Mining Emerging Sequential Patterns for Activity Recognition in Body Sensor Networks

Tao Gu[1], Liang Wang[1,2], Hanhua Chen[1,3],
Guimei Liu[4], Xianping Tao[2], and Jian Lu[2]

[1] University of Southern Denmark
{gu,wang,hhchen}@imada.sdu.dk
[2] Nanjing University
{txp,lj}@nju.edu.cn
[3] Huazhong University of Science and Technology
[4] National University of Singapore
liugm@comp.nus.edu.sg

Abstract. Body Sensor Networks offer many applications in healthcare, well-being and entertainment. One of the emerging applications is recognizing activities of daily living. In this paper, we introduce a novel knowledge pattern named Emerging Sequential Pattern (ESP)—a sequential pattern that discovers significant class differences—to recognize both simple (i.e., sequential) and complex (i.e., interleaved and concurrent) activities. Based on ESPs, we build our complex activity models directly upon the sequential model to recognize both activity types. We conduct comprehensive empirical studies to evaluate and compare our solution with the state-of-the-art solutions. The results demonstrate that our approach achieves an overall accuracy of 91.89%, outperforming the existing solutions.

Keywords: Body sensor networks, activity recognition, data mining.

1 Introduction

The recent advance of wireless sensing and the development of miniaturized sensors have led to the use of Body Sensor Networks (BSNs). A BSN consists of a number of sensor nodes, placed or implanted on a human body, which provide sensing and wireless communication capabilities. Such systems offer many promising applications in healthcare, assistive living, well-being, sports, and entertainment. One of the applications is recognizing activities of daily living. In such an application, user observations in the form of a continuous sensor data stream are collected from various sensor nodes and transmitted to a gateway device. Useful features will be first extracted from the sensor data to train an appropriate activity classifier, which can then be used to identify new observations.

P. Sénac, M. Ott, and A. Seneviratne (Eds.): MobiQuitous 2010, LNICST 73, pp. 102–113, 2012.

Recognizing activities using BSNs has attracted many research interests from academic researchers and industry participants. Most of the existing work focus on recognizing sequential activities (i.e., one activity after another in a timeline) in different settings [1–5]. However, the situations in real life are more complex since people often multitask when performing their daily activities. Such multitasking can 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. Little work has been done in addressing complex issues rise in recognizing sequential, interleaved and concurrent activities in a unified framework. Existing solutions, such as Interleaved Hidden Markov Model [6] and Factorial Conditional Random Field [7], rigidly model interleaved or concurrent activities using Markov chains which require proper training. However, in real life, there exists many different ways in which activities can be interleaved and performed concurrently, e.g., when performing an activity, one may start another activity any time interleavedly or concurrently. Hence, it may not be feasible, if not impossible, to construct the complete models for interleaved and concurrent activities through training.

In this paper, we investigate an efficient way to recognize both simple and complex activities. By analyzing the trace involving both simple and complex activities, we observe many unique feature sets for each sequential activity and there exists an inherent sequential order among these features. In addition, we observe that the sequential order of the feature sets will not be changed regardless of whether this activity is performed sequentially, interleavedly, or concurrently. Intuitively, such sequential orders correspond to the intermediate steps or sub-actions of an activity, e.g., in the *brushing teeth* activity, there exists a sequential order—*taking a toothbrush*, *squeezing toothpaste*, *brushing*, and *washing with water*. Discovering such sequential orders (complete or partial orders) provides an additional, useful discriminator to recognize the activities which are performed interleavedly or concurrently.

To address the aforementioned problem, we aim to eliminate the training process of interleaved and concurrent activities. To achieve this, we propose a novel knowledge pattern, named Emerging Sequential Pattern (ESP), to capture unique contrast *sequential* patterns among different activity classes, and build our activity models based on ESPs. While the sequential activity model requires a training process, our complex activity models are built directly upon the sequential model without training. In this way, our system has a great flexibility and applicability for real-life applications. We design and implement our activity models and recognition algorithms in our ESP-based Activity Recognizer (ESPar). Through comprehensive empirical and comparison studies, we demonstrate both the effectiveness and flexibility of our proposed system.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 describes ESP and the mining algorithm. We then present ESPar in Section 6. Section 7 reports our empirical studies, and finally, Section 8 concludes the paper.

