A Pattern Mining Approach to Sensor-Based Human Activity Recognition

Tao Gu, *Member*, *IEEE*, Liang Wang, *Student Member*, *IEEE*, Zhanqing Wu, Xianping Tao, *Member*, *IEEE*, and Jian Lu

Abstract—Recognizing human activities from sensor readings has recently attracted much research interest in pervasive computing due to its potential in many applications, such as assistive living and healthcare. This task is particularly challenging because human activities are often performed in not only a simple (i.e., sequential), but also a complex (i.e., interleaved or concurrent) manner in real life. Little work has been done in addressing complex issues in such a situation. The existing models of interleaved and concurrent activities are typically learning-based. Such models lack of flexibility in real life because activities can be interleaved and performed concurrently in many different ways. In this paper, we propose a novel pattern mining approach to recognize sequential, interleaved, and concurrent activities in a unified framework. We exploit Emerging Pattern—a discriminative pattern that describes significant changes between classes of data—to identify sensor features for classifying activities. Different from existing learning-based approaches which require different training data sets for building activity models, our activity models are built upon the sequential activity trace only and can be applied to recognize both simple and complex activities. We conduct our empirical studies by collecting real-world traces, evaluating the performance of our algorithm, and comparing our algorithm with static and temporal models. Our results demonstrate that, with a time slice of 15 seconds, we achieve an accuracy of 90.96 percent for sequential activity, 88.1 percent for interleaved activity, and 82.53 percent for concurrent activity.

Index Terms—Human activity recognition, pattern analysis, emerging pattern, classifier design and evaluation.

1 Introduction

With the rapid advances of sensors and wireless networks, recognizing human activity based on sensor readings has recently drawn much research interest in pervasive computing. Different from video-based activity recognition systems leveraging on video camera, in this paradigm, physical sensors are typically deployed to collect observations. Observations are used to train an appropriate activity model; the trained model can then be used to assign new observations with activity labels. A typical application is monitoring Activities of Daily Living (ADLs) [1] for the elderly and cognitively impaired people, and providing them with proactive assistance.

Human activity recognition is generally done at three different levels—action, ADL, and high level. At the action level, the physical actions of a user such as walking, sitting and running [2], [5], [9] are of concern. At the ADL level, the focus is on ADLs[1] and most of the existing work [5], [6], [7], [8], [10], [12], [13], [14], [15], [16], [17], [18], [20], [22], [23], [24], [25], [26] is done at this level. At the high level, ADLs are merged into several coarse-grained categories, such as Information/Leisure, Personal and Cleaning [16], [21]. The work reported in this paper concerns recognizing activities at the ADL level.

Humans usually perform ADLs in a sequential manner (i.e., one activity after another in a timeline), such as brushing teeth followed by washing face, and preparing a meal followed by eating the meal. However, in real life, there exist many complex situations since people often multitask when performing their activities. Such multitasking may occur in an interleaved (i.e., switching between the steps of two or more activities) or concurrent (i.e., performing two or more activities simultaneously) manner. Interleaved activities can be, e.g., while eating a meal in the dining room, a user may go to the living room to answer a phone call and come back to resume the meal; and concurrent activities can be, e.g., a user may drink coffee while watching TV. Recognizing activities in such a complex situation has a practical implication to real-life applications and motivates this work.

Activity recognition can be typically viewed as a classification problem where many traditional machine learning techniques can be applied to. Most of the existing work apply machine learning models to recognize only sequential activities in different settings [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [20], [24], [25], [26]. Little work has been done in addressing complex issues rise in recognizing sequential, interleaved, and concurrent activities.

Aiming to recognize both simple and complex activities[2] in a unified framework, in this paper, we propose a novel

1. We use ADLs, daily activities, and activities interchangeable throughout the paper.

2. We refer simple activity to sequential activity, complex activity to interleaved and concurrent activity in this paper.

• *T. Gu is with the Department of Mathematics and Computer Science, University of Southern Denmark, Campusvej 55, DK-5230 Odense M, Denmark. E-mail: gu@imada.sdu.dk.*
• *L. Wang, Z. Wu, X. Tao, and J. Lu are with the State Key Laboratory for Novel Software Technology, Department of Computer Science, Nanjing University, 22 Hankou Road, MMW Building, Nanjing 210093, China. E-mail: wl@smail.nju.edu.cn, {wuzhanqing, txp, lj}@nju.edu.cn.*

Manuscript received 17 Feb. 2009; revised 17 Aug. 2009; accepted 15 Apr. 2010; published online 23 Sept. 2010.
Recommended for acceptance by S. Parthasarathy.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2009-02-0074.
Digital Object Identifier no. 10.1109/TKDE.2010.184.

Emerging-Patterns-based approach to recognize sequential, interleaved, and concurrent activities. We formulate activity recognition as a pattern-based classification problem and build our activity models based on Emerging Pattern [43]—a discriminative pattern that describes significant changes between two classes of data. We apply Emerging Patterns to mine a set of unique, contrast sensor features for each activity class, and use these sets of features to recognize activities. Our interleaved and concurrent activity models are built directly from the sequential model, eliminating the training process; hence, it has a great flexibility and applicability for real-life applications. We also propose a novel trace segmentation algorithm based on the concept of feature relevance to segment the boundary of any two adjacent activities. We conduct a real-world trace collection done by four volunteers in a smart home. Through comprehensive experimental studies, we demonstrate both the effectiveness and flexibility of our proposed algorithms.

In summary, the paper provides the following contributions.

- *We formulate activity recognition as a pattern-based classification problem, and propose an Emerging-Patterns-based approach to recognize both simple and complex activities in a unified framework.* Differing from existing solutions that require an individual training process for each activity model, our interleaved and concurrent models are built directly from the sequential model.
- *We propose a novel trace segmentation algorithm based on feature relevance to segment the boundary of any two adjacent activities.* Our segmentation algorithm operates locally such that the detection of a boundary does not affect that of subsequent boundaries. This is in contrast to segmentation in previous work where an error in a segmentation process affects the subsequent ones.
- *We conduct a real-world trace collection and evaluate our algorithms through comprehensive experimental studies.* To the best of our knowledge, comparing to similar traces collected in [14], [27], our trace contains more comprehensive cases for studying interleaved and concurrent activities.

The rest of the paper is organized as follows: Section 2 discusses the related work. In Section 3, we describe our sensor platform and data preprocessing. Section 4 gives the background on Emerging Patterns, and the mining of Emerging Patterns. We then present our activity recognition algorithm in Section 5. Section 6 reports our empirical studies, and finally, Section 7 concludes the paper.

2 Related Work

Much work in human activity recognition [28], [29], [30], [31], [32], [33], [34], [35], [36], [37], [38], [39], [40], [41] has been done in computer vision. They leverage on video cameras as passive sensors to recognize people's actions from video sequences (i.e., activity recognition is done in the Action level). These works explore various tracking methods and spatiotemporal analysis to track moving objects and sense what a user is doing. A common theme in these works is the exploration of spatiotemporal video features [28], [31], [37], [39], [40], [41]. Temporal constraints on actions are usually addressed using probabilistic models such as Hidden Markov Models (HMMs) [32], [33], [34], [36], [37], [38] and Stochastic Context-Free Grammars [30], or explicit temporal correlation methods [41]. One of the earliest approaches to recognize human actions via HMMs was proposed by Yamato et al. [32] where they recognized tennis shots such as backhand stroke, backhand volley, smash, etc., by modeling a sequence of background subtracted images as outputs of class-specific HMMs. Huang et al. [39] used Dynamic Belief Networks (DBNs) for vision-based traffic monitoring. DBNs encode more complex conditional dependence relations among several random variables as opposed to just one hidden variable as in a traditional HMM.

In pervasive computing, researchers are recently interested in recognizing activities using sensors that directly measure user's movement, user location, living environment, and human-object interaction. Typical sensors can range from wall-mounted sensors (e.g., switch sensor [3], [4], [7], [11], [20], [27], infrared motion sensor [4], [11], and pressure sensor [11]) to wearable sensors (e.g., three-axis accelerometer [2], [5], [6], [8], [9], [12], [13], [17], [24], wrist-worn RFID sensor [4], [6], [10], [14], [15], [16], [17], [27], and temperature, humidity, and light sensors [5], [8], [12], [17], [24]).

There are two major models for recognizing human activities from artificial intelligence: logic-based approach and probabilistic approach. Early approaches such as [42] were based on logic. In this model, activities are described as a logical inference process of circumscription, and represented by a set of first-order statements called event hierarchy. However, logic-based approaches have limitations in distinguishing among consistent plans and have problems to handle uncertainty and noise in sensor data.

Probabilistic models are more appropriate and gain popularity as sensor readings are noisy and human activities are typically performed in a nondeterministic fashion. Probabilistic models can be generally categorized into static classification or temporal classification. In static classification, a variety of features is first extracted from sensor readings, and then a static classifier is applied to classify activities. Typical static classifiers include naïve Bayes used in [2], [3], [4], [6], decision tree used in [2], [4], [8], k-nearest neighbor used in [2], [4], [5], [8], and support vector machine used in [5]. Among these static classifiers, [2], [4] report that decision tree outperforms the rest. Multiple binary classifiers can be exploited to recognize interleaved and concurrent activities; however, this solution may not work properly because many activities share the common features.

In temporal classification, state-space models are typically used to enable the inference of hidden states (e.g., activity labels) given the observations (i.e., sensor readings). We name a few examples here: HMMs used in [12], [13], [14], [15], [16], [17], [18], [20], DBN used in [10], and Conditional Random Fields (CRFs) used in [18], [19], [20], [21], [22], [23]. Vail et al. [18] reported that CRFs score higher or on par with HMMs in terms of classification accuracy, and also demonstrated CRFs are able to incorporate features that are difficult to represent in an HMM model, namely features that link state transitions directly to observations. In a different setting, Kasteren et al. reported that CRFs outperform HMMs in all the cases with respect to time-slice accuracy, but HMMs achieve the overall highest class accuracy.

