunteer) to stay at that environment and perform the listed activities for a specified period of time (e.g. 4 weeks). During this period, the participant will annotate their activities using an experience sampling tool (ESM) [6, 29, 30]. When using the ESM tool, the participant need to carry either a PDA (with an ESM software installed on it) or wear a headset to report the current activity to a server (with an ESM software installed on it). For every specified period of time (e.g. 10 minutes), the tool provides the

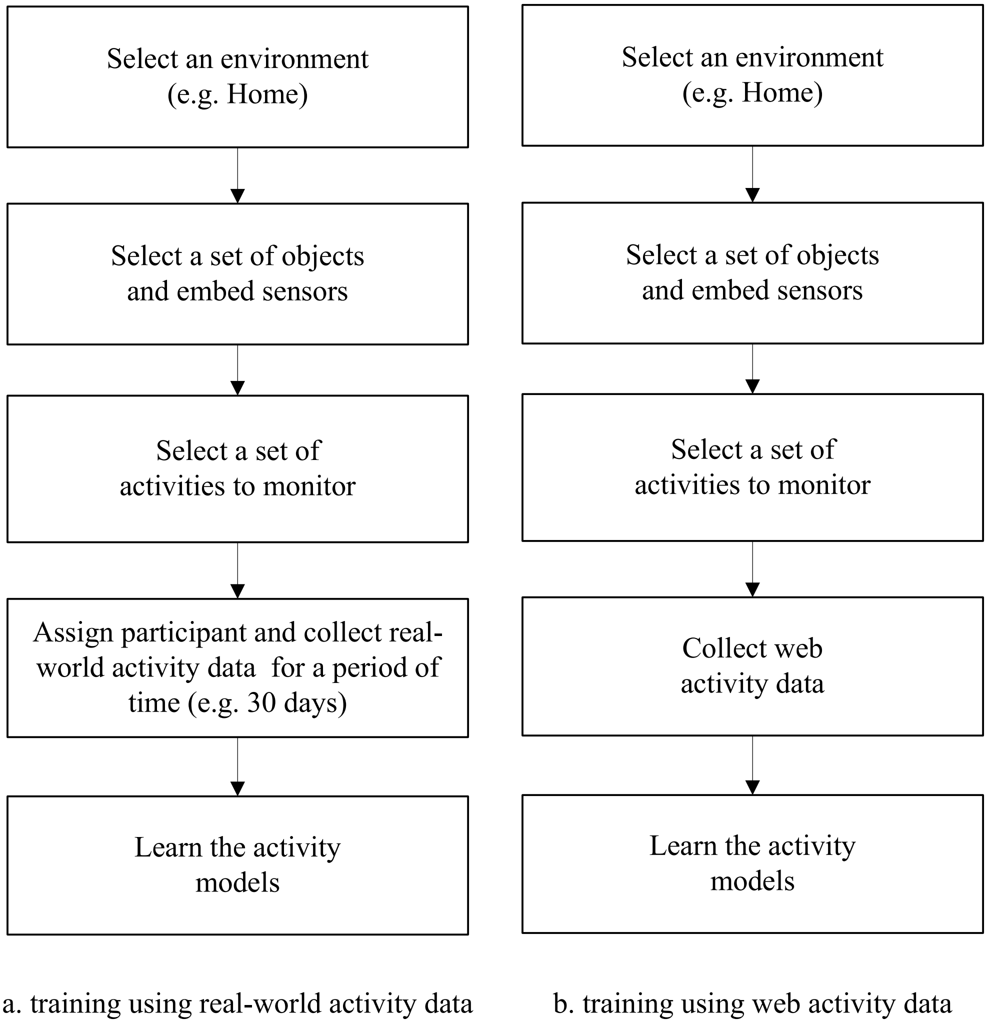


Figure 1.3: Two ways to train an activity recognition system: a. using real-world activity data, b. using web activity data.

list of activities to user for obtaining input from the participant, if he/she select an activity, the tool then stores this information along with the activated sensors (as the user interacted with the objects). This is called data collection period.

5. After the data collection period, the activity recognition system is trained with the help of collected data.

However, this approach is impractical because of the following reasons:

- First, using an ESM for real-world activity data collection has several disadvantages [31]: it may appear burdensome to the participants; and 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.

- Second, this approach is subject-dependent. Such a dependency greatly reduces the applicability and easiness of an activity recognition system. This system is not general purpose for three reasons,
  - First, an activity recognition system trained in an environment would not be applicable to another environment. It needs to be trained for every environment in which the system will be deployed.
  - Second, it would not be able to recognize new activities. If it is required to add new activities of interest, such a system needs to be trained again.
  - Third, it requires extensive interaction with the experts, especially in the training phase.

To overcome the aforementioned limitations, we need to ease the human dependency for data collection. We need an alternate source of the training data and an efficient method to extract such data. An alternate approach to train an activity recognition system is shown in Figure 1.3 (b). Advancement of the Internet and the World Wide Web (WWW) encourages millions of users to promote billions of web pages with varieties of contents [33]. A fraction of these pages either explicitly or implicitly provide information related to the activities of daily livings. For instance, the web page[1] [32] shown in Figure 1.4 provides information for how to prevent bathing injuries among elderly. Such web pages not only state an activity but also depict where to perform this activity and what objects to use and in what sequence. An activity recognition system would be broadly applicable and scalable by its very design if it can mine information from such web pages to learn the models. It would remove any subject's dependency and therefore make the system general-purpose.

[1] `http://www.ehow.com/how_2242133_prevent-bathing-injuries-among-elderly.html`

How to Prevent Bathing Injuries Among the Elderly
...
Step #1 Place slip-resistant **bath mats** both inside and outside the **tub**/shower area. Immediately clean up any water that may get splashed on the floor. Carpeting the **bathroom** floor is another option for safer footing.

Step #2 Hang **towel rods** that are rounded rather than rods with sharp corners to help prevent serious injuries from falls. Also, place a padded safety cover over the tub spout to protect against sharp edges. Wrapping a washcloth around the **tub faucet** will work just as well.

Step #3 Mount sturdy grab bars in the bathing area. Vinyl coated bars or bars with ridges provide for better grasp. Never use a towel rod for a grab bar, as they are not designed to bear weight. In addition, do not use glass shower **doors** for support. Be sure to place bath accessories within easy reach.

Step #4 Use adequate lighting in the **tub**/shower area. Low illumination in combination with poor vision can increase the risk of falling. A ceiling fixture is one way to provide plenty of light.

Step #5 Install an emergency **telephone** within easy reach.

Step #6 Set **water heater** thermostat to its lowest setting, typically not above 120 degrees F. Many people do not realize that hot water burns just like fire. A simple precaution is to purchase an anti-scalding device that shuts off water from a shower or **bathtub faucet** if the water temperature gets too hot. Found in hardware stores, these devices are easy to install.
...

Figure 1.4: An web page that provides information related to an activity [32].

A few efforts have been made to train an activity recognition system from the web (*Train From Web (TFW)*) rather than from the environment (*Train From Environment (TFE)*) [27,28]. The system in [27] 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 [28] 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 very low. One of the main reasons is that their activity model is only an *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, let us consider an environment in which the bathroom and the kitchen both have the following three objects: door, cabinet and faucet. Now, let us consider that a kitchen activity (e.g. washing dishes) is performed by a person using these three objects. It would be hard for an activity recognition system (that uses only OBMs) to recognize this activity correctly because there could be a bathroom activity (e.g. washing face) that usually requires the same set of objects as well. Therefore, only the OBM is not enough for an activity recognition system to produce high accurate recognition results.

The location (e.g. bedroom, kitchen) of a person in an environment can provide important context information for classification decisions and thus could be very helpful for activity recognition [34, 35]. It is common to use a specific location (e.g. the kitchen is for cooking and the bathroom is for bathing) to do an activity. The group of activities is limited for a given location. For example, preparing breakfast, preparing dinner etc. are kitchen activities, bathing, toileting, etc. are bathroom activities. It is possible to build an activity recognition system that uses only location-usage based model. However, such a system will only be able to provide high-level abstraction (or clue) of an activity (e.g. kitchen activity). It means that the system will not be able to recognize any specific goal-oriented activity (e.g. cooking).

It is possible to build a system that works in two-layer, in the first layer it classifies a group of activities (e.g. kitchen activity) using a location-usage based model and in the second layer it classifies the actual activity from that group using an object-usage based model. This is the main idea of the activity recognition system proposed in this research. However, location information by itself is not enough to classify the true activity group because a subject (or user) may switch among locations while performing an activity (e.g., moving back and forth between the living-room and kitchen while cooking). Such a situation could be named as location-confusion. It restricts the system's ability to recognize the actual activity group. Therefore, object with location in the first-layer is used in the proposed system to resolve any location-confusion which in turn helps the system to classify actual group.

It is also possible to classify an activity by using the combination of both object-usage and location-usage information. With this setting, it is desirable that such a system should provide high-level accuracy. However, it is important to stress here that such assumption might not be realistic in most cases. Because location-usage information, in one hand, can provide high-level abstraction (or clue) of an activity (e.g. the kitchen activity) and object-usage information, on the other hand, can provide low-level abstraction of an activity (e.g. cooking), the combination of these two can only generate mid-level clue. In practice, the low-level clue is primarily necessary to accurately classify a specific activity. Therefore, combining location and object information might not be suitable for accurately classifying an activity.

Furthermore, failure to classify activities with no specific location is another major drawback of the combined approach. For example, doing laundry (activity) is usually performed with a washing machine (object) which could be located in kitchen or in the foyer. Hence, doing laundry, in general, is an activity with no specific location, even though it has its own key object(s) i.e., washing machine. Let us term such activities as location-independent activities. It is highly probable that if the activity information is mined from the web, the location usage probability of a location-independent activity for a particular location would be relatively low compared to other activities specified for that location. For instance, the probability of using kitchen for doing laundry will be low with respect to that of other kitchen activities (e.g., dinner). Therefore, for a location-independent activity combination of both object-usage and location-usage information would not be appropriate, because using location with object in a model reduces the influence of object to that model. If the influence of such object(s) is reduced for an activity, it will eventually reduce the probability of classifying that activity. As a result, the system may become more prone to misclassification.

To overcome the aforementioned limitations, the motivation is to build an activity recognition system that first classifies a group of activities (e.g. kitchen activity) using a Location-and-Object-usage based model (LOBM) and then subsequently, classifies the actual activity (e.g. cooking) from that group using an object-usage based model. The system uses the web activity data to learn the models. This system will be applicable to a diverse set of environments and would provide high accurate recognition results. An end-user will be able to configure the system within a very short period of time and without the help of experts. He/she would even be able to add new activities or objects to the system by simply providing the name of the activities or objects.

1.2 Challenges

Although the need for the activity recognition research is addressed by many researchers past, it is not until recently that it has become one of the most active areas of research. A handful amount of researchers are working on building a variety of activity recognition systems. A lot of human behavior attributes (e.g. Multitasking, periodic variations, time scale) related challenges [36] for activity recognition have already been solved by the researcher. However, the challenges associated with the development of robust algorithms still exist. Additionally, there are a lot of challenges that exist to make an activity recognition system that uses web activity data to train the underlying classifier, since, few efforts have been made in this area. In this thesis the following challenges are dealt with:

- **Integrating location :** as discussed earlier, the motivation is to integrate location information along with object information to construct the robust models such that the system can use this to improve the activity recognition accuracy. However, in order to integrate location the following needs to be considered:
  - The set of object-usage for doing an activity may differ from location to location. For example, cleaning a bathroom is required a different set of object-usage than cleaning a kitchen.
  - It is required to generalize the activity models such that the system uses this will applicable to a diverse set of environments (i.e. not limited to an environment).
  - As the parameters (location and object) are integrated in a same model, a way to control the influence of each of the parameters to the model is required. Since, equal influence to each of these parameters may not appropriate in obtaining high-accurate recognition result.

  - A way to estimate the optimal influence of each of these parameters in the model is needed.

- **Activity grouping :** as described in section 1.1, the motivation is to use a two-layer classifier in which the first layer would classify the group of activities. To accomplish such goal, an efficient mechanism to group the activities is needed.

- **Web activity data mining :** Even though there are millions of pages that either explicitly or implicitly describe the activities of daily livings, identifying these pages would be the biggest challenge of any mining algorithm. How to extract activity knowledge from these pages would be the next big challenge. Above all, the hardest part would be to do the above tasks using a less amount of time.

1.3 Contributions

In this dissertation, an efficient activity recognition system that uses web activity data to train the underlying classifier is developed. It is applicable to diverse environments, it provides high accurate recognition result and it is possible to configure the system by an end-user with little expert knowledge. A high-accurate two-layer probabilistic classification integrating location and object-usage information is proposed. The first-layer uses a location and object-usage model to narrow down the scope for the recognition task, and the second-layer uses an object-usage based model for the actual activity 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.

The Naïve Bayesian (NB)-based classifier is used in both levels of classification. The parameter estimation for NB-based models uses the method of maximum likelihood (ML). Data sparsity is a major problem in estimating ML in AR because the size of the training data is relatively small in comparison with the other machine learning datasets. The distribution of the observed sensors in a dataset may not be always even between the activities. Additionally, some sensors would appear during testing but would not appear while training. This is called *zero-frequency* [37] problems which assign *zero-probability* of an *unseen object-usage* for an activity. To prevent such estimation problem, a *Smoothing Technique* is developed for adjusting the maximum likelihood of the probabilities to produce a more precise activity model.

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

Four experiments are performed with three real-world activity datasets to validate the system's performance. The proposed system achieved higher recognition accuracy and significantly reduces the mining time in comparison with the method of Wyatt et al. [28].

1.4 Organization

The thesis is organized as follows:

- Chapter 1 has presented a brief introduction of the concepts of activity recognition, and the problems associated with the current state-of-the-art systems that use real-world activity data. It has also described the importance and benefit of using web activity data to train an AR system. Finally, Chapter 1 has described the limitations of the existing systems that use web activity data and an overview of contributions in this thesis.

- Chapter 2 discussed the related work in the area of embedded sensors based activity recognition. It first describes the data collection frameworks and the essential tools for pursuing activity recognition research. Then the state-of-art probabilistic methods for activity recognition are described. Related work on embedded sensor based activity recognition systems that use real-world activity data is also discussed. Finally, the activity recognition systems that use web activity data are discussed with a set of differences between the proposed and the existing system.

- Chapter 3 presents an overview of the proposed activity recognition system. The system consists of five components. This chapter briefly describes each of the components. The activity classifier, which is the most important component of the system, is discussed in detail with algorithms.

- Chapter 4 gives a detail description of the proposed web activity data mining engine. It first defines the types of web activity pages, and the goal of the mining engine. Then a detail description of the mining algorithm is provided. This chapter also presents how to train the activity classifier using web activity data. Finally, it describes the noises associated with mined data, the challenges and/or difficulties of removing these noises and the noise reduction techniques.

- Chapter 5 demonstrate the experimental results to support the claims. The objectives of the experiments are defined first. Then the experimental setup to validate the system is discussed. Finally, the experimental results are described.

- Finally, chapter 6 concludes the thesis with a direction for future work.

Chapter 2

Related work

In this chapter, first, different types of sensor-based activity recognition systems are briefly discussed. Second, some of the real-world activity data collection frameworks for activity recognition research are described. Third, the well-known temporal probabilistic models for activity recognition are discussed. Finally, a set of activity recognition systems that use either real-world or web activity data to train the classifier are described.

2.1 Types of activity recognition systems

Activity recognition systems can be categorized into three types based on the type of sensor they use:

1. audio/video-based activity recognition system,
2. wearable sensor (e.g. accelerometer) -based activity recognition system,
3. simple and ubiquitous sensors (embedded sensors) -based activity recognition system,

In a video-based activity recognition system [11–19], sequences of video frames obtained from one or more video cameras are used to determine the activity. In a wearable sensor-based activity recognition system [15,20–24], acceleration signals in three axes (x, y, and z) are used to recognize a user's activity. In a simple sensor-based (or embedded sensor-based) activity recognition system [5,6,10,25–28], an activity is recognized through a stream of sensory data acquired from different sensors.

All the activity recognition systems based on the aforementioned sensors demonstrate the excitement and need for activity recognition systems. However, using these sensors there are some limitations. For example, video-based activity recognition system has the limitations of breaking user's privacy, whereas wearable sensor-based activity recognition system requires the user to wear sensors and their accuracy depends on the position of the attachments. Although, embedded sensor-based activity recognition system do not have these sorts of limitations, but it can only recognize an activity, if it is performed interacting with one or more objects with embedded sensors.

2.2 Real-world activity data collection frameworks

In order to collect real-world activity data, smart homes are required to monitor the interaction between users and their home environment. This is achieved by distributing a number of ambient sensors throughout the subjects living environment. Expertise and resources are needed to design and install the sensors, controllers, network components, and middleware to perform basic data collections [38]. Additionally, an effective way is needed to annotate subject's activities in an automatic and easy way.

In this section, first, a set of smart homes that are developed to collect real-world activity data are described. Second, a set of annotation techniques that are used to label subject's activity are described.

2.2.1 Smart homes around the world

Many research groups from all over the world have been investigating how to construct smart living environments that target medical care to the individual [39]. This Bold House [40] is the beginning in the world's most accessible house contains easy-grip handles, flat thresholds, and adjustable-height vanities. The Duke Smart House [41] is a dynamic "living laboratory" environment. It contributes to the innovation and demonstration of future residential building technology. 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 [42]. Georgia Institute of Technology builds an Aware Home as a prototype for an intelligent space [43]. Massachusetts

Institute of Technology (MIT) and TIAX are working on the PlaceLab initiative, which is a part of the House_n project [44]. 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.

2.2.2 Experience sampling methods

Experience Sampling Method (ESM) [6, 29, 30] or ecological momentary assessment (EMA) is a common tool to study a user's behavior [45]. It was first designed to study the experiences in the wild [46]. It has now been applied to study diverse fields such as educational and clinical research [47]. The ESM has recently been employed for interface design [29, 48–52]. In an electronic ESM, a user usually carries a portable computing device with an ESM installed on it. The user is prompt with a set of questions after every pre-specified period of time and the user provides his/her answer by tapping one choice. An example of ESM is shown in Figure 2.1

ESM is one of the most important tools for activity recognition research. An activity recognition algorithm requires a set of real-world activity examples for training. In order to collect such examples, a volunteer is needed to stay in a smart home for a specific period of time and annotated their activity using an ESM. In recent years, a considerable amount of ESM-based tools have been developed for data collection associated with the activities.

To the best of our knowledge, the first ever computerized ESM was introduced in [54]. Barrett et al. have shown that it is possible to use a PDA to expand the repertoire of experience-sampling techniques and reduced or eliminated some traditional problems associated with the pen-and-paper methods. They installed an ESM to PDA, it prompts participants with a set of questions and the prompting lasts for 10 seconds, and the participant has 60 seconds to respond. If the participant responses within the time frame, the item responses and associated reaction times are recorded. Otherwise it cues the participant again 5 minutes later. If the participant again does not respond, the trial is recorded as missed. The working principle of the most of the ESM tools (developed so far) is similar to this one.

In [29], Intille et al. designed an ESM tool so that it suits both researcher and subject needs. Their goal was to handover the PDA (with ESM software installed on it) to the participant and

Figure 2.1: A screen shot of an ESM [53].

takes it back after the study period with all the experiences and corresponding sensor data. It has the capabilities for standard multiple choice question experience sampling using a time-sampled protocol. They also included the protocol development flexibility which is not common in other open-source ESM software. Additionally, it has the ability to add a new protocol by simply modifying a comma-separated value file. Moreover, it is possible to configure the software such that it allows participants to leave answers via audio recording or by taking a picture.

MyExperience [55] is the latest and the most popular open source context-aware data collection platform for capturing objective and subjective data as it's experienced. MyExperience collects both quantitative and qualitative data on human activity, attitudes and behaviors by combining both sensing and self-report. MyExperience can record a wide range of data including sensors, images, video, audio and user surveys. Sensor data is automatically recorded with timestamped to a local SQL Compact Edition database running on the mobile phone. The initial version ships with 50 built-in sensors including support for GPS, GSM-based motion sensors, and device usage information [56].

2.3 Temporal probabilistic models

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 [5,8,57], the Hidden Markov Model (HMM) in [6,28,58,59] and the Conditional Random Field (CRF) in [2,3,6,60,61]. In this section, some of the popular probabilistic models are briefly described.

Before formally defining the models, a few terms are needed to be defined. Let $A = \{a_1, a_2, ..., a_m\}$ be the set of activities, $O = \{o_1, o_2\ \ ..., o_t\}$ be the set of objects and $L = \{l_1, l_2, ..., l_q\}$ be the set of locations in an 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 set of object-usage (interacted object) at a given time, where, n is the total number of object-usage.

2.3.1 The Hidden Markov Models (HMMs)

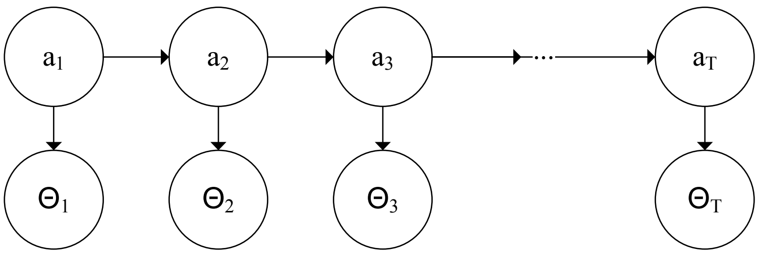


Figure 2.2: The Graphical model of a Hidden Markov Model.

HMM is a sequential model which is a probabilistic function of Markov chain as shown in Figure 2.2. It consists of a Hidden state (e.g. activity), a_t, and the observations (e.g. interacted objects), Θ_t, on each state. Hidden state at time, t, depends on the previous state at time, $t-1$. The observed variable at time, t, depends on the state at time, t. The goal is to find the joint probability distribution,

$$P(a,\theta) = \prod_{t=1}^{T} P(a_t|a_{t-1})P(\theta_t|a_t) \tag{2.1}$$

In HMM, a first-order Markov chain is used to generate a hidden state sequence. That is, given some probability of first state, a_1, and then given, a_1, second state, a_2, is generated, and so on. For each time, t, an output, Θ_t, is created, which is a function of state, a_t.

