epSICAR: An Emerging Patterns based Approach to Sequential, Interleaved and Concurrent Activity Recognition

Tao Gu [a], Zhanqing Wu [b], Xianping Tao [b], Hung Keng Pung [c], Jian Lu [b]

[a] Institute for Infocomm Research, Singapore

[b] State Key Laboratory for Novel Software Technology, Nanjing University, China

[c] School of Computing, National University of Singapore, Singapore

tgu@i2r.a-star.edu.sg; wzq@ics.nju.edu.cn; txp@ics.nju.edu.cn; dcsphk@nus.edu.sg; lj@nju.edu.cn

***Abstract*—Recognizing human activities from sensor readings has recently attracted much research interest in pervasive computing. 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 and concurrent) manner in real life. In this paper, we propose a novel Emerging Patterns based approach to Sequential, Interleaved and Concurrent Activity Recognition (*epSICAR*). We exploit Emerging Patterns as powerful discriminators to differentiate activities. Different from other learning-based models built upon the training dataset for complex activities, we build our activity models by mining a set of Emerging Patterns from the sequential activity trace only and apply these models in recognizing sequential, interleaved and concurrent activities. We conduct our empirical studies in a real smart home, and the evaluation results demonstrate that with a time slice of 15 seconds, we achieve an accuracy of 90.96% for sequential activity, 87.98% for interleaved activity and 78.58% for concurrent activity.**

Keywords- activity recognition; emerging patterns; sequential, interleaved and concurrent activities; wireless sensor networks

I. INTRODUCTION

With the advancement of wireless sensor network technology, recognizing human activities based on sensor readings has recently drawn much research interest from the pervasive computing community. Different from video-based activity recognition in computer vision, in this paradigm, there are typically different types of sensors deployed. These sensors may be wearable by a user or embedded in the environment. Sensor readings are then collected and interpreted to recognize various human activities. A typical application in healthcare is monitoring Activities of Daily Living (ADLs) [1] for the elderly and providing them with proactive assistance.

In real life, people perform ADLs in not only a sequential manner (i.e., performing one ADL after another), but also an interleaved (i.e., switching between the steps of two or more ADLs) or concurrent manner (i.e., performing two or more ADLs simultaneously). Recognizing activities in such a complex situation has practical implication to real-life pervasive computing applications; hence, it attracts our research interest and motivates this work.

Little work has been done in providing a unified solution to solve complex issues that arise in recognizing sequential, interleaved and concurrent activities. Activity recognition is typically viewed as a classification problem where many traditional machine learning techniques can be applied. Probabilistic-based models are popular due to its ability of handling noise and uncertainty in sensor readings. Some recent work showed that the variants of Conditional Random Field (CRF) [2-3] can be used to model interleaved and concurrent activities. However, these learning-based techniques require that a predicting instance (i.e., an interleaved or concurrent activity instance to be predicted) must have its model presented in the training dataset. Considering a great variety of ways in which daily activities can be interleaved and performed concurrently, the training dataset for learning such complex activity models has to be large enough. These solutions may lack flexibility in real-life deployment.

In this paper, we formulate activity recognition as a pattern based classification problem, and propose a novel Emerging Patterns based approach to recognize sequential, interleaved and concurrent activities. We build our activity models based on Emerging Patterns (EPs) [4] – a type of knowledge pattern that describes significant changes between two classes of data. We mine a set of EPs for each sequential activity from the training dataset, and use the sets of EPs obtained to recognize not only simple (i.e., sequential), but also complex (i.e., interleaved and concurrent) activities. This approach does not require training for complex activities; and hence it has a great flexibility and applicability for real-life pervasive computing applications. We conduct a real-world activity trace collection done by four volunteers in a smart home environment, and our evaluation results demonstrate both the effectiveness and flexibility of our solution.

In summary, the paper makes the following contributions.

- *We formulate activity recognition as a pattern based classification problem, and propose a novel Emerging Patterns based approach to recognize sequential, interleaved and concurrent activities in a unified solution.*
- *We propose a novel trace segmentation algorithm based on feature relevance to segment the boundary of any two adjacent activities during the recognition process.*

978-1-4244-3304-9/09/$25.00 ©2009 IEEE

- *We conduct a real-world trace collection using wireless sensors, evaluate our algorithm through comprehensive experiments and analyze our algorithm in detail.*

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 describes our sensor setup for data collection and Section 4 gives the background on EPs. We then describe the mining of EPs in Section 5, and present our activity recognition algorithm in Section 6. Section 7 reports our empirical studies. Finally, Section 8 concludes the paper.

II. Related Work

In pervasive computing, researchers are recently interested in recognizing activities using sensors. As sensor readings are usually noisy and activities are typically performed in a non-deterministic fashion, probabilistic learning-based methods are appropriate and gain more popularity.

Probabilistic-based models can be categorized into static classification or temporal classification. In static classification, sensor readings are first pre-processed into features, and then a static classifier is applied to classify the activities. Typical static classifiers include naïve Bayes used in [5-7], C4.5 decision trees used in [6-8], k-nearest neighbor (k-NN) used in [6, 8-9], and support vector machine (SVM) used in [9]. Multiple binary classifiers [5, 7] can be exploited to recognize interleaved and concurrent activities; however, this solution may not work properly because many activities share common features.

