Pervasive and Mobile Computing 6 (2010) 43–57

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

Object relevance weight pattern mining for activity recognition and segmentation[☆]

Paulito Palmes[a,*], Hung Keng Pung[b], Tao Gu[c], Wenwei Xue[d], Shaxun Chen[e]

[a] *National Neuroscience Institute, Singapore*
[b] *School of Computing, National University of Singapore, Singapore*
[c] *Department of Mathematics and Computer Science, University of Southern Denmark, Denmark*
[d] *Nokia Research Center, Beijing, China*
[e] *State Key Laboratory for Novel Software Technology, Nanjing University, China*

ARTICLE INFO

Article history:
Received 10 June 2008
Received in revised form 16 May 2009
Accepted 11 October 2009
Available online 23 October 2009

Keywords:
Activity recognition and segmentation
Web mining
Knowledge engineering
Relative term weighting

ABSTRACT

Monitoring daily activities of a person has many potential benefits in pervasive computing. These include providing proactive support for the elderly and monitoring anomalous behaviors. A typical approach in existing research on activity detection is to construct sequence-based models of low-level activity features based on the *order of object* usage. However, these models have poor accuracy, require many parameters to estimate, and demand excessive computational effort. Many other supervised learning approaches have been proposed but they all suffer from poor scalability due to the manual labeling involved in the training process. In this paper, we simplify the activity modeling process by relying on the relevance weights of objects as the basis of activity discrimination rather than on sequence information. For each activity, we mine the web to extract the most relevant objects according to their normalized usage frequency. We develop a *KeyExtract* algorithm for activity recognition and two algorithms, *MaxGap* and *MaxGain*, for activity segmentation with linear time complexities. Simulation results indicate that our proposed algorithms achieve high accuracy in the presence of different noise levels indicating their good potential in real-world deployment.

1. Introduction

With the unprecedented, enduring, and pervasive ageing of the human population [1], there is a growing interest in conducting research into pervasive computing to improve healthcare support for the elderly population [2,3]. One such promising research area that can generate a variety of useful applications for the elderly is the research in *Human Activity Recognition* (HAR). A typical application of HAR is to monitor Activities of Daily Living (ADLs) [4,5] of the elderly and cognitively impaired people, detect anomalous behavior, and provide them with proactive assistance [6–8]. Another potential application of HAR is an activity-based adaptation such as lowering the TV volume during phone calls and providing instructions or directions to first time users of unfamiliar appliances.

There are several ways to acquire data for HAR using different sensor systems:

(1) Analyze the signals generated by remote observation of the subject using audio and video sensors.
(2) Attach sensors to the body of the subject to track and identify primitive human actions.
(3) Attach sensors to objects and track their usage.

[☆] This research is supported by the Science and Engineering Research Council of Singapore (SERC) under the research grant R-255-000-299-305.
[*] Corresponding author. Tel.: +65 6357 7526.
E-mail addresses: Paulito_PALMES@nni.com.sg, ppalmes@yahoo.com (P. Palmes), dcsphk@nus.edu.sg (H.K. Pung), gu@imada.sdu.dk (T. Gu), wayne.xue@nokia.com (W. Xue), csx@ics.nju.edu.cn (S. Chen).

1574-1192/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.pmcj.2009.10.004

Option (1) [9–19] is rather complex to implement because audio and video information require processing of highly multidimensional data. Moreover, both types of processing are regarded as intrusive technologies. Option (2) is promising and requires an inexpensive technology for recognizing primitive sequences of movements [20,14,21–23,11,24,21] such as walking, jogging, sitting, jumping, etc. We refer to these basic movements as *human actions*. However, using signals from sensors in different parts of the body to identify goal-oriented activities and ADLs such as *cooking pasta* and *making tea* is more difficult to achieve in this context because the signals are highly variable. Since goal-oriented activities require interaction with objects in the environment, many recent papers [25,26,20,27,28] have shown that this problem is better addressed using option (3). This is the direction of research conducted in this paper. In our work, *human activities* refer to goal-oriented activities that involve the manipulation of objects (e.g. cooking, making tea) captured through the use of any object sensing technologies such as RFIDs.

The application of HAR to ADLs faces several challenges because of the large number of activities to be tracked. This is compounded by the observation that each activity can be performed in several ways. One major challenge is to find a robust model that can capture the most relevant information in mapping the low-level features to high-level concepts that can be optimally exploited by healthcare applications.

The straightforward way of obtaining activity models is to learn them from the data produced by each subject. The traditional recognition approach to acquire such models is to apply machine learning techniques to labeled and manually segmented sensor traces [29–31]. However, this approach is impractical in real-world scenarios as potential users have to spend a significant amount of time in generating and labeling the training data. Moreover, this approach does not directly address the segmentation problem which is a prerequisite step for activity recognition in most machine learning approaches.

Recent works [25,26,20,27,28,18,19] suggest an interesting and promising direction in feature detection. The approach is based on dense object-based sensing technology, where objects used in activities are tracked by RFID tags and the activity models are derived by mining the web [27]. The *order of objects* in performing activities with probabilistic distributions is extracted using supervised learning and used as the basis to represent activities either as Hidden Markov Models (HMM), Dynamic Bayesian Networks (DBN), or as Suffix Trees.

These models perform well at classifying hand-segmented *object use* traces. However, they rely on the *object order* that may fail to capture the intrinsic characteristics of any particular activity in real-world conditions. This is because an activity can be performed in several ways using different *order* and *cardinality* of objects. Failing to totally capture such characteristics of activities may significantly limit the accuracy and applicability of the model that relies particularly on object sequence.

In most cases of ADL, the lists of relevant objects for a particular activity are similar and do not vary significantly even when the activity is performed in different ways. Hence, it is reasonable to derive the model by extracting a complete set of relevant objects of an activity from a text corpus such as the web, rather than relying on the *order of objects*. This observation is the main motivation for the development of our activity recognition and segmentation algorithms.

In this paper, we propose unsupervised algorithms to segment the activity trace and recognize the corresponding activities. Our recognition and segmentation algorithms work under the following principles:

1. Each object has relatively *varying degrees* of relevance in different activities.
2. Certain objects are *highly discriminatory* and their presence can be used to recognize activities.
3. Comparing the *relative weights* of nearby objects in two adjacent activities can be used to detect their boundary.

We mined the web for each activity and extract the most relevant objects. An object's *relevance weight* is based on its normalized `tf-idf` (term frequency–inverse document frequency) scores [32–35]. In text mining and information retrieval communities, `tf-idf` is a common weighting scheme to determine the relative degree of importance of a term to a document in a corpus. In `tf-idf`, terms with high `tf-idf` has high term frequency occurrence in a group of documents but low global frequency in a corpus. For instance, in a collection of `howto` activities, the term "the" has a very low "tf-idf" score because it appears in almost all documents. In contrast, the term "coffee" will have a higher `tf-idf` score because it only has a high frequency in documents pertaining to "how to make coffee". A detailed discussion on how to compute the `tf-idf` score can be found in Section 3.2.

Our *KeyExtract* algorithm performs the recognition task by searching for highly discriminatory objects in an unsegmented *activity trace*. An *activity trace* contains a sequence of objects used in a set of consecutive non-interleaving activities. Furthermore, we propose two algorithms, *MaxGap* and *MaxGain*, to detect the boundary of any two adjacent activities in a trace based on the objects' relative weights. We analyze the effectiveness of our proposed algorithms using simulation and report our findings based on 100 randomly generated activity traces.

In summary, this paper makes three main contributions:

1. We propose the concept of relative weight discrimination by mining activities on the web. We extract objects involved in individual activities and assign corresponding weights to these objects based on their relevance to the activities. We then search these highly discriminatory objects to recognize activities.
2. We dissociate the recognition from the segmentation process. We propose a *KeyExtract* algorithm to recognize activities in a trace based on the *key objects* of individual activities, as well as two algorithms, *MaxGap* and *MaxGain*, to further segment the trace. A *key object* in an activity is the object with the highest `tf-idf` score.
3. We validate our algorithms using 100 random traces of 13 activities, and analyze their effectiveness and limitations.