2.3.2 The Conditional Random Fields (CRFs)

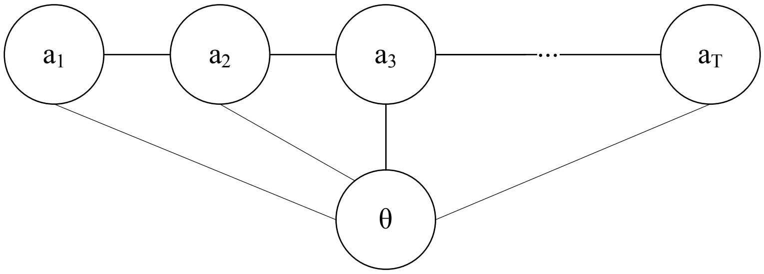


Figure 2.3: The Graphical model of a linear-chain Conditional Random Field.

A CRF is an undirected graphical model which models the conditional probability distribution over hidden states (e.g. activities) given the observations (e.g. interacted objects) [62]. Unlike HMM, a CRF is a discriminative probabilistic model, and makes no assumption that the observations are independent given the hidden state.

In a CRF, the cliques (the set of maximal fully connected subgraphs) play the key role in the conditional distribution. Let, C, be the set of all cliques in a given CRF. Then, a CRF factorizes the conditional distribution into a product of *clique potentials* $\phi_c(a_c,\Theta_c)$, where every $c \in C$ is a clique of and a_c and Θ_c are the observed and hidden nodes in such a clique. Clique potentials are functions that map variable configurations to nonnegative numbers. Using the click potentials, the conditional distribution can be written as,

$$P(a|\Theta) = \frac{1}{Z(\Theta)} \prod_{c \in C} \phi_c(a_c,\Theta_c) \tag{2.2}$$

where $Z(\Theta) = \sum_a \prod_{c \in C} \phi_c(a_c, \Theta_c)$ is the normalizing function which guarantees that the outcome is a probability.

Considering the clique potential as the log-linear combination of *feature functions*, $f_c()$, the Equation (2.2) can be written as,

$$P(a|\Theta) = \frac{1}{Z(\Theta)} \prod_{c \in C} exp\left\{W_c^T \cdot f_c(a_c, \Theta_c)\right\} \tag{2.3}$$

$$= \frac{1}{Z(\Theta)} exp\left\{\sum_{c \in C} W_c^T \cdot f_c(a_c, \Theta_c)\right\} \tag{2.4}$$

where W_c is the weight vector, $f_c(a_c, \Theta_c)$ is a function which will return 0 or 1 depending on the values of the input variable and therefore determines whether a click potential should be included in the calculation.

2.3.3 Naïve Bayes classifier

Studies comparing classification algorithms show that a simple Bayesian classifier known as the Naïve Bayesian (NB) classifier exhibits extremely good performance in various machine learning applications [63].

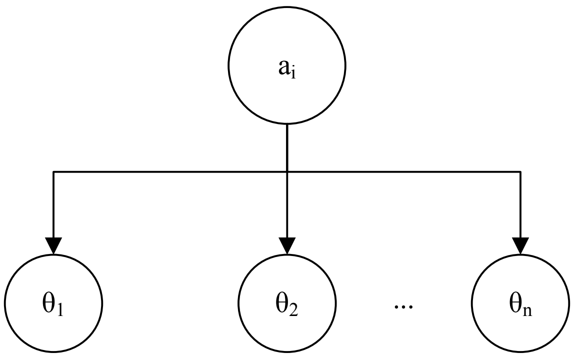


Figure 2.4: The Graphical model of a Naïve Bayes.

The NB-based activity classifier (shown in Figure 2.4) assumes that the effect of an object on a given activity is independent of the other object. This assumption is called activity conditional independence. For classification, the classifier computes the posterior probability, $P(A|\Theta)$, using the Bayes rule:

$$P(a_i|\Theta) \propto P(a_i) \prod_{k=1}^{|\Theta|} P(\theta_k|a_i) \tag{2.5}$$

where, $a_i \in A$ represents an activity (e.g. Bathing), A represents the set of activities and $P(a_i)$ is the *Prior Probability (PP)* of an activity. $P(\theta_k|a_i)$ is the likelihood of θ_k given an activity a_i, $\Theta \subset O$ is the vectors of activated sensors (as the subject interest with objects with embedded sensors) at a given time frame, O is the set of objects with embedded sensors.

During training the following probabilities are estimated:

$$P(\theta_1|a_i),\, P(\theta_2|a_i) \ldots P(\theta_k|a_i), P(a_i)\ \forall a_i \in A.$$

In order to classify the activity label of Θ, $P(a_i)P(\Theta|a_i)$ is evaluated for each activity a_i. The classifier predicts that the activity label of vector Θ is the activity a_i if and only if

$$P(a_i|\Theta) > P(a_j|\Theta)\ \ for\ 1 \leq j \leq m,\ j \neq i$$

where, m is the total number of activities. In other words, the classified activity label is the activity, a_i, for which, $P(a_i)$, is the maximum.

2.3.4 Naïve Bayes and smoothing

Parameter estimation plays the central role in the performance of a NB-based activity classifier. Data sparsity puts substantial challenges in parameter estimation because the sizes of the activity datasets are relatively small. The distribution of the sensors may not be even among the activity classes. Additionally, some of the sensors would appear during testing but would not appear while training, these are called *unseen sensors*. This is called *zero-frequency* [37] problem which assign *zero-probability* to an unseen sensor for a given activity. To prevent such estimation problem, *smoothing* is required to adjust the likelihood of a model to make it more accurate. At the very least, it is required to not assign zero probability to the unseen sensor. When estimating a likelihood based on a limited amount of sensors, such as a single activity instance, smoothing of the likelihood is extremely important.

A set of smoothing techniques have been proposed in the field of speech recognition (SR) and information retrieval (IR) [64]. The Jelinek-Mercer (JM) [65] (also referred to as the linear interpolation language model) and the Bayesian smoothing (BS) using Dirichlet priors [66] are two commonly used smoothing techniques used in IR to retrieve documents based on user's query. To the best of our knowledge, no smoothing techniques have been proposed in the field of AR. In this thesis, the JM based smoothing technique is adopted for activity recognition. In this section, various smoothing techniques proposed in the field of IR, are described.

2.3.4.1 Smoothing techniques in information retrieval

IR is the way to retrieve relevant documents based on the users query. In order to come up with a good query to retrieve the relevant documents, we need to think of the words (or terms) that would likely appear in these documents. In IR, the language modeling approach directly models this idea: If the document model is likely to generate a query, it will be a good match for the query, and it will happen if the document contains the query words often [37].

In other words, in the language modeling approach to IR, we can consider the probability of a query as being generated by a probabilistic model based on a document. For a query $q = q_1, q_2 \cdots, q_n$ and a document d, this probability is denoted by $p(q|d)$ [64]. In order to rank documents, the posterior probability $p(d|q)$ is estimated by the Bayes formula,

$$P(d|q) \propto P(q|d)P(d) \tag{2.6}$$

where $p(d)$ is the prior probability of a document for any query and $p(q|d)$ is the likelihood of the query given a document d. In IR, the $p(d)$ is considered to be uniform and therefore ignored. The likelihood, $p(q|d)$ is calculated as:

$$P(q|d) = \prod_{i=1}^{n} P(q_i|d) = \prod_{i=1}^{n} \frac{tf_{q_i,d}}{L_d} \tag{2.7}$$

where $tf_{q_i,d}$ is the *term frequency* of the term q_i in a document d, $L_d = \sum_{t\in d} tf_{t,d}$ is the length of the document and t is a term. This is called *query likelihood model* which is the original and basic method of language modeling in IR.

The classic problem of language modeling is one of estimation: The terms appear sparsely in the documents. In particular, if a query term q_i does not appear in the document then $P(q|d)$ will be 0. This is called *zero probability* estimation problem [37]. Such a problem leads to smooth probabilities in document language models to discount nonzero probabilities and to give some probability mass to unseen terms.

A wide variety of smoothing techniques have been proposed in the literatures. The Jelinek-Mercer (JM) [65] (also referred as the linear interpolation language model) and the Bayesian smoothing using Dirichlet priors (BS) [66] are two popular smoothing methods used in language models. The main idea behind these methods is to discount the probability of the words seen in the document and assign the extra probability mass to the unseen terms according to some “fallback” model.

Jelinek-Mercer (JM) smoothing: It is a simple idea but works extremely well in practice. It usages a mixture between a document-specific and entire collection-specific multinomial distribution:

$$P(t|d) = \lambda P_{mle}(t|M_d) + (1-\lambda)P_{mle}(t|M_c) \tag{2.8}$$

where $0 < \lambda < 1$ is the smoothing parameter and M_d and M_c are the language models derived from a document and from the entire document collection respectively.

Bayesian smoothing using Dirichlet priors (BS): An alternative of JM smoothing is to use a language model built from the whole collection as a prior Bayesian distribution in a Bayesian updating process. This is written as:

$$P(t|d) = \frac{tf_{t,d} + \mu P_{mle}(t|M_c)}{L_d + \mu} \tag{2.9}$$

where μ is the smoothing parameter. A large value of μ means more smoothing.

2.3.4.2 Other smoothing techniques in information retrieval

Laplace or additive smoothing [67] is the simplest smoothing method which works by adding an extra count to every term. The probability mass of a term, t, given a document, d, is calculated as,

$$P(t|d) = \frac{tf_{t,d} + 1}{L_d} \tag{2.10}$$

The problem of the Laplace smoothing is that it gives too much probability mass to unseen terms.

An improved smoothing method is the Good-turing smoothing [68] which re-estimate the frequency of the term t that occur tf times [69] as,

$$tf_t^* = (tf_t + 1)\frac{n_{tf_t+1}}{n_{tf_t}} \tag{2.11}$$

where n_{tf_t} is the number of terms that occur exactly tf_t times in the training data. Good-Turing is often used in combination with the backoff and interpolation algorithms rather than using it itself.

A more sophisticated smoothing technique known as Katz smoothing [70] extends Good-Turing estimation. The Katz smoothing method is a well known backoff method which works by discounting and redistributing probability mass only for the less common terms. Such a technique is popular in speech recognition.

Absolute discounting [71] is another smoothing method used in IR. The idea is similar to the interpolation method. It works by discounting the probability of seen terms by subtracting a constant instead of multiplying it.

2.4 Embedded sensor-based ARSs

Based on the type of training data an activity recognition system uses, the embedded sensor-based ARSs can be classified into two categories: train from environment (TFE) based activity recognition system that uses real-world activity data to train the classifier and train from Web (TFW) based activity recognition system that uses web activity data to train the classifier. In this section, some of the well known TFE-based and TFW-based ARSs are described.

2.4.1 Train From Environment (TFE)-based ARSs

A variety of simple and ubiquitous sensors based activity recognition systems have been proposed. For example, Tapia et al. [5] first employed such sensors for activity recognition. The authors provided the ESM in a PDA to the user to annotate their daily activities. NB-based 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 is 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 cannot 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.

2.4.2 Train From Web (TFW)-based ARSs

Perkowitz et al. [27] 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. Which in turn can be used in conjunction with *RFID* tags to recognize 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:

1. Select a set of websites like, `http://www.ehow.com/`, `http://www.epicurious.com/` that describes activities, and understands the HTML structure of such websites,
2. search for a page that describes an activity and extract the activity direction from this page,
3. set the title of the direction as the label of the activity,
4. parses and extract the object phrases from the direction,
5. removes the phrases that do not have noun sense,
6. 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.

7. finally filters 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 [28].

Wyatt et al. [28] 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:

1. 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 for the query.

2. 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 effectiveness of mining data is clearly related to z, the larger the value of z is the more effective the data would be. However, it will increase the mining time complexity.

3. It then determines $\tilde{P}$, a subset of P, as the activity pages using the genre classifier.

4. 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.

5. 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 $B_{ji} = p(object_i|activity_j)$.

This work is closely related to above two works. An activity recognition system using simple and ubiquitous sensors is developed, which is broadly applicable, and easy-to-use. This system also mine activity data from web to train its classifier. Despite these similarities, the proposed system has several differences which are summarized below:

1. It uses a high-accurate two-layer probabilistic classification integrating location and object-usage information. The classifier uses location-and-object-usage based model in the first-layer to classify a group of activities and object-usage based model in the second-layer to classify the actual activity.

2. The web activity data mining algorithm is efficient and simple. At first a parameter estimation model using web activity data is designed. It is efficiently implemented using advance operators of a search engine.

3. The proposed system is highly scalable. It is possible to add new activity or new object by simply giving activity name or the object (with embedded sensor) name. In other words, it has the ability to handle growing amounts of activities and objects in a graceful manner.

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

2.5 Summary

In this chapter, first, smart homes, data collection frameworks and essential tools that are developed for pursuing activity recognition research are discussed. Second, the state-of-art probabilistic methods for activity recognition are described. Third, the related work on embedded sensor based activity recognition systems that use real-world and web activity data is described. The differences between the existing and the proposed system are outlined. In the next chapter, the proposed activity recognition system is described.

Chapter 3

Activity recognition system using web activity data

This chapter presents the proposed activity recognition system. First, the general concept of each of the components of the system is described. Second, the detailed descriptions of each of the major components of the system are given.

3.1 Overview

A scalable and easy-to-use activity recognition system is developed that can recognize a large number of activities concerning different environments. Given a set of activities to monitor, object names and their corresponding locations, the proposed system mines the activity data from the web and stores them into repository. It then estimates the *location-usage* and *object-usage* likelihoods from the mined data. Once all the likelihoods are estimated, the system is ready to recognize activities in real-time. The overview of EARWD is shown in Figure 3.1.

The EARWD consists of five components:

1. The *environment* (e.g. home), in which the system will be deployed, consists of a number of appliances (or objects) such as, door, cabinet and desk. The strategy is to embed sensors to these objects to facilitate the activity recognition system in determining the state of that object when a person interacts with it.

2. The core of the system is the *Activity Classifier (AC)* which classifies (or infers) an activity based on a set of object-usage (interaction with an object) information, at a given time. The classifier is learned using the activity data mined from web. It uses a Naïve Bayes-based (NB) two-layer classifier to classify activities.

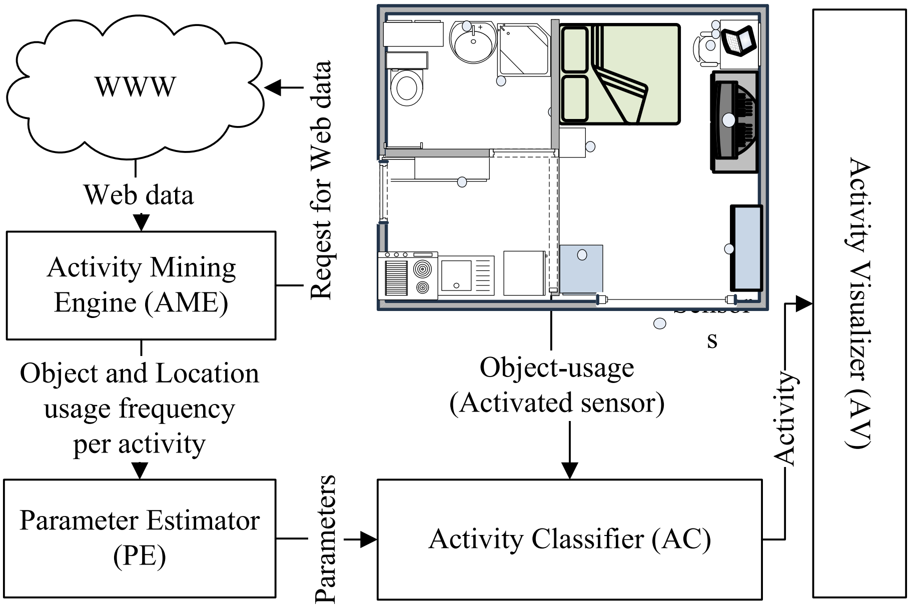


Figure 3.1: An overview of activity recognition system.

3. The *Activity Mining Engine (AME)* mines activity data from the web. It takes the following external inputs: a set of activities to monitor, object names with attached sensors and their corresponding locations. It gives object-usage and location-usage information for a given activity as output, such that the system can compute the model parameters (i.e. likelihoods).

4. The *Parameter Estimator* learns the model parameters and coefficients using the activity data mined by the mining engine.

5. The *Visualizing tool* is used to provide Graphical User Interface (GUI) to monitor the day-to-day activities. This is a web-based tool that shows detail (e.g. Activity label, Object used, date/time) of an activity such that an authorized person can access a secured website where he/she can scan a check-list.

3.2 Activity classifier

The activity classifier first classifies a group of activities (e.g. kitchen activity) using a Location-and-Object-usage based model (LOBM) and then subsequently, classifies the actual activity (e.g. cooking) from that group using an object-usage based model. In this section, the need for adopting

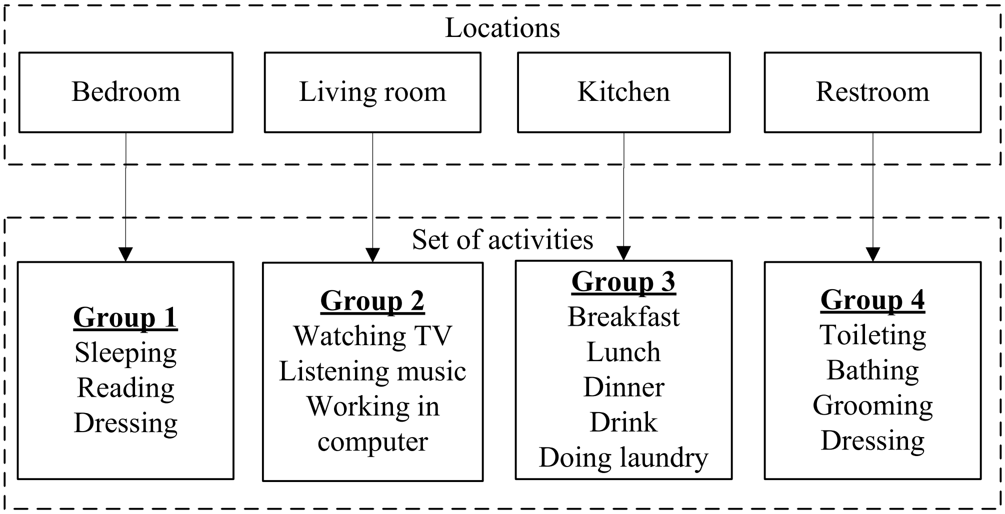


Figure 3.2: An example of location specific activities.

two activity models and two-layer classifier is justified first. Second, the activity models and the classification methods in each layer are described.

Most of the ACs [6, 27, 28, 72] utilize only the OBM to classify activities. As the number of activities to be monitored grow, the number of distinguishing objects between activities decreases. Hence, the downside of such approach is that they would produce more confusion between activities. Such systems would produce more confusion between activities. Therefore, only the OBMs would not be enough for 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 [34, 35]. 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 Figure 3.2).

A two-layer classifier is proposed, 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. The object information along with location information is used in the first-layer to resolve any location-confusion. A subject (or user) may switch among locations while performing 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 limits the system's ability to recognize the true activity group. Therefore, the EARWD 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 could be possible to design a one-layer classifier that uses object information along with the location information to classify an activity. However, if such a classifier is designed 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), might not have any obvious location. It could be performed in kitchen, foyer and bathroom. Using a one-layer classifier, this activity might not be well classified. The reason is, as activity information is mined from web, the probability of a location given an activity with no obvious location would be relatively low compared to other activities in that location. For example, the probability of using "kitchen" for "doing laundry" will be low with respect to those of other kitchen activities (e.g., dinner). For such activities with no specific location, both 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 the influence of such object(s) is reduced for an activity, the probability of classifying that activity will also be reduced.

Consider the following scenario that would help us for understanding why activities with no specific location may not be well classified using only location-and-object-usage based model. Let us consider that there is a probabilistic model that uses half of location-usage probability and half of object-usage probability. For the sake of simplicity, let us assume that there are two kitchen activities, "dinner" and "doing laundry". Let the probability of using "door" and "laundry dryer" be, 0.069 and 0.0032, and the probability of using "kitchen" be 0.51 for "doing laundry". Similarly, let the probability of using "door" and "laundry dryer" be respectively 0.063 and 0.000001, and the probability of using "kitchen" be 0.94 for "dinner".

Let us now assume that "doing laundry" is performed in "kitchen" using two objects, "door" and "laundry dryer". If the above model is used with a Naive Bayes classifier, the probability of "doing laundry" would be, $(1/2*0.069+1/2*0.51)*(1/2*0.0032+1/2*0.51) = 0.0742857$

and the probability of "dinner" in "kitchen" would be, $(1/2 * 0.063 + 1/2 * 0.94) * (1/2 * 0.000001 + 1/2 * 0.94) = 0.235705251$. Finally, the outcome of the classifier will be "dinner", since it shows comparatively higher probability. Although the object-usage probabilities are high for "doing laundry", it is not classified appropriately.

In order to overcome such situation, a two-layer classifier is used, 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.

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, the groups are constructed manually using user's preference of an activity/location and the environment. There are some activities that could be performed in multiple locations depending on the requirements of the activity. For example, the activity, "dressing" could be performed in "bedroom" and or it could be performed in "bathroom". Additionally, there are some activities having no specific location. For example, the activity, "cleaning home", does not have any specific location. In order to deal with such situations, the system allows the user to put one activity in multiple groups.

3.2.1 The goal of the classifier

Let $A = \{a_1, a_2, ..., a_m\}$ be the set of activities, $O = \{o_1, o_2 \; ..., o_t\}$ be the set of objects and $L = \left\{l_1, l_2, ..., l_q\right\}$ be the set of locations in an 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 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 object-usage. The goal is to map the observation sequence (i.e. object-usage sequence, Θ) into corresponding activity labels.

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

Figure 3.3 shows an overview of the two-layer classifier and an example is illustrated in Figure 3.4.

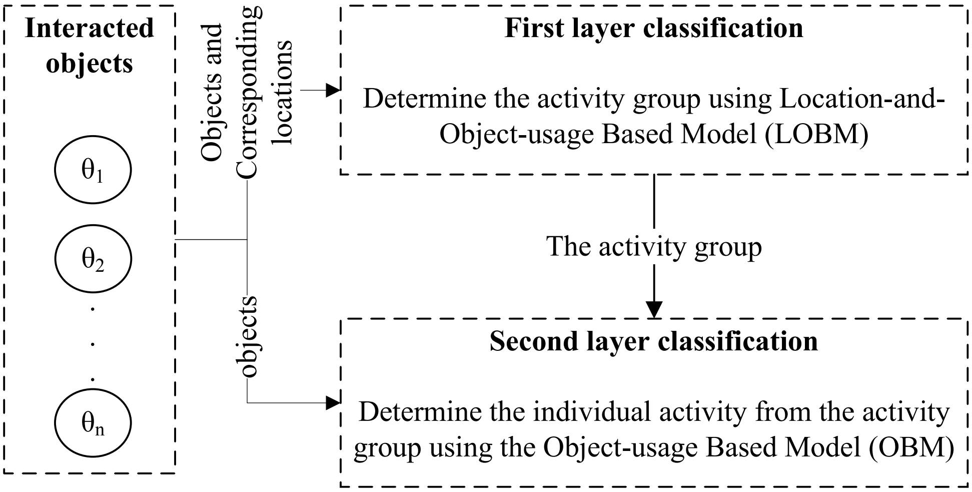


Figure 3.3: An overview of activity classification.