Recent work showed that Skip-Chain Conditional Random Field (SCCRF) [20], [21]—a variant of CRF—and Interleaved Hidden Markov Model (IHMM) [23]—a variant of HMM—can be used to model interleaved activities, and Factorial Conditional Random Field (FCRF) [19]—another variant of CRF—can be used to model concurrent activities. However, like other machine-learning-based techniques, they require predicting instances must have their models presented in the training data set. On one hand, this implies that the training data set has to be large enough to build the complete models for interleaved and concurrent activities. On the other hand, in real life, since there exists a great variety of ways in which activities can be interleaved and performed concurrently, it may not be possible to construct a complete model through the training process. Hence, the applicability and flexibility of these solutions [19], [20], [21], [23] remain questionable.

A different attempt to recognize activities is time-series-based classification, in which an activity is modeled as a sequence of discrete events [25], [26]. Activities are recognized through discovering and matching the Motif which is defined as the subsequences with similar behavior appeared frequently in time series data. However, this approach is sensitive to the order of the events as it rigidly models an activity sequence using its variable-length event subsequences over the entire continuum of their temporal scales.

Our approach is fundamentally different from the above because we use a discriminative knowledge pattern. The Emerging-Patterns-based classifier uses a set of multiattribute tests for each class, while most previous classifiers consider only one test on one attribute at a time. Different from the time-series-based classifier concerning the mining of regularities, we mine the abnormal growth among classes.

3 Sensor Platform and Data Preprocessing

In this section, we first present our wireless sensor platform, then describe data preprocessing.

3.1 Our Sensor Platform

We built our wireless sensor platform as shown in Fig. 1. It senses the four types of information—user's movement, environmental information, user location, and human-object interaction.

In this platform, a subject wears an iMote2 set on each hand and his waist. Each iMote2 set, as shown in Fig. 1b, consists of an IPR2400 processor/radio board and an ITS400 sensor board, capable of measuring motion data using a three-axis accelerometer, the temperature, humidity, and light level of the environment. A subject also wears an RFID wristband reader (Fig. 1c) on each hand. The wristband is custom-built and it incorporates a SkyeTek M1-mini RFID reader, a Mica2Dot module, and a rechargeable battery. The RFID reader is able to detect the presence of a tag within the range of 6 to 8 cm. HF RFID tags are attached to objects such as cups and spoons. In the case of metal objects, e.g., kettle, we tagged on its plastic handle. In the case of liquid objects,

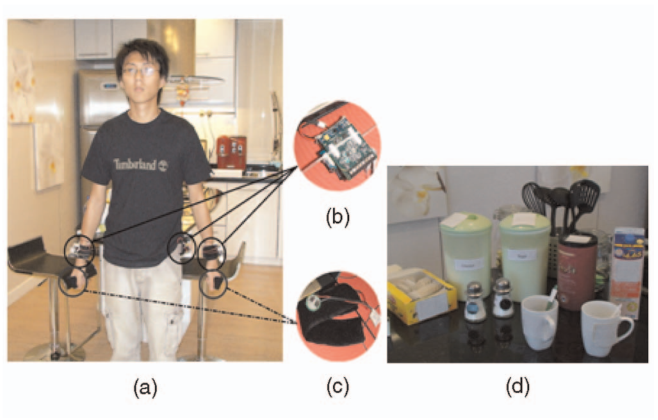


Fig. 1. (a) Our sensor platform consists of two RFID wristband readers and three iMote2 sets, (b) iMote2 set, (c) RFID wristband reader, and (d) tagged objects.

e.g., water, we tagged on the faucet with a special plastic handle to sense the use. Fig. 1d shows a screenshot of tagged objects in a kitchen. In addition, detecting user location is done in a simple way that an UHF RFID reader is located in each room to sense the proximity of a user wearing an UHF tag.

When a subject performs activities, the sensor readings from the three iMote2 sets are collected and transferred wirelessly to a local server. When a subject handles a tagged object, the wristband scans the tag ID and sends it wirelessly to another server that can map the ID to an object name. These sensor readings are recorded by the two servers separately, and will be merged into a single text file as the activity trace.

3.2 Feature Extraction

After obtaining the activity trace, we first extract appropriate sensor features from various sensor readings. We describe feature extraction for each type of sensors as follows:

For acceleration data, we compute features such as DC, variance, energy, frequency-domain entropy, and correlation. DC is the mean acceleration value. Variance is used to characterize the stability of a signal. Energy captures data periodicity, and it is calculated as the sum of the squared discrete FFT component magnitudes of a signal. Frequency-domain entropy helps to discriminate activities with similar energy values, and it is calculated as the normalized information entropy of the discrete FFT component magnitudes of a signal. Correlation is calculated between every two axes of each accelerometer reading and between all pairwise combinations of axes on different accelerometers. This feature aims to find out the correlation among the different axes of the three accelerometers. For temperature, humidity, and light level readings, we compute the mean value for each reading. For RFID and location readings, we use object name and location name as features. If no RFID reading is observed or in the presence of a corrupted tag ID, the feature value will be set to NULL.

In summary, we have a total number of 75 features in which 72 of them are numeric features and three of them are nominal features. In a fixed time interval which is set to one second in our experiments, we generate a 75-dimensional *feature vector*. A *feature vector* consists of many *feature items*, where a *feature item* refers to a feature-value pair in which a value can be numeric or nominal. We denote a numeric feature as $numfeature_i$. Suppose its range is $[x, y]$ and an interval $[a, b]$ (or in other forms, $(a, b]$, $[a, b)$, or (a, b)) is contained in $[x, y]$. We call $numfeature_i@[a, b]$ a *numeric feature item*, meaning that the value of $numfeature_i$ is limited inclusively between a and b. We denote a nominal attribute as $nomfeature_j$. Suppose its range is $v_1, v_2, \ldots, v_n$, we call $nomfeature_j@v_k$ a *nominal feature item*, meaning that the value of $nomfeature_j$ is v_k.

Numeric features will be discretized by the entropy-based discretization method [48]. The algorithm uses the class information entropy of candidate partitions to select binary boundaries, and uses the minimal entropy criteria to find multilevel cuts for each attribute. As a result, 72 numeric feature values are partitioned into 1,046 disjoint intervals. We then can directly combine feature name and its interval into a *numeric feature item*. For example, $accel\_body\_x@(-737.5\text{-} - 614.5]$ is an example of body acceleration feature in the x-axis in our training data set. For nominal feature, a feature name and its value are combined as a *nominal feature item*. For $left\_object$ and $right\_object$ features, we merge them into one feature by computing $left\_object \cup right\_object$ without losing any essential object due to user's handedness. For example, a *nominal feature item* can be *object@cup* or *location@bathroom*. In our current sensor setting, we have a total number of 1,133 *feature items*. They are indexed by a simple encoding scheme and will be used as inputs to the EPs mining process described in the next section.

4 Mining Emerging Patterns for Activity Recognition

4.1 Problem Statement

We formulate the problem of sequential, interleaved, and concurrent activity recognition as follows: Given a training data set that consists of a sequence of observations for sequential activities only (i.e., formally, a training trace O consists of T observations $O = \{o_1, o_2, \ldots, o_T\}$ associated with sequential activity labels $\{SA_1, SA_2, \ldots, SA_m\}$, where there are m sequential activities), our objective is to train a model that can assign each new observation with the correct activity label(s) and segment the new activity trace.

4.2 Emerging Pattern and Preliminaries

We provide the background of Emerging Pattern in this section. Emerging Pattern (EP) is a pattern that describes significant changes between two classes of data [43]. An EP is a set of items whose frequency changes significantly from one data set to another. Like other patterns or rules composed of conjunctive combinations of elements, EP can be easily understood and used directly by people. EP has been successfully used for predicting the likelihood of diseases [45] and discovering knowledge in gene expression data [46].

We first give some preliminary definitions. Suppose that a data set D consists of many instances. An instance contains a set of items (i.e., an item set), where an item is an attribute-value pair. The support of an item set X, $supp_D(X)$, is $count_D(X)/|D|$, where $count_D(X)$ is the number of instances in D containing X. A pattern is frequent if its support is no less than a predefined *minimum support threshold*. Unlike frequent pattern, EP is concerned with two classes of data.

TABLE 1
A Subset of EPs for the *Cleaning a Dining Table* Activity

EPs	Support(%)	Growth rate
$location@kichen$, $object@plate$	100	∞
$object@dining\_table$, $object@plate$	95.24	365.7
$object@cleanser$, $object@plate$, $object@wash\_cloth$, $location@kichen$	95.24	∞
$object@light\_power\_switch$, $object@plate$, $location@kichen$	95.24	∞
$object@wash\_cloth$, $object@bowl$, $object@plate$, $object@cleanser$, $location@kichen$	90.48	∞
$object@spoon$, $object@plate$, $object@bowl$, $object@dining\_table$, $location@kichen$	80.95	∞
$object@bowl$, $accel\_body\_x@(-155.25 \sim -52.25]$, $light@(24.5 \sim 28.5]$, $object@plate$	66.67	256
$object@bowl$, $accel\_left\_z@(-684.5 \sim -453.5]$, $light@(24.5 \sim 28.5]$, $object@plate$, $location@kichen$	66.67	∞

Definition 1. *Given two different classes of data sets D_1 and D_2, the growth rate of an item set X from D_1 to D_2 is defined as $GrowthRate(X) =$*

$$\begin{cases} 0, & if \quad supp_1(X) = 0 \quad and \quad supp_2(X) = 0, \\ \infty, & if \quad supp_1(X) = 0 \quad and \quad supp_2(X) > 0, \\ \frac{supp_2(X)}{supp_1(X)}, & otherwise. \end{cases}$$

Emerging Patterns are those item sets with large growth rates from D_1 to D_2.

Definition 2. *Given a growth rate threshold $\rho > 1$, an item set X is said to be a $\rho - EmergingPattern$ (or simply EP) from a background data set D_1 to a target data set D_2 if $GrowthRate(X) \geq \rho$.*

Since an EP has a high support in its target class and a low support in the contrasting class, it can be used as a powerful discriminator to differentiate the class membership of instances that contain the EP.