As our first research attempt on HAR, we realize that the proposed algorithms in this paper still have certain drawbacks. First, our approach can only recognize and segment non-interleaving human activities in a pre-determined list, similar to most previous approaches [25,27,28,18,19]. Second, in order to give a contextual help to the subject, we need to mine the web to construct a model for each activity in the list beforehand and be able to situate the subject's actions in relation to this model. Moreover, we assume each activity in the list has a unique *key object*. If this assumption does not hold true, our algorithm's accuracy will degrade in proportion to the number of these activities sharing common *key objects*. For instance, our algorithm has no problem discriminating "fry eggs" and "make coffee" activities because their respective key objects (eggs and coffee) are unique. However, if we add in the list of activities "make mocha cake" in which coffee or egg can be a *key object*, our algorithms' performance will have lower accuracy. Lastly, an activity can only be recognized in our approach after the subject touches the *key object* of this activity, which makes the key objects critical for both the accuracy and timeliness of the HAR performance. While it is beyond the scope of this paper, we consider all these issues to be very important research directions of our ongoing work. On the other hand, we believe our approach is quite suitable to segment and recognize ADL traces for research purpose, i.e. other researchers can mine the data already segmented by our approach and extract more information.

Similar to most previous approaches [25,27,28,18,19], our paper focuses on addressing the recognition and segmentation problem for non-interleaving human activities. Handling interleaving activities remains a challenging problem in HAR. While it is beyond the scope of this paper, we consider it to be a very important research direction in our future work.

The rest of the paper is organized as follows. We present related work in Section 2. We describe how we build our activity models through web mining in Section 3. We propose our algorithms for activity recognition and segmentation in Section 4 and evaluate their performance in Section 5. We conclude the paper with future research directions in Section 6.

2. Related work

This section reviews the major approaches to human activity recognition. For HAR based on *object usage*, let A and O be a set of all activities and all objects, respectively, such that $a_i = \{o_1, \dots, o_m\}$, $a_i \in A$, $o_{1 \le j \le m} \in O$, and $\{o\} \subset O$. For a typical activity trace input consisting of a sequence of objects $[o_1, o_2, \dots, o_w]$ representing consecutive activities, HAR has two important subproblems:

1. Segmentation Problem. Find the corresponding set of segments or clusters C of objects belonging to unknown activities. In other words, formulate a transformation from $[o_1, o_2, \dots, o_w] \rightarrow [c_1, \dots, c_x]$ with $\{o\}_{1 \le m \le x} = c_{1 \le m \le x}$, $c_{1 \le m \le x} \in C$, and $x \le w$.
2. Recognition Problem. Identify the corresponding unknown activities for each segment of objects in an activity trace. In other words, find the appropriate function f such that the mapping of the subset of objects to a particular activity a_k, $f : \{o\}_m \rightarrow a_k$ or $f : c_m \rightarrow a_k$, is correct.

Major approaches to HAR based on object-sensing technology can be roughly divided into two major classes:

- sequence-based models such as Hidden Markov Model (HMM) [27,36] or Dynamic Bayesian Networks (DBN) [25,15], and Conditional Random Fields (CRF) [37,38]
- machine learning algorithms [29,24] such as Naïve Bayes Classifier, Artificial Neural Networks (ANN), and Support Vector Machines (SVM).

Unlike in the *human action recognition* researches where both sequence-based and machine learning models are commonly used [39,14,16,21,23], researches in object-sensing technologies for HAR tend to favor sequence-based models such as HMM, DBN, and CRF. One reason is that the latter models work well because human activities are composed of finite sequences of discrete events or states. Moreover, most machine learning approaches work on the assumption that the activity trace input is already segmented which is not realistic. Manual segmentation is time-consuming and not practical in a real-world setting and automatic segmentation is a non-trivial problem in HAR.

A typical approach in machine learning is to model the mapping of an activity based on the collection of objects that are active disregarding temporal or sequence information. One typical model with this assumption was proposed by Tapia et al. [29]. They relied on a set of sensors to detect object usage for each ADL activity included in their research. They employed a Naïve Bayesian classifier to predict the activity labels using a manually segmented input of the activity trace. While the learning of the mapping function may be trivial, their approach fails to address the segmentation problem which is a prerequisite step for the processing of inputs using this model. Moreover, their supervised machine learning technique typically requires pre-training of manually segmented and labeled data which is a time-consuming process.

An alternative and popular approach is to learn the probabilities both of the object's usage and order in all the activities they are involved. Philipose et al. [25] proposed to use RFID tags attached to objects of interest and represented activities as a probabilistic *order of objects* used. These activity models were then converted into DBN. They reported good results but their approach also involves hand labeling of the training data which is a time-consuming process in a real-world condition.

Perkowitz et al. [27] proposed an interesting direction towards an unsupervised approach to activity recognition by extending the work of Philipose et al. [25]. Recognizing the problem of using supervised learning for the DBN, they proposed the idea to mine the web using Google's API to determine automatically the probability values of object usage and their sequence for the DBN. However, without segmentation, they have to rely on the sliding window technique for recognition which is noisy in cases where the window contains a boundary. Also, they fail to study the ideal window size for optimal recognition. Moreover, modeling sequence information is a permutation problem which suggests that their model may fail

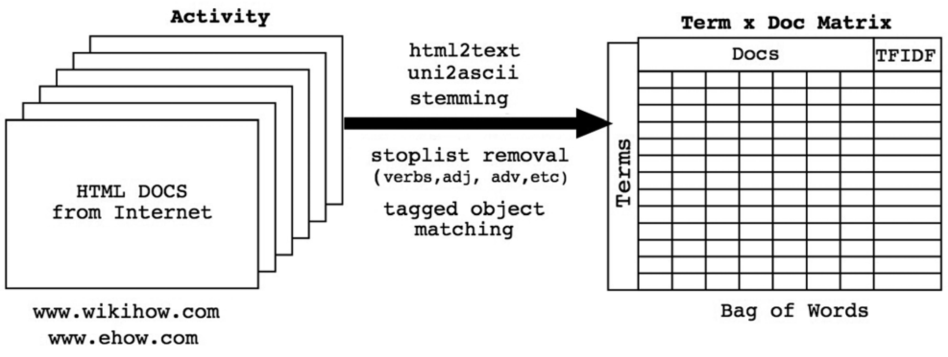


Fig. 1. Web-based activity mining.

to capture completely the sequence probabilities and the idiosyncrasy of certain activities. Failure to capture such intrinsic characteristics may limit their model's accuracy in real deployments.

Wyatt et al. [28] partially addressed the issue of completeness and idiosyncrasy by mining the web in a wider scope. They presented a bootstrap method that can produce labeled segmentations automatically. Their model abstraction uses a large number of labeled web pages as the training set in which human efforts are involved. They used these labeled web pages for the computation of the usage probabilities of objects which are needed to assemble an HMM for HAR. Then, they generate a generic model from the mined model using the Viterbi Algorithm and replace the HMM's observed probabilities with the computed Maximum Likelihood observation probabilities. Their segmentation algorithm is based on the likelihood that a particular observation is generated by a n-length path starting at a certain state and self-transitioning $n - 1$ times. They use Kullback-Leibler (KL) divergence for recognition by comparing the similarity between labeled activities with unlabeled ones. In this model, activity recognition is finding a match between labeled and unknown activities within a user-defined KL-distance threshold.

There are several shortcomings using Wyatt's proposed model. First, the Viterbi algorithm requires a significant amount of space to compute. Moreover, the KL-divergence between finite-state models cannot be solved mathematically and no computationally efficient algorithm exists. These issues limit the applicability of their model in real-world applications. Second, the segmentation is based on the duration of an activity that may vary greatly from one user to another. The accuracy may drop when their activity models were applied to real-world scenarios. Finally, although they used the generic mined models to segment the trace into labeled instances of activities, their segmentation process is sequential in nature such that any error in one segment may affect the segmentations of the subsequent traces.

The work in this paper borrows the idea of mining the web [28,27] to extract object usage information for activity modeling. Instead of using sequence information, however, our model relies on the appropriate weight assignment of relevant objects for discrimination. In this way, our model is not influenced by the idiosyncrasy of performing any activity by different individuals and is more practical for real-world deployment. The segmentation of the activity trace in two or more adjacent activities is an important and challenging issue in any activity recognition system. We address this problem by developing algorithms which operate locally within the context of two adjacent activities such that the segmentation process in one segment does not affect the segmentation process in other segments of the trace. This approach is in contrast to the sequential segmentation commonly employed in previous researches where an error in the segmentation process in one segment affects the subsequent ones. Using a simple model, our activity recognition and segmentation algorithms only require linear time complexity which is ideal for real-world deployment.