3.2.2 Activity models

The activity models are Naïve Bayes-based probabilistic models.

3.2.2.1 Location-and-Object-usage Based Model (LOBM)

Definition 3.1 *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.*

$$P_{LOBM}(A_j|\Theta) \propto \prod_{k=1}^{|\Theta|}(\alpha P(l_{\theta_k}|A_j) + (1-\alpha)P(\theta_k|A_j)) \tag{3.1}$$

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.*

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.

The probabilities estimation technique for this model and defining the value of α are described in Section 3.4.

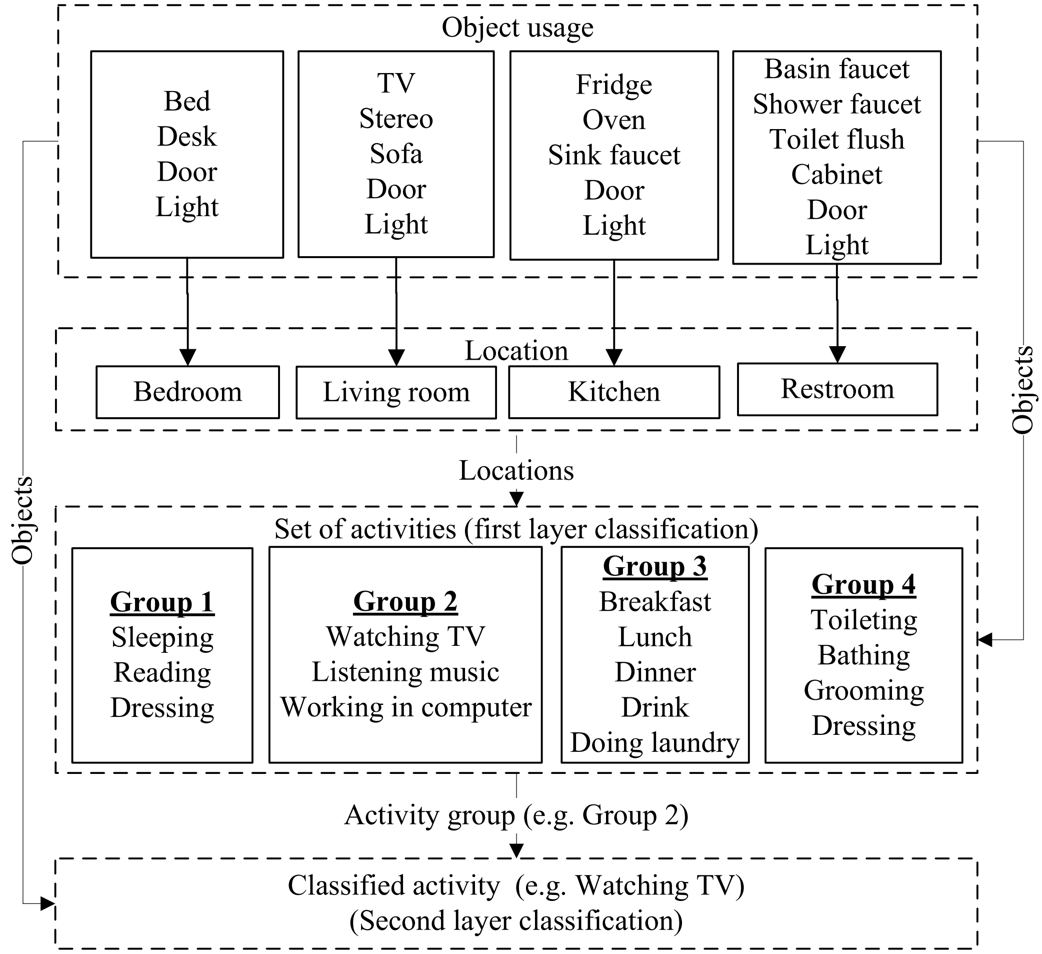


Figure 3.4: Two-layer activity classification: An example for the activity "Watching TV".

3.2.2.2 Object-usage Based Model (OBM)

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, a smoothing technique is proposed which is based on the Jelinek-Mercer (JM) [65] (described in Chapter 2) smoothing technique used in Information Retrieval.

Table 3.1: An example dataset: Each cell represents how many times an object is used for an activity.)

Activity \ Sensors	o_1	o_2	o_3	o_4	o_5	...	o_n
Leaving	2	17	21	1	5	...	2
Toileting	1	40	195	0	0	...	16
Showering	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

Before defining the smoothing method, it is required to define following two terms.

Definition 3.2 *An Activity Model (AM) =* $\{v_1, v_2, ..., v_n\}$ *is an observation vector of n number of objects for an activity, where,* v_i*, being the observed frequency of* i^{th} *object for an activity.*

Definition 3.3 *A Collective Model (CM) =* $\{AM_1, AM_2, ..., AM_m\}$ *is a collection of observation vectors of m number of activities, where,* AM_i*, being the activity model for* i^{th} *activity.*

Let us consider an activity dataset shown in Table 3.1. Each cell of the table represents the number of times t^{th} object, o_t, is used for an activity. Each row of this table represents an AM, M_{a_i}, for i^{th} activity, a_i. Entire table represents the CM, M_c, for the activity collection, c.

Definition 3.4 *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:*

$$P_{OBM}(a_i|\Theta) \propto P(a_i) \prod_{k=1}^{|\Theta|} (\lambda P(\theta_k|M_{a_i}) + (1-\lambda)P(\theta_k|M_c)) \tag{3.2}$$

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 equation (3.2), $0 < \lambda < 1$ 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. The value of λ can be set either to maximize the average performance of the classifier or to the average number of zero-frequencies in a dataset.

The probabilities estimation technique for this model and defining the value of λ are described in Section 3.4.

3.2.3 Classification of an activity group

In the first-layer, the classifier uses a NB-based method to classify a group of activities from a set activities based on the LOBM (shown in Equation (3.1)). 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

$$P_{LOBM}(A_i|\Theta) > P_{LOBM}(A_j|\Theta) \;\; for\ 1 \leq j \leq q, j \neq i \tag{3.3}$$

where, q, is the total number of activity groups.

3.2.4 Classification of an individual activity

Similarly, in the second-layer, the classifier uses a NB-based method to classify the individual activity from the activity group, A_j (output of the first-layer classifier), using only on the OBM (shown in Equation (3.2)). 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

$$P(a_i)P_{OBM}(a_i|\Theta) > P(a_j)P_{OBM}(a_j|\Theta) \;\; for\ 1 \leq j \leq g, j \neq i \tag{3.4}$$

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

3.3 Activity mining engine

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)*. An EACP explicitly describe the steps required to perform an activity. In contrast, an IACP provides some steps that are related to an activity or it would influence an activity without directly specifying how to perform an activity.

Most of the activity mining engines [27, 28] proposed so far only mine information from EACPs. The mining engine proposed in this thesis considers both types (i.e. EACP and IACP) of pages. The reason is that even though an IACP does not provide steps to perform an activity, however, it provides activity related information. Therefore, the purpose of activity mining engine is to mine activity data from all relevant web pages. It provides object-usage and location-usage information for a given activity as output, which are used to compute the model parameters.

In this thesis, the total number of IACPs and EACPs in the web is termed as, activity pages indexed (API). The number of pages within these APIs that contains an object name is termed as, object pages indexed (OPI). Similarly, the number of pages within these APIs that contains a location name is termed as, location pages indexed (LPI).

The existing activity mining engines need to load (from web) a set of these pages (EACPs, IACPs) to classify the EACPs and to mine activity data (API, OPI and LPI). On the other hand, the mining engine proposed in this thesis doesn't need to load these pages, but instead, it employs the search engines (already mine information from these pages) for this purpose. Employing the search engines for this purpose makes the mining system exceptionally fast. The details about the mining engine are provided in next chapter.

3.4 Parameter estimator

The main task of the parameter estimator (PE) is to transform the activity data into likelihoods such that the classifier can use them to classify activities. Additionally, the PE estimates the coefficients associated with the models.

3.4.1 Estimation of likelihoods

The PE uses the following formulas to calculate the likelihoods:

$$P(o_j|A_i) = \frac{\sum_{a_k \in A_i} freq(o_j|a_k)}{\sum_{a_k \in A_i,\, o_c \in O} freq(o_c|a_k)} \tag{3.5}$$

$$P(l_j|A_i) = \frac{\sum_{a_k \in A_i} freq(l_j|a_k)}{\sum_{a_k \in A_i,\, l_c \in L} freq(l_c|a_k)} \tag{3.6}$$

$$P(o_j|M_{a_i}) = \frac{freq(o_j|a_i)}{\sum_{o_c \in O} freq(o_c|a_i)} \tag{3.7}$$

$$P(o_j|M_c) = \frac{\sum_{a_i \in A} freq(o_j|M_{a_i})}{\sum_{a_k \in A,\, o_c \in O} freq(o_c|M_{a_k})} \tag{3.8}$$

where, $freq(o_j|a_k)$ and $freq(l_j|a_k)$ are the relative frequencies of object, o_j, and location, l_j in a dataset for a given activity, a_k.

3.4.2 Estimation of prior probability

If the classifier is trained using real world activity data, then the prior probability, $P(a_i)$ of an activity, a_i, is estimated as,

$$P(a_i) = \frac{freq(a_i)}{\sum_{a_j \in A} freq(a_j)} \tag{3.9}$$

However, if the classifier is trained using web activity data, $P(a_i)$ is estimated based on the external input. The frequency of pursuing an activity in a time frame (e.g., two showers per day) is the input, which is then converted into per minute to get the prior probability. It is to be noted that there is no suitable way (not yet at least) to mine and estimate prior probabilities from the web because it is *highly subject dependent*. If it is considered to be, $P(a_i) = \frac{API(a_i)}{\sum_{a_j \in A} API(a_j)}$, which might not be an accurate measurement, because it would be biased to the number of pages indexed by the search engines. 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 $<<$ then “Bathing”. In real life, frequency of “Toileting” is lot more than “Bathing”. On the other hand, if some existing datasets are used to compute the prior probabilities, it would limit the applicability and scalability of the system. This is because the degree of pursuing an activity is totally depended on the user.

3.4.3 Estimation of coefficients

In equation (3.1), the coefficient, $0 < \alpha < 1$, is used to control the influence of location. It is needed to estimate how much influence would be optimal (or nearly optimal) for a given dataset. In other words, it is required to calculate the importance of the locations for all the activity groups. If the sum of average number of times the locations appeared in the activity documents is calculated, it would give the importance. Therefore, it is calculated as,

$$\alpha = \frac{\sum_{i=1}^{q} \frac{\sum_{a_k \in A_i, l_c \in L} freq(l_c|a_k)}{\sum_{a_k \in A_i} freq(a_k)}}{q} \tag{3.10}$$

Where, $freq(a_k) = \sum_{o_j \in O} freq(o_j|a_k) + \sum_{l_c \in L} freq(l_c|a_k)$ and q is the number of activity groups.

In equation (3.2), the coefficient, $0 < \lambda < 1$, is used 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 a dataset has, the more smoothing is required. Therefore, if λ can be calculated as the distance of the average of the average number of objects with zero-frequencies in each activity from 1, it will give us the optimal (or nearly optimal) λ. That is,

$$\lambda = 1 - \frac{\sum_{a_i \in A} \frac{\sum_{o_c \in O} \delta(freq(o_c|a_i))}{t}}{m}, \left\{ \delta = \begin{cases} 1 \ if \ freq(o_c|a_i) == 0 \\ 0 \ otherwise \end{cases} \right. \tag{3.11}$$

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

3.5 Visualizing tool

The *Visualizing tool* is a web-based tool that shows in detail (e.g. Activity label, Object used, date/time, etc.) of an activity such that an authenticated person can access a secured website where he/she can scan a check-list. Figure 3.5 shows an example of output of the visualizing tool. It represents the activity log of Mr. John for a day (Friday, May 22, 2009). It consists of four

Activity log of Mr. John for Friday, May 22, 2009

Activity	Start time	End time	Objects
Toileting	07:12am	07:28am	Click to see
Breakfast	07:29am	07:50am	Click to see
Taking medication	07:51am	07:52am	Click to see
Reading newspaper	07:53am	08:20am	Click to see
Bathing	08:21am	08:38am	Click to see
Dressing	08:39am	08:50am	Click to see
Going out	08:51am	12:02pm	Click to see
Idle	12:03pm	12:07pm	Click to see
Toileting	12:07pm	12:15pm	Click to see
Lunch	12:16pm	12:55pm	Click to see
Taking medication	12:56pm	12:57pm	Click to see
Watching TV	12:58pm	01:22pm	Click to see
Resting	01:23pm	03:30pm	Click to see
Toileting	03:31pm	03:38pm	Click to see
Going out	03:39pm	05:58pm	Click to see
Toileting	05:59pm	06:07pm	Click to see
Cooking	06:08pm	07:12pm	Click to see
Dinner	07:13pm	07:30pm	Click to see
Watching TV	07:31pm	08:45pm	Click to see

Figure 3.5: An example of the output of visualizing tool.

columns: the name of the activity, the start-time, end-time of the activity and a link to a page in which the set of object-usage for that activity will be found.

3.6 The summary of the terms and the algorithms

This section summarizes the terms and equations of this chapter. It also discuss the learning and inference algorithms.

3.6.1 The summary of the terms

Let $A = \{a_1, a_2, ..., a_m\}$ be the set of activities, $O = \{o_1, o_2 \; ..., o_t\}$ be the set of objects and $L = \{l_1, l_2, ..., l_q\}$ be the set of locations in an 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 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 object-usage. The goal is to map the observation sequence (i.e.

object-usage sequence, Θ) into predefined activity labels.

In the first-layer, the classifier classifies a group of activities, $A_j \in A$, using the LOBM shown below:

$$P_{LOBM}(A_j|\Theta) \propto \prod_{k=1}^{|\Theta|}(\alpha P(l_{\theta_k}|A_j) + (1-\alpha)P(\theta_k|A_j)) \tag{3.12}$$

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.

The individual activity, $a_i \in A_j$, is classified in second-layer using the OBM shown below:

$$P_{OBM}(a_i|\Theta) \propto P(a_i)\prod_{k=1}^{|\Theta|}(\lambda P(\theta_k|M_{a_i}) + (1-\lambda)P(\theta_k|M_c)) \tag{3.13}$$

During the training the model parameters are estimated as,

$$P(\theta_j|A_i) = \frac{\sum_{a_k \in A_i} freq(\theta_j|a_k)}{\sum_{a_k \in A_i,\, o_c \in O} freq(o_c|a_k)} \tag{3.14}$$

$$P(l_j|A_i) = \frac{\sum_{a_k \in A_i} freq(l_{\theta_j}|a_k)}{\sum_{a_k \in A_i,\, l_c \in L} freq(l_c|a_k)} \tag{3.15}$$

$$P(\theta_j|M_{a_i}) = \frac{freq(\theta_j|a_i)}{\sum_{o_c \in O} freq(o_c|a_i)} \tag{3.16}$$

$$P(\theta_j|M_c) = \frac{\sum_{a_i \in A} freq(\theta_j|M_{a_i})}{\sum_{a_k \in A,\, o_c \in O} freq(o_c|M_{a_k})} \tag{3.17}$$

$$\alpha = \frac{\sum_{i=1}^{q} \frac{\sum_{a_k \in A_i, l_c \in L} freq(l_c|a_k)}{\sum_{a_k \in A_i} freq(a_k)}}{q} \tag{3.18}$$

$$\lambda = 1 - \frac{\sum_{a_i \in A} \frac{\sum_{o_c \in O} \delta(freq(o_c|a_i))}{t}}{m}, \left\{ \delta = \begin{cases} 1 \ if \ freq(o_c|a_i) == 0 \\ 0 \ otherwise \end{cases} \right. \tag{3.19}$$

where, $freq(o_j|a_k)$ and $freq(l_j|a_k)$ are the relative frequencies of object, o_j, and location, l_j in a dataset for a given activity, a_k, q is the number of activity groups, m and t are the number of activities and objects respectively.

3.6.2 The algorithm for classification

Algorithm 3.1: ActivityClassifier(A, Θ, $P(l|A)$, $P(\theta|A)$, α, λ). The activity classifier.

Data: List of activities, A, List of object-usage, Θ, at a given time, $P(l|A)$, $P(\theta|A)$, α, λ
Result: An activity, a

```
/* Location-and-object-usage based first-layer classifier */;
```

A_j = FirstLayerClassifier(A, Θ, $P(l|A)$, $P(\theta|A)$, λ);

```
/* Object-usage based second-layer classifier */;
```

a = SecondLayerClassifier(A_j, Θ, $P(\theta|A)$, α);

The system would be ready to recognize activity in real time, as soon as all the model parameters have been estimated. This is called inference phase. In this phase, activities are inferred by an inference engine (or classifier) which uses sensory data coming from the objects (as human interact with the objects with embedded sensors).

The classifier (or the inference engine) is shown in Algorithm 3.1. It takes following inputs: list of activities A, list of object-usage, Θ, at a given time, $P(l|A)$, $P(\theta|A)$, α and λ. It classifies the most probable activity as output in two-layers.

In the first-layer, the classifier infers a group of activities, $A_j \in A$, using Equation 3.12. The corresponding algorithms is shown in Algorithm 3.2. It takes following inputs: A, Θ, $P(l|A)$, $P(\theta|A)$, α and provides an activity group, A_j, as output.

Algorithm 3.2: FirstLayerClssifier(A, Θ, $P(l|A)$, $P(\theta|A)$, α). The activity group classifier.

Data: List of activity groups, A, List of object-usage, Θ, List of location-usage probabilities, $P(l|A)$, list of object-usage probabilities, $P(\theta|A)$, and α

Result: An activity group, A_g

$A_g = A_1$; `/* First-layer's output */`

for $j = 1$ *to* q **do**

  `/*` q `is the total number of activity groups in` A `*/;`

  $P_{LOBM}(A_j|\Theta) = 0$;

  for *each object-usage,* $\theta_k \in \Theta$ **do**

    if $P_{LOBM}(A_j|\Theta) > 0$ **then**

      $P_{LOBM}(A_j|\Theta) = P_{LOBM}(A_j|\Theta) * (\alpha P(l_{\theta_k}|A_j) + (1-\alpha)P(\theta_k|A_j))$;

    else

      $P_{LOBM}(A_j|\Theta) = \alpha P(l_{\theta_k}|A_j) + (1-\alpha)P(\theta_k|A_j)$;

    end

  end

  if $j > 1$ *AND* $P_{LOBM}(A_j|\Theta) > P_{LOBM}(A_{j-1}|\Theta)$ **then**

    $A_g = A_j$;

  end

end

Algorithm 3.3: SecondLayerClssifier(A, Θ, $P(\theta|A)$, λ). The activity classifier.

Data: List of activities, A, List of object-usage, Θ, list of object-usage probabilities, $P(\theta|A)$, and λ

Result: An activity, a

$a = a_1$; `/* Second-layer's output */`

for $i = 1$ *to* $length(A)$ **do**

  `/*` $length(A)$ `is the total number of activities in` A `*/;`

  $P_{OBM}(a_i|\Theta) = 0$;

  for *each object-usage,* $\theta_k \in \Theta$ **do**

    if $P_{OBM}(a_i|\Theta) > 0$ **then**

      $P_{OBM}(a_i|\Theta) = P_{OBM}(a_i|\Theta) * (\lambda P(\theta_k|M_{a_i}) + (1-\lambda)P(\theta_k|M_c))$;

    else

      $P_{OBM}(a_i|\Theta) = \lambda P(\theta_k|M_{a_i}) + (1-\lambda)P(\theta_k|M_c)$;

    end

  end

  if $i > 1$ *AND* $P_{OBM}(a_i|\Theta) > P_{OBM}(a_{i-1}|\Theta)$ **then**

    $a = a_i$;

  end

end

In the second-layer, the classifier infers the actual activity, $a_i \in A$, using the equation 3.13. The corresponding algorithms is shown in Algorithm 3.3. It takes following inputs: A, Θ, $P(\theta|A)$, λ and provides an activity, a_i, as output.

3.7 Summary

In this chapter, first, an overview of the system is briefly described. Second, importance of location information along with object information for activity recognition is described. Third, a two-layer classifier is proposed, in which, the first-layer classifies the group of activities (e.g., kitchen activities) using the location-and-object-usage based model, and the second-layer classifies the individual activity (e.g., cooking) within the activity group using the object-usage based model. Fourth, the characteristics of the different types of activity pages available in web are described. Finally, the mechanism to mine web activity data from these pages using the search engines, and the mechanism to learn the model parameters are described.

Chapter 4

Activity mining engine

4.1 Introduction

Advancement of the Internet and the World Wide Web (WWW) encourages millions of users to promote billions of web pages with varieties of contents [33]. A fraction of these pages either explicitly or implicitly provide information related to the activities of daily livings. Such web pages not only state an activity but also depict where to perform this activity and what objects to use and in what sequence. An activity recognition system would be broadly applicable and scalable by its very design if it can mine information from such web pages to learn the models. The purpose of activity mining engine is to provide enough information such that the system can use these for training. Before describing any further how activity mining engine works, it would be useful to define the *web activity data mining* first.

Definition 4.1 *The web activity data mining can be defined as a type of web mining technique, under the category of recourse finding and/or extracting. More specifically, activity data mining is the process of retrieving (either on-line or off-line) the data related to human activities available in the web and the process transforming the above data into a form such that it can be used to train an activity classifier.*

Therefore, the goal of the activity mining engine is to mine web activity data, such that these can be used to train the system. Before explaining the activity mining engine, it is essential to provide some facts related to the web pages that describe activities of daily livings.

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 4.1: An EACP that provides information related to object-usage for an activity [73].

4.1.1 Types of activity pages in the 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 4.2 *Explicit Activity Catalog Page (EACP): A web page is called an Explicit Activity Catalog Page (EACP) if it provides instructions in detail, like 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 sequence and location-usage for that activity.*

For example, the web page[1] [73] shown in Figure 4.1 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 4.3 *Implicit Activity Catalog Page (IACP): A web page is called an Implicit Activity Catalog Page (IACP) if it does not directly define 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.*

[1] `http://www.articlealley.com/article_108210_28.html`

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 4.2: An IACP that provides information related to object-usage for an activity [74].