In temporal classification, state-space models are typically used to enable the inference of hidden states (i.e., activity labels) given the observations (i.e., sensor readings). We name a few examples here: Hidden Markov Model (HMM) used in [10-13], Dynamic Bayesian Network (DBN) used in [14-15] and CRF used in [16]. Recent work showed that SCCRF [3] – a variant of CRF – and IHMM [17] – a variant of HMM – can be used to model interleaved activities, and FCRF [2] – another variant of CRF – can be used to model concurrent activities. However, like other learning-based techniques, they require that a predicting instance must have its model presented in the training dataset. On one hand, this implies that the training dataset has to be large enough to build the complete models for interleaved and concurrent activities. Any partial model will result in a loss of recognition accuracy. On the other hand, in real life there exists a great variety of ways in which daily activities can be interleaved and performed concurrently. It may be not possible to construct a complete model by training. Hence, the flexibility and scalability of the solutions in [2-3, 17] are limited.

A different attempt to recognize activities is Time Series based classification, in which an activity is modeled as a sequence of discrete events [18]. 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. The EPs based classifier model uses a set of multi-attribute 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. Our activity models are built upon the sequential activity instances only and will be applied to recognize interleaved and concurrent activities.

III. Our Sensor Setup

We built a wireless sensor platform from off-the-shelf sensors to collect sensor information. Our sensor platform measures a user's movement (i.e., left hand, right hand and body movements), user location, the living environment (i.e., temperature, humidity and light), and the human-object interaction (i.e., objects a user touches).

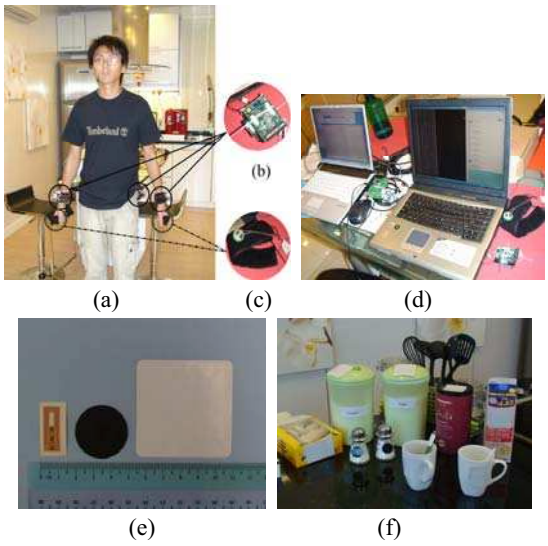

Figure 1. (a) Wearable sensors consist of 2 RFID wristband readers and 3 iMote2 sets, (b) iMote2 set, (c) RFID wristband reader, (d) servers, (e) HF RFID tags, and (f) HF RFID tagged objects.

Figures 1a, 1b and 1c illustrate our wearable sensors. A subject wears a Crossbow iMote2 set on each of his hands and his waist. Each iMote2 set (Figure 1b) consists of an IMote2 IPR2400 processor/radio board and an ITS400 sensor board, capable of measuring the movement of a user's hands and body (3-axis accelerometer), the temperature, humidity and light of the environment.

A subject also wears an RFID wristband reader (Figure 1c) on each of his hands. The custom-built wristband incorporates a SkyeTek M1-mini RFID reader, a Crossbow Mica2Dot module and a rechargeable battery. The wristband is able to detect the presence of a tag within the range of 6 to 8 cm. HF RFID tags are attached to day-to-day objects such as cups, teaspoons and books in a smart home. Figure 1e shows three types of such tags we used and they operate on 13.56MHz. Figure 1f shows a screen shot 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 daily activities, the sensor readings from the three iMote2 sets are transferred wirelessly to a local server (Figure 1d, left) which runs on a laptop PC with an iMote2 IPR2400 board connected through its USB port. When a subject handles a tagged object, the wristband scans the tag ID and sends it wirelessly to another server (Figure 1d, right) that can map the ID to an object name. This server runs on a Linux-based laptop PC with a MIB510CA serial interface board and a Mica2Dot module connected through its serial port. The sensor readings are recorded by the two servers separately, and will be merged into a single txt file as the activity trace. Our objective is to design an algorithm to recognize the activities in the activity trace.

IV. Emerging Patterns and Preliminaries

We provide the background of EPs in this section. Discovery of powerful distinguishing features between datasets is an important objective in data mining. Emerging Pattern is a new type of knowledge pattern that describes significant changes (differences or trends) between two classes of data [4]. An EP is a set of items whose frequency changes significantly from one dataset to another. Like other patterns or rules composed of conjunctive combinations of elements, EPs can be easily understood and used directly.

We first give some preliminary definitions. Suppose that a dataset D consists of many instances. An instance contains a set of items (i.e., an itemset), where an item is an attribute-value pair. The support of an itemset 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 patterns, Emerging Patterns are concerned with two classes of data.