2 Related Work

To classify activities in BSNs, probabilistic models are typically used due to the non-deterministic nature of human activity. Probabilistic models can be categorized into static and temporal classification. In the static model, features are first extracted from sensor readings, and then a static classifier is applied for classification. Typical static classifiers include naïve Bayes used in [1,2], decision tree used in [1, 9], and k-nearest neighbor used in [1, 2]. 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 activity labels. We name a few examples here: Hidden Markov Models (HMMs) used in [3, 4] and Conditional Random Fields (CRFs) used in [4, 6]. Recent work showed that Interleaved Hidden Markov Model [6] can be used to model interleaved activities and Factorial Conditional Random Field [7] can be used to model concurrent activities. However, they require 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 loss of accuracy. On the other hand, in real life, there exists a great variety of ways in which activities can be interleaved and performed concurrently by different individuals. As a consequence, it may not be possible to obtain the complete models for interleaved or concurrent activities through training. Hence, the applicability and flexibility of these solutions are limited.

The solution presented in this paper is fundamentally different than the existing work by introducing a new knowledge pattern—ESP. We use ESP to build the models for interleaved or concurrent activities directly on the sequential activity model without training. In this way, our system is more easy to train. ESPs extend our earlier approach of Emerging Pattern based model [8] by taking the sequential order of observations into account, and mining unique contrast, sequential patterns to achieve better classification accuracy.

3 Emerging Sequential Pattern

In this section, we introduce ESP and present an algorithm to efficiently mine ESPs. ESP is motivated by the concept of Emerging Pattern [11] which is a kind of contrast patterns among unordered features/items. ESP greatly extends Emerging Pattern by considering the inherent sequential information among sequences and discovering the unique contrast sequential patterns.

3.1 Definitions

Let $I = \{i_1, i_2, \cdots, i_n\}$ be a set of items and $\mathcal{C} = \{C_1, C_2, \cdots, C_k\}$ be a set of class labels. An itemset is a subset of I. A sequence is an ordered list of non-empty itemsets, denoted by $s = \langle e_1, e_2, \cdots, e_m \rangle$, where e_i is a non-empty itemset

and it is also called an element of s. Each sequence has a class label $C_i \in \mathcal{C}$. We use $label(s)$ to denote the class label of a sequence s. In the context of activity recognition, items are feature items, elements are feature vectors, sequences are segments of traces within a continuous period of time that belong to the same activities, and the class labels of the sequences are the activities they belong to.

Given two sequences $s_1 = \langle a_1, a_2, \cdots, a_n \rangle$ and $s_2 = \langle b_1, b_2, \cdots, b_m \rangle$, if there exists integers $1 \leq i_1 < i_2 < \cdots < i_n \leq m$ such that $a_1 \subseteq b_{i_1}$, $a_2 \subseteq b_{i_2}$, $\cdots$, $a_n \subseteq b_{i_n}$, then we say s_1 is a subsequence of s_2, and s_2 is a super-sequence of s_1, denoted as $s_1 \prec s_2$. If $s_1 \prec s_2$, we also say s_2 contains s_1.

Definition 1. *(**Support of a Sequential Pattern**) Given a sequence database D, the support of a sequence s in D is defined as the number of sequences in D that contain s, denoted as $sup_D(s) = |\{s_i | s \prec s_i, s_i \in D\}|$.*

Note that the support of a pattern is defined as the number of distinct sequences containing it instead of the total number of occurrences. The support of s in D with respect to a particular class C is defined as the number of sequences in D with label C that contain s, that is, $sup_D(s, C) = |\{s_i | s \prec s_i, s_i \in D, label(s_i) = C\}|$.

Definition 2. *(**Frequent Sequential Pattern**) Given a user-specified minimum support threshold min_sup, if $sup_D(s) \geq min\_sup$, then we say s is a frequent sequential pattern in D.*

We are interested in sequential patterns that can distinguish sequences of different classes. We expect such patterns to occur frequently in one class and rarely in other classes. The discriminative power of a sequential pattern s with respect to a class C is defined as follows:

$$dpower(s, C) = p(C|s) \cdot p(\overline{C}|\overline{s}) = \frac{sup_D(s, C)}{sup_D(s)} \cdot \frac{(|D| - sup_D(s)) - (sup_D(C) - sup_D(s, C))}{|D| - sup_D(s)}$$

where $p(C|s)$ is the probability of class C if s is present, and $p(\overline{C}|\overline{s})$ is the probability of other classes if s is absent. The range of $dpower(s, C)$ is [0,1]. The higher the *dpower* value is, the more discriminative the pattern is. In the ideal case, all sequences containing s belong to class C ($p(C|s)$=1), and all the sequences that do not contain s belong to classes other than C ($p(\overline{C}|\overline{s})$=1), and we have $dpower(s, C)$=1.