For example, the web page[2] [74] shown in Figure 4.2 is an IACP that describes the list of steps required to make bathing safe 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

[2] `http://ezinearticles.com/?How-to-Make-Bathing-Safe-For-Independent-Seniors&id=2148355`

Table 4.1: 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

steps required for bathing but it provide information that is related to bathing.

The web-based activity recognition systems proposed in the literature, only consider EACP as the source of activity information. However, IACPs also provide information that is extremely helpful for activity recognition. In this research, both of these pages are considered to be the source of activity data.

4.1.2 Goals of the mining engine

We can see from Equations (3.1) and (3.2) that during training, it is required 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 web activity data such that these can be used by the PE to estimate the likelihoods.

In other words, the goals of an AME is to find the EACPs and IACPs 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 4.1.

All the mining engines proposed in the literature so far need to load a set of pages to determine the set of EACPs. It makes the mining engines extremely time consuming. Therefore, the motivation is to develop a fast mining engine that will mine activity data from web without loading any pages.

4.2 Mining web activity data

The proposed mining engine is based on the popular web search engines available today. A web search engine is developed to find a piece of information on the web. If we search with a specific term(s) or sentence(s), the search engine will provide information that is related to the term(s)

Table 4.2: Search engine's Query modifiers and operators.

Name	Description
“”	The quotes forces the search engine 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, a search engine 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.
intext	Is the Google search syntax for searching only in the body text of documents and ignoring links, anchors, URLs, and titles [75]. That is, if we include [intext:] in our query, a search engine would return all the web pages containing the word in the *body* text of the web pages. For instance, the query [intext:“Washing machine”] would find all the web pages that have “Washing machine” in their body text.
inbody	Is the Yahoo and Bing search syntax, which is same as “intext”.

or sentence(s). The information may consist of web pages, images, information and other types of files (e.g. pdf or doc). Some search engines also mine data available in databases or open directories [76].

The AME uses the existing search engines (e.g. Google, Yahoo and Bing) to search such pages and determines the PCs. Searching using a search engine is simple, choosing the appropriate search terms is the key to find the required information [77]. Most of the search engines support a bunch of advanced operators, which are query words and have special meaning for the engine. It is possible to modify the search in some way, or even instruct the search engine for a different search [78]. 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 4.2 shows the modifiers and operators that are used to mine the web activity data.

Figure 4.3 shows how the AME mines activity data from the web using a search engine. The corresponding algorithm is shown in Algorithm 4.1. For each activity, $a_i \in A$, the AME first searches the number of potential pages that describe a_i, using a query, $intitle$:“a_i”$intext$:“a_i. Let the set of *activity pages indexed (API)* by the search engine be Ω for a given query. The cardinality of Ω is denoted by $n = |\Omega|$. Next step is to determine the number of pages indexed by the search engine for a location *(or location pages indexed (LPI))* $l_j \in L$ within the activity

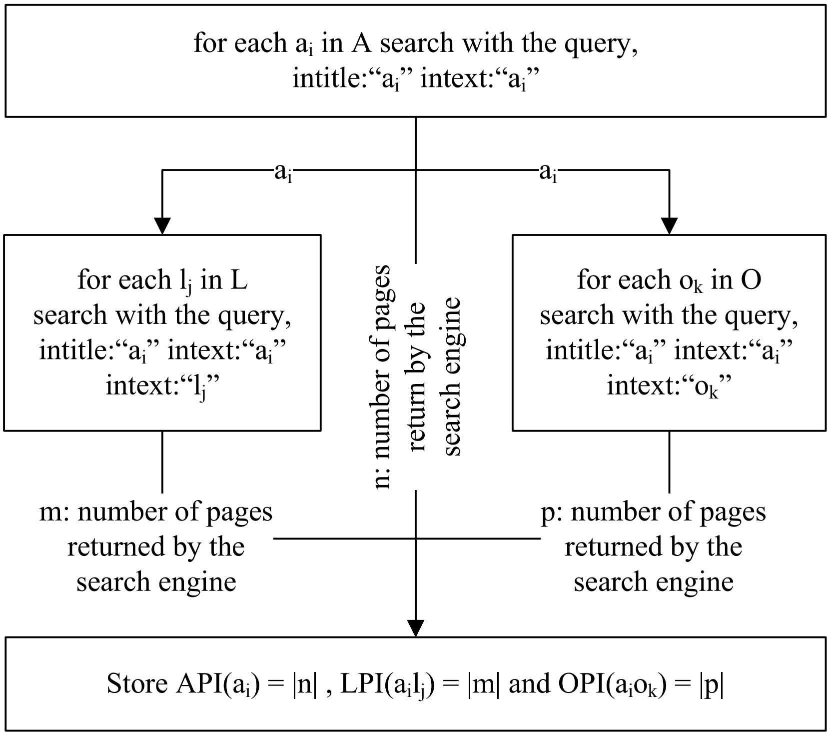


Figure 4.3: Activity Mining Engine.

Algorithm 4.1: AME(A, O, L). The activity mining engine.

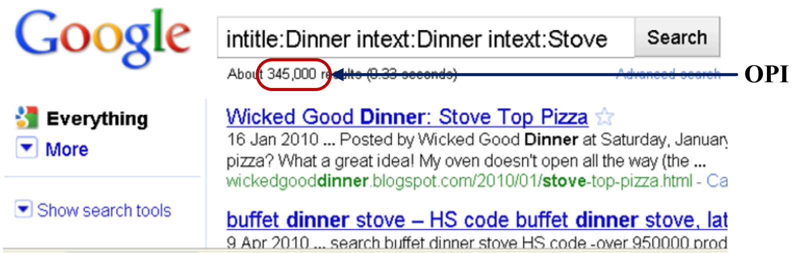
Data: List of activities A, List of objects O, List of locations L
Result: Activity Pages Indexed (API), Location Pages Indexed (LPI) and Object Pages Indexed (OPI)

for $i \leftarrow 1$ **to** $length(A)$ **do**
  API_i = `this` ←SG("*intitle* :"a_i"*intext* :"a_i""); `/* SE (Search Engine) would return the number of pages indexed by the search engine for the given query */;`
  for $j \leftarrow 1$ **to** $length(L)$ **do**
    LPI_{ij} = `this` ←SE("*intitle* :"a_i"*intext* :"a_i" *intext* :"l_j"");
  end
  for $k \leftarrow 1$ **to** $length(O)$ **do**
    OPI_{ik} = `this` ←SE("*intitle* :"a_i"*intext* :"a_i" *intext* :"o_k"");
  end
end

pages. The AME uses the query, *intitle* :"a_i"*intext* :"a_i *intext* :"l_j" (for Bing and Yahoo AME uses *intitle* :"a_i" *inbody* :"a_i *inbody* :"l_j") to return the number of pages containing the l_j in their text for a given activity pages. Let the search engine return $m \subseteq \Omega$ pages that contain an occurrence of l_j. Similarly, let $p \subseteq \Omega$ be the pages returned by the search engine for an object *(or object pages indexed (OPI))*, if the AME searches with the query, $intitle$:"a_i" $intext$:"a_i $intext$:"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. In Figure 4.4, some examples of API, OPI and LPI are shown.

a. an example of Activity Page Indexed (API).

b. an example of Location Page Indexed (LPI)



c. an example of Object Page Indexed (OPI)

Figure 4.4: Mining examples using Google (the search engine).

4.3 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 web activity data is:

$$r = m + m(q+t); \tag{4.1}$$

where, m, t, and q are the total number of activities, objects, and locations respectively. As we can see in Algorithm 4.1, for m activities, the AME requires m queries to mine *APIs*, for q locations and m activities, the AME requires mq queries to mine *LPIs*, and for t objects and m activities, the AME 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 AME would need 260 queries in total.

4.4 Training the system using web activity data

In section 3.4, the purpose of the parameter estimator is to estimate likelihoods and the coefficients from the training data is described. In this section, how the parameter estimator estimates them from mined data is shown.

4.4.1 Estimation of likelihoods

In order for better understanding, the likelihood estimation equations (as shown in Equations 3.5 - 3.8) are rephrased here.

$$P(o_j|A_i) = \frac{\sum_{a_k \in A_i} OPI(o_j|a_k)}{\sum_{a_k \in A_i,\, o_c \in O} OPI(o_c|a_k)} \tag{4.2}$$

$$P(l_j|A_i) = \frac{\sum_{a_k \in A_i} LPI(l_j|a_k)}{\sum_{a_k \in A_i,\, l_c \in L} LPI(l_c|a_k)} \tag{4.3}$$

$$P(o_j|M_{a_i}) = \frac{OPI(o_j|a_i)}{\sum_{o_c \in O} OPI(o_c|a_i)} \tag{4.4}$$

$$P(o_j|M_c) = \frac{\sum_{a_i \in A} OPI(o_j|M_{a_i})}{\sum_{a_k \in A,\, o_c \in O} OPI(o_c|M_{a_k})} \tag{4.5}$$

where, $OPI(o_j|a_k)$ and $LPI(l_j|a_k)$ are the relative frequencies of an object, o_j, and a location, l_j in a dataset for a given activity, a_k.

4.4.2 Estimation of coefficients

In order for better understanding, the coefficient estimation equations (as shown in Equations 3.10 and 3.11) are rephrased here.

$$\alpha = \frac{\sum_{i=1}^{q} \frac{\sum_{a_k \in A_i,\, l_c \in L} LPI(l_c|a_k)}{\sum_{a_k \in A_i} (\sum_{l_c \in L} LPI(l_c|a_k) + \sum_{o_j \in O} OPI(o_j|a_k))}}{q} \tag{4.6}$$

where, q is the number of activity groups.

$$\lambda = \frac{\sum_{a_i \in A} \frac{\sum_{o_c \in O} \delta(OPI(o_c|a_i))}{t}}{m}, \left\{ \delta = \begin{cases} 1 & if\ OPI(o_c|a_i) == 0 \\ 0 & otherwise \end{cases} \right. \tag{4.7}$$

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

4.5 Noise associated with the mined data

There could be much noise associated with the mined data, since the mining technique is based on the search engines. In other words, as the mining is based on syntactic meaning rather than semantic meaning, there will be various types of noises:

- Noises associated with the activity title: There could be a page that is not an activity page (i.e. EACP) nor a relevant activity page (i.e. IACP), but has an activity name in its title. Such page can be categorized into two types:
  - Invalid activity page: Although the activity name appears in the document's title, it is not guaranteed that only the activity manual page contains such title.
  - Inappropriate title: Although the activity name appears in the document's title and the page is an activity manual page, it is not guaranteed that the information is related to the activity we are looking for.
- Noises associated with the object or location usage: A set of activity manual pages may contain unexpected (noisy) object or location usage information. Such pages could be categorized into three types:
  - Different activity context: There could be activity pages that provide object or location usage information whose context is different from the expected.
  - Irrelevant object or location usage information: A web page typically contains many information blocks. Apart from the main content blocks, it could have blocks like, navigation panels, copyright and privacy notices, and advertisements. Such blocks may provide irrelevant object or location usage information for an activity.
  - Object names with different meanings: There could be many object names with multiple meanings. For instance, the object "Pan" has at least two meanings: cooking utensil consisting of a wide metal vessel, and commode pans. Activity pages may contain such objects which are different from the object we are searching for.

In the following subsections, a brief description of these noises is given.

4.5.1 Noises associated with the activity title

There can be a page having activity name on its title which is neither an activity page nor a page providing relevant information. Such a page could be termed as invalid activity page.

4.5.1.1 Invalid activity page

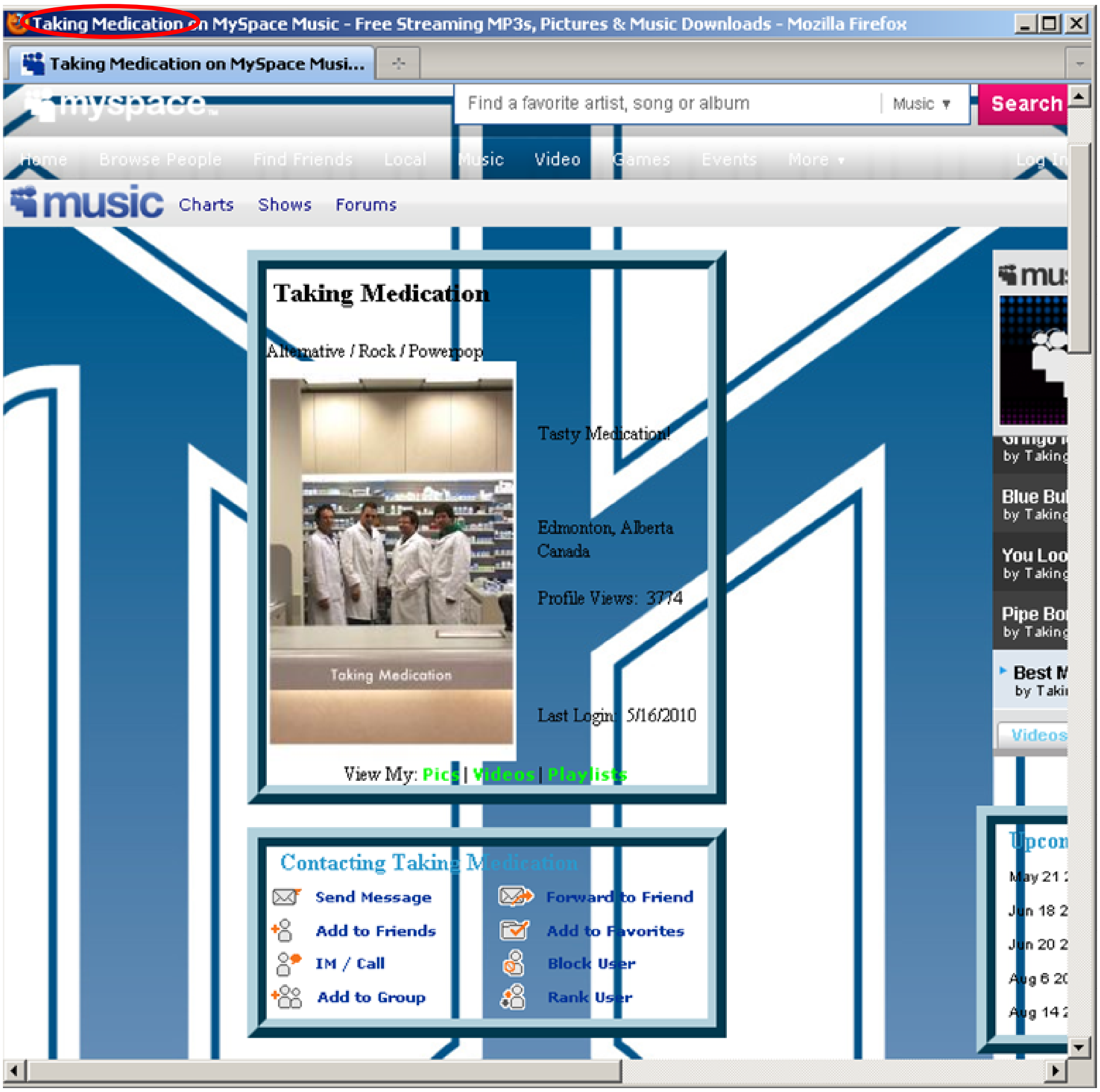


Figure 4.5: An example: A noisy web page [79], contains a noisy activity name in the title.

The mining engine uses the keyword, "intitle:" to determine the set of activity pages. Using such a keyword to find the activity pages, would only return the pages that have the activity name on their title. It is not guaranteed that only the activity manual pages (EACP or IACP) will be returned. For example, as shown in Figure 4.5, the web page is not an activity page, even though it has an activity name (i.e. taking medication) in the title.

4.5.1.2 Inappropriate title

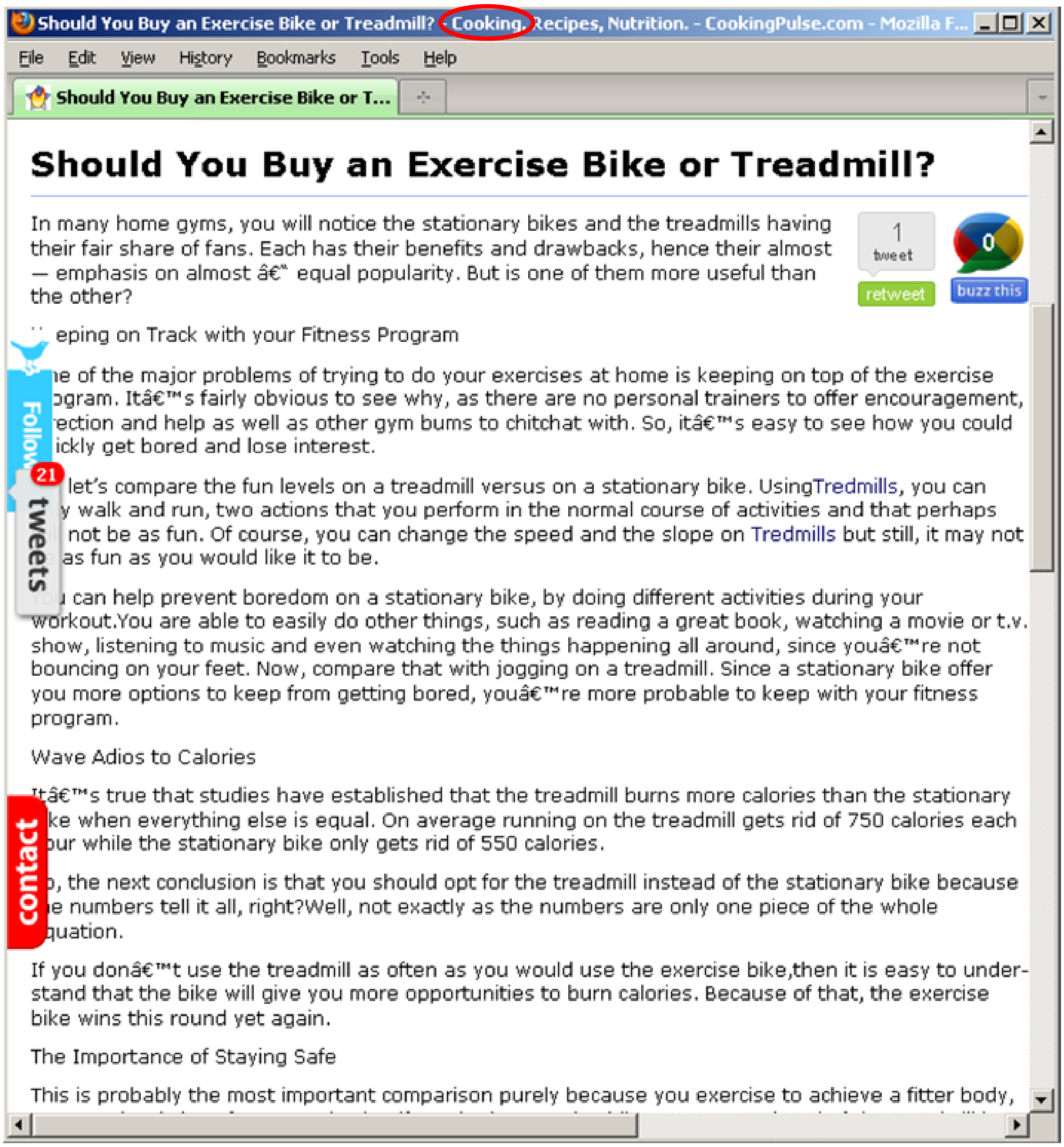


Figure 4.6: An example: A noisy web page [80], contains a different activity name in the title.

Although the activity name appears in the document title and the page is related to an activity it is not guaranteed that the information is related to the activity we are looking for. For example, as shown in Figure 4.6, the title of this page contains cooking but body does not contain the instruction related to “cooking” instead it has information related to “exercise”. If the search is performed to get the activity pages related to cooking, this page would appear.

4.5.2 Noises associated with the object or location usage

A set of activity manual pages may contain unexpected (noisy) object or location usage information. Such pages could be categorized in three types:different activity context, irrelevant object or location usage information and Object/location names with different meanings.

4.5.2.1 Different activity context

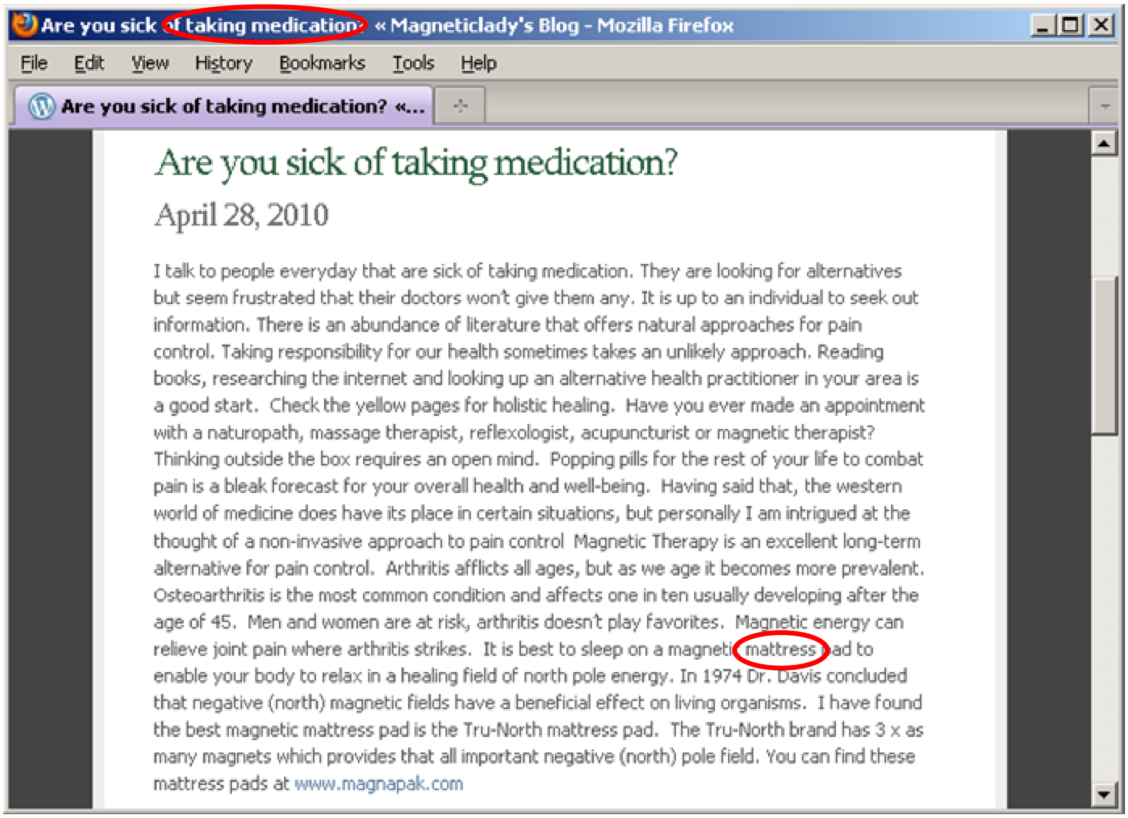


Figure 4.7: An example: A noisy web page [81] contains a noisy activity name in the title and an object name in the text.