Definition 1: Given two different classes of datasets D_1 and D_2, the growth rate of an itemset X from D_1 to D_2 is defined as $GrowthRate(X) =$

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

EPs are those itemsets with large growth rates from D_1 to D_2.

Definition 2: Given a growth rate threshold $\rho > 1$, an itemset X is said to be a *ρ-Emerging Pattern* (or simply EP) from a background dataset D_1 to a target dataset D_2 if $GrowthRate(X) \geq \rho$.

An EP with high support in its target class and low support in the contrasting class can be seen as a strong signal indicating the class of a test instance containing it.

Example 1. Consider the following training dataset with two classes, P and N (this is an encoding of the Saturday morning activity example from [22]).

P	***N***
{ 2,6,7,10 } { 3,5,7,10 } { 3,4,8,10 } { 2,4,8,9 } { 1,4,8,10 } { 3,5,8,10 } { 1,5,8,9 } { 2,5,7,9 } { 2,6,8,10 }	{ 1,6,7,10 } { 1,6,7,9 } { 3,4,8,9 } { 1,5,7,10 } { 3,5,7,9 }

Then {1, 9} is an EP from class P to class N with a growth rate $\frac{9}{5}$; it is also a ρ-EP for any $1 < \rho \leq \frac{9}{5}$.

V. Mining Emerging Patterns from the Sequential Activity Trace

A. Problem Statement

We formulate the problem of sequential, interleaved and concurrent activity recognition as follows. Given a training dataset that consists of an observation sequence for sequential activities only (i.e., formally, a training dataset 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.

B. Data Preprocessing

Before we introduce the mining of EPs, we need to select appropriate sensor features and discretize their readings. We convert the sensor readings to a series of *observation vectors* where each *observation vector* consists of 15 features as shown below.

o = [*accel_body_x*, *accel_body_y*, *accel_body_z*, *accel_right_x*, *accel_right_y*, *accel_right_z*, *accel_left_x*, *accel_left_y*, *accel_left_z*, *temperature*, *humidity*, *light*, *location*, *left_object*, *right_object*]

We compute each *observation vector* in a fixed time interval which is set to one second in our experiments. Corresponding to the type of sensors, these features have two types of values. For an RFID reading, we use the object name (e.g., cup, toothpaste, etc.) as its feature value. Since a subject is unlikely to touch two or more objects in a one-second interval when performing activities, we choose the first object for each RFID wristband reader in such an interval. If no RFID reading is observed or in the presence of a corrupted tag ID, a *null* value will be given. For numeric sensor readings, we take the average of all the readings in this interval as the feature value. When the sensor readings are not observed in the interval due to sensor errors, we use the value from the previous interval, where we assume the states of a subject and the environment will not be changed in such a short interval.

We then transform these *observation vectors* into *feature vectors*. A *feature vector* consists of many *feature items*, where a *feature item* refers to a feature-value pair in which a feature 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.

To discretize numeric features, we use the entropy-based discretization method [19] which partitions a range of continuous values into a number of disjoint intervals such that the entropy of the partition is minimal.

The discretization method partitions 12 numeric feature values into a total of 267 disjoint intervals. Then we can directly combine the feature name and its interval into a *numeric feature item.* For example, *accel_body_x@(-737.5~-614.5*] is one such example in our training dataset. For nominal feature, the feature name and its value are combined as a *nominal feature item*. For the *left_object* and *right_object* features, we merge them into one feature by computing *left_object* ∪ *right_object* without losing any essential objects during the user-object interaction due to user's handedness. For example, a *nominal feature item* can be *object@cup* or *location@bathroom*. Based on our current sensor setup, we have a total of 354 *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.

C. Mining Emerging Patterns

In our training dataset, each observation is assigned with an activity label(s). We first identify the instances for each sequential activity in the training dataset. 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, denoted as 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 dataset. We refer EP_{SA_i} as the EPs of SA_i.

We discover the EPs by an efficient algorithm described in [20]. After computation, we get m sets of EPs, one set per sequential activity. Table 1 presents an example of the top 8 EPs of the *cleaning a dining table* activity. Column 1 shows the EPs. For example, the EP "*object@cleanser, object@plate, object@wash_cloth, location@kitchen*" has a support of 95.24% 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~-453.5*], *light@(24.5-28.5*], *object@plate, location@kitchen*" has a support of 66.67% 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 the left hand movement when a subject performed this activity.