4.3 Mining Emerging Patterns from Sequential Activity Instances

We use sequential activity instances for training. Note that the instances of interleaved and concurrent activities are not used in our mining process. An instance here refers to a union of all the observations that belong to a sequential activity during a continuous period of time.

For each sequential activity class SA_i, we mine a set of EPs to contrast its instances, D_{SA_i}, against all other activity instances D'_{SA_i}, where $D'_{SA_i} = D - D_{SA_i}$ and D is the entire sequential activity data set. We refer EP_{SA_i} as the EPs of sequential activity SA_i. We discover the EPs by an efficient algorithm described in [47]. The algorithm mines closed patterns and generators simultaneously under one depth-first search scheme.

After computation, we get n sets of EPs, one set per sequential activity class. Table 1 presents an example of the

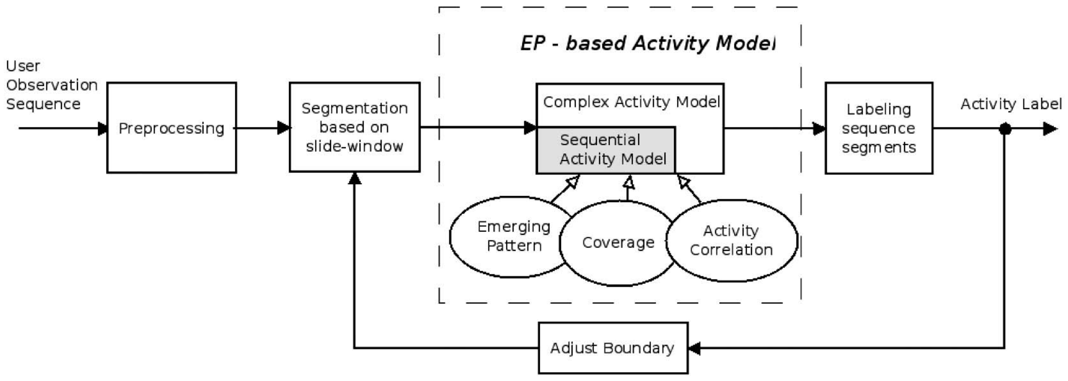

EP - based Activity Model
User Observation Sequence
Preprocessing
Segmentation based on slide-window
Complex Activity Model
Sequential Activity Model
Emerging Pattern
Coverage
Activity Correlation
Labeling sequence segments
Activity Label
Adjust Boundary

Fig. 2. Overview of our activity recognition system.

top eight EPs of the *cleaning a dining table* activity. For example, the EP

$$\{object@cleanser, object@plate, object@wash\_cloth, location@kichen\}$$

has a support of 95.24 percent and a growth rate of ∞. It has an intuitive meaning that cleanser, plate, and wash_cloth are the common objects which are involved in this activity, and this activity usually occurs in the kitchen. In fact, one of the advantages of EPs is easy to understand. Another interesting phenomenon is that the EPs

$$\{object@bowl, accel\_left\_z@(-684.5\text{-}-453.5], light@(24.5\text{-}28.5], object@plate, location@kichen\}$$

has a support of 66.67 percent only, where $accel\_left\_z$ stands for the acceleration data of a subject's left hand in the z-axis. This probably can be explained that there exist some variances of his left-hand movement when the subject performs this activity.

5 Our Activity Recognition Algorithm

5.1 Overview

We first give an overview of our activity recognition system as illustrated in Fig. 2. Given a new observation sequence from time $t = 0$ to T, our algorithm aims to assign correct activity label(s) to each observation. At time t, for each possible activity A_i, we first obtain a test instance $S_{t\sim t+L_{A_i}}$ using a window L_{A_i} (the length of L_{A_i} is the average duration of A_i and can be obtained from the training data). We then compute the likelihood of A_i based on our activity model. The activity with the maximum likelihood will be assigned to this instance. In the next step, the window moves on and we obtain the next test instance. The same computation follows to obtain an activity with the maximum likelihood. We then apply a segmentation algorithm to detect and adjust the boundary between these two adjacent activities. The above processes run recursively until the end, i.e., $t = T$.

The score function measures the likelihood of a possible activity for a given test instance. Its design is critical in our algorithm. In our design, we consider three probability elements: EP score, coverage score, and correlation score. We describe each of them in the following sections.

5.2 Score Function for Sequential Activity

5.2.1 EP Score

Given a test instance $S_{t\sim t+L_{SA_i}}$ for a possible activity SA_i, the EP score measures the likelihood of a set of SA_i's EPs contained in this instance. It provides a probabilistic measurement on the fraction of EP_{SA_i} (i.e., the discriminating features of SA_i) contained in $S_{t\sim t+L_{SA_i}}$. To make use of each EP set, we combine the strength of each set of EPs based on the aggregation method described in [44].

Suppose an instance $S_{t\sim t+L_{SA_i}}$ contains an EP, X, where $X \in EP_{SA_i}$, then the odds that $S_{t\sim t+L_{SA_i}}$ belongs to SA_i is defined as $\frac{growth\_rate(X)}{growth\_rate(X)+1}$. The differentiating power of a single EP is then defined by the odds and the fraction of the population of class that contain the EP. More specifically, the differentiating power of X is given by $\frac{growth\_rate(X)}{growth\_rate(X)+1} * supp_{SA_i}(X)$. The aggregated EP score of $S_{t\sim t+_{SA_i}}$ for SA_i is defined as follows:

$$aggregated\_score(SA_i, S_{t\sim t+_{SA_i}}) = \sum_{X \subseteq S_{t\sim t+_{SA_i}}, X \in EP_{SA_i}} \frac{growth\_rate(X)}{growth\_rate(X)+1} * supp_{SA_i}(X), \quad (1)$$

where $supp_{SA_i}(X)$ is the support of X in class SA_i, and $growth\_rate(X)$ is $supp_{SA_i}(X)$ divided by the X's support in $non-SA_i$ class. The EP score of each activity is then normalized to all the training instances of SA_i. Finally, we define the EP score as follows:

$$ep\_score(SA_i, S_{t\sim t+L_{SA_i}}) = \frac{aggregated\_score(SA_i, S_{t\sim t+L_{SA_i}})}{base\_score(SA_i)}, \quad (2)$$

where $base\_score(SA_i)$ is the median of the values of $aggregated\_score(SA_i, S_{t\sim t+L_{SA_i}})$ in the training data.

Example 1. Given a test instance $S = \{1, 2, 3, 4, 6, 7, 8, 10, 11\}$, and a possible activity SA_i which has four EP sets: ({1, 3}, 100 percent, ∞), ({2, 3}, 95 percent, 150), ({1, 2, 3, 8}, 81 percent, ∞), and ({1, 7}, 62 percent, 80). Now, we know the differentiating power of the four EPs is 1.0, 0.95, 0.81, and 0.62, respectively. The aggregated EP score can be obtained by summing up all the EPs, hence we get $aggregated\_score(SA_i, S) = 3.38$. Suppose there are seven training data sets for SA_i, we compute the aggregated

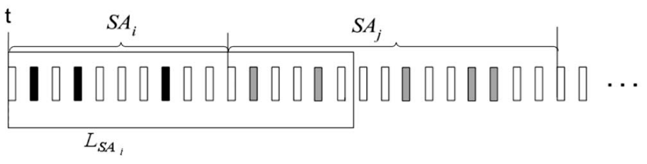


Fig. 3. Illustration of the coverage score (each bar represents a feature vector, black bars represent a subset of EP_{SA_i} and gray bars represent a subset of EP_{SA_j}).

score for each training data set. For example, we obtain 2.76, 3.38, 3.56, 3.66, 3.85, 3.85, and 4.12, respectively, and we obtain the median 3.66. Finally, the EP score for SA_i is obtained by $ep\_score(SA_i, S) = 3.38/3.66 = 0.92$.

5.2.2 Coverage Score

Given a test instance $S_{t\sim t+L_{SA_i}}$ for a possible activity SA_i, the Sliding-Window Coverage score (coverage score for short) measures a fraction of irrelevant observations contained in this instance with respect to SA_i.

Fig. 3 illustrates this concept. A sequence of feature vectors from time t is labeled with the ground truth activity SA_i, followed by SA_j. During prediction, a sliding window L_{SA_i} is used to get instance $S_{t\sim t+L_{SA_i}}$. Since the length of L_{SA_i} is an approximation, it could be longer or shorter than the actual length of $S_{t\sim t+L_{SA_i}}$. For example, shown in this example, although $S_{t\sim t+L_{SA_i}}$ includes the fractions of EP_{SA_i}, it covers some observations of the adjacent activity SA_j as well. Therefore, we introduce the coverage score to measure the fraction of irrelevant observations in a sliding window. The lower the percentage of irrelevant observations covered; the larger the coverage score is obtained.

We denote $coverage\_score(SA_i, S_{t\sim t+L_{SA_i}})$ as the coverage score of instance $S_{t\sim t+L_{SA_i}}$ for SA_i. This score is computed based on a function, $relevance(SA_i, f_p)$, where f_p is a feature vector contained in L_{SA_i}. Recall that a feature vector is a set of feature items. We first compute $relevance(SA_i, item_h)$ for each $item_h \in f_p$, we then aggregate their scores for computing $relevance(f_p, SA_i)$.

We compute $relevance(SA_i, item_h)$ based on the following equation:

$$relevance(SA_i, item_h) = P(item_h|SA_i) + \sum_{item_h \in X, X \in EP_{SA_i}} supp_{SA_i}(X), \quad (3)$$

where the probability $P(item_h|SA_i)$ is obtained from the training data, and $\sum_{item_h \in X, X \in EP_{SA_i}} supp_{SA_i}(X)$ indicates that more weights are given to an item which appears in EP_{SA_i}.