3. Activity model

To recognize an activity being performed by a subject, we first need to obtain the activity model. Given a set of activities $A = \{a_1, a_2, \dots, a_p\}$, we build our activity models consisting of a set of all terms $T = \{t_1, t_2, \dots, t_m\}$ used for each activity $a \in A$, together with their associated usage weights W. This process has two major steps as presented in the following subsections.

3.1. Mining the web

First, we obtain a set of terms $T = \{t_1, t_2, \dots, t_m\}$ used for defining each activity $a \in A$ from a set of `howto` instructions in the web. Similar to [27,28], we extract relevant instructions for a particular activity using search engines such as `Google`, or specific web sites such as `www.ehow.com`. Since the relevance of these instructions returned by search engines may vary significantly, we focus on two specific websites: `wikihow` (`www.wikihow.com`) and `ehow` (`www.ehow.com`). Both sites provide comprehensive instructions for many household activities such as *make tea*, *make coffee*, *brush teeth*, *take pills*, etc.

Fig. 1 shows a brief description of the processes involved in extracting relevant terms for a particular activity. Each `howto` activity has a collection of documents describing the different ways contributors of `ehow` and `wikihow` perform this activity.

Algorithm 1: Term Extraction Algorithm

```
Input: Objects: $O = \{o_1, o_2, o_3, \dots, o_n\}$ ;
       Howtos: $HWS = \{H_1, H_2, H_3, \dots, H_p\}$;
         where: $H_i = \{html \mid html \in \texttt{Activity } i\}, \quad i = 1, \dots, p$ ;
       StopList $S = \{s_1, s_2, s_3, \dots, s_q\}$;
         where: $s_i \in \{verbs, adjectives, pronoun, preposition\}$
Result: Terms $T = \{t_1, t_2, t_3, \dots, t_m\}$;
begin
    foreach $H \in HWS$ do
        foreach $html \in H$ do
            $words = html.tokenize()$;
            $words.stem()$ ;
            $words.remove(S)$;
            $words.match(O)$;
            T.$push(words)$;
    return T.unique;
end
```

First, a collection of these html `howtos` is transformed into plain text and stemmed using Porter's *Stemming Algorithm* [40]. In stemming, morphological variants of terms (e.g., singular vs. plural) which have similar semantic interpretations are considered to be equivalent and reduced to their stemmed or root forms. This process reduces the number of distinct terms needed to represent any activity, thereby saving processing time and storage space. Second, the number of relevant terms is further reduced by removing terms appearing in the stoplist [40–43,32] (e.g., verbs, adjectives, pronouns, adverbs, false nouns, etc.). Finally, only terms found in the database of objects are retained. The database of objects can be easily built by extracting information from a server that has stored the IDs of objects in the physical space. The filtering process makes sure that false nouns (e.g., water, switch, etc.) but with object mapping in the physical space do not appear in the stoplist. The simplifying assumption of using unordered collection of terms or words, disregarding sequence or grammar is popularly know as *bag-of-words* model in the text mining community [19]. The term extraction process is summarized in Algorithm 1. Note that in the algorithm, the final collection of terms has corresponding equivalent objects in the real world (filtered by *words.match()*).

3.2. Determining object relevance weight

Next, we identify the usage weights for each term obtained in the previous step for each activity. By observing that the occurrence frequency of a term in a particular instruction closely parallels the weight of the corresponding object in real usage, we determine the relevance weight W of each term $t \in a \in A$ by computing its `tf-idf` score [44,34,35].

The `tf-idf` computation is based on two factors: locality and generality of occurrence of the term in given documents. Locality (`tf`) is measured based on the total number of occurrences of the term over all documents. On the other hand, term's generality is based on the frequency of documents where the term occurs. Relevant terms have high term frequency but low document frequency [44,34,35]:

$$\texttt{tf-idf}_i^a = |t_i^a| \times \log \frac{|D^a|}{|d^a : t_i^a \in d^a| + 1} \tag{1}$$

where:

$$\begin{aligned} \texttt{tf-idf}_i^a &\rightarrow \texttt{tf} - \texttt{idf} \text{ of term } i \text{ in activity } a \in A; \\ |t_i^a| &\rightarrow \text{term frequency of term } i \text{ in activity } a \in A; \\ |D^a| &\rightarrow \text{total number of documents in activity } a \in A; \\ |d^a : t_i^a \in d^a| &\rightarrow \text{number of docs where term } i \text{ appears in activity } a \in A. \end{aligned}$$

Eq. (1) implies that if a term is too common, it occurs in almost all documents and will have a very low `tf-idf` score. The factor, $\log \frac{|D^a|}{|d^a : t_i^a \in d^a| + 1}$, demotes words if they occur in almost all documents (too general terms) and promotes words that occur in a limited number of documents (specific terms).

Since weight computation varies significantly from one activity to another because of heterogeneous sources, normalization is needed to establish a common basis of comparison among relative term weights in different activities. Eq. (2) describes the normalization technique used in this paper:

$$W_i^a = \frac{\log(\texttt{tf-idf}_i^a)}{\max_{j=1}^{n^a}\{\log(\texttt{tf-idf}_j^a)\}} \tag{2}$$

Table 1
A partial view of mined ADL models.

Make tea		Make coffee		Make pasta		Fry egg	
Object	Weight	Object	Weight	Object	Weight	Object	Weight
Tea	1.00	Coffee	1.00	Pasta	1.00	Egg	1.00
Water	0.85	Water	0.86	Flour	0.88	Pan	0.99
Cup	0.83	Cup	0.85	Pepper	0.85	Oil	0.78
Sugar	0.75	Pot	0.82	Water	0.84	Burner	0.76
Teapot	0.75	Grinder	0.80	Sauce	0.83	Spatula	0.69
Pot	0.74	Filter	0.79	Tomato	0.81	Lid	0.66
Bowl	0.72	Sugar	0.76	Cheese	0.80	Water	0.65
Lemon	0.70	Coffeemaker	0.73	Garlic	0.80	Bowl	0.60
Kettle	0.70	Creamer	0.72	Oil	0.80	Butter	0.60
Microwave	0.67	Tablespoon	0.72	Pot	0.79	Dish	0.60

Note: All weights are normalized using the log smoothing function.

where:

$$\begin{aligned} W_i^a &\rightarrow \text{normalized weight of term } i \text{ in activity } a \in A; \\ \texttt{tf-idf}_i^a &\rightarrow \texttt{tf} - \texttt{idf} \text{ of term } i \text{ in activity } a \in A; \\ n^a &\rightarrow \text{total number of terms in activity } a \in A. \end{aligned}$$

This normalization ensures that the topmost term has 1.0 weight and the relative distances of succeeding terms based on their weights do not have a high variability due to the smoothing effect of the `log` transformation. Moreover, this transformation lessens the strong bias in weights of the topmost terms.

We have mined 30 ADL models, and Table 1 lists a partial set of activity models mined from `ehow` and `wikihow` with their corresponding top 10 terms ordered by their normalized weights.

4. Algorithms for activity recognition and segmentation

Based on activity models obtained in the previous section, we can now apply these models for the development of activity recognition and segmentation algorithms.

4.1. Recognizing activities based on relevance weights

The activity models obtained (Table 1) reveal the sets of relevant terms used in activities. The topmost term has the highest relevance for each particular activity. Another important observation is that the topmost terms are unique among all the activity models. These observations hold true in most of the ADL models we mined. This suggests that recognition of activities can be based on searching the set of topmost terms with their normalized weights equal to 1.0. These topmost terms are used as the *key objects* to discriminate various activities.

Algorithm 2 describes the steps involved in our activity recognition algorithm (*KeyExtract*) based on the above observations. The occurrence of one of the *key objects* in a sequence of objects signifies the presence of an activity in the neighborhood of these objects. The *KeyExtract* algorithm iterates through the entire trace to search for other *key objects*.

In a real application, a subject may touch other *key objects* unintentionally (noise) while a particular activity is being performed. In this case, we examine the relevance of nearby objects based on their weights in the activity corresponding to the current *key object*. If the sum of all the weights including the *key object* is 1.0, it reveals that only the *key object* weight contributed to the summation while the rest of its neighbors have zero weights. This *key object* is considered as noise and removed in the trace. While there may exist a set of *key objects* for each activity $a \in A$, we use only one *key object* in this paper. The use of several objects as discriminants will be the subject of our future investigation.