Definition 3. *(**Emerging Sequential Pattern**) Given a user-specified minimum support threshold min_sup and minimum discriminative power threshold ρ, if $sup(s) \geq min\_sup$ and $dpower(s, C) \geq \rho$, then we say s is an emerging sequential pattern with respect to class C.*

Our task here is to enumerate the ESPs with $sup \geq min\_sup$ and $dpower \geq \rho$ from training sequences, and then use them to recognize the activities of new sequences. To mine ESPs, we use the pattern growth approach and propose an algorithm named ESPMiner presented in the next section.

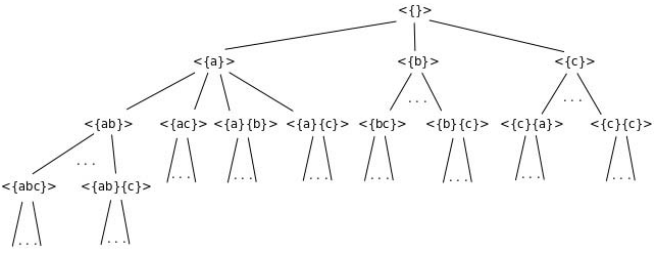

Fig. 1. Our mining framework

3.2 Mining ESPs

The search space of the sequential pattern mining problem can be represented as a prefix tree as shown in Fig. 1. In this tree, each node represents a sequential pattern, and it is a prefix of its children nodes. A sequential pattern $s = \langle e_1, e_2, \cdots, e_k \rangle$ can be extended in two ways: (1) element extension: one new item is added to the last element e_k; and (2) sequence extension: one new element containing only one item is added after e_k. For element extension, we sort the items in lexicographical order, and an item can be added to an element only if the item is larger than all the existing items in the element. For example, an element $e = \{b\}$ can be extended by item c but cannot be extended by item a. The purpose of this restriction is to avoid generating the same pattern more than once.

ESPMiner explores the search space in depth-first order, and it generates a sequential pattern from its prefix. The support of sequential patterns has the anti-monotone property. That is, the support of a sequential pattern is always no larger than that of its sub-sequences. ESPMiner uses this property to prune the search space. It starts from the empty pattern $\langle\rangle$ and recursively adds items to the pattern until the support of the pattern is below the minimum support threshold.

Algorithm 1 illustrates ESPMiner. When it is first called, $s{=}\langle\rangle$ and $D_s = D$. For each sequential pattern s, ESPMiner scans its projected database D_s to find its frequent element extensions and sequence extensions (line 1). An item x is a frequent element extension of $s = \langle e_1, e_2, \cdots, e_k \rangle$ if $s' = \langle e_1, e_2, \cdots, e_k \cup \{x\} \rangle$ is frequent. An item x is a frequent sequence extension of s if $s' = \langle e_1, e_2, \cdots, e_k, \{x\} \rangle$ is frequent. ESPMiner constructs the project databases of extensions of s from D_s (line 2). If a sequence in D_s contains $s' = \langle e_1, e_2, \cdots, e_k \cup \{x\} \rangle$ or $s' = \langle e_1, e_2, \cdots, e_k, \{x\} \rangle$), then the sequence is put into the projected database of s'. Next, ESPMiner outputs s' if $dpower(s', C) \geq \rho$ (lines 5-7, lines 12-14), and then extends s' recursively (line 8, line 15).

3.3 Mining ESPs for Activity Recognition

We then apply ESPMiner to mine the ESPs for activity recognition. Before the mining process, we first need to convert the data stream to a sequence of

Algorithm 1. The ESPMiner Algorithm

Input: $s = \langle e_1, e_2, \cdots, e_k \rangle$ is the frequent sequential pattern currently being processed;
D_s is the projected database of s;
min_sup is the minimum support threshold;
ρ is the minimum discriminative power threshold;
Output: Mined ESPs

Scan D_s to find frequent extensions of s, denoted as $F_{elm}(s)$ and $F_{seq}(s)$, where $F_{elm}(s)$ contains frequent element extensions of s, and $F_{seq}(s)$ contains frequent sequence extensions of s;
Scan D_s a second time to construct projected databases for super-sequences of s that extend s by one new item in $F_{elm}(s)$ or $F_{seq}(s)$;
for all items $x \in F_{elm}(s)$ **do**
    $s' = \langle e_1, e_2, \cdots, e_k \cup \{x\} \rangle$;
    if $dpower(s') \geq \rho$ **then**
        output s';
    end if
    DFSMine(s', $D_{s'}$, min_sup, ρ);
end for
for all items $x \in F_{seq}(s)$ **do**
    $s' = \langle e_1, e_2, \cdots, e_k, \{x\} \rangle$;
    if $dpower(s') \geq \rho$ **then**
        output s';
    end if
    DFSMine(s', $D_{s'}$, min_sup, ρ);
end for

observation vectors by concatenating all of the raw data in a fix time interval (set to one second in our experiments), and then we extract useful features as follows.