TABLE I. A SUBSET OF EPS FOR THE *CLEANING A DINING TABLE* ACTIVITY

EPs	Support (%)	Growth rate
location@kitchen, object@plate	100	∞
object@dining_table, object@plate	95.24	365.7
object@cleanser, object@plate, object@wash_cloth, location@kitchen	95.24	∞
object@light_power_switch, object@plate, location@kitchen	95.24	∞
object@wash_cloth, object@bowl, object@plate, object@cleanser, location@kitchen	90.48	∞
object@spoon, object@plate, object@bowl, object@dining_table, location@kitchen	80.95	∞
object@bowl, accel_body_x@(-155.25--52.25], light@(24.5-28.5], object@plate	66.67	256
object@bowl, accel_left_z@(-684.5~-453.5], light@(24.5-28.5], object@plate, location@kitchen	66.67	∞

VI. THE *epSICAR* ALGORITHM

We first give an overview of our proposed activity recognition algorithm, as illustrated in Figure 2. Given a new observation sequence from time $t = 0$ to T, our algorithm aims to assign the correct activity label(s) to each observation. At time t, for each activity A_i, we first obtain a test instance $S_{t\sim t+L_{A_i}}$ by computing the union of all the observations from t to $t + L_{A_i}$ using a slide window L_{A_i} which is the average length of all the instances of A_i. We compute the score for A_i based on a score function, and obtain an activity label $A_{previous}$ that yields the highest score. The same process is repeated to obtain another activity label $A_{current}$ that yields the highest score in the next slide window. We then apply our trace segmentation algorithm to detect and adjust the boundary between $A_{previous}$ and $A_{current}$. Finally, we compute the score again and assign the label $A_{candidate}$, yielding the highest score, to $S_{t\sim t+L_{A_i}}$. The above process is repeated until the end of the sequence, i.e., $t = T$.

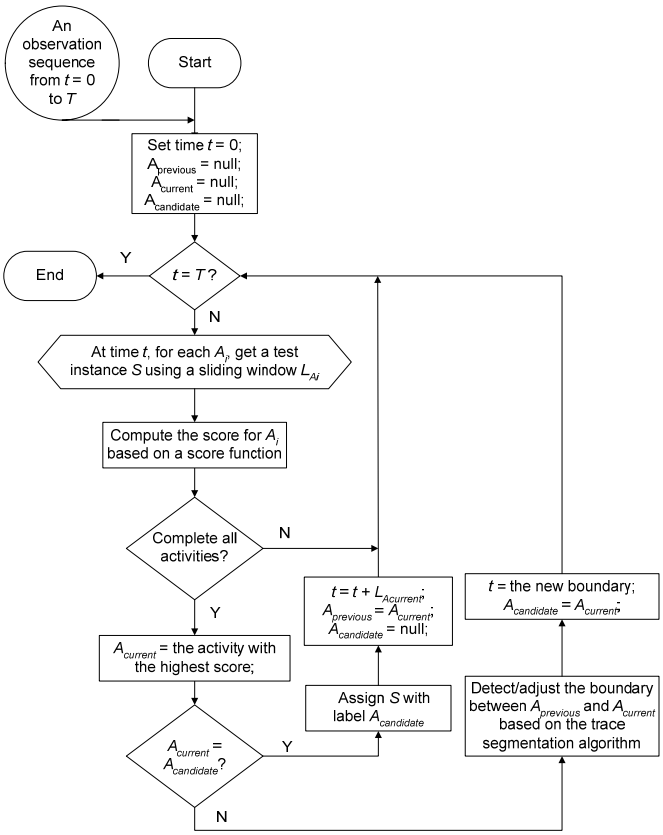

Figure 2. Flow chart of the *epSICAR* algorithm

We design the score function based on three types of scores: *EP score*, *slide-window coverage score* and *activity-correlation score*. In the remaining of this section, we first describe the score function for sequential activity, followed by the score computation for interleaved and concurrent activities. We then describe our *epSICAR* algorithm in detail.

A. The Score Function for Sequential Activity

1) EP Score

A single EP can sharply differentiate the class membership of a fraction of the test instance $S_{t\sim t+L_{SA_i}}$ which contains the EP. To make use of each set of EP to achieve good overall accuracy, we combine the strength of EPs based on the aggregation method described in [21]. The aggregated score of $S_{t\sim t+L_{SA_i}}$ for SA_i is defined as follows.

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

where $supp_{SA_i}(X)$ is the support of X in SA_i, and $growth\_rate(X)$ is $supp_{SA_i}(X)$ divided by the X's support in non-SA_i classes. The EP scores of each activity are then "normalized" by dividing them using the median of the scores of the training instances of that activity. Finally, the *EP score* is defined as follows.

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

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 2. For an instance S = {1, 3, 4, 6, 7, 8, 10, 11} in SA_i which contains 4 EPs: ({1, 3}, 100%, ∞), ({2, 3}, 95%, 150), ({1, 2, 3, 8}, 81%, ∞), ({1, 7}, 62%, 80). The differentiating power of the 4 EPs is computed as 1.0, 0.95, 0.81, 0.62 respectively, then its $aggregated\_score(SA_i, S)$ = 3.38. The aggregated scores of SA_i for all its training instances are 2.76, 3.38, 3.56, 3.66, 3.85, 3.85, 4.12, and the median is 3.66. Finally, we obtain $ep\_score(SA_i, S)$ = 0.92.

2) Slide-window Coverage score

The *EP score*, $ep\_score(SA_i, S_{t\sim t+L_{SA_i}})$, provides a measurement on the fraction of EP_{SA_i} (i.e., the discriminating features of SA_i) that is contained in the test instance $S_{t\sim t+L_{SA_i}}$. However, $S_{t\sim t+L_{SA_i}}$ may contain other observations that do not belong to SA_i. We introduce the *Slide-Window Coverage score* (i.e., *coverage score* for short) to describe a fraction of irrelevant observations to SA_i which are contained in $S_{t\sim t+L_{SA_i}}$.