4.2. Object-based activity segmentation

Once the activities have been identified by *KeyExtract*, the next task is to group objects in the trace according to their corresponding activities. In a one-dimensional trace of non-interleaving activities, the problem of clustering can be transformed into finding the boundary where the *object use* of one activity ends and the *object use* of the other activity begins. This problem is referred to as the *boundary detection problem* or *segmentation problem* which is depicted in Fig. 2.

To address this problem, we propose two segmentation algorithms which work in an unsupervised manner. We observe from our activity models that an object will have a high weight for a particular activity if it is important to this activity. On the other hand, the same object will have a low weight for activities it has less or no relevance. Further, the weights of adjacent objects in the same activity (e.g., object **b** and **c** in Fig. 2) do not significantly vary compared to the weights of two adjacent objects belonging to two different activities (e.g., objects **a** and **b** in Fig. 2). Based on these heuristics, we introduce our first segmentation algorithm: *MaxGap*.

Algorithm 2: Activity Recognition Algorithm (*KeyExtract*)

```
Input: Object Sequence: $O = [o_1, o_2, o_3, \ldots, o_n]$;
       Terms: $T = \{t_1, t_2, t_3, \ldots, t_m\}$;
       Activities: $A = \{a_1, a_2, a_3, \ldots, a_p\}$;
        where: $a_j = \{t \mid t \in T\}, \; j = 1, \ldots, p$
Result: DetectedActivities = $\{i \mid W_i^a = 1\}$ for certain activity $a$
begin
    foreach $o_i \in O$ do
        foreach $a \in A$ do
            foreach $t \in a$ do
                if $(t = o_i \mid W_i^a = 1.0)$ then
                    if $\sum_{j=i-1}^{i+1} W_j^a \geq 1.0 + e, \; e > 0$ (to handle noise) then
                        DetectedActivities.push($i$)
    return DetectedActivities;
end
```

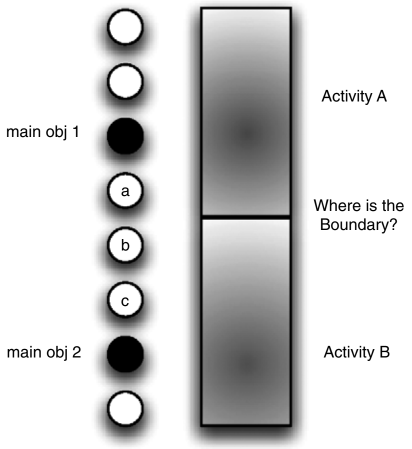
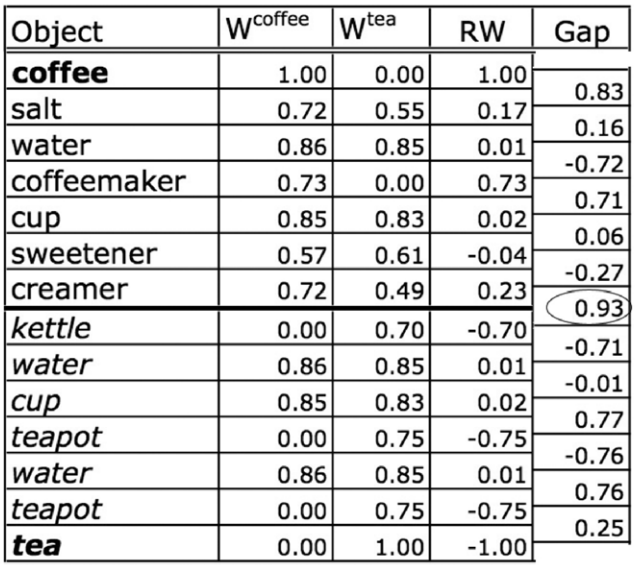


Fig. 2. Boundary detection problem.

Algorithm 3: MaxGap Algorithm

```
Input: Objects: $O = \{o_1, o_2, o_3, \ldots, o_n\}$ ;
       Terms: $T = \{t_1, t_2, t_3, \ldots, t_m\}$;
       Activities: $A = \{a_1, a_2, a_3, \ldots, a_p\}$;
        where: $a_j = \{t \mid t \in T\}, \; j = 1, \ldots, p$ ;
       DetectedActivities = $\{i \mid W_i^a = 1\}$ for certain activity $a$
Result: Boundaries
begin
    foreach $(x, y) \in$ DetectedActivities s.t. $o_x, o_y$ are adjacent do
        for $ctr = x$ to $y$ do
            $RW_{ctr} = W_{ctr}^x - W_{ctr}^y$;
        for $ctr = x$ to $y - 1$ do
            $GAP_{ctr} = RW_{ctr} - RW_{ctr+1}$
        Boundaries.push($ctr$) such that $GAP_{ctr}$ is maximum ;
    return Boundaries;
end
```



Object	W^{coffee}	W^{tea}	RW	Gap
coffee	1.00	0.00	1.00	
				0.83
salt	0.72	0.55	0.17	
				0.16
water	0.86	0.85	0.01	
				-0.72
coffeemaker	0.73	0.00	0.73	
				0.71
cup	0.85	0.83	0.02	
				0.06
sweetener	0.57	0.61	-0.04	
				-0.27
creamer	0.72	0.49	0.23	
				0.93
kettle	0.00	0.70	-0.70	
				-0.71
water	0.86	0.85	0.01	
				-0.01
cup	0.85	0.83	0.02	
				0.77
teapot	0.00	0.75	-0.75	
				-0.76
water	0.86	0.85	0.01	
				0.76
teapot	0.00	0.75	-0.75	
				0.25
tea	0.00	1.00	-1.00	

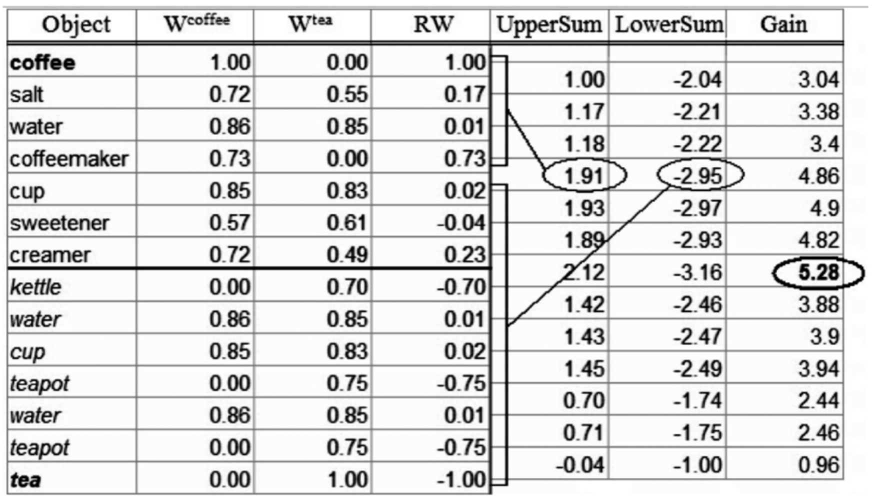
Fig. 3. Computing MaxGap.

To detect the boundary between two adjacent activities, **A** and **B**, the *MaxGap* algorithm computes the difference between the weight of each object in activity **A** and its weight in activity **B** (*RW*: Relative Weight). If the object is more relevant to **A** than **B**, then *RW* will be positive while the reverse, will be negative. It then computes the difference of each consecutive *RW* pairs (gap), and the maximum gap is the boundary for these two activities. Algorithm 3 outlines the *MaxGap* approach. The input of the algorithm is a sequence of objects, two ends of which are two different *key objects*. The output of the algorithm is the location of an object where we should separate the two activities. Fig. 3 shows an example of finding the boundary between two activities using *MaxGap*. The complexity of *MaxGap* is linear.

However, in cases where the two adjacent activities, **A** and **B**, share common objects, boundary detection will be complicated if the common objects are located nearby the boundary. This is because their *RWs* will be close to zero. Fig. 4 illustrates such case in which *cup* and *water* are shared by both *make tea* and *make coffee* activities. The *MaxGap* algorithm may fail to work in this case.