There are pages that provide activity information but the context of the pages may be different from the expected, for example, as shown in Figure 4.7, the page is something related to “taking medication" while the context is different. It talks about alternative information of "taking medication". However, it has object-usage information like, "mattress".

4.5.2.2 Irrelevant object or location usage information

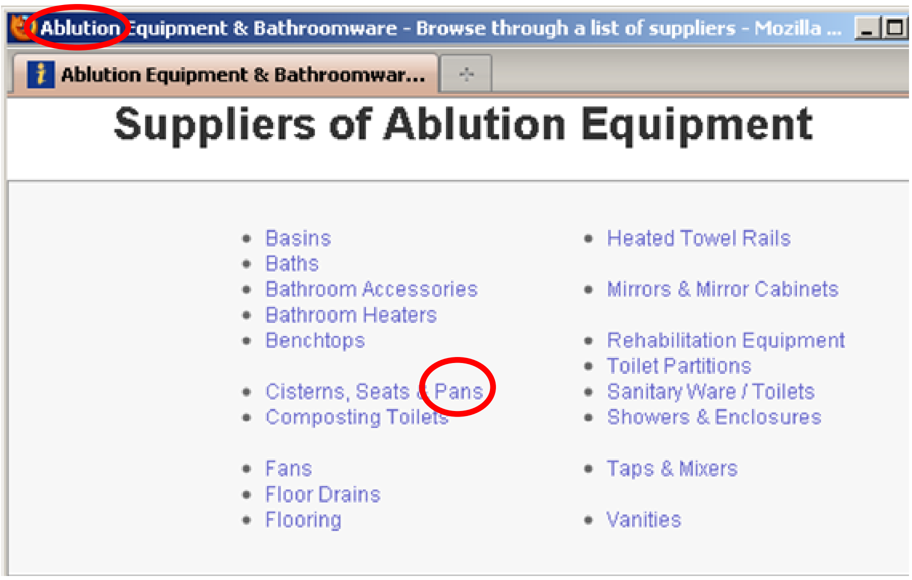
Figure 4.8: An example: A noisy web page [82], contains a activity name in the title and a noisy object name in the navigation panels.

A web page typically contains many information blocks. Apart from the main content blocks, it could have blocks like, navigation panels, copyright and privacy notices, and advertisements. Such blocks may provide irrelevant object or location usage information for an activity. For example, as shown in Figure 4.8, the page is something related to "taking medication" in the navigation panel it has object-usage information like, "TV".

4.5.2.3 Object/location names with different meanings

There could be object/location names with multiple meanings. Activity pages may contain such objects which is different to the object we are searching for. For instance, as shown in Figure 4.9, the object, "Pan", has at least two meanings: cooking utensil consisting of a wide metal vessel and commode pans. If the object, "Pan" (in kitchen, as cooking utensil) is embedded with sensors and information are mined based on this then the page shown in 4.9 (b) is noisy.

a. An example: Pans as cooking utensil



b. An example: Pans as ablution utensil

Figure 4.9: An example: An object with different meanings a. Pans as cooking utensil [83], b. Pans as ablution utensil [84].

4.6 Challenges and difficulties to remove noises

4.6.1 Identifying the true activity pages

In order to identify a true activity page, it may not be sufficient to find only the pages that mention the activity name in their title, but rather the subset of those pages that contain detailed descriptions of the activity being performed. Identifying this subset of the pages is a problem known as genre classification [28]. Therefore, the first thing we need is the genre classifier. However, there are several disadvantages of using a genre classifier for this:

1. It is required to load each of the pages (returned by a search engine) to determine the genre of the page. This is not realistic, since there could be millions of pages. Loading each of these pages would take huge amount of time. For example, let us consider we need to classify the genre of the pages that contain “cooking” in their title. We searched for the potential set of pages using the query, “intitle:cooking”, the search engine returned around 8 millions of pages. This is not realistic to load and classify the genre of these pages, since it could be very expensive or could take months.

2. The accuracy of the genre classifier should be very high, because the higher the accuracy is the lower the noises would be.

4.6.2 Identifying the object-usage and location-usage

After identifying the true activity pages, the next task is to identify the terms that denote objects and locations from these pages. For this purpose we need a parts-of-speech tagger (POS tagger or POST) to identify the nouns and an object (or a location) tagger to identify the objects (or locations) from these nouns. However, the accuracy of such tagger should also be very high.

4.7 Reducing the effect of mining noises

Rather than eliminating the noises during mining, it is more appropriate to reduce the consequences of such noises while training with the mined data, since it will not impact mining time. The reduction is performed in following two ways.

4.7.1 Reducing the effect of activity title related noises

To reduce the effect of the noises associated with the activity title, only the pages that contain at least one object/location name in their text are considered to be the true activity pages. Such a technique removes the unwanted pages that have inappropriate/irrelevant title.

That is, to estimate the conditional probability, $\sum_{o_c \in O} OPI(o_c|a_i)$ and $\sum_{a_k \in A,\, o_c \in O} OPI(o_c|M_{a_k})$ are used instead of $API(a_i)$ and $\sum_{a_k \in A_i} API(a_k)$ respectively, as shown in Equations 4.4 and 4.5.

For example, let "Preparing Breakfast", "Preparing dinner" be two activities, and "Fridge", "Oven" be two objects. After mining the web activity data, 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 parameter estimator considers only the activity pages to be the true pages if it contains either a specified object or location. Such a technique removes most of the pages that are not activity pages or not related activity pages.

4.7.2 Reducing the effect of object/location usage noise

The noises associated with the object and location usage will not impact the activity recognition accuracy significantly, because such noises will be relatively low. However, to reduce the effect of such noise, the proposed classifier combines the location and object usage information together in a model. Such a combination reduces the influence of each other (location and object). That is, even though the noise is associated with the object-usage information, by using location-usage information with it, reduces the influence of object-usage noise and vice versa. The noise will only be significant if both object and location usage information are noisy. Experimental results are provided to validate the claim.

4.8 Summary

In this chapter, a brief description of the activity mining engine is given. The activity mining engine uses the advanced search engine's operators to mine activity information from the web. Using such techniques reduce the mining time dramatically. However, it also introduces noises to the mined data. The description of such noises and what are the challenges and difficulties to remove such noises is also provided. Finally, a method is proposed to reduce the effect of the mining noise.

Chapter 5

Experimental results and analysis

The objective of this chapter is to validate the performance of the EARWD. Four experiments are performed to validate the system's performance: First, the efficiency of mining method is verified by checking the likelihoods estimated by the parameter estimator (PE) with the help of the activity mining engine (AME). Second, the classifier's performance in classifying activities of three datasets is evaluated. Third, the impact of the coefficients (α and λ) in activity classification is analyzed, and evaluate the proposed methods for estimating these coefficients. Finally, the comparison results of different classifiers and different mining engines are shown.

The chapter is organized as follows, in section 5.1, a procedure of acquiring a real-world activity dataset is described. In section 5.2, the setup for mining web activity data and for evaluating system's performance is explained. In section 5.3-5.6, the results of four experiments are provided. In section 5.7, important issues associated with system are discussed.

5.1 A framework for real-world activity data collection

Activity recognition algorithms require a large variety of real-world activity datasets for evaluating the algorithm. In order to collect such data, smart homes are developed which can monitor the interaction between users and their home environment. This is achieved by distributing a number of ambient sensors throughout the subject's living environment. There is a tremendous amount of overhead in constructing such a testbed [38]. Expertise and resources are needed to design and install the sensors, controllers, network components, and middleware just to perform basic data collections. Therefore, acquiring real-world activity datasets is expensive. Another difficulty in promoting the use of such systems is to find effective ways to annotate subject's activities in an automatic and easy way. As a result, very few physical testbeds exist. In the cases where real sensor data have been collected and analyzed, only rarely is this data made available to the research community.

Moreover, the range and variety of such datasets is limited to only one or two environments. In other words, the diversity of such datasets is extremely limited. In activity recognition research, it is important to validate the performance of the system with a diverse set of real-world activity examples. The algorithm tested with the datasets acquired from varieties of environment would be robust in general. It would therefore show superior performance (irrespective of the environment) in comparison with the algorithms validated with the datasets acquired from a single environment.

Additionally, the datasets acquired in real-world environment are relatively small. There are a variety of reasons for the datasets being smaller: 1) it is difficult to find volunteers who would stay in a testbed and annotate their own activities, 2) even if such volunteers are found, they are required to stay there for long time to generate a reasonable amount of data. For example, in order to have 100 instances of an activity, participant(s) might need to stay in the testbed for 100 days, 3) the volunteers have to be focused all the time for accurate annotation of an activity 4) it could be very expensive to get a large dataset.

Thus it is desirable to have a data collection method that is inexpensive, flexible, and user-friendly but is capable of providing large and diverse activity datasets. In this thesis, a solution is proposed to this problem by implementing a data collection tool which is inexpensive but is capable of providing large variety of activity datasets. The proposed tool is web-based and can be used to create a replica of any home environment, according to the system's requirements. It can collect data inexpensively as long as the users want.

5.1.1 Setup a virtual environment for data collection

In order to setup an activity recognition environment, first, it is required to define the set of activities to be annotated. It is then require choosing objects that can be embedded with sensors. In order to add activities and objects, two interfaces are developed which are simple and therefore could be used by a user with little computer's knowledge. In the following two subsections, two interfaces are described.

5.1.1.1 Add the activities to monitor

The first step is to add the number of activities to monitor. In this step a user is prompt with a set of activities and locations using the interface shown in Figure 5.1. A user is expected to choose an activity and their corresponding location(s) (e.g. the room(s) where an activity is usually performed).

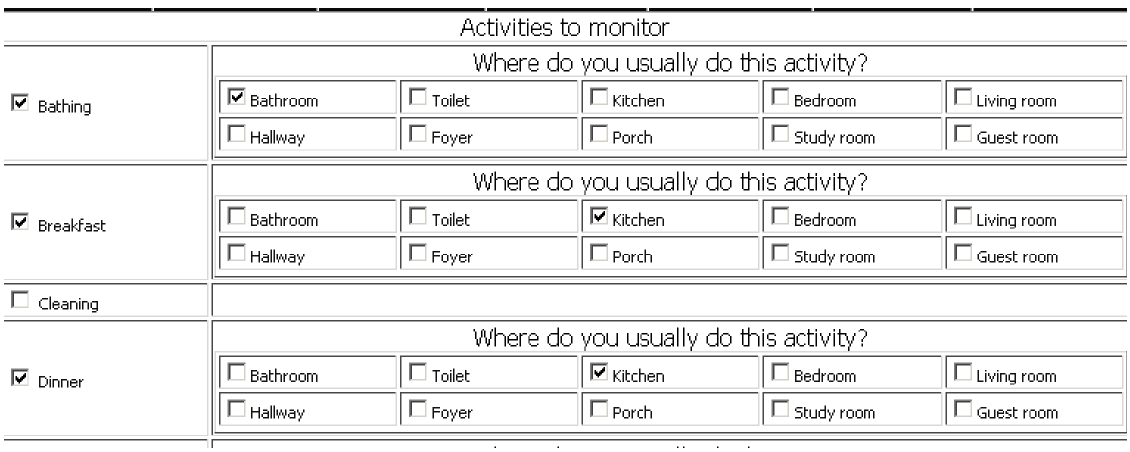


Figure 5.1: Add the set of activities to monitor.

5.1.1.2 Add the set of objects

After acquiring the activities information, the next step is to add the set of key objects (i.e. most important objects) that a person have in the environment. The key objects are the objects which are most frequently used for doing an activity. Only the key objects should be chosen since it would be very expensive to embed sensors to all the objects in an environment.

In this step a user is prompt with a set of the key objects given a location through the interface shown in Figure 5.2. The set of locations shown in this interface are the locations which are chosen during the activity addition step (referring Figure 5.1). A user is expected to select one or more objects per locations.

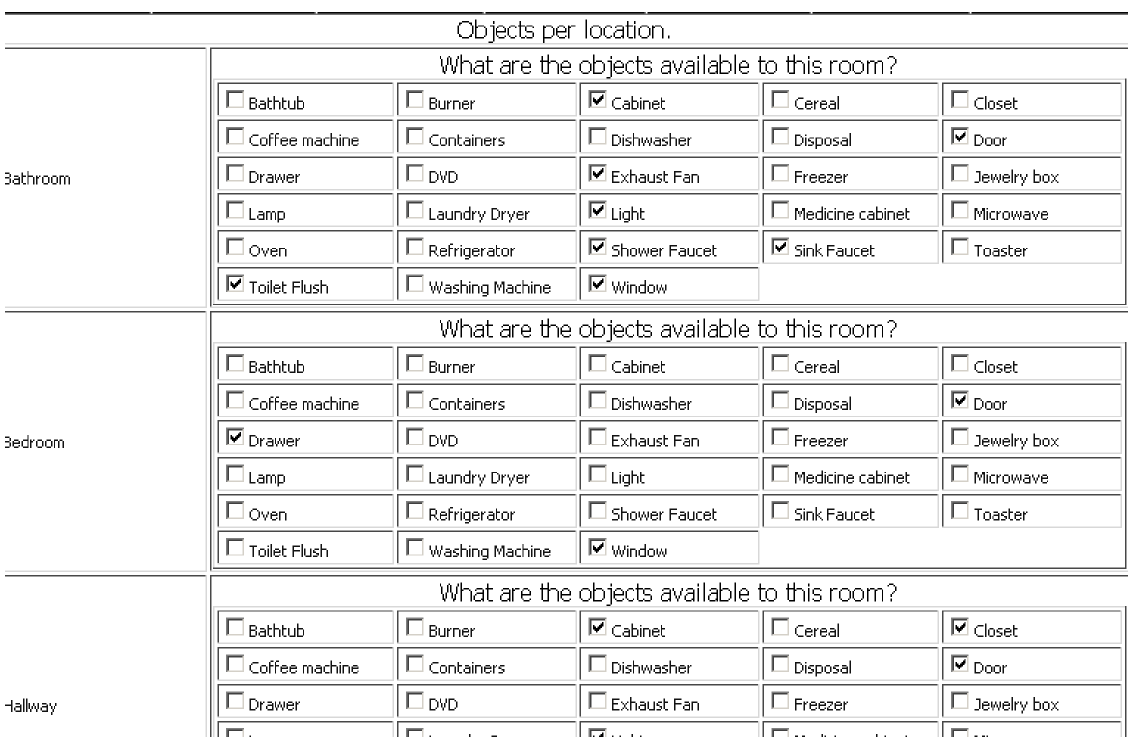


Figure 5.2: Select a set of objects to embed sensors.

5.1.2 Web-based Experience Sampling Method

In this thesis a web-based inexpensive ESM is proposed to collect a set of large dataset from different environments and therefore, this research refrained using any special device that may not available in all environments. Additionally, the web-based solution makes the data collection procedure ubiquitous.

The screen shot of the ESM is shown in Figure 5.3. The ESM consists of a combo box (as shown at the left hand side of the Figure 5.3) to which participant can select the activity he/she performed. The participant can choose the duration of the activity from two combo boxes (i.e. Hour and Minutes). Upon selection of the duration from the combo boxes, the two text boxes (i.e. Start time and End time) will be filled automatically, however, participant can modify the “Start time” and “End time” too. In addition to these, participant is expected to provide the set of object he/she uses along with the time interval.

The set of objects are the objects available to the location(s) to which the selected activity is

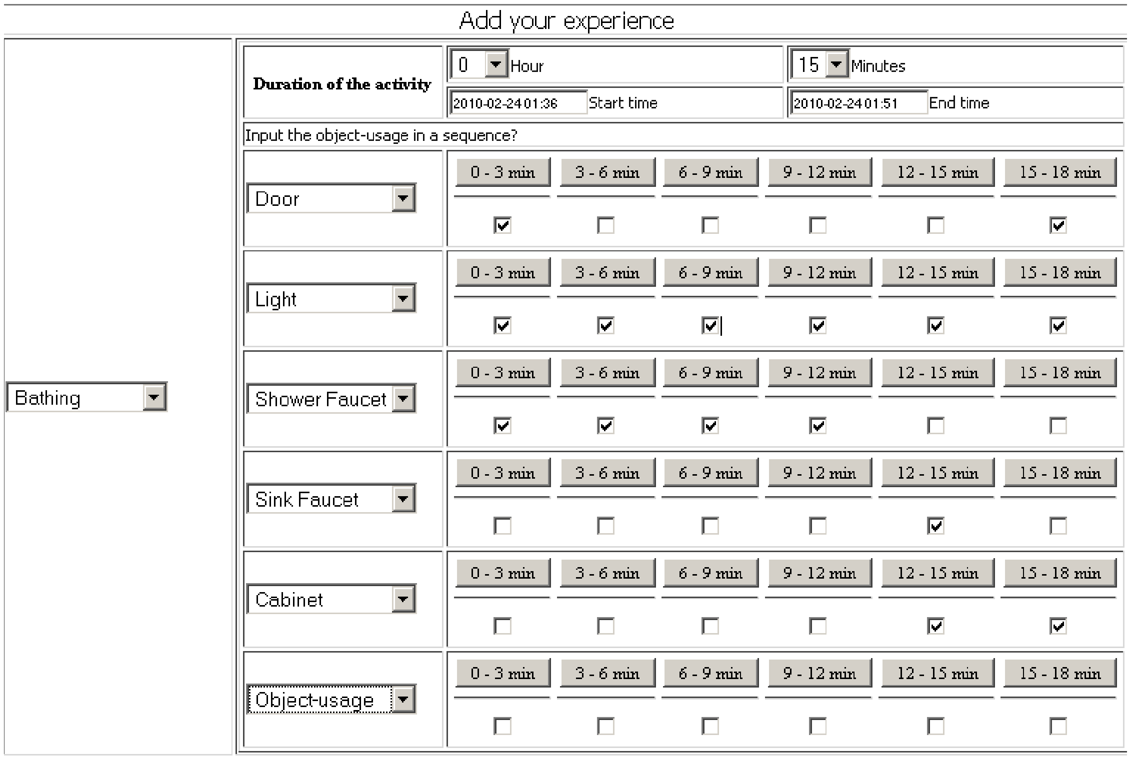


Figure 5.3: The experience sampling tool.

performed. Therefore, the set of objects is subject to automatically changed when the selected activity is changed. After the selection of an activity, only one row of the object-usage will be visible. The next row will only be visible if the participant chooses an object from the combo box and the next will be shown if he/she select another object and so on. The maximum number of object-usage would be the total number of objects available to the location to which the activity is performed. In this way, the redundant stuffs are made invisible from the participants.

There are 6 check boxes per object-usage through which the participant can provide the object-usage interval. The corresponding 6 buttons are used to label the time intervals. To reduce the participant's effort, the number of intervals is decided to be 6. The label of the buttons (e.g. 0-3 min, 3-6 min) is automatically changed based on the selected duration. For example, if the participant provides 10 minutes as the duration of the activity, the labels of the buttons will be 0 - 2, 2-4, ..., 10 - 12.

After providing all the information, participants can add his/her experience. The ESM stores all the given object-usage and their corresponding intervals. Additionally, it generates and stores random noises. The ESM would store an additional object-usage that is not on the list as the random noise. The random noise makes the data as close as possible to real-world. In real-world activity scenario, it is possible to use one or more objects inadvertently (e.g. living room's light is on while toileting).

Although it is possible to add experiences anytime (e.g. at the end of each day) the participant wants. But it is expected that participant(s) will add his or her experience at the end of an activity. The ESM tool will beep every 10 minutes to remind the participant to input his or her activity if finished. The reason to choose such technique is that as the object-usage sequence are taken from participants, it would be best to add activities at the end such that he or she can provide a complete picture of the object-usage sequence for that activity. It also reduces the participant's effort in annotating their experience.

5.2 Experimental setup

Before presenting the analysis, the experimental setup is described. Two sorts of setup are required to mine web activity data and to evaluate the performance of the system, which are described in this section.

5.2.1 Setup for mining

As described earlier, the AME uses three search engines: Google, Yahoo and Bing to mine web activity data.

To mine activity data using Google, 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/. For example, to mine the API for "Cooking", the AME will send a query as, `http://ajax.googleapis.com/ajax/services/search/web?v=1.0&q=Cooking`. In response, Google will return a page that would contain the formatted results like, the estimatedResultCount (i.e. the number of pages indexed by Google), 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.

Similarly, to mine activity data using Yahoo, AME uses the site, `http://boss.yahooapis.com/ysearch/web/v1/`. For example, to mine the API for "Cooking", the AME will send a query as, `http://boss.yahooapis.com/ysearch/web/v1/intitle:Cooking?appid=<apiid>&format=xml&start=1&count=1`. In response, Yahoo will return a XML page that would contain the field, <resultset_web...>, in which there is a parameter, totalhits, which is the number of pages indexed by Yahoo.

Similarly, to mine activity data using Bing, AME uses the site, `http://api.search.live.net/xml.aspx`. For example, to mine the API for "Cooking", the AME will send a query as, `http://api.search.live.net/xml.aspx?Appid=<apiid>&query=intitle:Cooking&sources=web+image`. In response, Bing will return a XML page that contains the field, <web:Total>, which is the number of pages indexed by Bing.

It is to be noted here that Google, Yahoo and Bing would not allow automated search using their original sites.

5.2.2 Setup for evaluating system's performance

Three datasets have been used to evaluate the system's performance, which are described in this section. Additionally, the formulas to measure the activity recognition accuracy are described.

5.2.2.1 The datasets

To evaluate the performance of EARWD, two real-worlds and a semi real-world datasets have been used. Real-worlds datasets are gathered by Tapia et al. [5] in MIT PlaceLab (it is called PlaceLab dataset), and by Kasteren et al. [6] at Intelligent Systems Lab Amsterdam (ISLA) (it is called ISLA dataset). The semi real-world dataset is gathered by using the tool described in previous section.