We now aggregate the values of $relevance(SA_i, item_h)$ for all $item_h \in f_p$. The aggregation can be simply done using $\sum_{item_h \in f_p} relevance(SA_i, item_h)$. However, if EP_{SA_i} has many more items than EP_{SA_j}, then a feature vector usually gets higher scores for SA_i than SA_j even for the feature vectors of SA_j. Hence, we need a normalized scheme. The normalized $relevance(SA_i, f_p)$ is computed as follows:

$$relevance(SA_i, f_p) = \frac{unnorm\_relevance(SA_i, f_p)}{base\_relevance(SA_i)}, \quad (4)$$

where $unnorm\_relevance(SA_i,\ f_p) = \sum_{item_h \in f_p} relevance(SA_i,\ item_h)$ and $base\_relevance(SA_i)$ be the median of the values of $unnorm\_relevance(SA_i, f_p)$ in the training data.

We now can compute $coverage\_score(SA_i, S_{t\sim t+L_{SA_i}})$. A simply way is to sum up all the $relevance(SA_i, f_p)$ in L_{SA_i}. However, it may bias toward longer activities. Hence, we compute $coverage\_score(SA_i, S_{t\sim t+L_{SA_i}})$ by averaging all the $relevance(SA_i, f_p)$ as follows:

$$coverage\_score(SA_i, S_{t\sim t+L_{SA_i}}) = \frac{1}{L_{SA_i}} \sum_{f_p \in L_{SA_i}} relevance(SA_i, f_p). \quad (5)$$

5.2.3 Correlation Score

Human activities are usually performed in a nondeterministic fashion. However, there exist some correlations between them, i.e., when an activity SA_j has been performed, the probability of another activity SA_i being performed. For example, in a daily routine, a user usually brushes his teeth, followed by washing his face; cleans the dining table after eating his meal. We use condition probability to model such correlations between activities, i.e., $P(SA_i|SA_j)$ which is the conditional probability of SA_i given SA_j. We can easily obtain such probabilities from training data set. Note that the initial value is set to zero, i.e., $P(SA_i|NULL) = 0$.

5.3 Score Function for Interleaved and Concurrent Activities

We now describe how to compute the scores of interleaved and concurrent activities. We set the number of single activities involved in interleaved or concurrent activities to two for illustrations although in theory it can be more than two.

We denote CA_i as both interleaved activities (i.e., in this case, we denote CA_i as $SA_a \& SA_b$) and concurrent activities (i.e., in this case, we denote CA_i as $SA_a + SA_b$), where two single activities SA_a and SA_b are involved in. We define the sliding-window length of CA_i as $L_{CA_i} = L_{SA_a} + L_{SA_b}$, and use L_{CA_i} to get the test instance $S_{t\sim t+L_{CA_i}}$. Since an instance of CA_i containing both EP_{SA_a} and EP_{SA_b} (i.e., some of the steps that belong to SA_a and SA_b, respectively, are interleaved or overlapped), we compute the EP score of CA_i as follows:

$$\begin{aligned} &ep\_score(CA_i, S_{t\sim t+L_{CA_i}}) \\ &\quad = max[ep\_score(SA_a, S_{t\sim t+L_{CA_i}}), \\ &\qquad ep\_score(SA_b, S_{t\sim t+L_{CA_i}})]. \end{aligned} \quad (6)$$

When computing the coverage score of CA_i, we choose the higher score from $relevance(SA_a, f_p)$ and $relevance(SA_b, f_p)$ since CA_i contains both the observations of SA_a and SA_b in $S_{t\sim t+L_{CA_i}}$. The rational behind is that a feature vector f_p that belongs to an activity usually has a higher relevance to this activity. Hence, we have

$$\begin{aligned} relevance(CA_i, f_p) = max(&relevance(SA_a, f_p), \\ &relevance(SA_b, f_p)). \end{aligned}$$

Then, the coverage score of CA_i can be computed as follows:

$$coverage\_score(CA_i, S_{t\sim t+L_{CA_i}}) = \frac{1}{L_{CA_i}} \sum_{f_p \in L_{CA_i}} relevance(CA_i, f_p). \quad (7)$$

The computation of correlation score can be quite complex in interleaved and concurrent activities. There are three situations: a sequential activity followed by an interleaved or a concurrent activity, an interleaved or a concurrent activity followed by a sequential activity, and an interleaved or a concurrent activity followed by another interleaved or concurrent activity. Given the rational that higher condition probability implies a stronger activity correlation, we choose the maximum value of all possible condition probabilities for all these cases. To illustrate, given CA_j, where $CA_i = SA_a\&SA_b$ or $CA_i = SA_a + SA_b$, such probability can be computed as follows:

$$P(CA_i|CA_j) = max(P(SA_a|SA_c), P(SA_a|SA_d), P(SA_b|SA_c), P(SA_b|SA_d)). \quad (8)$$

The computation of $P(SA_i|CA_j)$ and $P(CA_i|SA_j)$ follows a similar method, and we use $P(A_i|A_j)$ to denote all the three cases. In addition, we apply correlation analysis to measure the likelihood of SA_a and SA_b appeared in an interleaved ($SA_a\&SA_b$) or concurrent ($SA_a + SA_b$) activity. This probability can be computed as $P(SA_aSA_b)$. Finally, we define the correlation score as follows:

$$correlation(A_i) = P(A_i|A_j)P(SA_aSA_b), \quad (9)$$

where A_i can be CA_i or SA_i, A_j can be CA_j or SA_j, and SA_a and SA_b are two single activities involved in CA_i. When A_i is SA_i, $P(SA_aSA_b)$ is defined as $P(A_i)$.

In summary, the score function for sequential, interleaved, and concurrent activities is defined as follows:

Definition 3. *Given a time t, and an activity A_j which ends at t, for each activity A_i, a test instance $S_{t\sim t+L_{A_i}}$ is obtained from t to $t + L_{A_i}$, the likelihood of A_i is computed as follows:*

$$score(A_i, A_j, S_{t\sim t+L_{A_i}}) = c_1 * ep\_score(A_i, S_{t\sim t+L_{A_i}}) + c_2 * coverage\_score(A_i, S_{t\sim t+L_{A_i}}) + c_3 * correlation(A_i), \quad (10)$$

where c_1, c_2, and c_3 are coefficients, representing the weight of each individual score. These coefficients have different implications. For example, a higher c_1 implies that the subject always performs his activities in a consistent manner. A higher c_2 implies that all the instances of the activity are performed in a constant duration whereas a lower c_2 implies that the variance of the instances can be large. A higher c_3 implies that the subject usually performs his activities in certain order. These weights reflect a subject's habit in his daily routine.

5.4 Sliding-Window-Based Algorithm

We now design our recognition algorithm. We first apply a sliding-window-based algorithm which is commonly used in time series data analysis. Given m sequential activities, the maximum number of interleaved and concurrent activities can be computed by $m(m-1)$. The total number of activities is then m^2. We define L_{max} as $max\{L_{A_k}\}$, where $k = 1, 2, \ldots, m^2$. A simple recognition method is to test each possible activity label using its corresponding sliding window and the one with the highest score wins out. Algorithm 1 describes this method, and it returns two activity labels corresponding to the top two scores. Note that the activity label with the second highest score will be used in Section 5.6.

Algorithm 1. Sliding-window-based Recognition Algorithm **-slidingWinRecog**
Input: feature vector of length
L_{max}: $F = \{f_t, f_{t+1}, f_{t+2}, \ldots, f_{t+L_{max}}\}$,
where prediction starts at time t,
predicted activity A_j in the previous sliding window.
Output: An ordered pair $<A_i, A'_i>$, where A_i is the activity label with the highest score and A'_i is the activity label with the second highest score.

```
1: foreach activity $A_i$, $i = 1, 2, \ldots, m^2$ do
2:     get instance $S_{t\sim t+L_{A_i}} = \cup_{p=t}^{t+L_{A_i}} f_p$;
3:     compute $score(A_i, A_j, S_{t\sim t+L_{A_i}})$;
4: end for
5: return $<A_i, A'_i>$;
```

5.5 The Trace Segmentation Algorithm

The simple sliding-window-based algorithm presented in Algorithm 1 can be applied recursively to recognize activities in a given activity trace. However, it has a shortcoming. Since the sliding-window length L_{A_i} of each activity is an approximation of the actual length, the segmentation may not accurate. Moreover, any error in one segment may affect the segmentations of the subsequent trace. This error may accumulate resulting in poor performance.

Aiming to segment the trace accurately, we propose our trace segmentation algorithm as presented in Algorithm 2. Given two candidate activity labels (e.g., A_j followed by A_i) based on Algorithm 1, our segmentation algorithm aims to determine the boundary between A_j and A_i so that the next sliding window can be applied from this boundary.

Algorithm 2. Trace Segmentation Algorithm **-adjustBoundary**
Input: feature vector of length $L_{A_j} + L_{A_i}$:
$F = \{f_{t-L_{A_j}}, \ldots, f_{t+L_{t+L_{A_i}}}\}$,
where t is the existing boundary,
predicted activity A_j followed by A_i based on Algorithm 1.
Output: the boundary between A_j and A_i.

```
1: foreach $p$ from $t - L_{A_j}$ to $t + L_{A_i}$ do
2:     $RW[p] = relevance(A_j, f_p) - relevance(A_i, f_p)$;
3: end for
4: foreach $p$ from $t - L_{A_j}$ to $t + L_{A_i}$ do
5:     $upperSum =$ sum of all $RWs$ from $t - L_{A_j}$ to $p$;
6:     $lowerSum =$ sum of all $RWs$ from $p$ to $t + L_{A_i}$;
7:     $GAIN[p] = upperSum$ - $lowerSum$;
8: end for
9:     boundary $= p$ such that $GAIN[p]$ is maximum;
10: return boundary;
```

The intuition behind our design is, given an activity instance, its feature vectors obtained by preprocessing its observations usually have a higher relevance to this activity than all other activities. Furthermore, the relevance of its feature vectors in the same activity instance does not vary significantly as compared to the relevance of two feature vectors belonging to two different activities. The algorithm makes use of the weight difference, i.e., Relative Weight (RW), of each feature vector between the two adjacent activities.

5.6 Method for Dealing with Sliding Window

Combining our basic sliding-window recognition algorithm with the trace segmentation algorithm can be very efficient provided a testing instance is well covered by a sliding window. However, the actual length of an instance for each particular activity varies from one to another in reality. Such variety may affect the recognition performance seriously.