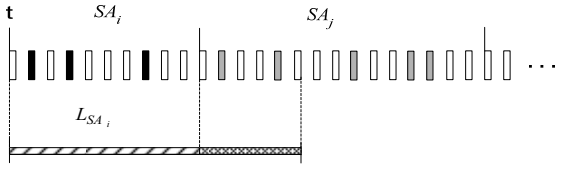

Figure 3. Illustration of Slide-window coverage

Figure 3 illustrates this concept. The bars in the figure represent the feature vectors computed from the observations. The feature vectors from time t are labeled with the ground truth SA_i, followed by SA_j. The black bars represent a subset of EP_{SA_i}, while the gray bars represent a subset of EP_{SA_j}. During the prediction step, a slide window L_{SA_i} is applied to get instance $S_{t\sim t+L_{SA_i}}$. In this example, although $S_{t\sim t+L_{SA_i}}$ includes the fractions of EP_{SA_i}, it covers some observations of its adjacent activity SA_j as well. Therefore, we introduce the *coverage score* to measure the fraction of irrelevant observations in a slide window. The lower the percentage of irrelevant observations covered; the larger the *coverage score* 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 $relevance(SA_i, f_p)$, where a feature vector f_p 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 to compute $relevance(SA_i, f_p)$.

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

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

where the probability $P(item_h \mid SA_i)$ is obtained from the training data, and $\sum_{item_h\in X, X\in EP_{SA_i}} supp_{SA_i}(X)$ indicates more weights 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 $\sum_{item_h\in f_k} relevance(SA_i, item_h)$. Then the normalized $relevance(SA_i, f_p)$ is computed as follows.

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

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.

Finally, $coverage\_score(SA_i, S_{t\sim t+L_{SA_i}})$ can be computed by averaging all the $relevance(SA_i, f_p)$ as follows.

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

3) Activity-correlation score

Human activities are usually performed in a non-deterministic fashion. However, there exist some correlations between them, i.e., when SA_j has been performed, the probability of 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 correlations between activities. We define the *activity-correlation score* of SA_i as $P(SA_i \mid SA_j)$, which is the conditional probability of SA_i given SA_j. We can easily obtain the *activity-correlation scores* for all activities from the training dataset. Note that the initial value is set to zero, i.e., $P(SA_i \mid NULL) = 0$.

B. The Score Function for Interleaved and Concurrent Activities

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

We denote CA_i as both interleaved activities (i.e., CA_i is represented as SA_a & SA_b in this case) and concurrent activities (i.e., CA_i is represented as SA_a + SA_b in this case), where SA_a and SA_b are involved in. We define the slide-window length L_{CA_i} of CA_i as $L_{SA_a} + L_{SA_b}$, 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.

$$ep\_score(CA_i, S_{t\sim t+L_{CA_i}}) = \frac{1}{2}[ep\_score(SA_a, S_{t\sim t+L_{CA_i}}) + ep\_score(SA_b, S_{t\sim t+L_{CA_i}})]$$

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 rationale behind this is that a feature vector f_p, which belongs to an activity usually, has a higher relevance to this activity. Hence, we have

$$relevance(CA_i, f_p) = \max(relevance(SA_a, f_p), relevance(SA_b, f_p))$$

Then the *coverage score* of CA_i can be computed as follows.

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

The computation of *activity-correlation score* for interleaved and concurrent activities can be quite complex. There are three situations: a sequential activity followed by interleaved or concurrent activities, interleaved or concurrent activities followed by a sequential activity, and interleaved or concurrent activities followed by other interleaved or concurrent activities. Given the rationale that a 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 $CA_i = SA_a + SA_b$, the *activity-correlation score* of CA_i, where $CA_j = SA_c$ & SA_d or $CA_j = SA_c + SA_d$, can be computed as follows.

$$P(CA_i \mid CA_j) = \max(P(SA_a \mid SA_c), P(SA_a \mid SA_d), P(SA_b \mid SA_c), P(SA_b \mid SA_d))$$

The computation of $P(SA_i \mid CA_j)$ and $P(CA_i \mid SA_j)$ follows the same method.

C. Total Score

In summary, the computation of the total score 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 score function of A_i is then defined as follows.

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

where c_1, c_2 and c_3 are the coefficients. The coefficient represents the importance of an individual score, and its value implies different meanings. For example, if c_1 is high, it implies that the activities always follow certain patterns well. 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. If c_3 is high, it implies that there exist strong correlations among the activities performed. These weights reflect the activity habit of a subject.