Tapia et al. utilized 77 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. In

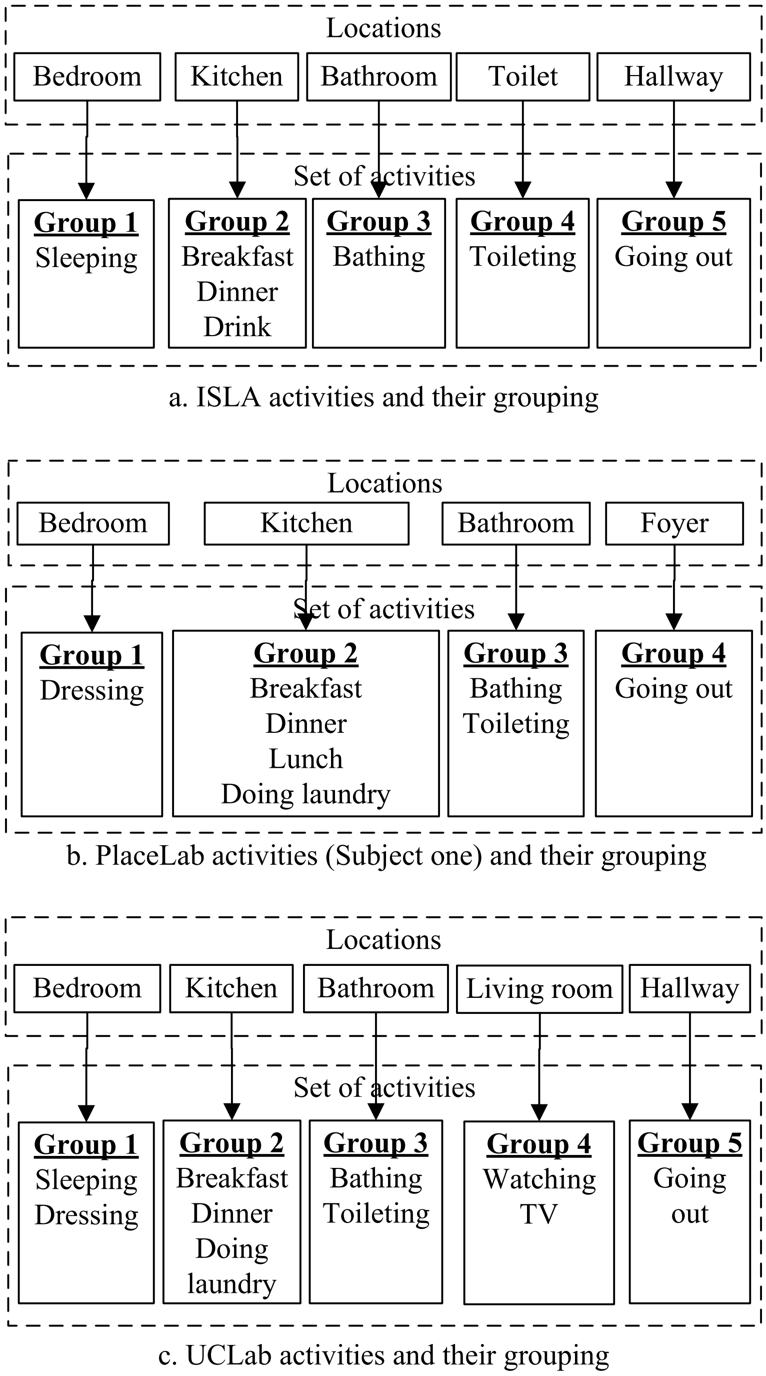


Figure 5.4: The activities and their grouping.

this thesis, only the Subject one's dataset is used.

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 [85].

In order to gather semi real-world dataset, a virtual environment is created (using the tool described in section 5.1) which is identical to a 28-year-old girl's environment. Thirty objects are virtually embedded with thirty state-change sensors and nine activities are considered to be monitored. During the time of conducting the experiments, 1 day data was annotated.

Figure 5.4 shows the ISLA, PlaceLab and UCLab activities and their grouping is used to validate the system's performance.

5.2.2.2 Accuracy measurement formulas

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

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

and the class Accuracy is measured by,

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

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.

Although 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 will be $\approx$ 92%, whereas the class accuracy will be $\approx$ 68%. Therefore, the class accuracy should be the primary way to evaluate the activity classifiers performance. However, in this dissertation both the time-slice and the class accuracy is reported.

It is to be noted here that if the number of instances of activities are equal then time-slice accuracy and class accuracy are equal.

5.3 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. The mining would be effective, if the likelihoods estimated from the mined data are realistic. Table 5.1 and 5.2 show the calculated object-usage and location-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 5.1, the likelihoods of using a "Microwave" for preparing "breakfast" or preparing "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.

Table 5.1: The likelihoods of object-usage generated by the PE in conjunction with the AME for ISLA activities

Activities / Objects	Going out	Toileting	Bathing	Sleeping	Breakfast	Dinner	Drink
Microwave	0.011690	0.002967	0.018108	0.023292	0.093776	0.069610	0.040820
Door	0.271218	0.169139	0.206099	0.201636	0.189941	0.118653	0.139415
Cups	0.024783	0.086053	0.201334	0.037974	0.066897	0.129529	0.201873
Fridge	0.016600	0.000001	0.009768	0.031555	0.096165	0.043901	0.068815
Plate	0.064531	0.038575	0.048248	0.037171	0.061522	0.254115	0.114878
Dishwasher	0.007715	0.005934	0.007386	0.016259	0.017023	0.042814	0.016172
Flush	0.009352	0.097922	0.022873	0.012996	0.004479	0.004706	0.026544
Freezer	0.011690	0.000001	0.004884	0.014868	0.028551	0.046472	0.044055
Pans	0.007949	0.109792	0.012747	0.014761	0.012543	0.026499	0.014945
Groceries	0.017301	0.000001	0.009292	0.009306	0.009676	0.019577	0.042493

However, there are some noises too. For example, the "Pans" usage likelihood is reasonably higher for "Toileting", but this is a kitchen appliance and the likelihood of this object usage should be high for "Breakfast" or "Dinner". The reason is, the term "Pan" have at least two meanings: cooking utensil consisting of a wide metal vessel and commode pans.

Table 5.2: The likelihoods of location-usage generated by the PE in conjunction with the AME for ISLA activities

Activities / Locations	Going out	Toileting	Bathing	Sleeping	Breakfast	Dinner	Drink
Kitcen	0.098658	0.063457	0.085161	0.096515	0.093393	0.107818	0.104163
Toilet	0.019917	0.170678	0.062832	0.020528	0.012939	0.004612	0.012626
Bathroom	0.029803	0.062910	0.086459	0.036005	0.072523	0.010522	0.019570
Hallway	0.002158	0.000547	0.000270	0.003364	0.002278	0.000896	0.001479
Bedroom	0.040285	0.024070	0.021160	0.050944	0.058784	0.008996	0.016233

5.4 Experiment 2: Activity recognition accuracy

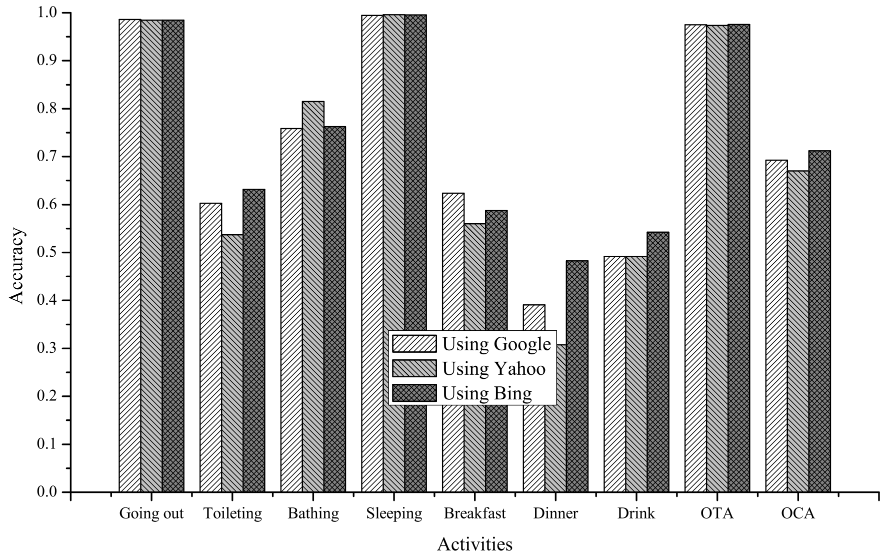


Figure 5.5: The accuracies per class for ISLA dataset the rightmost two pairs of clusters compare the overall timeslice accuracy (OTA) and the overall class accuracy (OCA).

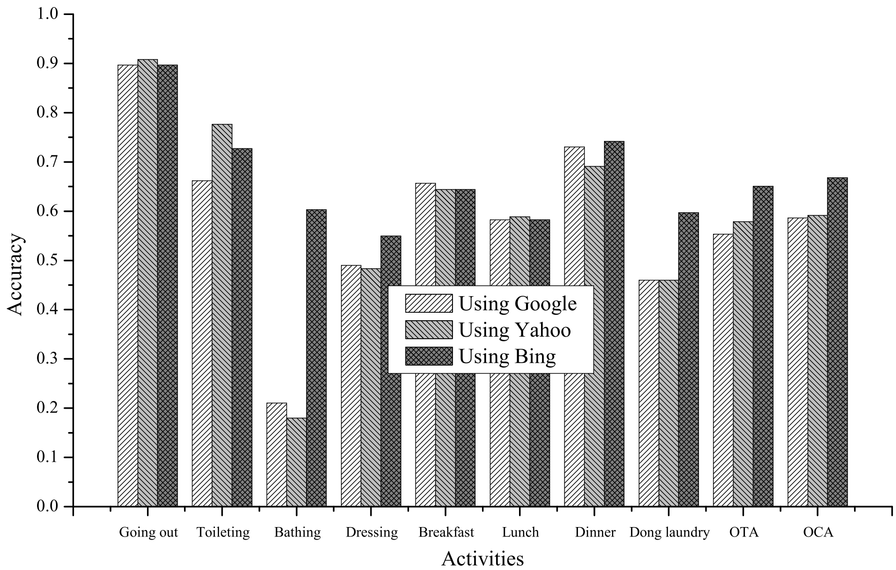


Figure 5.6: The accuracies per class for PlaceLab dataset (Subject one), the rightmost two pairs of clusters compare the overall timeslice accuracy (OTA) and the overall class accuracy (OCA).

The purpose of this experiment is to see how accurate the proposed method is to classify the

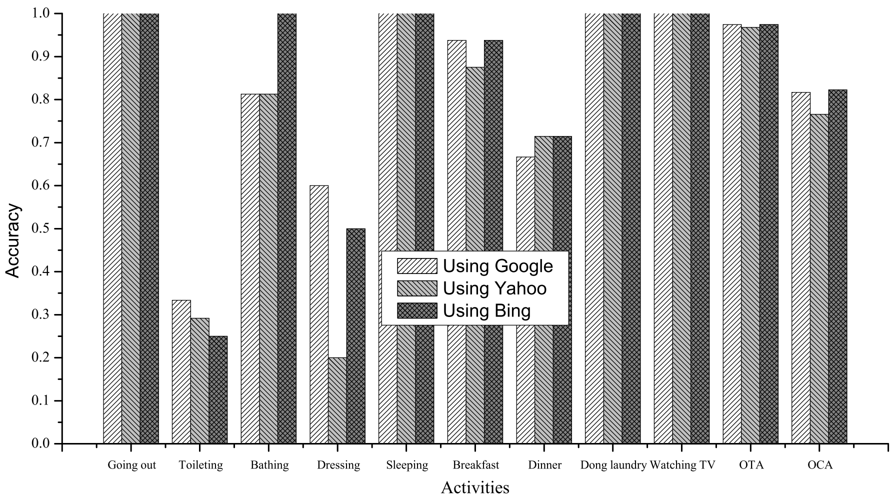


Figure 5.7: The accuracies per class for UCLab dataset, the rightmost two pairs of clusters compare the overall timeslice accuracy (OTA) and the overall class accuracy (OCA).

activities.

Figures 5.5, 5.6 and 5.7 summarize the accuracies per class for three datasets. Each of the three bars in a cluster represents (from left to right) the accuracy of each activity using Google, Yahoo and Bing respectively. The rightmost two clusters compare the overall time slice accuracy (OTA) and the overall class accuracy (OCA).

The estimated coefficients are shown in Table 5.3. In section 5.5, more about the effect of α and λ is discussed.

Table 5.3: The estimated α and λ for three datasets.

Datasets	α			λ		
	Google	Yahoo	Bing	Google	Yahoo	Bing
ISLA	0.5354	0.5726	0.5823	0.9694	1.0000	1.0000
PlaceLab	0.4605	0.4834	0.4968	0.8650	0.9550	0.9350
UCLab	0.4134	0.4381	0.4210	0.8370	0.9519	0.9296

5.4.1 Summary of the accuracies

In this section, the summarization of the activity recognition results of the system (learned using web activity data as well as real-world activity data) is given. The results are shown in Figure

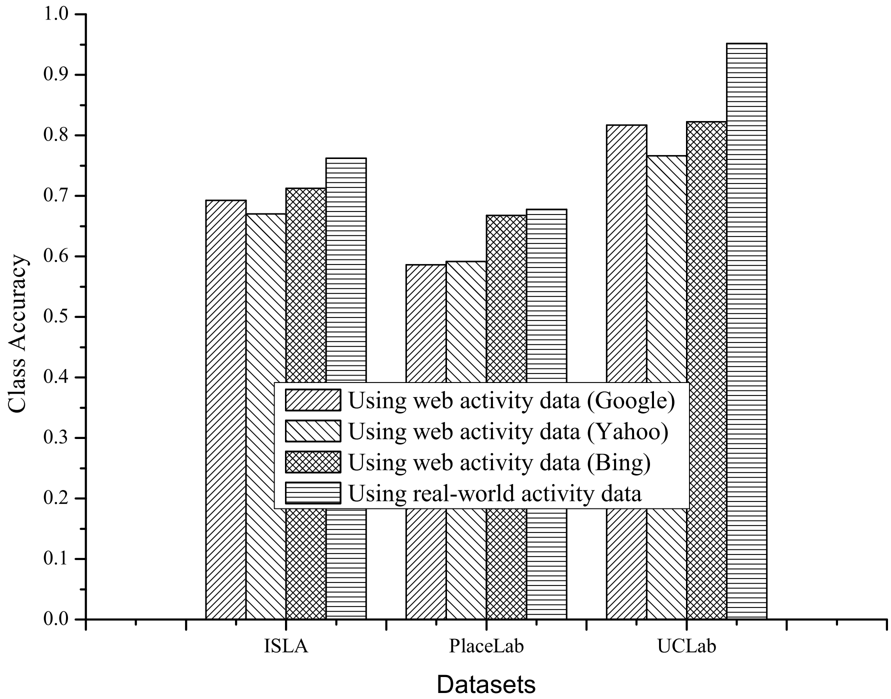


Figure 5.8: The summary of the accuracies for all the datasets.

5.8. Each of the four bars in a cluster represents (from left to right) the overall class accuracy of the system when it is learned using both web activity data (first three bars in a cluster) and the real-world activity data (last bar in a cluster). Detail description of each of the accuracy bars are given below:

- Using web activity data

  1. Google: The system achieves overall class accuracy 69.25% (time slice accuracy is 97.50%), 58.60% (time slice accuracy is 55.35%) and 81.67% (time slice accuracy is 97.46%) for the ISLA, PlaceLab and UCLab data set respectively when learned using the knowledge mined by Google.
  2. Yahoo: The system achieves overall class accuracy 67.01% (time slice accuracy is 97.34%), 59.14% (time slice accuracy is 57.86%) and 76.59% (time slice accuracy is 96.80%) for the ISLA, PlaceLab and UCLab data set respectively when learned using the knowledge mined by Yahoo.

3. Bing: The system achieves overall class accuracy 71.23% (time slice accuracy is 97.57%), 66.76% (time slice accuracy is 65.06%) and 82.24% (time slice accuracy is 97.46%) for the ISLA, PlaceLab and UCLab data set respectively when learned using the knowledge mined by Bing.

- Using real-world activity data: The system achieves overall class accuracy 76.21% (time slice accuracy is 97.66%), 67.75% (time slice accuracy is 66.71%) and 95.18% (time slice accuracy is 99.27%) for the ISLA, PlaceLab and UCLab data set respectively when learned using the real world activity data.

5.4.2 Confusion matrices

The corresponding confusion matrices are shown in Tables 5.4, 5.5, 5.6. Each of these tables contains three matrices, each of which is a n-by-n confusion matrix (excluding the activity name) generated by the system for a dataset when it is learned using a search engine (e.g. Google). The i^{th} row, and the j^{th} column represents the percentage of times an activity, a_i, is recognized as activity, a_j.

For the ISLA dataset, as shown in 5.4, the system exhibits similar performance in classifying individual activities when it is learned through any of the search engines. However, the performance of the system is worse for classifying "Toileting" when it is learned through Bing in comparison with Google and Yahoo.

Similarly, for the PlaceLab dataset, as shown in 5.5, the system exhibits similar performance in classifying individual activities when it is learned through any of the search engines. However, the performance of the system is better for classifying "Bathing" when it is learned through Bing in comparison with Google and Yahoo.

Finally, for the UCLab dataset, as shown in 5.6, the system exhibits similar performance in classifying individual activities when it is learned through any of the search engines. However, the performance of the system is worse for classifying "Toileting" when it is learned through Google in comparison with Yahoo and Bing.

Table 5.4: The Confusion matrix for the ISLA dataset.

	Going out	Toileting	Bathing	Sleeping	Breakfast	Dinner	Drink
Using Google							
Going out	0.99	0.01	0.00	0.00	0.00	0.00	0.01
Toileting	0.12	0.60	0.12	0.14	0.00	0.01	0.01
Bathing	0.05	0.18	0.76	0.00	0.01	0.00	0.00
Sleeping	0.00	0.01	0.00	0.99	0.00	0.00	0.00
Breakfast	0.09	0.17	0.03	0.07	0.62	0.00	0.01
Dinner	0.12	0.25	0.01	0.01	0.00	0.39	0.23
Drink	0.05	0.08	0.07	0.00	0.03	0.27	0.49
Using Yahoo							
Going out	0.98	0.01	0.00	0.00	0.00	0.00	0.01
Toileting	0.04	0.54	0.23	0.15	0.01	0.01	0.01
Bathing	0.01	0.17	0.82	0.00	0.01	0.00	0.00
Sleeping	0.00	0.00	0.00	1.00	0.00	0.00	0.00
Breakfast	0.16	0.17	0.05	0.06	0.56	0.00	0.01
Dinner	0.24	0.25	0.00	0.01	0.00	0.31	0.20
Drink	0.05	0.08	0.07	0.00	0.03	0.27	0.49
Using Bing							
Going out	0.98	0.01	0.00	0.00	0.00	0.00	0.01
Toileting	0.08	0.63	0.13	0.14	0.00	0.01	0.01
Bathing	0.04	0.19	0.76	0.00	0.01	0.00	0.00
Sleeping	0.00	0.00	0.00	1.00	0.00	0.00	0.00
Breakfast	0.11	0.17	0.06	0.07	0.59	0.00	0.01
Dinner	0.12	0.01	0.00	0.01	0.00	0.48	0.38
Drink	0.05	0.05	0.03	0.02	0.03	0.27	0.54

5.4.3 Discussion

The activity recognition system 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 5.5, the classifier made more confusion between "Toileting" and "Bathing" because, these two activities were performed in a same location (i.e. "Bathroom"), and the number of distinguishing objects are low.

It is observed that the activities in an environment are more accurately recognized, if the most

Table 5.5: The Confusion matrix for the PlaceLab dataset.

	Going out	Toileting	Bathing	Dressing	Breakfast	Lunch	Dinner	Doing laundry
Using Google								
Going out	0.90	0.07	0.01	0.01	0.00	0.01	0.00	0.00
Toileting	0.09	0.66	0.14	0.02	0.01	0.05	0.02	0.02
Bathing	0.08	0.63	0.21	0.02	0.01	0.01	0.03	0.00
Dressing	0.06	0.23	0.13	0.49	0.01	0.07	0.03	0.00
Breakfast	0.24	0.03	0.04	0.03	0.66	0.00	0.00	0.00
Lunch	0.13	0.04	0.00	0.16	0.00	0.58	0.07	0.01
Dinner	0.07	0.06	0.08	0.04	0.00	0.00	0.73	0.02
Doing laundry	0.42	0.04	0.03	0.03	0.01	0.02	0.00	0.46
Using Yahoo								
Going out	0.91	0.06	0.01	0.01	0.00	0.01	0.00	0.00
Toileting	0.09	0.78	0.02	0.02	0.01	0.05	0.02	0.02
Bathing	0.09	0.65	0.18	0.03	0.01	0.02	0.02	0.00
Dressing	0.06	0.30	0.05	0.48	0.01	0.07	0.03	0.01
Breakfast	0.24	0.04	0.01	0.06	0.64	0.00	0.00	0.00
Lunch	0.13	0.04	0.00	0.15	0.00	0.59	0.07	0.01
Dinner	0.07	0.04	0.07	0.03	0.00	0.06	0.69	0.03
Doing laundry	0.42	0.06	0.00	0.03	0.01	0.02	0.00	0.46
Using Bing								
Going out	0.90	0.03	0.05	0.01	0.00	0.01	0.00	0.00
Toileting	0.07	0.73	0.06	0.05	0.01	0.03	0.02	0.03
Bathing	0.08	0.21	0.60	0.05	0.01	0.01	0.03	0.00
Dressing	0.04	0.17	0.14	0.55	0.01	0.07	0.03	0.00
Breakfast	0.22	0.05	0.02	0.05	0.64	0.00	0.00	0.01
Lunch	0.12	0.03	0.01	0.16	0.00	0.58	0.07	0.03
Dinner	0.06	0.02	0.11	0.03	0.00	0.00	0.74	0.04
Doing laundry	0.07	0.04	0.03	0.24	0.01	0.02	0.00	0.60

important objects (or the key objects that are more generally used) for that activities are embedded with sensors rather than the less important objects. For example, the activities of ISLA and UCLab datasets are more accurately recognized than the PlaceLab dataset, because the numbers of important objects are relatively high in ISLA and UCLab.

The system's performance is proportional to the number of objects utilized in an environment. The more objects are utilized the more likely it is to create confusion. For example, the classifier

Table 5.6: The Confusion matrix for the UCLab dataset.