Algorithm 4: MaxGain Algorithm

```
Input: Objects: $O = \{o_1, o_2, o_3, \dots, o_n\}$ ;
       Terms: $T = \{t_1, t_2, t_3, \dots, t_m\}$;
       Activities: $A = \{a_1, a_2, a_3, \dots, a_p\}$;
         where: $a_i = \{t \mid t \in Terms\}$,   $i = 1, \dots, p$ ;
       DetectedActivities $= \{i \mid W_i^a = 1\}$ for certain objects $o_i$ in activity $a$
Result: Boundaries
begin
    foreach $(x, y) \in$ DetectedActivities | $o_x, o_y$ are adjacent do
        for $ctr = x$ to $y$ do
            $RW_{ctr} = W^x(o_{ctr}) - W^y(o_{ctr})$;
        for $ctr = x$ to $y$ do
            UpperSum=0; LowerSum=0;
            for $upper = x$ to $ctr$ do
                UpperSum += $RW_{upper}$;
            for $lower = ctr + 1$ to $y$ do
                LowerSum += $RW_{lower}$;
            $GAIN_{ctr}$ = UpperSum - LowerSum;
        Boundaries.push($ctr$) such that $GAIN_{ctr}$ is maximum ;
    return Boundaries;
end
```



Object	W^{coffee}	W^{tea}	RW	UpperSum	LowerSum	Gain
coffee	1.00	0.00	1.00	1.00	-2.04	3.04
salt	0.72	0.55	0.17	1.17	-2.21	3.38
water	0.86	0.85	0.01	1.18	-2.22	3.4
coffeemaker	0.73	0.00	0.73	1.91	-2.95	4.86
cup	0.85	0.83	0.02	1.93	-2.97	4.9
sweetener	0.57	0.61	-0.04	1.89	-2.93	4.82
creamer	0.72	0.49	0.23	2.12	-3.16	**5.28**
kettle	0.00	0.70	-0.70	1.42	-2.46	3.88
water	0.86	0.85	0.01	1.43	-2.47	3.9
cup	0.85	0.83	0.02	1.45	-2.49	3.94
teapot	0.00	0.75	-0.75	0.70	-1.74	2.44
water	0.86	0.85	0.01	0.71	-1.75	2.46
teapot	0.00	0.75	-0.75	-0.04	-1.00	0.96
tea	0.00	1.00	-1.00			

Fig. 4. A walk-through example for MaxGain.

Table 2
ADLs used in our experiments.

1	Make coffee	8	Use computer
2	Make tea	9	Take pills
3	Make pasta	10	Laundry
4	Make oatmeal	11	Read books
5	Fry eggs	12	Watch TV
6	Make orange juice	13	Make phone phone call
7	Brush teeth		

To address this issue, we propose *MaxGain* which is outlined in Algorithm 4. The input is the same as that of *MaxGap*. We walk through the *MaxGain* algorithm using an example shown in Fig. 4. The first column is a segment of objects extracted from a trace in one of our experiments described in the next section. *Coffee* and *tea* are the *key objects* for the *make coffee* and *make tea* activities, respectively. For each possible object X_i (a candidate boundary), we compute the *UpperSum* and *LowerSum* as shown in columns 4 and 5, respectively. The *UpperSum* is the sum of all *RWs* from *coffee* to X_i, while the *LowerSum* is the sum of all *RWs* from X_{i+1} to *tea*. We then compute the *Gain* for each candidate boundary, where *Gain* is defined as the difference between *UpperSum* and *LowerSum*.

The result is shown in the last column. The object with the maximum value of *Gain* (*MaxGain*) is the boundary. In the example shown in Fig. 4, the *creamer* object yields the maximum *Gain* of 5.28 which also indicates the location of the boundary in the ground truth.

Compared to *MaxGap*, *MaxGain* considers the interplay of the group of objects between two adjacent activities. *MaxGap* only makes use of the relationship between two adjacent objects. Intuitively, *MaxGain* tends to be more accurate and noise-tolerant since it is "globally optimized". The complexity of *MaxGain* is also linear. When calculating the *UpperSum* (or *LowerSum*) of each candidate boundary X_i, we can simply add (or subtract) the *RW* of X_{i-1}.

5. Evaluation and results

To evaluate our proposed algorithms, we randomly extract from `wikihow` or `ehow` websites a sample of 13 ADLs as shown in Table 2. They can generate more than 6 billion possible activity traces because in each activity, some object traces may be missed and missing object can influence the segmentation process. We use 100 random traces out of the 6 billion possibilities since the variability of the mean performances of the proposed algorithms based on their standard errors is relatively small at this size.

To evaluate the robustness of our proposed algorithms in the presence of noise, we generate two kinds of noise, namely: *dummy* and *neighborhood* noises. A *dummy* noise is an object with no weight [`tf-idf` $= 0$] added to the original activity trace. On the other hand, *neighborhood* noises are randomly selected neighboring objects added to the original trace.

The level of noise is controlled using a probability value from 0 (i.e., no noise) to 0.50. For each level of noise with 100 activity traces, the evaluation of each proposed algorithm is subjected to 20 trials to minimize the standard error of their mean performances. The performances of the proposed algorithms, *MaxGap* and *MaxGain*, are then compared to that of the *Random* and *MidPoint* algorithms.

As its namesake suggests, the *Random* algorithm randomly selects the location of the boundary between two key objects. It is expected to provide the worst performance among the four algorithms. If the average number of objects between any two key objects is M, *Random* algorithm has $1/M$ chance of hitting the correct boundary of two adjacent activities. On the

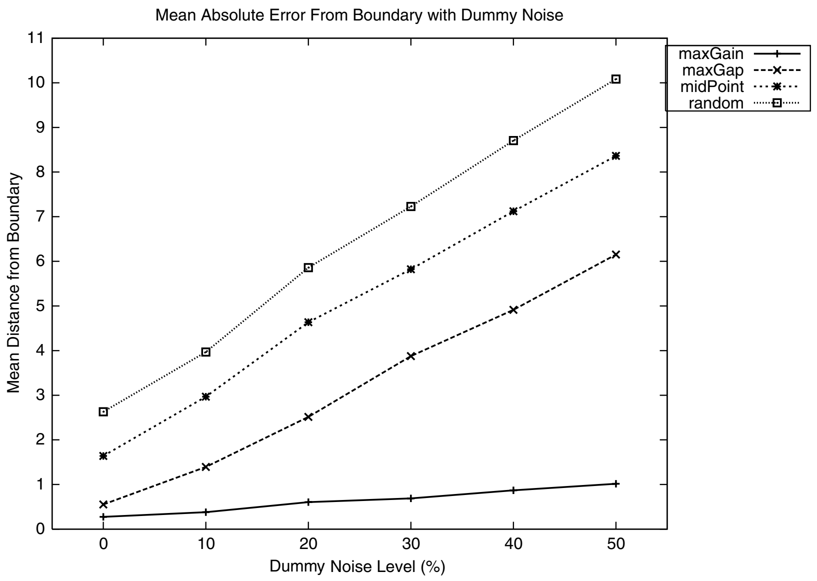


Fig. 5. Mean Absolute Error From Boundaries (MAEBs).

other hand, the *MidPoint* algorithm works on the heuristic that the boundary is always located midway between the two key objects. If the distribution of the true boundary is random, both the *MidPoint* and *Random* algorithms will have similar results.

The performance comparison is based on the two metrics, namely: (i) Mean Absolute Error from the true Boundary (MAEB) and (ii) Mean Percentage of the true Boundaries detected (MPB). MAEB is a continuous value that measures how many objects away is the algorithm's boundary from the true boundary. A good algorithm must have a MAEB value near zero. MPB, on the other hand, is the ratio between the number of true boundaries detected by the algorithm and the total number of boundaries. A good algorithm has a MPB value near 100%. MAEB and MPB are summarized in the following equations:

$$\text{MAEB} = \frac{\sum_{j=1}^{T}\sum_{i=1}^{N} |trB_{ij} - algB_{ij}|}{N * T} \tag{3}$$

$$\text{MPB} = \frac{\sum_{j=1}^{T} \frac{D_j}{B_j} \times 100}{T} \tag{4}$$

where:

trB	$\rightarrow$	true boundary;
$algB$	$\rightarrow$	algorithm boundary;
N	$\rightarrow$	total number of boundaries;
T	$\rightarrow$	number of traces;
D_j	$\rightarrow$	number of correctly detected boundaries;
B_j	$\rightarrow$	total number of true boundaries in a trace

5.1. *Results with dummy noise*

Fig. 5 shows the MAEB trend (averaged in 20 trials) of each algorithm as the level of *dummy* noise increases. *MaxGain* has the best performance followed by *MaxGap* with both *Random* and *MidPoint* performing the worst. *MaxGain*'s MAEB is only slightly affected by the increasing noise level. Its error from the boundary with no noise is 0.27 (less than one object) and at 50% noise level is only around 1 object. On the other hand, the performances of the rest of the algorithms significantly worsened with the increasing noise level.