Basically, there exist two cases—the sliding-window length (i.e., $L_{A_i} = \alpha \times \overline{L_{A_i}}$) is either shorter or longer than the actual length of an instance. In the former case, one or more items in the EPs of activity A_i will not be contained in L_{A_i}, resulting in a lower EP score for A_i. To overcome this issue, we select an appropriate α (where $\alpha > 1$) to achieve the best performance. We will evaluate the effect of α in our experiment presented in the next section.

In the later case, it is likely that L_{A_i} covers the observations that belong to the next activity, resulting in a possibility of missed detection. To address this issue, we propose a solution as follows: We assume the basic sliding-window recognition algorithm starts at time t. In L_{A_i}, other than the candidate activity A_i computed (i.e., the score of A_i should be the highest among all the activities in L_{A_i}), we select another candidate activity with the second highest score, A'_i. We can then apply the trace segmentation algorithm to adjust the boundary between these two candidate activities so that the next prediction can be executed from the correct boundary. This method is summarized in Algorithm 3 (Lines 5-6).

Algorithm 3. *epSICAR* Activity Recognition Algorithm

```
Input: an observation sequence $O = \{o_1, o_2, \dots, o_T\}$ with
    a length of T;
    $m$ sequential activities $\{SA_1, SA_2, \dots, SA_m\}$.
Output: assign the activity label to each observation.
 1: preprocess $O$ to obtain feature vectors
      $F = \{f_1, f_2, \dots, f_T\}$;
 2: $t = 1$;
 3: $A_{previous} = null$; $A_{current} = null$; $A_{candidate} = null$;
 4: while $t \leq T$
 5:     $<A_{current}, A'_{current}> =$
            slidingWinRecog($F_{t,t+L_{max}}, A_{previous}$);
 6: $L_{A_{current}} =$
      $adjustBoundary(F_{t,t+L_{max}}, A_{current}, A'_{current}) - t$;
 7: if $t = 1$ or $A_{current} = A_{candidate}$
 8:     Assign label $A_{current}$ to $o_t \sim o_{t+L_{A_{current}}}$;
 9:     $t = t + L_{A_{current}}$;
10:     $A_{previous} = A_{current}$;
11:     $A_{candidate} = null$;
12: else if $A_{candidate} \neq A_{current}$
13:     $t =$ adjustBoundary($F_{t-L_{A_{previous}},t+L_{A_{current}}}$,
                         $A_{previous}$, $A_{current}$);
14:     $A_{candidate} = A_{current}$;
15:   end if
16: end while
```

Fig. 4. (a) A smart home, (b) a snapshot of our video recording shows a subject is frying eggs in the kitchen, and (c) another snapshot shows a subject is doing vacuuming in the living room.

This computation does not affect the case where L_{A_i} does not cover the observations of another activity A_i including the first case mentioned in this section. In this case, our trace segmentation algorithm is able to adjust the boundary to the end of L_{A_i} since the observations in L_{A_i} are not relevant to A'_i. As a result, the instances L_{A_i} will be labeled as A_i, and the next prediction can start from the end of L_{A_i}.

5.7 The Entire Process

Putting the above algorithms together, we summarize the entire process in the *epSICAR* algorithm as shown in Algorithm 3.

We now analyze the complexity of *epSICAR*. Assuming that computing a score for an activity requires time $O(S)$ and adjusting the boundary between two activities needs time $O(B)$, where both $O(S)$ and $O(B)$ are linear to the length of a slice window. Given the total number of activities m^2, and assuming that the observation sequence O contains n activity instances and the average loop for confirming an activity instance is k, then the total complexity of the *epSICAR* algorithm can be computed as $n(k \cdot m^2 \cdot O(S) + m \cdot O(S) + (k-1) \cdot O(B))$. Based on the measurement result in our experiments, the value of k falls in the interval [2.5, 3.0]. Therefore, the time complexity of the *epSICAR* algorithm is finally $n(m^2 \cdot O(S) + O(B))$.

6 Empirical Studies

We now move to evaluate our proposed algorithm. There are no public available data sets containing sequential, interleaved, and concurrent activities across a variety of daily activities in a real-world situation for our study. The data set collected in [14] contains only RFID tagged objects. It has only 11 interleaved activities limited to the morning section only. The House_n PLIA1 [27] data set contains only four hours data, and the number of interleaved and concurrent activities is also limited. Aiming to collect a more comprehensive data set for our study, we conducted our own trace collection described in the next section. We then present and discuss the evaluation results obtained from a series of experiments.

6.1 Trace Collection and Evaluation Methodology

Trace collection was done in a smart home as shown in Fig. 4a. We first conducted a survey on common ADLs performed in one's daily life. Out of them, we randomly selected 26 ADLs as summarized in Table 2. Among them, there are many possible combinations for interleaved and concurrent activities. We randomly chose 15 interleaved activities and 16 concurrent activities, e.g., *using computer* and *using phone*

TABLE 2
Sequential Activities Performed

0	making coffee	13	ironing
1	making tea	14	eating meal
2	making oatmeal	15	drinking
3	frying eggs	16	taking medication
4	making a drink	17	cleaning a dining table
5	applying makeup	18	vacuuming
6	brushing hair	19	taking out trash
7	shaving	20	using phone
8	toileting	21	watching TV
9	brushing teeth	22	watching DVD/movies
10	washing hands	23	using computer
11	washing face	24	reading book/magazine
12	washing clothes	25	listening music/radio

("23 & 20", interleaved), and *brushing teeth* while *listening music/radio* ("9 + 25", concurrent). The data were collected over a period of two weeks. We had four volunteers. Each day, each of them performed these activities at their choices in any order they like, resembling the situations in their daily routines. However, the duration of each activity is shorter as compared to that in their normal lives in order to collect more instances. There was only one subject performing activities at any given time to reduce annotation efforts. One of the volunteers annotated the trace to establish the ground truth together with video recording. Figs. 4b and 4c show two snapshots in our video recording.

Table 3 shows a total number of 532 activity instances we collected and a breakdown by three activity cases. We use 10-fold cross validation for our evaluation and divide the entire trace into 10 data sets as shown in Fig. 5. Most of data sets consist of sequential, interleaved, and concurrent activities except Data sets 1, 6, and 7 which contain only sequential activities. In our evaluation, we use the sequential activity instances from any nine data sets for training, and use all the remaining instances for testing.

We evaluate the performance of our algorithm using time-slice accuracy which is a typical technique in time series data analysis. The time-slice accuracy represents the percentage of correctly labeled time slices. The length of time slice Δt is set to 15 seconds as our experiment shows different Δt does not affect accuracy much. This time-slice duration is short enough to provide precise measurements

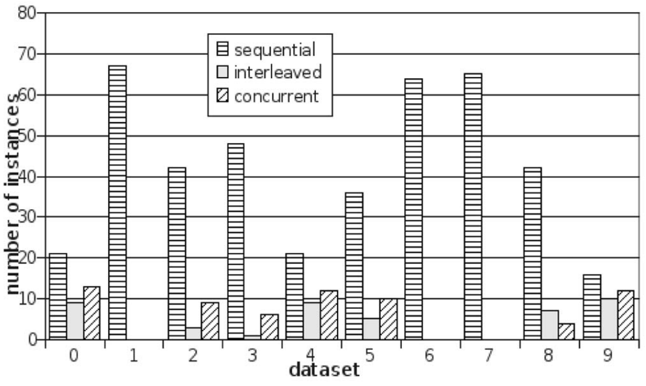


Fig. 5. Instance breakdown by three activity cases.

for most of activity recognition applications. The metric of the time-slice accuracy is defined as follows: We first denote LB as the label(s) in a time slice, where LB can be $\{SA_i\}$ in the case of sequential activity and LB can be $\{SA_i, SA_j\}$ for interleaved or concurrent activities, SA_i, $SA_j \in (SA_1, SA_2, \ldots, SA_m)$. We denote LB_G as the ground truth label(s), and LB_R as the predicted label(s). The time-slice accuracy is defined as follows:

$$Slice\_Accuracy = \frac{\sum_{SA_i \in LB_G \cap LB_R} L_{SA_i}}{\sum_{SA_i \in LB_G \cup LB_R} L_{SA_i}}. \quad (11)$$

Example 2. We now give an example as follows: Given $LB_G = \{9, 25\}$, if the predicted labels $LB_R = \{9, 25\}$, then $Slice\_Accuracy$ is 1 since $LB_G \cap LB_R = LB_G \cup LB_R$. If $LB_R = 10$, then $Slice\_Accuracy$ is 0 since $LB_G \cap LB_R = \phi$. If $LB_R = 9$, then $Slice\_Accuracy$ can be computed as $\frac{L_9}{L_9 + L_{25}}$.

The total time-slice accuracy is defined as follows:

$$Total\_Accuracy = \frac{\sum_1^{\frac{t}{\Delta t}} Slice\_Accuracy}{\frac{T}{\Delta t}}. \quad (12)$$

6.2 Experiment 1: Accuracy Performance

In this experiment, we evaluate time-slice accuracy for different activity cases. Table 4 shows the average accuracies of sequential, interleaved, and concurrent activities, respectively, and the overall accuracy.

The accuracy of sequential activity is the highest among all the three cases, while the accuracies of interleaved and concurrent activities are lower. The result probably can be explained as follows: First, we have four volunteers, and

TABLE 3
Number of Instances Collected

Type of Activities	Number of Instances
sequential	422
interleaved	44
concurrent	66
total	532

TABLE 4
Overall Accuracy

Type of Activities	Time-slice Accuracy
sequential	90.96%
interleaved	88.1%
concurrent	82.53%
total	88.67%

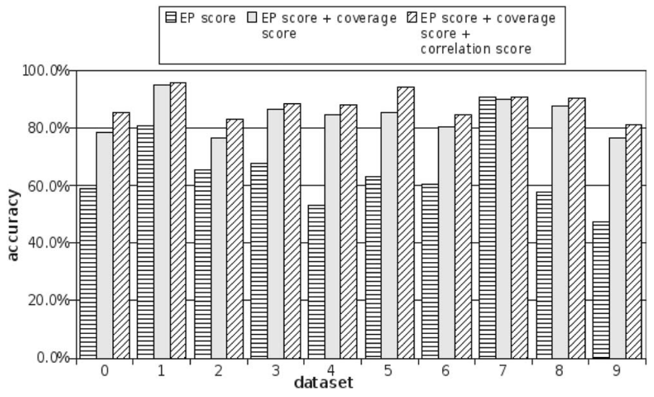


Fig. 6. Analysis of the score function.