	Going out	Toileting	Bathing	Dressing	Sleeping	Breakfast	Dinner	Doing laundry	Watching TV
Using Google									
Going out	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Toileting	0.04	0.33	0.58	0.00	0.04	0.00	0.00	0.00	0.00
Bathing	0.00	0.19	0.81	0.00	0.00	0.00	0.00	0.00	0.00
Dressing	0.35	0.00	0.05	0.60	0.00	0.00	0.00	0.00	0.00
Sleeping	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Breakfast	0.00	0.00	0.00	0.00	0.06	0.94	0.00	0.00	0.00
Dinner	0.00	0.00	0.19	0.05	0.00	0.10	0.67	0.00	0.00
Doing laundry	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Watching TV	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00
Using Yahoo									
Going out	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Toileting	0.04	0.29	0.58	0.00	0.04	0.00	0.00	0.00	0.04
Bathing	0.00	0.19	0.81	0.00	0.00	0.00	0.00	0.00	0.00
Dressing	0.35	0.30	0.15	0.20	0.00	0.00	0.00	0.00	0.00
Sleeping	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Breakfast	0.00	0.00	0.00	0.00	0.00	0.88	0.00	0.13	0.00
Dinner	0.00	0.00	0.19	0.00	0.00	0.10	0.71	0.00	0.00
Doing laundry	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Watching TV	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00
Using Bing									
Going out	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Toileting	0.04	0.25	0.67	0.00	0.04	0.00	0.00	0.00	0.00
Bathing	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Dressing	0.35	0.05	0.00	0.50	0.10	0.00	0.00	0.00	0.00
Sleeping	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Breakfast	0.00	0.00	0.00	0.00	0.06	0.94	0.00	0.00	0.00
Dinner	0.00	0.00	0.19	0.00	0.00	0.10	0.71	0.00	0.00
Doing laundry	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Watching TV	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

performs best in classifying the ISLA activities, with a classification accuracy of 71.20%. This is because in their experiment they only utilized 14 objects, and the number of objects per location was limited. Nevertheless, for the PlaceLab dataset, the accuracy of classification is 66.88%. They have utilized 84 objects for their setup.

A group of similar activities is more distinguishable if performed in different locations. For example, “Toileting” and “Bathing” in the ISLA dataset are more distinguishable (as shown in Table 5.4) than in the PlaceLab or the UCLab datasets (as shown in Table 5.5 and 5.6) because “Toilet” and “Bathroom” are two different locations in the ISLA environment.

The system usually performs better if it is learned with the activity knowledge mined using Bing in comparison with Google and Yahoo. For example, as we can see in Figure 5.8, the classification results of the system are better for PlaceLab and UCLab datasets when using the knowledge that is mined using Bing. The reason is that the activity knowledge mined using Bing is less noisy then other search engines.

5.5 Experiment 3: Varying the model coefficients

The purpose of this experiment is two-folds: Analyze the impact of the coefficients (α and λ) in accuracy of activity classification and to observe whether the proposed methods of estimating the coefficients can determine the nearly optimal values or not. The experiment is performed 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.

5.5.1 Varying the coefficient, α

The activity recognition results for varying the coefficient, α, are shown in Figure 5.9, 5.10 and 5.11 for the ISLA, PlaceLab and UCLab datasets respectively.

As expected, all the three datasets are sensitive to the α values. For example, as we can see in Figure 5.9-5.11, for α = 0.0, the accuracies of activity classification are relatively low with respect to α = 0.1. It indicates that incorporating the LOBM significantly improve the activity classification accuracy. But only the LBM is not always sufficient for first-layer classification. For example, the accuracies are relatively low when α is set to 1.0.

The estimated coefficients, α, for all the datasets as shown in Table 5.3, are near their optimal values. For example, as we can see in Figure 5.9 that the maximum performance for the ISLA dataset is observed for $\alpha = 0.6$ and the estimated α for this dataset are 0.5354, 0.5726,

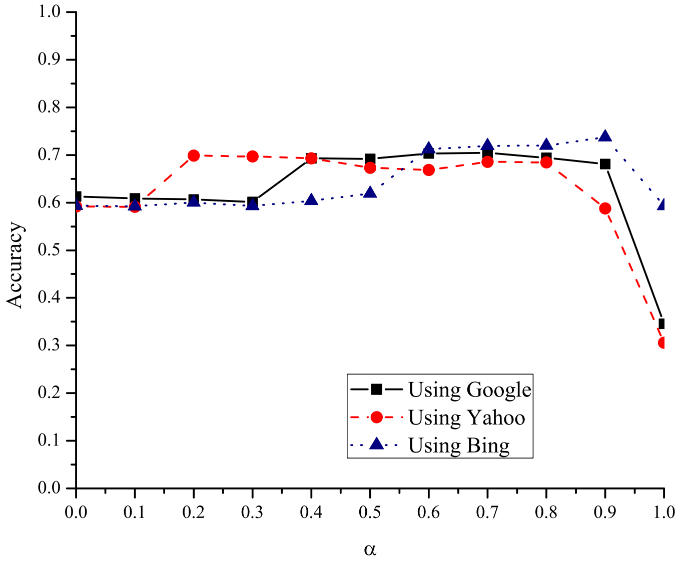


Figure 5.9: Activity recognition accuracy with different α settings for the ISLA dataset.

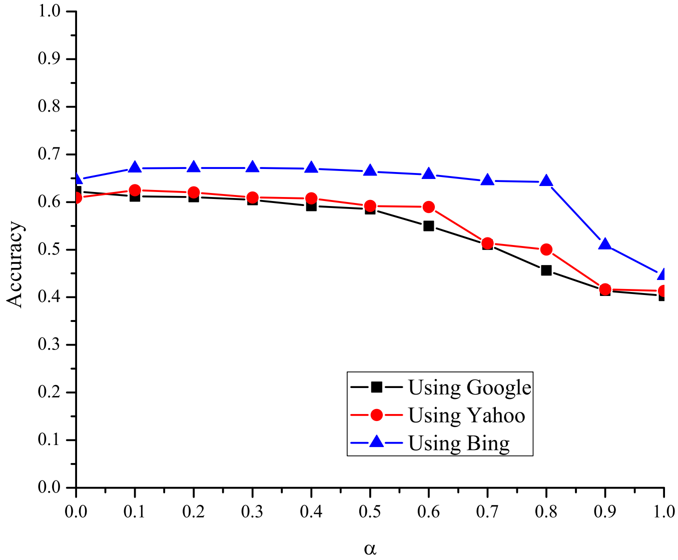


Figure 5.10: Activity recognition accuracy with different α settings for the PlaceLab dataset.

0.5823(using Google, Yahoo and Bing respectively), which are near their optimal values.

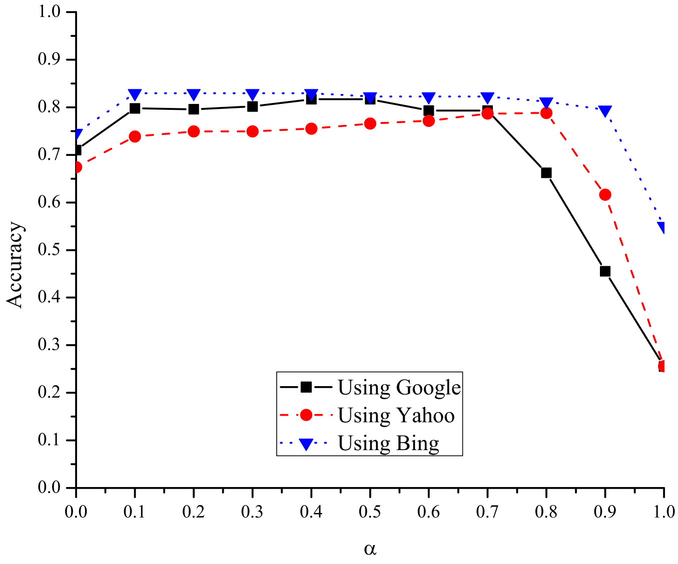


Figure 5.11: Activity recognition accuracy with different α settings for the UCLab dataset.

5.5.2 Varying the coefficient, λ

The activity recognition results for varying the coefficient, λ, are shown in Figure 5.12, 5.13 and 5.14 for the ISLA, PlaceLab and UCLab datasets respectively.

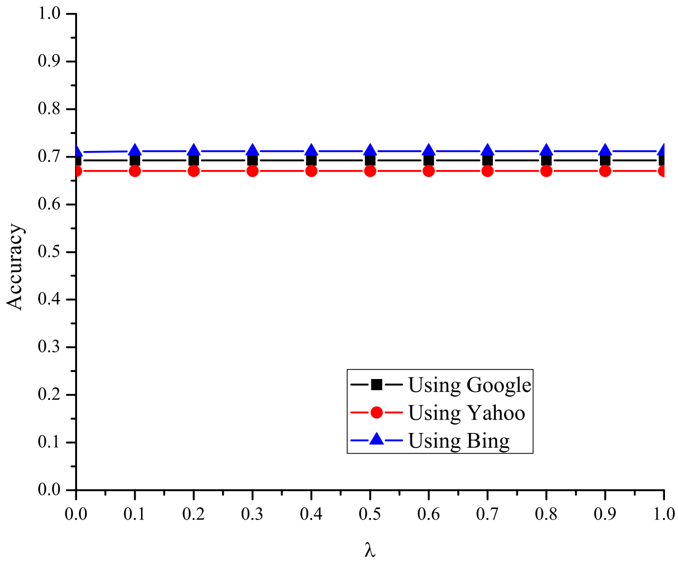


Figure 5.12: Activity recognition accuracy with different λ settings for the ISLA dataset.

The activity recognition accuracy is sensitive to λ values for all the datasets except the ISLA

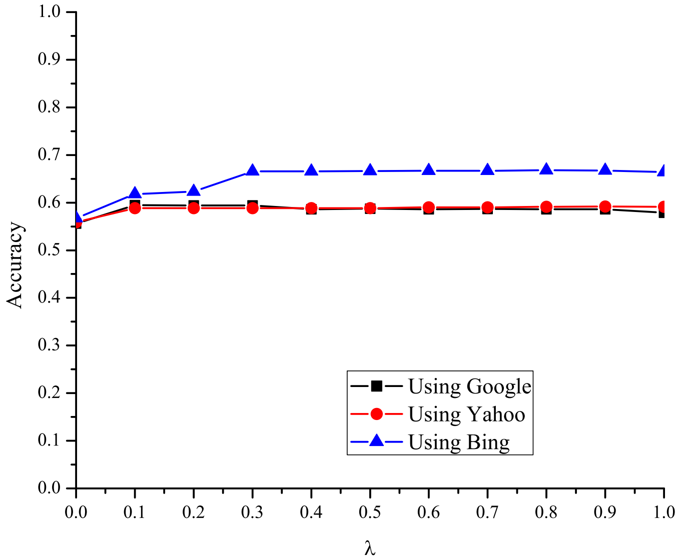


Figure 5.13: Activity recognition accuracy with different λ settings for the PlaceLab dataset.

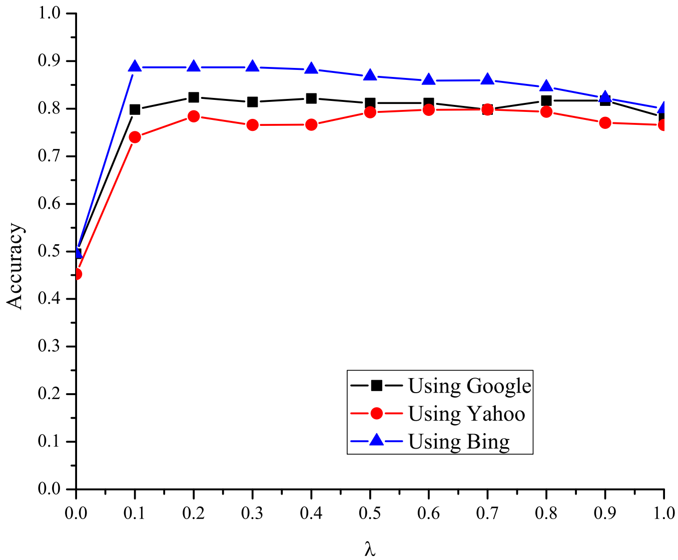


Figure 5.14: Activity recognition accuracy with different λ settings for the UCLab dataset.

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 is almost zero).

The estimated coefficient, λ, as shown in Table 5.3, are near their optimal values. For example, in Figure 5.13, we can see that the maximum performance for the PlaceLab dataset is observed when the λ = 0.9 and the estimated λ for this dataset is 0.9550, which is nearly optimal.

5.6 Experiment 4: Comparison with the other methods

The goal of this experiment is two-fold:

1. Compare the performance of the EARWD in classifying the activities with UARS [28].
2. Compare the time complexity of the proposed mining technique with the mining technique proposed in [28].

5.6.1 Performance comparison of the classifiers

The performance of the proposed classifier is compared with a TFW based classifier, proposed by Wyatt et al. in [28], which uses HMM as the classifier. The comparison results are shown in Figure 5.15. It is observed that the EARWD achieved superior performance for all the datasets. For example, for the PlaceLab datset, the observed improvement is 60.35% when learned with the mined data using Bing.

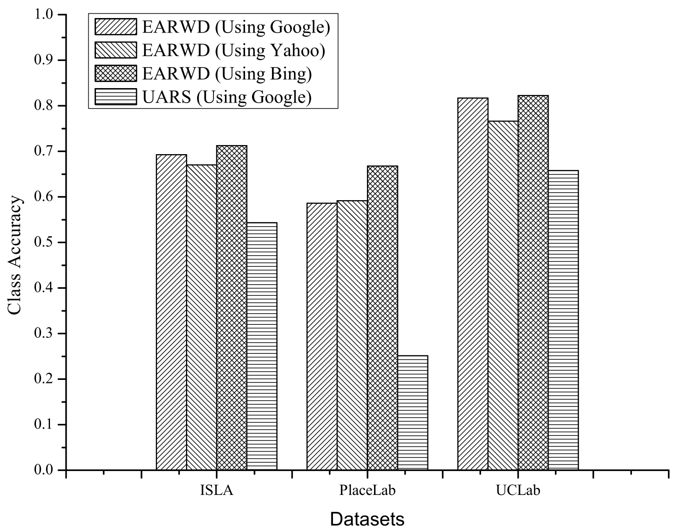


Figure 5.15: Comparison with other methods (class accuracies are used to compare).

5.6.2 Mining time comparison

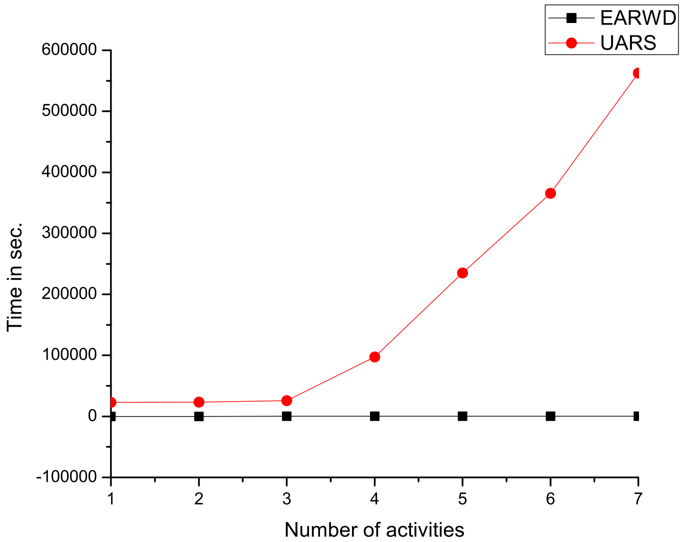


Figure 5.16: Mining time comparison between the EARWD and the UARS for the ISLA dataset.

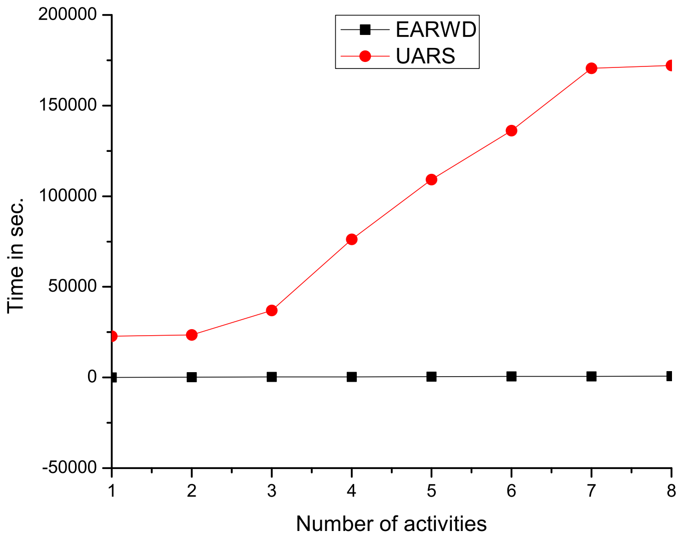


Figure 5.17: Mining time comparison between the EARWD and the UARS for the PlaceLab dataset.

To the best of our knowledge, only two systems [27, 28], have been proposed to train an AR system from the web. The proposed mining technique is compared with the technique used in [28] (Their system is named as, UARS). It is not feasible to perform a direct comparison to the

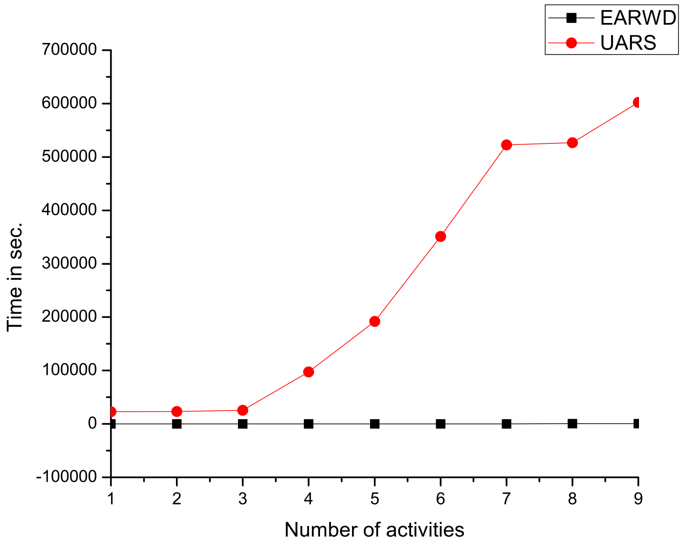


Figure 5.18: Mining time comparison between the EARWD and the UARS for the UCLab dataset.

technique used in [27]. 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 [28]. The method has a strict one-to-one match between the activities. It is not required to map the tagged object names with the mined object names since the information associated with the tagged object is mined.

The mining time required by EARWD and UARS for each of the datasets are shown in Figure 5.16, 5.17 and 5.18. As expected, the EARWD outperforms for all the datasets.

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

The EARWD would take $t = 1+1(5+20) = 26$ (using the Equation (4.1)) seconds to mine activity information regarding "Going out", assuming that the search engine would take 1 second to provide the search result for each query.

The total time, t, UARS would take to the mine activity knowledge is calculated using the following steps (in chapter 2.4.2, the mining algorithm is described).

1. The UARS first search web with the query "How to" "Going out". The search engine would return $\hat{P}$ pages. Let us assume that $|\hat{P}| = 10,700,000$ and we set $t = 1$ (assuming that the

search engine 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, total time, t, would be, $t = 1 + 10700 * 2 = 21401$.

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, total time, t, will be 21615.

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

5.7 Discussion

5.7.1 Providing activity name, object name and location name

One of the most important components of the EARWD is to choose the name of the activities, objects and locations because the efficiency of the AME depends on choosing appropriate names. For the current version of the EARWD the names are chosen manually. For example, for the PlaceLab dataset the exact activity names are used as they used in their paper. But for the ISLA dataset the activity "Leaving" is changed to "Going out" to make it more sensible and consistent to other datasets. It is preferred to provide object name in one word, however, multiple words (e.g. "Washing machine", "Shower faucet") is also fine.

5.7.2 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, embedding sensors to both of these objects might improve the classification accuracy.

5.7.3 Affect of web data changing

The web is a dynamic information environment. Web contents are constantly changing and therefore, new activity page can be added to web. How much these changes would affect the activity recognition? To answer this question, the performance of the system is measured for different days. Three datasets are used, ISLA, PlaceLab and UCLab. The results are shown in Table 5.7. The experiment is performed for 3 days and as we can see in the Table that the web data changing affect is negligible. The biggest fluctuation (around 4%) of activity recognition accuracy is observed between June 03, 2010 and June 05, 2010, for the UCLAB activities, when the system is learned with the datasets mined using Google.

Table 5.7: The affect of web data changing (G− >Google, Y− >Yahoo, B− >Bing).

Date / Datasets	June 03, 2010			June 05, 2010			June 06, 2010		
	G	Y	B	G	Y	B	G	Y	B
ISLA	0.69	0.67	0.71	0.70	0.69	0.69	0.69	0.69	0.71
PlaceLab	0.59	0.59	0.67	0.59	0.59	0.67	0.57	0.59	0.68
UCLab	0.82	0.77	0.82	0.86	0.75	0.83	0.86	0.75	0.83

5.8 Summary

In this chapter, the performance of the EARWD is validated. For this purpose, four experiments are performed: First, the efficiency of mining method is verified by checking the likelihoods estimated by the parameter estimator with the help of the activity mining engine. Second, the classifier's performance is evaluated in classifying activities of three datasets. Third, the impact of the coefficients (α and λ) in activity classification is analyzed, the proposed methods of estimating these coefficients is also analyzed. Finally, the comparison results of different classifiers and different mining engines are shown.

Chapter 6

Conclusion and future work

6.1 Conclusion

The goal of this research is to develop an efficient activity recognition system using web activity data that is broadly applicable, easy-to-use and highly accurate. For this purpose an environment in considered in which a set of simple and ubiquitous sensors (or state-change sensors) is embedded with a set of daily life objects. The sensors are embedded in a way such that the system can determine the state of the object when a user interacts with it. The system recognizes the activities of daily livings based on a set of object-usage for a period of time.

An activity recognition system requires training with representative examples for all the activities it has to recognize. There are two ways to train an activity recognition system: using real-world activity data and using web activity data. In this thesis, the problems of using real-world activity data to train the system are addressed and the benefits of using web activity as the alternate source of activity data are described. The problems associated with the current state of the art techniques to mine web activity data and to train an AR system using such data are addressed. Finally, a novel way to mine human activity data from web is described. How to train an activity classifier using such data are also described. 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.

The proposed activity recognition system (i.e. EARWD) uses a high-accurate two-layer Naïve Bayesian based probabilistic classifier, that utilizes both location-usage and object-usage information to classify an activity. The first-layer uses a location-and-object-usage based model to narrow down the scope of the classification task by classifying a group of activities (location specific) from a set of activities. The second-layer uses an object-usage based model to classify the actual activity from that group (classified by the first-layer).