Moreover, *MaxGain* accuracy in detecting true boundary as shown in Fig. 6 is consistently around 88% MPB independent of the noise level. *MaxGap* best performance is 79% average detection but decreases significantly with the increasing noise level. Both the *Random* and *MidPoint* algorithms have worst performances in all noise levels.

5.2. *Results with neighborhood noise*

Fig. 7 shows the same trend from the previous plots, i.e., *MaxGain* has the best MAEB followed by *MaxGap*. Again, both *Random* and *MidPoint* algorithms have consistently poor performances in all noise levels.

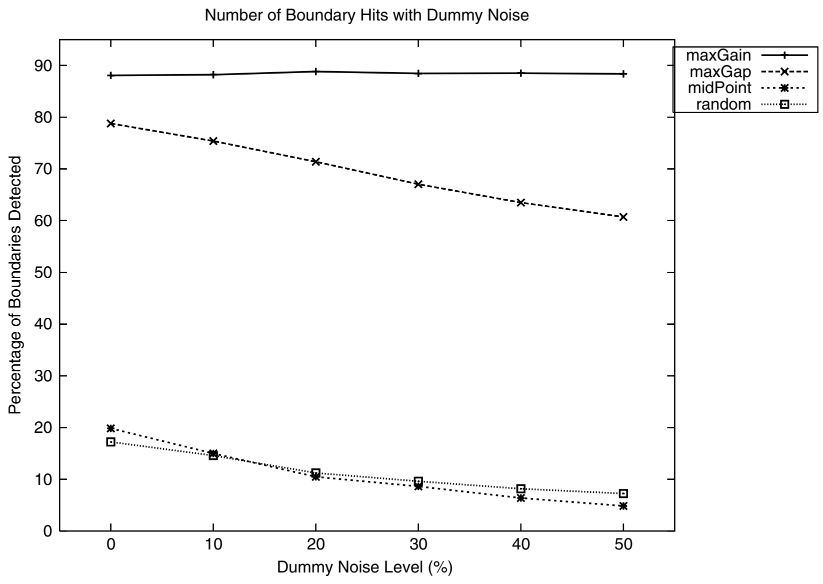


Fig. 6. Mean Percentage of Detected Boundaries (MPBs).

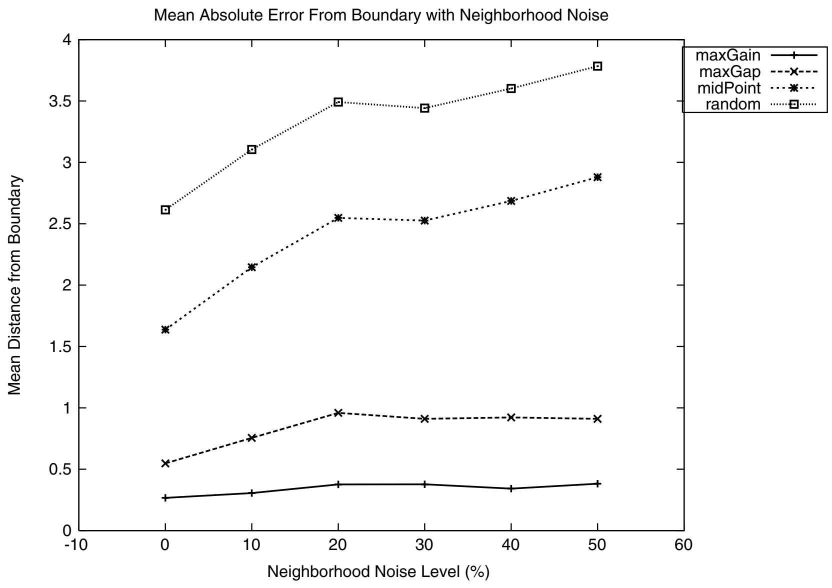


Fig. 7. Mean Absolute Error From Boundaries (MAEBs).

Comparing *MaxGain*'s MAEBs in Figs. 5 *and* 7 indicate that its performance in the presence of the *neighborhood* noise is better than its performance with the *dummy* noise. Its MAEB value for the *neighborhood* noise is consistently less than 0.4 in all noise levels. The same observation is true with the *MaxGap* performance, i.e., its MAEB with the *neighborhood* noise is better than its MAEB with the *dummy* noise. Its MAEB value is consistently above 0.5 but does not exceed 1.0 in all noise levels. Also, both *Random* and *MidPoint* MAEBs are better in the presence of *neighborhood* noise than with the *dummy* noise.

Fig. 8 shows that the MPBs of the four approaches are independent of the noise levels. Again, *MaxGain* has the best mean boundary detection rate (89%) followed by *MaxGap* (79%) with *Random* and *Midpoint* performing the worst (both less than 23%).

Comparing the performances of the algorithms between *dummy* and *neighborhood* noises suggest that the latter has lesser negative effect on the performance of four approaches. This is encouraging because in many cases, *neighborhood* noises are more common than *dummy* noises in the real setting. More often during data gathering of sensor output, noises are generated by unintentionally touching neighboring objects or touching the same object several times. While the presence of *dummy* noises always increases the search space between two adjacent *key objects*, the presence of neighborhood noises may actually reduce the search space. This is due to the possibility that the *key object* may appear several times in one activity. Since

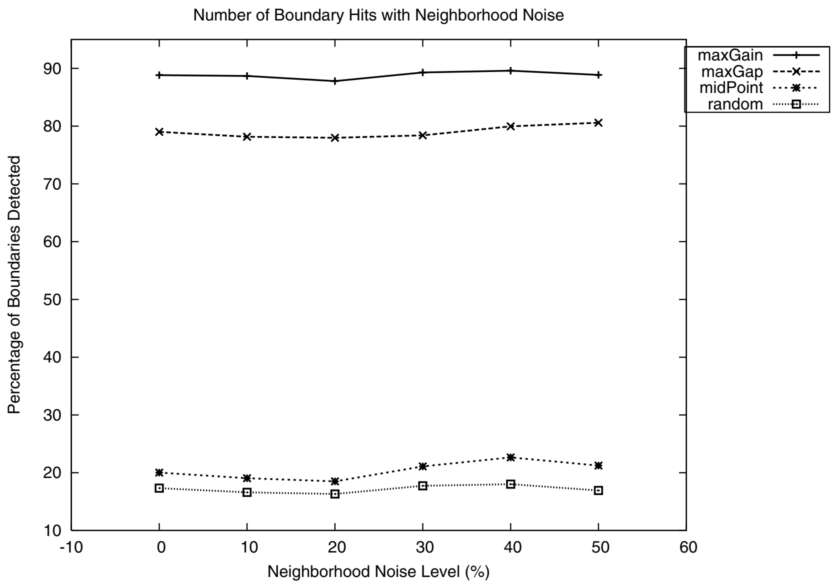


Fig. 8. Mean Percentage of Detected Boundaries (MPBs).

the four algorithms share the common technique of skipping the search space if the two adjacent activities are the same, it reduces the search space in cases where the addition of neighboring objects include the *key object*. However, this technique only works on the assumption that the *neighborhood* noise does not contain the *key object* belonging to another activity.

5.3. Discussion

In examining the behavior of the above four boundary detection algorithms, we made sure that *precision* and *recall* of the *KeyExtract* algorithm was maintained at 100% to maximize the number of boundaries to be detected. In real scenarios, however, *precision* and *recall* are affected based on the two types of noises: *False negative* (*fn*) noise and *False positive* (*fp*) noise:

$$precision = \frac{tp}{tp + fp} \tag{5}$$

$$recall = \frac{tp}{tp + fn} \tag{6}$$

where:

$tp \rightarrow$ true positive;

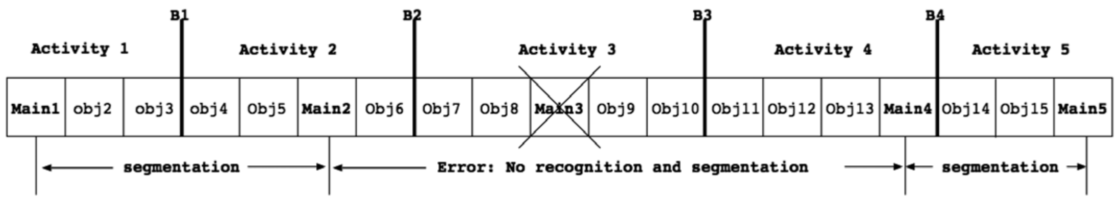
$fp \rightarrow$ false positive;

$fn \rightarrow$ false negative.