D. Recognizing Activity using a Slide Window

Algorithm 1: The Slide-window based Recognition Algorithm

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 slide window.
Output: the activity label starts from time t

1: **foreach** activity A_i, $i = 1, 2, \ldots, m^2$ **do**
2: get instance $S_{t\sim t+L_{A_i}} = \bigcup_{p=t}^{t+L_{A_i}} f_p$;
3: compute $score(A_i, A_j, S_{t\sim t+L_{A_i}})$;
4: **end for**
5: return A_i with the highest score;

We now can apply the score function to recognize the activity label in the test instance obtained using a slide window method. Given m sequential activities, the number of interleaved and concurrent activities can be computed by $m(m-1)$. Then the total number of activities is m^2. We define L_{max} as $\max\{L_{A_k}\}$, where $k = 1, 2, \ldots, m^2$. A straightforward method is to test each activity label using its corresponding slide window and the one with the highest score wins out. Algorithm 1 describes this approach.

E. The Trace Segmentation Algorithm

The simple slide-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 slide window length L_{A_i} of each activity is an approximation of the actual length, the segmentation may not be accurate. Moreover, any error in one segment may affect the recognition of the subsequent trace. This error may accumulate and affect the recognition accuracy seriously.

To overcome this limitation, we propose our trace segmentation algorithm as presented in Algorithm 2. We first obtain two predicted activity labels (e.g., A_j followed by A_i) based on Algorithm 1. The task of our segmentation algorithm is to determine the boundary between A_j and A_i accurately so that the next slide window can be applied from this boundary.

Intuitively, given an activity instance, its feature vectors obtained by pre-processing 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. Based on these heuristics, we introduce our trace segmentation algorithm as described in Algorithm 2. This algorithm makes use of the weight difference, i.e., *Relative Weight* (*RW*), of each feature vector between the two adjacent activities.

Algorithm 2: The Trace Segmentation Algorithm

Input: feature vector of length $L_{A_j} + L_{A_i} : F = \{ f_{t-L_{A_j}}, \ldots, f_{t+L_{A_i}} \}$,
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

```
foreach p from t- L_{A_j} to t+ L_{A_i}  do
    RW[p] = relevance(A_j, f_p) - relevance(A_i, f_p) ;
end for
foreach p from t- L_{A_j} to t+ L_{A_i}  do
    upperSum = sum of all RWs from t- L_{A_j}  to p ;
    lowerSum = sum of all RWs from p to t+ L_{A_i} ;
    GAIN[p] = upperSum – lowerSum ;
end for
    boundary = p such that GAIN[p] is maximum ;
return boundary;
```

VII. Empirical Studies

We now move to evaluate our proposed algorithm. We conducted our own trace collection in a complex, real-world situation. In this section, we first describe our experimental setup and metric, then present and discuss the results obtained from a series of experiments.

A. Experimental Setup and Metric

Figure 4. (a) StarHome – a real smart home, (b) a video snapshot shows that a subject is performing an activity in the kitchen, (c) another video snapshot shows that a subject is performing an activity in the living room.

Trace collection was done in StarHome – a real smart home built by our organization – as shown in Figure 4a. We deployed our sensor platform and tagged over 100 objects. We randomly selected 26 activities out of common ADLs as summarized in Table 2. We then chose 15 interleaved activities and 16 concurrent activities, e.g., *using computer* and *using phone* ("23 & 20", interleaved), *brushing teeth* while *listening music/radio* ("9 + 25", concurrent), etc. The data were collected by four volunteers over a period of two weeks. Each day, each of them performed these activities at his choice in his own way. However, the duration of each activity is shorter as compared to the actual duration in his daily life in order to collect more instances. A sequence of trace was logged in the server. There was only one subject performing activities at any given time. Location information was recorded partially by hand due to the limitation of our indoor location system. One of the volunteers annotated the traces to establish the ground truth together with video recording. Figures 3b and 3c show two video snapshots in our data collection.

TABLE II. 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

Table 3 shows a total number of 532 activity instances collected and a breakdown in the three types of activities.

TABLE III. Number of Instances Collected

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

We use ten-fold cross-validation for our evaluation and evaluate the performance of our algorithm using the time-slice accuracy which is a typical technique in time-series 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 doest not affect the accuracy much. This time slice duration is short enough to provide precise measurements 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 can be defined as

$$Slice\_Accuracy = \sum_{SA_i \in LB_G \cap LB_R} L_{SA_i} \Big/ \sum_{SA_i \in LB_G \cup LB_R} L_{SA_i}$$

We now give a few examples to illustrate the time-slice accuracy. Given $LB_G = \{9, 25\}$, if the predicted labels $LB_R = \{9, 25\}$, then *Slice_Accuracy* = 1 since $LB_G \cap LB_R = LB_G \cup LB_R$. If $LB_R = \{10\}$, then *Slice_Accuracy* = 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 computed as follows.

$$\text{Total\_ Accuracy} = \frac{\sum_{1}^{T/\Delta t} Slice\_Accuracy}{T/\Delta t}$$

B. *Experiment 1: Accuracy Analysis*

In this experiment, we evaluate the accuracies of different types of activities. Table 4 shows the average accuracies of sequential, interleaved and concurrent activities, respectively, and the overall accuracy. Figure 5 shows the breakdown of the types of activities in the ten datasets.