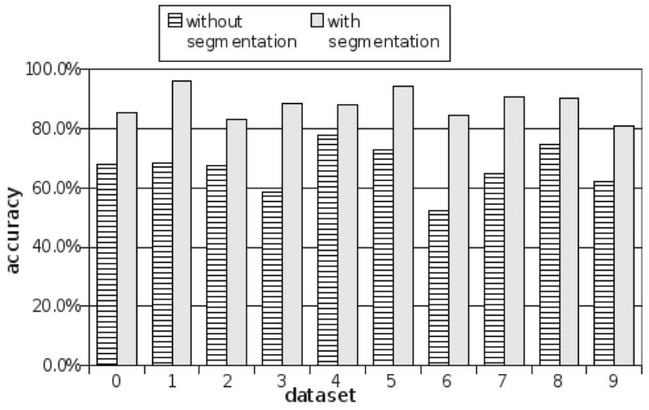


Fig. 7. Effect of the trace segmentation algorithm.

they are instructed to perform their activities in their own ways during trace collection. Each subject may perform his interleaved or concurrent activities in a different manner. This difference does not influence the sequential activity recognition much because the training process captures the common characteristics of all the four volunteers. However, some of the specific characteristics of interleaved and concurrent activities performed by different individuals may not be captured by our model. As a result, the accuracies of interleaved and concurrent activities are lower and their results fluctuate from one data set to another, depending on different subjects. Nevertheless, the result is reasonable good, and we achieve our objective of building a unified activity model to recognize all the three activity cases based on sequential activity training data only.

Second, the accuracy of concurrent activity is 5.57 percent lower than that of interleaved activity while the accuracy of interleaved activity is close to that of sequential activity. It is probably due to sliding-window length. In our model, we apply $L_{CA_i} = L_{SA_a} + L_{SA_b}$ to calculate the sliding-window length of CA_i. This estimation seems work well in the case of interleaved activity as the observations of SA_a and SA_b do not overlap each other. However, for concurrent activity, there exists some overlapped steps between SA_a and SA_b, hence L_{CA_i} should be much shorter than $L_{SA_a} + L_{SA_b}$.

6.3 Experiment 2: Model Analysis

In this experiment, we evaluate and analyze our proposed activity model. We first evaluate *epSICAR* with respect to our score function. Fig. 6 shows that the accuracies of *epSICAR* with EP score, EP $\mathrm{score} + \mathrm{coverage}$ score, and EP $\mathrm{score} + \mathrm{coverage\ score} + \mathrm{correlation}$ score, respectively. As shown in the figure, *epSICAR* achieves an accuracy of 66 percent on average with the EP score only, demonstrating that the concept of EPs works effectively in recognizing both simple and complex activities. However, the effectiveness of the EP score is not as high as we expect and there exist some variations. We analyze this case and suggest three reasons. First, the use of EPs in activity recognition is leveraged on mining discriminating sensor features. The more discriminating features are collected, the better EPs are mined and the better results are obtained. Our current sensor platform has limited types of sensors. More sensor features can be developed, which we leave for our future work. Second, we currently deploy a simple aggregation method to sum up the contribution of each set of EPs. We plan to further investigate this method in our future work. Third, EP score works better for sequential activity than interleaved and concurrent activities as suggested by the result. This is probably because the EP sets for each activity are mined from sequential activity instances only. Mining all the activity instances may achieve a better result; however, it will limit the flexibility of this model in real life.

Fig. 6 also suggests that, by introducing the coverage score, the accuracy is improved significantly by about 19 percent and the variance also decreases. This demonstrates that the coverage score enhances the robustness of our algorithm significantly. We also observe that the accuracy can be further improved by about 3 percent when adding the correlation score. This contribution is relatively trivial in our data set for the reason that the four subjects may have their own habits and perform their activities in different sequences, resulting in a weak correlation between these activities. In the situation of one subject performing his/her daily routine activities, the correlation score may play an important role.

Next, we evaluate the effect of our segmentation algorithm. Fig. 7 shows the results for *epSICAR* with and without segmentation. As expected, *epSICAR* with segmentation achieves a much better accuracy (i.e., 25 percent improvement on average) in all the data sets. Without segmentation, *epSICAR* only achieves an accuracy of 60 percent on average. This demonstrates that, on one hand, a sliding-window-based method has a limitation in truncating an activity instance with its correct length for prediction; more seriously, any error in a boundary detection may affect the recognition of the subsequent trace. The errors may accumulate one after another affecting recognition accuracy seriously. On the other hand, our trace segmentation algorithm works well to segment the two adjacent activities, and improves accuracy significantly.

6.4 Experiment 3: Parameter Analysis

In this experiment, we evaluate and analyze the effect of important parameters used in our model. We first evaluate the effect of the length of time slice, Δt. Fig. 8 shows the average accuracy for our data sets using different lengths of time slice. We can see that the accuracy remains quite stable with the length of Δt varies from 1 to 30 seconds. It demonstrates that the metrics we defined is effective, and

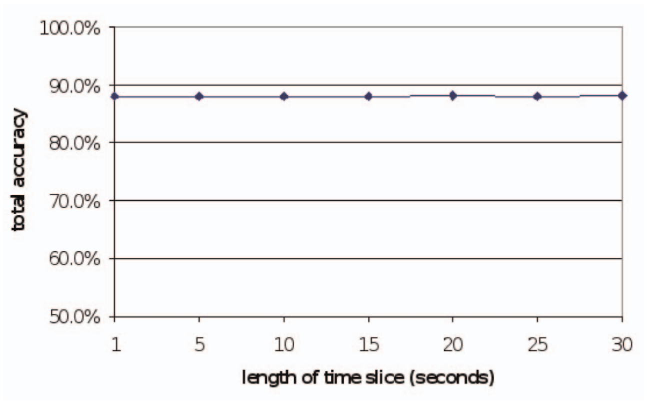


Fig. 8. Effect of the time-slice length Δt.

the accuracy does not fluctuate with different Δt. We actually set Δt to 15 seconds for all other experiments.

In this experiment, we evaluate the effect of coefficients (i.e., c_1, c_2, and c_3) used in our cost function. Table 5 shows the accuracy results using different sets of coefficient. The value of each coefficient represents the importance of an individual score in the cost function for a given data set. The result reveals that there are certain patterns in our data set, i.e., a subject seems always to perform each activity in a constant duration, and it shows less activity correlation since we have four subjects. Table 5 also shows that the set ($c_1 = 1.0$, $c_2 = 1.5$, and $c_3 = 0.7$) achieves the best performance, and we actually use this set for all other experiments.

Finally, we evaluate and analyze how the sliding-window length affects the accuracy. Choosing an appropriate sliding-window length is critical in our model as we discussed. In a typical sliding-window method, a too-long window may include more than two activities' observations resulting in difficulty of segmentation. On the other hand, a too-short window may break an activity instance into too many fractions resulting in difficulty of recognition. Recall that the length of the sliding window L_{A_i} is calculated based on $L_{A_i} = \alpha \times \overline{L_{A_i}}$, where $\overline{L_{A_i}}$ is the average length of all activity instances in the training data set. Fig. 9 shows the effect of different α values with respect to the average accuracy, and the effect of our method of dealing with the sliding-window length (named *Enhanced Detection* algorithm for illustration). Fig. 9 shows that the highest accuracy is achieved at $\alpha = 1.1$.

TABLE 5
Effect of Coefficients

Coefficients (c_1, c_2, c_3)	**Sequential**	**Interleaved**	**Concurrent**	**Overall**
(1.0, 1.0, 1.0)	90.68%	86.49%	75.92%	87.12%
(1.0, 1.5, 0.7)	90.96%	88.10%	82.53%	88.67%
(0.8, 1.5, 0.7)	90.33%	87.02%	77.81%	87.43%
(1.2, 1.5, 0.7)	90.72%	86.38%	78.23%	87.58%
(1.0, 1.3, 0.7)	90.56%	86.49%	78.58%	87.62%
(1.0, 1.7, 0.7)	90.53%	86.22%	78.31%	87.52%
(1.0, 1.5, 0.5)	90.96%	87.87%	77.67%	87.98%
(1.0, 1.5, 0.9)	90.68%	87.81%	77.03%	87.66%

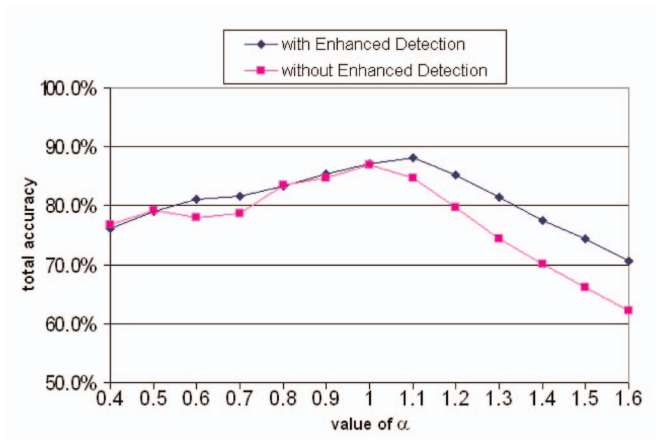


Fig. 9. Effect of α.