The EARWD uses an activity mining engine to mine activity data from web. The mining engine takes the following external inputs: a set of activities to monitor, object names with attached sensors and their corresponding locations. It generates object-usage and location-usage information for the given activities as output by utilizing web search engines (e.g. Google, Yahoo and Bing). These information are used by the AR system to train the underlying classifier.

In order to validate the system's performance, four experiments have been conducted. It is shown that the proposed method can classify the activities with high accuracy. The comparison results of different classifiers and different mining engines are also shown. It is observed that the proposed mechanism yields *significant improvement* in comparison to the existing systems in the literature.

6.2 Future work

As discussed in chapter 5, the activities in an environment are more accurately recognized by EARWD if the most important objects (or the key objects that are more generally used) for that activities are embedded with sensors rather than the less important objects. Therefore, it will be helpful for the system if a recommendation system is developed that can recommend the set of key objects from a given set of objects. That is, given a set of activities to monitor, a set of objects of the environment, the recommendation system would suggest the set of objects to which the sensors should be embedded. Therefore, one of the future goals will be to develop such a recommendation system.

Additionally, current research is focused on human activity recognition in a single person environment. However, there could be an environment in which we need to recognize activities of two or more person. For example, we may need to know how our parents are doing while we are away from home. Therefore, the next focus would be to make an efficient activity recognition system using web activity data that would be able to recognize multi-user activities from sensor readings in a smart home environment. Some of the challenges in building such system are stated below:

- **Object-usage:** At first we need to determine who is using the object. If we consider a multi-user environment, in current configuration there is no way to distinguish who uses the object. We can either use wearable GPS sensors or we can use RFID tags with a wearable RFID reader rather than using a video-camera.

- **Collective-effort:** Activities are often performed by multiple users involving interactions between them. How to recognize these activities would be another major challenge.

- **Activity dataset:** To the best of our knowledge, there is no real-life activity dataset that would reflect activities in a multi-user environment. Therefore, getting or generating such a dataset and subsequently checking the validity of the algorithm would be another challenge.

References

[1] M. R. Hodges and M. E. Pollack, "An 'object-use fingerprint': the use of electronic sensors for human identification," in *Proc. Ubicomp*, ser. Lecture Notes in Computer Science, J. Krumm, G. D. Abowd, A. Seneviratne, and T. Strang, Eds., vol. 4717. Springer, 2007, pp. 289–303.

[2] 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.* New York, NY, USA: ACM, 2008, pp. 30–39.

[3] D. H. Hu and Q. Yang, "Cigar: concurrent and interleaving goal and activity recognition," in *AAAI*, D. Fox and C. P. Gomes, Eds. AAAI Press, 2008, pp. 1363–1368.

[4] D. J. Patterson, L. Liao, D. Fox, and H. A. Kautz, "Inferring high-level behavior from low-level sensors," in *Proc. Ubicomp*, ser. Lecture Notes in Computer Science, A. K. Dey, A. Schmidt, and J. F. McCarthy, Eds., vol. 2864. Springer, 2003, pp. 73–89.

[5] E. M. Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," in *Pervasive*, ser. Lecture Notes in Computer Science, A. Ferscha and F. Mattern, Eds., vol. 3001. Springer, 2004, pp. 158–175.

[6] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, "Accurate activity recognition in a home setting," in *UbiComp '08: Proceedings of the 10th international conference on Ubiquitous computing*. New York, NY, USA: ACM, 2008, pp. 1–9.

[7] T. Choudhury, G. Borriello, S. Consolvo, D. Haehnel, B. Harrison, B. Hemingway, J. Hightower, P. P. Klasnja, K. Koscher, A. LaMarca, J. A. Landay, L. LeGrand, J. Lester, A. Rahimi,

A. Rea, and D. Wyatt, "The mobile sensing platform: an embedded activity recognition system," *IEEE Pervasive Computing*, vol. 7, no. 2, pp. 32–41, 2008.

[8] A. M. J. Sarkar, Y.-K. Lee, and S. Lee, "A smoothed naïve bayes-based classifier for activity recognition," *IETE Technical Review*, vol. 27, no. 2, pp. 107–119, 2010.

[9] J. Yang, B. N. Schilit, and D. W. McDonald, "Activity recognition for the digital home," *Computer*, vol. 41, no. 4, pp. 102–104, 2008.

[10] P. Palmes, H. K. Pung, T. Gu, W. Xue, and S. Chen, "Object relevance weight pattern mining for activity recognition and segmentation," *Pervasive Mob. Comput.*, vol. 6, no. 1, pp. 43–57, 2010.

[11] J. Ben-Arie, Z. Wang, P. Pandit, and S. Rajaram, "Human activity recognition using multidimensional indexing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 8, pp. 1091–1104, 2002.

[12] V. Kellokumpu, M. Pietikäinen, and J. Heikkilä, "Human activity recognition using sequences of postures," in *MVA*, 2005, pp. 570–573.

[13] J. K. Aggarwal and Q. Cai, "Human motion analysis: A review." *Computer Vision and Image Understanding*, vol. 73, no. 3, pp. 428–440, 1999. [Online]. Available: http://dblp.uni-trier.de/db/journals/cviu/cviu73.html#AggarwalC99

[14] R. Polana and R. Nelson, "Detecting activities," in *Journal of Visual Communication and Image Representation*, 1993, pp. 2–7.

[15] S. M.-J. A. Ward, M.-P. Lukowicz, S. M.-G. Troster, and M.-T. E. Starner, "Activity recognition of assembly tasks using body-worn microphones and accelerometers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 10, pp. 1553–1567, 2006.

[16] J. Muncaster and Y. Ma, "Activity recognition using dynamic bayesian networks with automatic state selection," in *WMVC '07: Proceedings of the IEEE Workshop on Motion and Video Computing*. Washington, DC, USA: IEEE Computer Society, 2007, p. 30.

[17] C. Sminchisescu, A. Kanaujia, and D. Metaxas, "Conditional models for contextual human motion recognition," *Comput. Vis. Image Underst.*, vol. 104, no. 2, pp. 210–220, 2006.

[18] R. Hamid, S. Maddi, A. Bobick, and I. Essa, "Unsupervised analysis of activity sequences using event-motifs," in *VSSN '06: Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*. New York, NY, USA: ACM, 2006, pp. 71–78.

[19] R. Hamid, S. Maddi, A. F. Bobick, and I. A. Essa, "Structure from statistics - unsupervised activity analysis using suffix trees," in *ICCV*. IEEE, 2007, pp. 1–8.

[20] B. Logan, J. Healey, M. Philipose, E. M. Tapia, and S. S. Intille, "A long-term evaluation of sensing modalities for activity recognition," in *Proc. of Ubicomp07*, 2007, pp. 483–500.

[21] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *Pervasive*, ser. Lecture Notes in Computer Science, A. Ferscha and F. Mattern, Eds., vol. 3001. Springer, 2004, pp. 1–17.

[22] T. Huynh and B. Schiele, "Unsupervised discovery of structure in activity data using multiple eigenspaces," in *2nd International Workshop on Location- and Context- Awareness (LoCA 2006)*, Dublin, Ireland, May 2006.

[23] ——, "Towards less supervision in activity recognition from wearable sensors," *Wearable Computers, IEEE International Symposium*, vol. 0, pp. 3–10, 2006.

[24] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," in *IAAI'05: Proceedings of the 17th conference on Innovative applications of artificial intelligence*. AAAI Press, 2005, pp. 1541–1546.

[25] M. Philipose, K. P. Fishkin, M. Perkowitz, D. J. Patterson, D. Fox, H. Kautz, and D. Hahnel, "Inferring activities from interactions with objects," *IEEE Pervasive Computing*, vol. 3, no. 4, pp. 50–57, 2004.

[26] D. J. Patterson, D. Fox, H. Kautz, and M. Philipose, "Fine-grained activity recognition by aggregating abstract object usage," in *ISWC '05: Proceedings of the Ninth IEEE International*

Symposium on Wearable Computers. Washington, DC, USA: IEEE Computer Society, 2005, pp. 44–51.

[27] 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*. New York, NY, USA: ACM, 2004, pp. 573–582.

[28] D. Wyatt, M. Philipose, and T. Choudhury, "Unsupervised activity recognition using automatically mined common sense," in *AAAI'05: Proceedings of the 20th national conference on Artificial intelligence*. AAAI Press, 2005, pp. 21–27.

[29] S. S. Intille, J. Rondoni, C. Kukla, I. Ancona, and L. Bao, "A context-aware experience sampling tool," in *CHI '03: CHI '03 extended abstracts on Human factors in computing systems*. New York, NY, USA: ACM, 2003, pp. 972–973.

[30] S. S. Intille, E. M. Tapia, J. Rondoni, J. Beaudin, C. Kukla, S. Agarwal, L. Bao, and K. Larson, "Tools for studying behavior and technology in natural settings," in *Proc. Ubicomp*, ser. Lecture Notes in Computer Science, A. K. Dey, A. Schmidt, and J. F. McCarthy, Eds., vol. 2864. Springer, 2003, pp. 157–174.

[31] S. Consolvo and M. Walker, "Using the experience sampling method to evaluate ubicomp applications," *IEEE Pervasive Computing*, vol. 2, no. 2, pp. 24–31, 2003.

[32] A. Keefer. How to prevent bathing injuries among the elderly (accessed: 2010, May 04). [Online]. Available: http://www.ehow.com/how_2242133_prevent-bathing-injuries-among-elderly.html

[33] R. Cilibrasi and P. M. B. Vitányi, "The google similarity distance," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 370–383, 2007.

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

[35] L. Liao, D. Fox, and H. A. Kautz, "Location-based activity recognition using relational markov networks," in *IJCAI*, L. P. Kaelbling and A. Saffiotti, Eds. Professional Book Center, 2005, pp. 773–778. [Online]. Available: http://www.informatik.uni-trier.de/~ley/db/conf/ijcai/ijcai2005.html#LiaoFK05

[36] E. M. Tapia, "Activity recognition in the home setting using simple and ubiquitous sensors," Master's thesis, Media Arts and Sciences, Massachusetts Institute of Technology, 2003.

[37] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to information retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[38] D. Cook, M. Schmitter-Edgecombe, A. Crandall, C. Sanders, and B. Thomas, "Collecting and disseminating smart home sensor data in the casas project," in *Proc. of CHI09 Workshop on Developing Shared Home Behavior Datasets to Advance HCI and Ubiquitous Computing Research*, 2009.

[39] D. E. Comeau. Smart houses info (accessed: 2010, April 08). [Online]. Available: http://gero-tech.net/smart-homes.html

[40] B. Luscombe. This bold house (accessed: 2010, April 10). [Online]. Available: http://www.aarpmagazine.org/lifestyle/Articles/a2003-08-28-bold_house.html

[41] J. G. Dean Chen. Duke university smart home program (accessed: 2010, April 08). [Online]. Available: http://www.smarthome.duke.edu/

[42] U. of Rochester Medical Center. Smart medical home research laboratory (accessed: 2010, April 08). [Online]. Available: http://www.futurehealth.rochester.edu/smart_home/

[43] B. Jones. The aware home research initiative (accessed: 2010, April 08). [Online]. Available: http://awarehome.imtc.gatech.edu/

[44] H. R. Group. Mit house_n (accessed: 2010, April 08). [Online]. Available: http://architecture.mit.edu/house_n/

[45] G. Hsieh, I. Li, A. Dey, J. Forlizzi, and S. E. Hudson, “Using visualizations to increase compliance in experience sampling,” in *UbiComp ’08: Proceedings of the 10th international conference on Ubiquitous computing*. New York, NY, USA: ACM, 2008, pp. 164–167.

[46] Csikszentmihalyi, M, Larson, Reed, and Prescott, Suzanne, “The ecology of adolescent activity and experience,” *Journal of Youth and Adolescence*, vol. 6, no. 3, pp. 281–294, 1977.

[47] J. M. Hektner, J. A. Schmidt, and M. Csikszentmihalyi, *Experience sampling method: measuring the quality of everyday life.* Sage Publications Inc, 2007.

[48] S. S. Intille, E. M. Tapia, J. Rondoni, J. Beaudin, C. Kukla, S. Agarwal, L. Bao, and K. Larson, “Tools for studying behavior and technology in natural settings,” in *In Proceedings of UBICOMP 2003*. Springer, 2003, pp. 157–174.

[49] S. Intille, C. Kukla, and X. Ma, “Eliciting user preferences using image-based experience sampling and reflection,” in *CHI ’02: CHI ’02 extended abstracts on Human factors in computing systems*. New York, NY, USA: ACM, 2002, pp. 738–739.

[50] J. M. Hudson, J. Christensen, W. A. Kellogg, and T. Erickson, “”i’d be overwhelmed, but it’s just one more thing to do”: availability and interruption in research management,” in *CHI ’02: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM, 2002, pp. 97–104.

[51] S. Consolvo, L. Arnstein, and B. R. Franza, “User study techniques in the design and evaluation of a ubicomp environment,” in *UbiComp ’02: Proceedings of the 4th international conference on Ubiquitous Computing*. London, UK: Springer-Verlag, 2002, pp. 73–90.

[52] S. Hudson, J. Fogarty, C. Atkeson, D. Avrahami, J. Forlizzi, S. Kiesler, J. Lee, and J. Yang, “Predicting human interruptibility with sensors: a wizard of oz feasibility study,” in *CHI ’03: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM, 2003, pp. 257–264.

[53] J. Sarkar, “Experience sampling tool,” Work in progress, 2010.

[54] L. F. Barrett and D. J. Barrett, "An introduction to computerized experience sampling in psychology," *Soc. Sci. Comput. Rev.*, vol. 19, no. 2, pp. 175–185, 2001.

[55] J. Froehlich, M. Y. Chen, S. Consolvo, B. Harrison, and J. A. Landay, "Myexperience: a system for in situ tracing and capturing of user feedback on mobile phones," in *MobiSys '07: Proceedings of the 5th international conference on Mobile systems, applications and services*. New York, NY, USA: ACM, 2007, pp. 57–70.

[56] J. Froehlich. The myexperience tool (accessed: 2010, April 08). [Online]. Available: http://myexperience.sourceforge.net/

[57] P. Korpipää, M. Koskinen, J. Peltola, S.-M. Mäkelä, and T. Seppänen, "Bayesian approach to sensor-based context awareness," *Personal Ubiquitous Comput.*, vol. 7, no. 2, pp. 113–124, 2003.

[58] D. H. Hu, X.-X. Zhang, J. Yin, V. W. Zheng, and Q. Yang, "Abnormal activity recognition based on hdp-hmm models," in *IJCAI'09: Proceedings of the 21st international jont conference on Artifical intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2009, pp. 1715–1720.

[59] M. Buettner, R. Prasad, M. Philipose, and D. Wetherall, "Recognizing daily activities with rfid-based sensors," in *Ubicomp '09: Proceedings of the 11th international conference on Ubiquitous computing*. New York, NY, USA: ACM, 2009, pp. 51–60.

[60] C.-C. Lian and J. Y.-J. Hsu, "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, 2008, pp. 1814–1815.

[61] D. L. Vail, M. M. Veloso, and J. D. Lafferty, "Conditional random fields for activity recognition," in *AAMAS '07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*. New York, NY, USA: ACM, 2007, pp. 1–8.

[62] L. Liao, D. Fox, and H. A. Kautz, "Hierarchical conditional random fields for gps-based activity recognition," in *ISRR*, ser. Springer Tracts in Advanced Robotics, S. Thrun, R. A. Brooks, and H. F. Durrant-Whyte, Eds., vol. 28. Springer, 2005, pp. 487–506.

[63] J. Han, *Data Mining: Concepts and Techniques*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005.

[64] C. Zhai and J. Lafferty, "A study of smoothing methods for language models applied to information retrieval," *ACM Trans. Inf. Syst.*, vol. 22, no. 2, pp. 179–214, 2004.

[65] F. Jelinek and R. L. Mercer, "Interpolated estimation of Markov source parameters from sparse data," in *Proceedings, Workshop on Pattern Recognition in Practice*, E. S. Gelsema and L. N. Kanal, Eds. Amsterdam: North Holland, 1980, pp. 381–397.

[66] D. J. C. Mackay and L. C. B. Petoy, "A hierarchical dirichlet language model," *Natural Language Engineering*, vol. 1, pp. 1–19, 1995.

[67] Lidstone and G. J., "Note on the general case of the bayes-laplace formula for inductive or a posteriori probabilities," *Transactions of the Faculty of Actuaries*, vol. 8, pp. 182–192, 1920.

[68] I. J. Good, "The population frequencies of species and the estimation of population parameters," *Biometrika*, vol. 40, no. parts 3 and 4, pp. 237–264, 1953.

[69] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proceedings of the 34th annual meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1996, pp. 310–318.

[70] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 35, no. 3, pp. 400–401, Mar 1987.

[71] H. Ney, U. Essen, and R. Kneser, "On structuring probabilistic dependencies in stochastic language modeling," *Computer Speech and Language*, vol. 8, pp. 1–38, 1994.

[72] 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 *Pervasive*, ser. Lecture Notes in Computer Science, K. P. Fishkin, B. Schiele, P. Nixon, and A. J. Quigley, Eds., vol. 3968. Springer, 2006, pp. 349–365.

[73] L. Mercer. Bathing in style - the art of bathing well (accessed: 2010, May 04). Available: http://www.articlealley.com/article_108210_28.html.

[74] R. McIntire. How to make bathing safe for independent seniors (accessed: 2010, May 04). Available: http://ezinearticles.com/?How-to-Make-Bathing-Safe-For-Independent-Seniors&id=2148355.

[75] About.com. Google intext: Command (accessed: 2010, May 25). [Online]. Available: http://google.about.com/od/i/g/intextdef.htm

[76] Wikipedia. Web search engine (accessed: 2010, May 15). [Online]. Available: http://en.wikipedia.org/wiki/Web_search_engine

[77] Google. The essentials of google search (accessed: 2010, April 08). [Online]. Available: http://www.google.com/support/bin/static.py?page=searchguides.html&ctx=basics

[78] ——. Google search basics : More search help - web search help (accessed: 2010, April 08). [Online]. Available: http://www.google.com/support/websearch/bin/answer.py?hl=en&answer=136861

[79] Myspace. Taking medication on myspace music - free streaming mp3s, pictures & music downloads (accessed: 2010, May 18). [Online]. Available: http://www.myspace.com/takingmedication

[80] CookingPlus. Should you buy an exercise bike or treadmill? - cooking, recipes, nutrition. - cookingpulse.com (accessed: 2010, May 18). [Online]. Available: http://cookingpulse.com/2010/04/should-you-buy-an-exercise-bike-or-treadmill/

[81] Magneticlady. Are you sick of taking medication? (accessed: 2010, May 18). [Online]. Available: http://magneticlady.wordpress.com/2010/04/28/are-you-sick-of-taking-medication/

[82] Webmd. Taking medication safely: Medication chart and tips (accessed: 2010, May 18). [Online]. Available: http://www.webmd.com/a-to-z-guides/medication-safety-tips-dos-and-donts

[83] A. Albert. Preview - choosing pots and pans to improve your cooking - fine cooking recipes, techniques and tips (accessed: 2010, May 26). [Online]. Available: http://www.finecooking.com/articles/pots-and-pans-to-improve-cooking.aspx?ac=ts&ra=fp

[84] Ralenti Management Limited. Ablution equipment & bathroomware - browse through a list of suppliers (accessed: 2010, May 26). [Online]. Available: http://www.ralenti.co.nz/build_school/ablution_equip.html

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

[86] A. K. Dey, A. Schmidt, and J. F. McCarthy, Eds., *UbiComp 2003: Ubiquitous Computing, 5th International Conference, Seattle, WA, USA, October 12-15, 2003, Proceedings*, ser. Lecture Notes in Computer Science, vol. 2864. Springer, 2003.

[87] A. Ferscha and F. Mattern, Eds., *Pervasive Computing, Second International Conference, PERVASIVE 2004, Vienna, Austria, April 21-23, 2004, Proceedings*, ser. Lecture Notes in Computer Science, vol. 3001. Springer, 2004.

Appendix A

List of Publications

International Journal Papers:

[1] **Jehad Sarkar**, La The Vihn, Young-Koo Lee, Sungyoung Lee. GPARS: a general-purpose activity recognition system. Applied Intelligence. March 2010. [Online]. Available:http://dx.doi.org/10.1007/s10489-010-0217-4

[2] **Jehad Sarkar**, Young-Koo Lee, Sungyoung Lee. A smoothed Naïve Bayes-based classifier for activity recognition. IETE Technical Review 27(2), 107119 (2010). DOI 10.4103/0256-4602.60164.

International Conference Papers:

[3] Asad Masood Khattak, Zeeshan Pervez, **Jehad Sarkar**, Sungyoung Lee, Young-Koo Lee, Service Level Semantic Interoperability, International Workshop on Computing Technologies and Business Strategies for u-Healthcare (CBuH 2010), Seoul, Korea, July 2010 (Accepted for publication).

[4] **Jehad Sarkar**, Young-Koo Lee, Sungyoung Lee. ARHMAM: an activity recognition system based on Hidden Markov minded activity model. The Fourth International Conference on Ubiquitous Information Management and Communication (ICUIMC'10), pp: 484-492, 2010, SKKU, Suwon, South Korea.

[5] **Jehad Sarkar**, Kamrul Hasan, Young-Koo Lee, Sungyoung Lee, Salauddin Zabir. Distributed activity recognition using key sensors. 11th International Conference on Advanced Communication Technology, pp: 2245-2250, 2009, Phoenix Park, South Korea.

[6] **Jehad Sarkar**, Phan Tran Ho Truc, Young-Koo Lee and Sungyoung Lee. Statistical language modeling approach to activity recognition. In Proceedings of the 5th International Conference on Ubiquitous Healthcare, pp: 148-152, 2008,, Busan, South Korea.

[7] Khandoker Tarik-ul Islam, **Jehad Sarkar**, Kamrul Hassan, Mohammad Rezwanul Huq, Andrey Gavrilov, Sungyoung Lee, and Young-Koo Lee. A framework for smart object and its collaboration in smart environment. 10th International Conference on Advanced Communication Technology, pp:852-855, 2008, Phoenix Park, South Korea.

Domestic Conference Papers:

[8] Syed Khairuzzaman Tanbeer, **Jehad Sarkar**, Byeng-Soo Jeong,, Young-Koo Lee and Sungyoung Lee. I-Tree: A frequent patterns mining Approach without candidate generation or support constraint. In Proceedings of the 27th KIPS (Korean Information Processing Society) conference, pp:31-33 Kyung won University, Korea, May 11-12, 2007.