TABLE IV. OVERALL ACCURACY

Type of Activity	Time-slice Accuracy
sequential	90.96%
interleaved	87.98%
concurrent	78.58%
total	88.11%

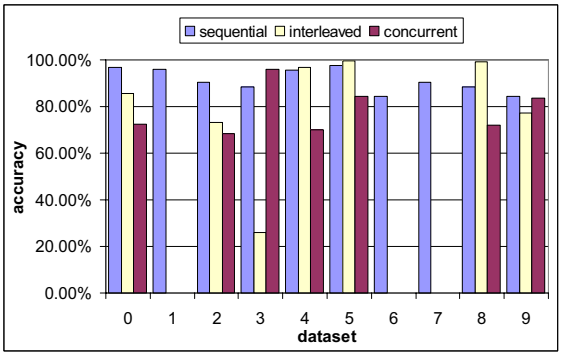

Figure 5. Accuracy breakdown in the ten datasets

The accuracy of sequential activity is the highest among all the three types. The accuracies of interleaved and concurrent activities are lower; it probably can be explained as follows. Firstly, we have four subjects, and they are instructed to perform their activities in their own ways. 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 subjects. However, some of the specific characteristics of interleaved and concurrent activities performed by different subjects may not be captured by our model. Nevertheless, the result is reasonably good, and we achieved our objective of building the activity model to recognize all types of activities based on the sequential activity training data only.

Second, the accuracy of concurrent activity is 9.4%, lower than that of interleaved activity, while the accuracy of interleaved activity is close to that of sequential activity. It is probably due to the slide-window length. In our model, we apply $L_{CA_i} = L_{SA_a} + L_{SA_b}$ to calculate the slide-window length of CA_i. This estimation seems to work well in the case of interleaved activity because 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 $L_{SA_j} + L_{SA_k}$.

Another observation from Figure 5 is that the variances of interleaved and concurrent activities are larger than that of sequential activity in the ten datasets. This may be caused by the imbalance of instances in our datasets. The number of interleaved and concurrent activity instances is much smaller than that of sequential activity instances as shown in Table 3.

C. *Experiment 2: Model Analysis*

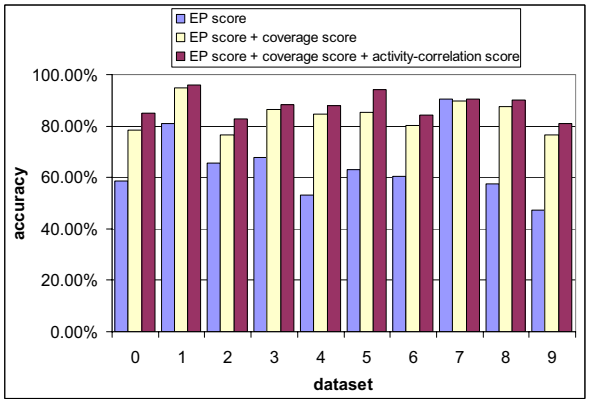

Figure 6. Analysis of the score function

In this experiment, we evaluate and analyze the effectiveness of different models in our algorithm. We first evaluate the *epSICAR* algorithm with respect to our score function. Figure 6 shows that the accuracies of the *epSICAR* algorithm with *EP score*, *EP score* + *coverage score*, and *EP score* + *coverage score* + *activity-correlation score*, respectively. The figure suggests that *epSICAR* achieves an accuracy of 66% on average with the *EP score* only, demonstrating that the concept of *EPs* works effectively. However, the effectiveness of the *EP score* is not as high as expected and there exists some variations. We analyze this case and suggest two reasons. First, the use of *EPs* in activity recognition is leveraged on mining discriminating features. The more discriminating features collected, the better *EPs* mined and the better results obtained. Our current sensor platform provides only 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 the contribution of each set of *EPs*. We will further improve this method in our future work. Figure 6 also suggests that, by introducing the *coverage score*, the accuracy is improved significantly by about 19% and the variance also decreases. We also observe that the accuracy is further improved by about 3% when adding in the *activity-correlation score*.

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

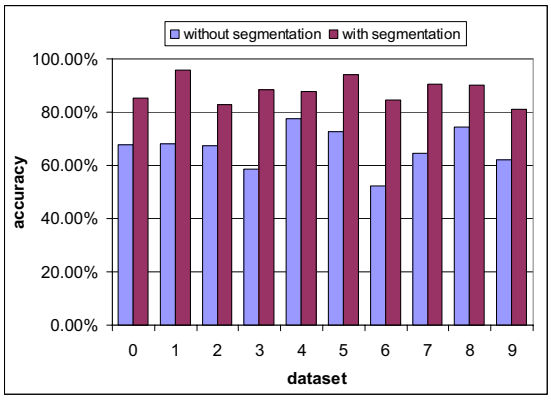

Figure 7. Effect of the trace segmentation alrogithm

VIII. CONCLUSION

In this paper, we studied the problem of human activity recognition based on sensor readings in a pervasive computing environment. We deployed our sensor platform and conducted a real-world trace collection done by four volunteers in a smart home for two weeks. Our dataset contains comprehensive activity instances for our study. We then investigated a challenging problem of how we can apply a model, which can be learnt from sequential activity instances only, in recognizing both simple (i.e., sequential) and complex (i.e., interleaved and concurrent) activities. We exploited EPs as powerful discriminators to differentiate activities and proposed the *epSICAR* algorithm. We conducted comprehensive evaluation studies using our dataset and analyzed our algorithm in detail. The results demonstrate both the effectiveness and flexibility of our algorithm.