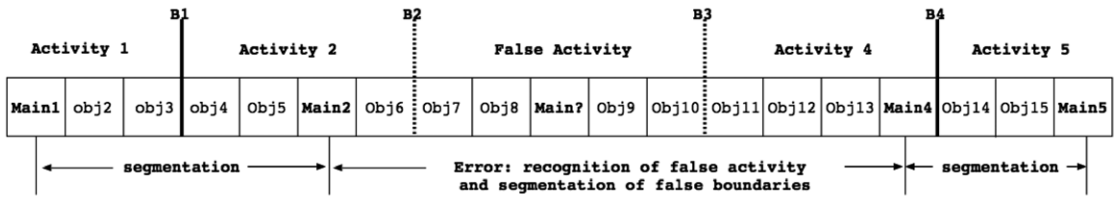
False negative (*fn*) noise happens when a key object in the original trace is lost in the transmission or reported falsely as another ordinary object by the sensor. This happens for example when an RFID reader fails to report the tag of the key object or it receives corrupted IDs. This type of error is a common problem among all the sensor-based activity recognition systems. Efforts to improve the reliability of sensor deployment and acquisition process require further investigation. Fig. 9a illustrates this particular case. Assuming that the original trace has 5 activities identified by 5 *key objects*, error in reporting *Main3* would result in a failure of detecting *Activity 3*. The effect of noise, however, is local and has no influence on the segmentation and recognition of activities in the other parts of the trace.

False positive (*fp*) noise happens when an ordinary object in the original trace is falsely reported by the sensor as a *key object*. Another case of false recognition occurs when a subject unintentionally touches another key object while an activity is being performed. Fig. 9b illustrates this particular case. Assuming that the original trace contains 4 activities identified by the 4 *key objects*, an object between *obj8* and *obj9*, reported by the sensor as a *key object* would result in a false recognition of additional activity. Similar to the observation above, the effect of noise is local and does not affect the performance of segmentation and recognition in the other parts of the trace.

Computation of *precision* and *recall* require actual experiments that will be the subject of our future investigation. Similar to existing approaches, our *KeyExtract* algorithm depends on the reliability of the sensor deployment which is the subject of ongoing investigation in many research laboratories. Currently, there are still many issues on the reliability of the RFID tag readers and other sensors. One current option to offset this shortcoming and to maximize *recall* will be to install redundant sensors on the *key objects* to ensure that their usage will be properly detected and accurately reported. Also, *precision*


B1
B2
B3
B4
Activity 1
Activity 2
Activity 3
Activity 4
Activity 5
Main1 obj2 obj3 obj4 Obj5 Main2 Obj6 Obj7 Obj8 Main3 Obj9 Obj10 Obj11 Obj12 Obj13 Main4 Obj14 Obj15 Main5
segmentation
Error: No recognition and segmentation
segmentation

(a) False negative.


B1
B2
B3
B4
Activity 1
Activity 2
False Activity
Activity 4
Activity 5
Main1 obj2 obj3 obj4 Obj5 Main2 Obj6 Obj7 Obj8 Main? Obj9 Obj10 Obj11 Obj12 Obj13 Main4 Obj14 Obj15 Main5
segmentation
Error: recognition of false activity and segmentation of false boundaries
segmentation

(b) False positive.

Fig. 9. False negative and false positive noises.

can be improved by carefully planning an environmental setup that minimizes background noise due to unintentional touching of unrelated objects in performing a particular activity. These shortcomings will hopefully be overcome as the sensor technology matures in the near future.

6. Conclusion and future work

We propose an unsupervised approach to human activity recognition and segmentation in this paper. Instead of relying on the *order of object use*, our approach exploits the discriminative trait of the *usage frequency* of objects in different activities. We construct activity models by mining `howto` web pages containing instructions of household activities and extract relevant objects based on their weights (normalized `tf-idf` scores). The weights are then utilized to recognize and segment an activity trace containing a sequence of objects used in a number of consecutive and non-interleaving activities.

We divide the activity recognition and segmentation into two separate processes. Based on a consistent pattern observed in real-world traces, we propose an activity recognition algorithm, *KeyExtract*, which uses the list of discriminatory *key objects* from all activities to identify the activities present in a trace. We further propose two heuristic segmentation algorithms, *MaxGap* and *MaxGain*, to detect the boundary between each pair of activities identified by *KeyExtract*. The boundary detection is based on the calculation, aggregation, and comparison of the relative weights of all objects sandwiched in any two *key objects* representing adjacent activities in a trace.

We have conducted simulation studies to validate the effectiveness of the proposed algorithms using 100 random traces of 13 activities. The results demonstrate that our algorithms achieve much better accuracy in activity boundary detection than *Random* and *MidPoint* algorithms under different noise levels.

One important issue that needs special attention in our future endeavor is how to deal with *interleaving* activities. The majority of supervised machine learning approaches suffer from the strong assumption that the activities have already been segmented without discussing the process. On the other hand, temporal classification models such as CRF (Conditional Random Fields) and HMM (Hidden-Markov Models) assume that the activities are performed sequentially. Addressing the problems in *interleaving* activities has important implications in real deployment setting. More often, incomplete activities especially for aged people occur due to interruptions in performing another activity that needs urgent attention such as receiving a phone call while boiling water or cooking. Detecting these interrupted and not completed activities are vital for the safety of aged people living alone. We believe that adding more features such as sound [45], location [46], and other context information may help address this problem in our future research work.

One direction of research we are investigating is how to effectively use several objects as discriminants [47,48] to make the system more robust and improve recognition accuracy. Our undergoing research also tackles the problem in interleaving activities. The challenge is to identify the different concurrent activities and develop a suitable clustering algorithm to group objects into their corresponding activities. To approach this problem, we extend the idea of relevance weighting discrimination by incorporating pairwise probability sequence information and apply a variant of K-Nearest Neighbor (KNN) clustering with dynamic backtracking to enable cluster membership correction. Our preliminary results for two concurrent activities are encouraging but more work has to be done to tackle three or more concurrent activities.

Acknowledgements

We would like to thank the editors and anonymous reviewers for their valuable comments and suggestions to improve the readability and technical quality of our paper.