However, we can still achieve above 70 percent at $\alpha = 1.6$, and the accuracy trend shows a linear relation. It also demonstrates that our trace segmentation algorithm works effectively and correct segmentations can prevent a serious loss in accuracy. We also observe that the total accuracy without the *Enhanced Detection* algorithm decreases when $\alpha > 1$, demonstrating that the *Enhanced Detection* algorithm works effectively in dealing with the case where the sliding window covers the observations of another activity. On the other hand, we observe a similar result in the cases of with and without the *Enhanced Detection* algorithm when $\alpha < 1$, demonstrating that the *Enhanced Detection* algorithm does not affect our prediction process as we analyzed in Section 5.6. To summarize, this experiment demonstrates that the length of sliding window does affect the accuracy of our algorithm, however, but its influence is limited in our model and *epSICAR* deals with various cases effectively.

6.5 Experiment 4: Comparison Study

This section reports our comparison study. We compare the performance of our algorithm with both static models (i.e., C4.5 and Naïve Bayes) and temporal models (i.e., HMM and CRF). HMM is a generative probabilistic model consisting of a hidden variable and an observable variable at each time step as illustrated in Fig. 10a. CRF is a discriminative probabilistic model, and a linear-chain CRF model can be used for activity recognition, as illustrated in Fig. 10b.

Since C4.5, Naïve Bayes, HMM, and CRF cannot be directly applied to recognize interleaved and concurrent activities, we compare the performances of different models based on sequential activity only in this experiment. For

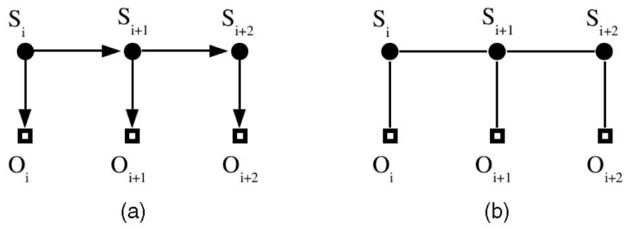


Fig. 10. (a) Graphic structure of HMM and (b) linear-chain CRF, where the round-shaped nodes represent the hidden states and the square-shaped nodes represent the observations.

TABLE 6
Comparison Results

Classifier	Accuracy				With boundary detection	Remark
	Sequential ADL	Interleaved ADL	Concurrent ADL	Overall		
epSICAR	90.96%	88.10%	82.53%	88.67%	YES	pattern based model
Naïve Bayes	60.45%	N.A.	N.A.	60.45%	NO	static probabilistic model
C4.5	47.96%	N.A.	N.A.	47.96%	NO	static model
HMM	70.34%	N.A.	N.A.	70.34%	NO	temporal probabilistic model
CRF	83.27%	N.A.	N.A.	83.27%	NO	temporal probabilistic model

each of the 10 data sets, an HMM was trained and the Viterbi algorithm was used to recover the state sequence, which is the recognized label sequence. Similarly, for each data set, a CRF was trained and the state sequence was recovered as the recognized label sequence. Both the HMM and CRF were evaluated on the same training and testing data sets generated using 10-fold cross validation. Table 6 shows the comparison results of different algorithms.

The average accuracy of *epSICAR* is 90.96 percent—the highest amount three, followed by 83.27 percent for CRF and 70.34 percent for HMM. Our EP-based algorithm outperforms both static and temporal models. As we know, static classifiers are concerned about mining the regularities among training data while our EP-based algorithm mines the differences between classes. In a discriminative task, such as sensor-based activity recognition, mining the differences is more straightforward and effective as suggested by the result.

In addition, we have the following analysis. First, the amount of training data is still relatively small comparing to the complexity of real-life activities. In this circumstance, mining the differences between classes is more effective for building a discriminative model. Although CRF is also a discriminative model, it focuses on mining the regularities in which typically a large amount of training data is required. This comparison result reveals that our EP-based approach tends to be more efficient with the same amount of training data. HMM, as a generative joint model, is least effective due to the known shortages, such as overfitting training data and strong independence assumptions. The *epSICAR* algorithm as compared to HMM and CRF is more fault-tolerant. HMM and CRF are basically directed-graph-based probabilistic models, a prediction error made in one state may cause a series of errors in the state sequence. In this experiment, we notice that it typically takes 20 steps for the HMM model to fall back to the correct state after an error occurred in the current step, whereas *epSICAR* can quickly turn into the correct state in two or three steps after an error. Furthermore, our EP-based algorithm is more noise-tolerant as compared to both static and temporal models. This is because mining the differences of classes will not include noise patterns provided the noise distribution is random. The noise-tolerant feature is particularly important in sensor-based activity recognition as it is inevitable that sensor data contain noise.

In addition, *epSICAR* is capable of handling interleaved and concurrent activities even if the training instances only contain sequential activity. From this aspect, *epSICAR* has a good advantage of handling complex activities and great applicability for real-life applications.

7 Conclusion

To conclude, we first summarize the paper, and then discuss the limitations of our approach, and outline our future work.

7.1 Summary

To summarize, in this paper, we study the problem of activity recognition based on sensor readings in a pervasive computing environment. We deploy wireless sensors and conduct a real-world trace collection. We then investigate a challenging problem that how we can apply a model, which can be learned from sequential activity instances only, in recognizing both simple and complex activities. We exploit Emerging Patterns as powerful discriminators to differentiate activities, and propose the *epSICAR* algorithm. Our comprehensive evaluation results demonstrate both the effectiveness and flexibility of our algorithm.

7.2 Limitations and Future Work

As initial exploration of a pattern-based approach to sensor-based activity recognition, we demonstrate that Emerging Pattern can be effectively applied to recognize activities. Mining a strong EP set for each activity is important as we leverage on a score function to differentiate activities. An EP describes the significant changes between two activity classes; and ideally, there should exist many distinguishing features among different activity classes. In our current sensor platform, we only capture four types of information. There are many other useful sensor features can be developed, e.g., the audio, users orientation, etc. The audio feature has been proven to be very useful in recognizing human activities [12], [13] because it captures the sounds produced when performing activities and there exist many different audio features among different activities. We will incorporate audio sensor into our platform in our future work.

Another limitation is on our data collection. While the data sets we collected provide many comprehensive cases for studying and analyzing our algorithm, it was done in a mock scenario. A more nature collection should be conducted in a real home, and it will be subject to our financial and resource constraints. We are seeking budget support and a possibility of collaborating with other partners for such further study.

As we mentioned in Section 6.4, the coefficients of our score function reflect a users habit in her/his daily routine. It is very interesting to further study their effects to discover activity patterns with respect to different users and activities under different circumstances. Although they are currently obtained though our experiments, a better approach is to learn from training data sets through statistical learning methods that we will study in our future work.

The computation of EP score was based on a simple aggregation method which does not have any solid statistical foundation. We will also look into a better method for aggregating the contribution of each EP set. There are many potential methods can be applied in this case, such as Bayes theorem. Finally, our goal is to develop an efficient, real-time sensor-based recognition system capable of recognizing various activities under real-life scenarios.

Acknowledgments

This work was supported by the Danish Council for Independent Research, Natural Science under Grant 09-073281, and partially supported by National 973 program of China under Grant 2009CB320702 and National Natural Science Foundation of China under Grants 60736015, 60721002, and 61073031.

References

[1] S. Katz, A.B. Ford, R.W. Moskowitz, B.A. Jackson, and M.W. Jaffe, "Studies of Illness in the Aged. The Index of ADL: A Standardized Measure of Biological and Psychological Function," *J. Am. Medical Assoc.*, vol. 185, pp. 914-919, Sept. 1963.

[2] L. Bao and S.S. Intille, "Activity Recognition from User-Annotated Acceleration Data," *Proc. Second Int'l Conf. Pervasive Computing (PERVASIVE '04)*, pp. 1-17, 2004.

[3] E.M. Tapia, S.S. Intille, and K. Larson, "Activity Recognition in the Home Setting Using Simple and Ubiquitous Sensors," *Proc. Second Int'l Conf. Pervasive Computing (PERVASIVE '04)*, pp. 158-175, 2004.

[4] B. Logan, J. Healey, M. Philipose, E.M. Tapia, and S.S. Intille, "A Long-Term Evaluation of Sensing Modalities for Activity Recognition," *Proc. Ninth Int'l Conf. Ubiquitous Computing (UbiComp)*, Sept. 2007.

[5] T. Huynh, U. Blanke, and B. Schiele, "Scalable Recognition of Daily Activities from Wearable Sensors," *Proc. Int'l Symp. Location and Context-Awareness (LoCA)*, Sept. 2007.

[6] M. Stikic, T. Huynh, K. Van-Laerhoven, and B. Schiele, "ADL Recognition Based on the Combination of RFID and Accelerometer Sensing," *Proc. Int'l Conf. Pervasive Computing Technologies for Healthcare*, 2008.

[7] Y. Nakauchi, K. Noguchi, P. Somwong, and T. Matsubara, "Human Intention Detection and Activity Support System for Ubiquitous Sensor Room," *J. Robotics and Mechatronics*, vol. 16, no. 5, pp. 545-551, 2004.

[8] C. Lombriser, N.B. Bharatula, D. Roggen, and G. Tröster, "On-Body Activity Recognition in a Dynamic Sensor Network," *Proc. Int'l Conf. Body Area Networks (BodyNets)*, 2007.

[9] J.B.J. Bussmann, W.L.J. Martens, J.H.M. Tulen, F. Schasfoort, H.J.G. van den Berg-Emons, and H. Stam, "Measuring Daily Behavior Using Ambulatory Accelerometry: The Activity Monitor," *Behavior Research Methods, Instruments, and Computers*, vol. 33, no. 3, pp. 349-356, 2001.

[10] M. Philipose, K.P. Fishkin, M. Perkowitz, D.J. Patterson, D. Fox, H. Kautz, and D. Hähnel, "Inferring Activities from Interactions with Objects," *IEEE Pervasive Computing*, vol. 3, no. 4, pp. 50-57, Oct. 2004.

[11] D. Wilson and C. Atkeson, "Simultaneous Tracking and Activity Recognition (STAR) Using Many Anonymous, Binary Sensors," *Proc. Int'l Conf. Pervasive Computing*, pp. 62-79, 2005.