For our future work, we will further develop our sensor platform to include more sensor features and fully explore the discriminating power of EPs. We will also look into a better method for aggregating the contribution of each set of EPs.

ACKNOWLEDGMENT

This work is funded by the Science and Engineering Research Council of Singapore under the research grant SERC 0521210083, and partially supported by National Natural Science Foundation of China under the research grants NSFC 60721002, NSFC 60736015. We thank all the reviewers for their constructive comments.

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. Journal of the American Medical Association, 185: 914-919, September 1963.

[2] Tsu-yu Wu, Chia-chun Lian, and Jane Yung-jen Hsu, "Joint recognition of multiple concurrent activities using factorial conditional random fields," In Proceedings of AAAI Workshop on Plan, Activity, and Intent Recognition, California, July 2007.

[3] Derek Hao Hu and Qiang Yang, "CIGAR: concurrent and interleaving goal and activity recognition," In Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI 2008), Chicago, Illinois, USA, 2008.

[4] Guozhu Dong and Jinyan Li, "Efficient mining of emerging patterns: discovering trends and differences," In Proc. 5th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD'99), pages 43–52, San Diego, CA, USA, Aug 1999.

[5] E. Munguia Tapia, S. S. Intille, and K. Larson, "Activity recognition in the home setting using simple and ubiquitous sensors," In Proceedings of PERVASIVE 2004, vol. LNCS 3001, pp. 158-175, 2004.

[6] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," In Proceedings of PERVASIVE 2004, vol. LNCS 3001, pp. 1-17, 2004.

[7] B. Logan, J. Healey, M. Philipose, E. Munguia-Tapia, S. Intille, "A long-term evaluation of sensing modalities for activity recognition," In Proceedings of Ubicomp 2007. Innsbruck, Austria, September 2007.

[8] C. Lombriser, N. B. Bharatula, D. Roggen, and G. Tröster, "On-Body activity recognition in a dynamic sensor network," In 2nd Int. Conf. on Body Area Networks (BodyNets), 2007.

[9] Tâm Huynh, Ulf Blanke and Bernt Schiele, "Scalable recognition of daily activities from wearable sensors," 3rd International Symposium on Location and Context-Awareness (LoCA), Germany, September 2007.

[10] D. Patterson, D. Fox, H. Kautz, M. Philipose, "Fine-Grained activity recognition by aggregating abstract object usage," Proceedings of ISWC 2005, Osaka, October 2005.

[11] S. Wang, W. Pentney, A.-M. Popescu, T. Choudhury, M. Philipose, "Common sense based joint training of human activity recognizers," In Proceedings of IJCAI 2007. Hyderabad, January 2007.

[12] Jonathan Lester, Tanzeem Choudhury, and Gaetano Borriello, "A practical approach to recognizing physical activities," In Proceedings of International Conference on Pervasive Computing, 2006.

[13] Jamie A. Ward, Paul Lukowicz, Gerhard Tro¨ ster, and Thad E. Starner, "Activity recognition of assembly tasks using body-worn microphones and accelerometers," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 28, no. 10, October 2006.

[14] 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, 3(4): 50-57, October 2004.

[15] Daniel Wilson and Chris Atkeson, "Simultaneous tracking and activity recognition (STAR) using many anonymous, binary sensors," In proceedings of the third International Conference on Pervasive Computing, pp 62-79, Munich, Germany, 2005.

[16] Douglas L. Vail, Manuela M. Veloso, and John D. Lafferty, "Conditional random fields for activity recognition," International Conference on Autonomous Agents and Multi-agent Systems (AAMAS), 2007.

[17] Joseph Modayil, Tongxin Bai, and Henry Kautz, "Improving the recognition of interleaved activities," Research note, in Proc. of the Tenth International Conference on Ubiquitous Computing (UBICOMP08), Seoul, South Korea, September 2008.

[18] R. Hamid, S. Maddi, A. Johnson, A. Bobick, I. Essa, C. Isbell, "A novel sequence representation for unsupervised analysis of human activities," Artificial Intelligence Journal, 2008, in press.

[19] Fayyad, U. and Irani, K, "Multi-interval discretization of continuous-valued attributes for classification learning," In proceedings of 13th International Joint Conference on Artificial Intelligence. San Francisco, 1993.

[20] Jinyan Li, Guimei Liu and Limsoon Wong, "Mining statistically important equivalence classes and delta-discriminative emerging patterns," In the Proceedings of 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2007), pages 430-439, San Jose, 2007.

[21] G. Dong, X. Zhang, L. Wong, and J. Li. CAEP: Classification by Aggregating Emerging Patterns. In DS'99 (LNCS 1721), Japan, Dec. 1999.

[22] J. R. Quinlan. Induction of decision trees. In Machine Learning, Vol 1, pages 81-106, 1986.