References

[1] World Population Ageing: 1950-2050. URL: http://www.un.org/esa/population/publications/worldageing19502050/.
[2] E. Mynatt, I. Essa, W. Rogers, Increasing the opportunities for aging in place, in: ACM Conference on Universal Usability, ACM Press, 2000, pp. 65–71.
[3] W. Mann, S. Helal, Pervasive computing research on aging, disability and independence, in: International Symposium on Applications and the Internet Workshops, 2004, pp. 244–246.
[4] S. Katz, R.W. Moskowitz, A.B. Ford, B.A. Jackson, M.W. Jaffe, Studies of illness in the aged. the index of adl: A standardized measure of biological and psychological function, Journal of the American Medical Association 185 (1963) 914–919.
[5] I. McDowell, C. Newell, Measuring Health: A Guide to Rating Scales and Questionnaires, 2nd edition, Oxford University Press, New York, USA, 1996.
[6] S. Consolvo, P. Roessler, B. Shelton, A. LaMarca, B. Schilit, S. Bly, Technology for care networks of elders, Pervasive Computing, IEEE 3 (2) (2004) 22–29.
[7] K.Z. Haigh, J. Phelps, C.W. Geib, An open agent architecture for assisting elder independence, in: The First International Joint Conference on Autonomous Agents and MultiAgent Systems, AAMAS, 2002, pp. 578–586.
[8] S. Intille, K. Larson, Designing and evaluating supportive technology for homes, in: Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics, IEEE Press, 2003.
[9] J. Ben-Arie, Z. Wang, P. Pandit, S. Rajaram, Human activity recognition using multidimensional indexing, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (8) (2002) 1091–1104.
[10] V. Kellokumpu, M. Pietikäinen, J. Heikkilä, Human activity recognition using sequences of postures, in: IAPR Conference on Machine Vision Applications, MVA, Tsukuba Science City, Japan, 2005, pp. 570–573.
[11] J.K. Aggarwal, Q. Cai, Human motion analysis: A review, Computer Vision and Image Understanding: CVIU 73 (3) (1999) 428–440.
[12] R. Polana, R. Nelson, Detecting activities, in: DARPA93, 1993, pp. 569–574.
[13] Y. Yacoob, M.J. Black, Parameterized modeling and recognition of activities, in: ICCV, 1998, pp. 120–127.
[14] J. Ward, P. Lukowicz, G. Troester, T. Starner, Activity recognition of assembly tasks using body-worn microphones and accelerometers, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (10) (2006) 1553–1567.
[15] J. Muncaster, Y. Ma, Activity recognition using dynamic bayesian networks with automatic state selection, in: IEEE Workshop on Motion and Video Computing, 2007.
[16] C. Sminchisescu, A. Kanaujia, Z. Li, D. Metaxas, Conditional models for contextual human motion recognition, in: IEEE International Conference on Computer Vision, ICCV, vol. 2, 2005, pp. 1808–1815.
[17] M. Aoba, Y. Takefuji, Motion feature extraction using second-order neural network and self-organizing map for gesture recognition, in: IPSJ Digital Courier, vol. 1, 2005, pp. 268–281.
[18] R. Hamid, S. Maddi, A. Bobick, 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, ACM, New York, NY, USA, 2006, pp. 71–78. doi:10.1145/1178782.1178794.
[19] R. Hamid, S. Maddi, A. Bobick, M. Essa, Structure from statistics — Unsupervised activity analysis using suffix trees, in: IEEE 11th International Conference on Computer Vision, ICCV 2007, 2007, pp. 1–8.
[20] B. Logan, J. Healey, M. Philipose, E. Munguia-Tapia, S. Intille, A long-term evaluation of sensing modalities for activity recognition, in: Proc. of Ubicomp 2007, 2007, pp. 483–500.
[21] L. Bao, S.S. Intille, Activity recognition from user-annotated acceleration data, in: Proc. of Pervasive 2004, 2004, pp. 1–17.
[22] T. Huỳnh, B. Schiele, Unsupervised discovery of structure in activity data using multiple eigenspaces, in: Proc. of the 2nd International Workshop on Location and Context-Awareness, LoCA 2006, 2006, pp. 151–167.
[23] T. Huỳnh, B. Schiele, Towards less supervision in activity recognition from wearable sensors, in: Proc of the 10th IEEE International Symposium on Wearable Computing, ISWC 2006, 2006, pp. 3–10.
[24] N. Ravi, N. Dandekar, P. Mysore, M.L. Littman, Activity recognition from accelerometer data, American Association for Artificial Intelligence.
[25] M. Philipose, K.P. Fishkin, M. Perkowitz, D.J. Patterson, D. Fox, H. Kautz, D. Hähnel, Inferring activities from interactions with objects, IEEE Pervasive Computing 3 (4) (2004) 50–57.
[26] D. Patterson, D. Fox, H. Kautz, M. Philipose, Fine-grained activity recognition by aggregating abstract object usage, in: Proc. of ISWC 2005, 2005, pp. 44–51.
[27] M. Perkowitz, M. Philipose, D.J. Patterson, K.P. Fishkin, Mining models of human activities from the web, in: Proc. of the 13th International World Wide Web Conference, WWW 2004, 2004, pp. 573–582.
[28] D. Wyatt, M. Philipose, T. Choudhury, Unsupervised activity recognition using automatically mined common sense, in: Proc. of AAAI 2005, 2005, pp. 21–27.
[29] E. Tapia, S.S. Intille, K. Larson, Activity recognition in the home using simple and ubiquitous sensors, in: Proc. of Pervasive 2004, 2004, pp. 158–175.
[30] E.M. Tapia, S.S. Intille, K. Larson, Portable wireless sensors for object usage sensing in the home: Challenges and practicalities, in: AmI, 2007, pp. 19–37.
[31] Y. Matsuo, N. Okazaki, K. Izumi, Y. Nakamura, T. Nishimura, K. Hasida, H. Nakashima, Inferring long-term user property based on users' location history, in: Inferring Long-term User Property Based on Users' Location History, 2007, pp. 2159–2165.
[32] W.B. Frakes, R.A. Baeza-Yates, Information Retrieval: Data Structures & Algorithms, Prentice-Hall, 1992.
[33] S. Usui, P. Palmes, K. Nagata, T. Taniguchi, N. Ueda, Keyword extraction, ranking, and organization for the neuroinformatics platform, Biosystems 88 (3) (2007) 334–342.
[34] G. Salton, Developments in automatic text retrieval, Science 253 (1991) 974–979.
[35] G. Salton, M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, 1983.
[36] J. Gao, A. Hauptmann, A. Bharucha, H. Wactlar, Dining activity analysis using a hidden markov model, in: 17th International Conference on Pattern Recognition, ICPR, vol. 2, 2004, pp. 915–918.
[37] J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in: Proc. 18th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA, 2001, pp. 282–289.
[38] C. Sutton, A. McCallum, K. Rohanimanesh, Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data, The Journal of Machine Learning Research 8 (2007) 693–723.
[39] M. Brand, N. Oliver, A. Pentland, Coupled hidden markov models for complex action recognition, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997, pp. 994–999.
[40] M. Porter, An algorithm for suffix stripping. program, Program 14 (3) (1980) 130–137.
[41] Stopword List. URL: http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words.
[42] English Club. URL: http://www.englishclub.com/.
[43] Adjective List. URL: http://www.keepandshare.com/doc/view.php?u=12894.
[44] M. Berry, Survey of Text Mining: Clustering, Classification, and Retrieval, Springer-Verlag, 2004.
[45] F. Kraft, R. Malkin, T. Schaaf, A. Waibel, Temporal ica for classification of acoustic events in a kitchen environment, in: Proc. Interspeech 2005, pp. 2689–2692.
[46] G. Yavas, D. Katsaros, Özgür Ulusoy, Y. Manolopoulos, A data mining approach for location prediction in mobile environments, Data & Knowledge Engineering 54 (2) (2005) 121–146. doi:10.1016/j.datak.2004.09.004.
[47] R. Agrawal, R. Srikant, Fast algorithms for mining association rules, in: J.B. Bocca, M. Jarke, C. Zaniolo (Eds.), Proc. 20th Int. Conf. Very Large Data Bases, VLDB, Morgan Kaufmann, 1994, pp. 487–499. URL: http://citeseer.ist.psu.edu/agrawal94fast.html.
[48] Y. Li, S.M. Chung, Text document clustering based on frequent word sequences, in: CIKM'05: Proceedings of the 14th ACM International Conference on Information and Knowledge Management, ACM, New York, NY, USA, 2005, pp. 293–294. doi:10.1145/1099554.1099633.

Paulito Palmes received his Doctor of Engineering degree from the Toyohashi University of Technology (TUT), Japan. He currently works in the Research Department of the National Neuroscience Institute in Singapore. His research interests are in the fields of pattern mining and recognition, machine learning, evolutionary neural networks, genetic algorithms, and more recently in fMRI/DTI neuroimaging analysis.

Hung-Keng Pung (http://www.comp.nus.edu.sg/punghk) is an Associate Professor of the Department of Computer Science at the National University of Singapore (NUS). He received his Ph.D. from the University of Kent at Canterbury (UK) before joining NUS in 1986 as a faculty. He heads the Networks Systems and Services Laboratory and held a joint appointment as a Principal Scientist at Institute of Infocomm Research (Singapore) for several years. His research interests are context-aware systems; service oriented computing; quality of service management, protocol design and networking.

Tao Gu was a research scientist of the Institute for Infocomm Research (Singapore) before joining the Department of Mathematics and Computer Science at University of Southern Denmark recently, as an Assistant Professor. He received his Ph.D. in computer science from National University of Singapore in 2005 under the supervision of Prof Pung. His current research interests involve various aspects of ubiquitous and pervasive computing including architectures, tools, distributed lookup algorithms, and activity detection.

Wenwei Xue received his Ph.D. degree in Computer Science from the Hong Kong University of Science and Technology in May 2007. He currently works as a member of research staff in the Scalable Rich Context Data Processing team, Nokia Research Center, Beijing. His research interests include context-aware middleware infrastructure, peer-to-peer networks, sensor query processing and application-driven systems design for pervasive computing.

Shaxun Chen received his B.S. and M.Sc. degrees in Computer Science from Nanjing University, China, in 2005 and 2008, respectively. He was a research assitant in Institute for Infocomm Research, Singapore, in 2007. He is now a Ph.D. student at UC Davis.