[12] J. Lester, T. Choudhury, and G. Borriello, "A Practical Approach to Recognizing Physical Activities," *Proc. Int'l Conf. Pervasive Computing*, 2006.

[13] J.A. Ward, P. Lukowicz, G. Tröster, and T.E. Starner, "Activity Recognition of Assembly Tasks Using Body-Worn Microphones and Accelerometers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1553-1567, Oct. 2006.

[14] D. Patterson, D. Fox, H. Kautz, and M. Philipose, "Fine-Grained Activity Recognition by Aggregating Abstract Object Usage," *Proc. IEEE Int'l Symp. Wearable Computers*, Oct. 2005.

[15] D. Wyatt, M. Philipose, and T. Choudhury, "Unsupervised Activity Recognition Using Automatically Mined Common Sense," *Proc. Am. Assoc. for Artificial Intelligence (AAAI) Conf.*, July 2005.

[16] W. Pentney, A.M. Popescu, S. Wang, H. Kautz, and M. Philipose, "Sensor-Based Understanding of Daily Life via Large-Scale Use of Common Sense," *Proc. Am. Assoc. for Artificial Intelligence (AAAI) Conf.*, July 2006.

[17] S. Wang, W. Pentney, A.M. Popescu, T. Choudhury, and M. Philipose, "Common Sense Based Joint Training of Human Activity Recognizers," *Proc. Int'l Joint Conf. Artificial Intelligence*, Jan. 2007.

[18] D.L. Vail, M.M. Veloso, and J.D. Lafferty, "Conditional Random Fields for Activity Recognition," *Proc. Int'l Conf. Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2007.

[19] T.Y. Wu, C.C. Lian, and J.Y. Hsu, "Joint Recognition of Multiple Concurrent Activities Using Factorial Conditional Random Fields," *Proc. Am. Assoc. for Artificial Intelligence (AAAI) Workshop Plan, Activity, and Intent Recognition*, July 2007.

[20] T.L.M. van Kasteren, A.K. Noulas, G. Englebienne, and B.J.A. Kröse, "Accurate Activity Recognition in a Home Setting," *Proc. 10th Int'l Conf. Ubiquitous Computing (UbiComp)*, Sept. 2008.

[21] D.H. Hu and Q. Yang, "CIGAR: Concurrent and Interleaving Goal and Activity Recognition," *Proc. Am. Assoc. for Artificial Intelligence (AAAI) Conf.*, 2008.

[22] D.H. Hu, S.J. Pan, V.W. Zheng, N.N. Liu, and Q. Yang, "Real World Activity Recognition with Multiple Goals," *Proc. 10th Int'l Conf. Ubiquitous Computing (UbiComp)*, Sept. 2008.

[23] J. Modayil, T.X. Bai, and H. Kautz, "Improving the Recognition of Interleaved Activities Research Note," *Proc. 10th Int'l Conf. Ubiquitous Computing (UbiComp)*, Sept. 2008.

[24] T. Huynh, M. Fritz, and B. Schiele, "Discovery of Activity Patterns Using Topic Models," *Proc. 10th Int'l Conf. Ubiquitous Computing (UbiComp)*, Sept. 2008.

[25] R. Hamid, S. Maddi, A. Bobick, and I. Essa, "Unsupervised Analysis of Activity Sequences Using Event Motifs," *Proc. ACM Int'l Workshop Video Surveillance and Sensor*, 2006.

[26] R. Hamid, S. Maddi, A. Johnson, A. Bobick, I. Essa, and C. Isbell, "A Novel Sequence Representation for Unsupervised Analysis of Human Activities," *Artificial Intelligence*, vol. 173, no. 14, pp. 1221-1244, 2009.

[27] 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," *Proc. Int'l Conf. Pervasive Computing*, pp. 349-365, 2006.

[28] Y. Yacoob and M.J. Black, "Parameterized Modeling and Recognition of Activities," *Proc. IEEE Int'l Conf. Computer Vision*, 1998.

[29] D. Moore, I. Essa, and M. Hayes, "Exploiting Human Actions and Object Context for Recognition Tasks," *Proc. IEEE Int'l Conf. Computer Vision*, 1999.

[30] Y.A. Ivanov and A.F. Bobick, "Recognition of Visual Activities and Interactions by Stochastic Parsing," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 852-872, Aug. 2000.

[31] I. Haritaoglu, D. Harwood, and L.S. Davis, "W4: Real-Time Surveillance of People and Their Activities," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809-830, Aug. 2000.

[32] J. Yamato, J. Ohya, and K. Ishii, "Recognizing Human Action in Time-Sequential Images Using Hidden Markov Model," *Proc. Int'l IEEE CS Conf. Computer Vision and Pattern Recognition*, 1992.

[33] N. Oliver, A. Garg, and E. Horvitz, "Layered Representations for Learning and Inferring Office Activity from Multiple Sensory Channels," *Computer Vision and Image Understanding*, vol. 96, no. 2, pp. 163-180, 2004.

[34] X.D. Sun, C.W. Chen, and B.S. Manjunath, “Probabilistic Motion Parameter Models for Human Activity Recognition,” *Proc. IEEE Int'l Conf. Pattern Recognition,* 2002.

[35] J.B. Arie, Z.Q. Wang, P. Pandit, and S. Rajaram, “Human Activity Recognition Using Multidimensional Indexing,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 8, pp. 1091-1104, Aug. 2002.

[36] M. Brand and V. Kettnaker, “Discovery and Segmentation of Activities in Video,” *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 8, pp. 844-851, Aug. 2000.

[37] N. Robertson and I. Reid, “A General Method for Human Activity Recognition in Video,” *Computer Vision and Image Understanding,* vol. 104, no. 2, pp. 232-248, 2006.

[38] F. Fusier, V. Valentin, F. Brémond, M. Thonnat, M. Borg, D. Thirde, and J. Ferryman, “Video Understanding for Complex Activity Recognition,” *Machine Vision and Applications,* vol. 18, nos. 3/4, pp. 167-188, Aug. 2007.

[39] T. Huang, D. Koller, J. Malik, G.H. Ogasawara, B. Rao, S.J. Russell, and J. Weber, “Automatic Symbolic Traffic Scene Analysis Using Belief Networks,” *Proc. Nat'l Conf. Artificial Intelligence,* 1994.

[40] A.A. Efros, A.C. Berg, G. Mori, and J. Malik, “Recognizing Action at a Distance,” *Proc. IEEE Int'l Conf. Computer Vision,* pp. 726-733, 2003.

[41] E. Shechtman and M. Irani, “Space-Time Behavior Based Correlation,” *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition,* pp. 405-412, 2005.

[42] H. Kautz, “A Formal Theory of Plan Recognition,” PhD dissertation, Univ. of Rochester, 1987.

[43] G.Z. Dong and J.Y. Li, “Efficient Mining of Emerging Patterns: Discovering Trends and Differences,” *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining,* pp. 43-52, Aug. 1999.

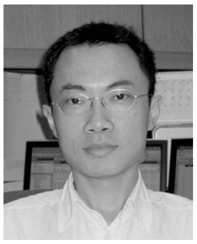
[44] G.Z. Dong, X. Zhang, L. Wong, and J.Y. Li, “CAEP: Classification by Aggregating Emerging Patterns,” *Proc. Second Int'l Conf. Discovery Science,* Dec. 1999.

[45] J.Y. Li, H.Q. Liu, J.R. Downing, A.E. Yeoh, and L. Wong, “Simple Rules Underlying Gene Expression Profiles of More Than Six Subtypes of Acute Lymphoblastic Leukemia (ALL) Patients,” *Bioinformatics,* vol. 19, no. 1, p. 71-78, 2003.

[46] J.Y. Li, H.Q. Liu, S.K. Ng, and L. Wong, “Discovery of Significant Rules for Classifying Cancer Diagnosis Data,” *Bioinformatics,* vol. 19, pp. ii93-ii102, 2003.

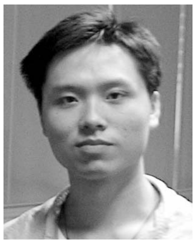
[47] J.Y. Li, G.M. Liu, and L. Wong, “Mining Statistically Important Equivalence Classes and Delta-Discriminative Emerging Patterns,” *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining,* pp. 430-439, 2007.

[48] U. Fayyad and K. Irani, “Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning,” *Proc. Int'l Joint Conf. Artificial Intelligence,* 1993.



Tao Gu received the BEng degree from Huazhong University of Science and Technology, the MSc degree from Nanyang Technological University, Singapore, and the PhD degree in computer science from the National University of Singapore. He is currently an assistant professor in the Department of Mathematics and Computer Science at the University of Southern Denmark (SDU). He is also the funding director of the Advanced Network Systems Lab at SDU. His research interests include pervasive computing, wireless sensor networks, and distributed computing. He is a member of the IEEE and the ACM.

Liang Wang received the BSc degree in computer science from Nanjing University in 2007. He is currently a PhD candidate in the Department of Computer Science at Nanjing University, supervised by Prof. Jian Lu and Dr. Tao Gu. His research interests include pervasive computing and wireless sensor networks. He is a student member of the IEEE.



Zhanqing Wu received the BSc degree in computer science from Nanjing University in 2006. He is a MS student in the Department of Computer Science at Nanjing University, supervised by Prof. Xianping Tao and Dr. Tao Gu. His research interests include pervasive computing and wireless sensor networks.

Xianping Tao received the MSc and PhD degrees in computer science from Nanjing University in 1994 and 2001, respectively. He is currently a professor in the Department of Computer Science at Nanjing University. His research interests include software agents, middleware systems, Internetware methodology, and pervasive computing. He is a member of the IEEE.

Jian Lu received the BSc, MSc, and PhD degrees in computer science from Nanjing University in 1982, 1984, and 1988, respectively. He is currently a professor in the Department of Computer Science at Nanjing University. He is also the director of the State Key Laboratory for Novel Software Technology and the vice director of the Institute of Software Technology at Nanjing University. His research interests include programming methodology, pervasive computing, software agent, and middleware. He is a member of the ACM.