For acceleration data, we compute five common features—DC mean, variance, energy, frequency-domain entropy and correlation. The DC mean is the mean acceleration value in a time interval. Variance is used to characterize the stability of a signal. Energy captures the 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 between axes is especially useful for discriminating between activities that involve translation in just one dimension. It is computed for every two axes of each accelerometer and all pair-wise axes combinations of two different accelerometers. For RFID reading or location information, we use object name or location name as features. 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. *Feature vectors* are indexed by a simple encoding scheme and will be used as inputs to the mining process.

The mining process leverages on sequential activity instances only. Specifically, for each sequential activity class SA_i, we mine a set of ESPs 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. After mining, we obtain a set of ESPs for each sequential activity. Table 1 presents an ESP subset for *making oatmeal*. Columns 3 and 4 show the corresponding values of support and dpower, respectively.

Table 1. An ESP Example for *making oatmeal*

ESP	Support	dpower
$\{ACCEL\_BODY\_X\_MEAN@(-900.75 \sim -822.75]$, $ACCEL\_LEFT\_X\_MEAN@(214.39 \sim 352.87]$, $LOCATION@kitchen\}$; $\{ACCEL\_LEFT\_X\_MEAN@(543.27 \sim 843.19]$, $ACCEL\_LEFT\_Z\_MEAN@(441.53 \sim 847.59]\}$; $\{ACCEL\_RIGHT\_Z\_MEAN@(467.36 \sim 701.17]$, $OBJECT@tablespoon\}$; $\{LOCATION@kitchen\}$; $\{OBJECT@burner\}$	75.0%	1.0

4 ESP-Based Activity Recognition

4.1 ESPar Overview

We first give an overview of ESPar. The input is the sensor data stream which will be first pre-processed into a sequence of feature vectors. ESPar operates in two phases—model training and activity recognition. In the training phase, a training dataset consisting of sequential activity instances will be used to train our activity models. In the recognition phase, given a sequence of feature vectors (i.e., S_t, $t = 0 \sim T$), we first segment its sequence using a sliding window (i.e., L_{A_i}) to obtain a test instance (i.e., $S_{t \sim t+L_{A_i}}$, $t = 0 \sim T$), and then we apply our recognition algorithm to label this sequence segment. This process will be performed recursively, and each sequence segment will be assigned with a candidate label. For each pair of consecutive sequence segments we label, we apply a boundary detection algorithm to detect the boundary, and adjust the length of each sequence segment according to the new boundary. The above recognition processes will be performed recursively until the end of the sequence. In the following sections, we first describe our activity models. We then present the ESPar algorithm.

4.2 Sequential Activity Model

We design the activity model for each sequential activity SA_i based on a set of ESPs (i.e., ESP_{SA_i}) we mined from sequential activity instances. Each set of ESPs contains many subsets in which a single ESP can sharply differentiate the class membership of a fraction of the test instance $S_{t \sim t+L_{SA_i}}$ that contains the ESP. To make use of each subset of ESPs to achieve good overall accuracy, we combine the strength of each ESP based on the aggregation method described as follows.

Suppose an instance $S_{t \sim t+L_{SA_i}}$ contains an ESP, X, where $X \in ESP_{SA_i}$, then the odds that $S_{t \sim t+L_{SA_i}}$ belongs to SA_i is defined as $dpower(X)$. The differentiating power of a single ESP is then defined by the odds and the fraction of the population of class that contain the ESP. More specifically, the differentiating power of X is given by $dpower(X) * sup_{SA_i}(X)$. The aggregated probability of $S_{t \sim t+_{SA_i}}$ belongs to SA_i is defined as follows.

$$aggr\_p(SA_i, S_{t \sim t+_{SA_i}}) = \sum_{X \subseteq S_{t \sim t+_{SA_i}}, X \in ESP_{SA_i}} dpower(X) * sup_{SA_i}(X) \quad (1)$$

where $sup_{SA_i}(X)$ is the support of X in SA_i. The ESP measurement of each activity are then normalized by dividing them using the median probability value in the training instances of that activity. Finally, the ESP measurement is defined as follows.

$$esp(SA_i, S_{t\sim t+L_{SA_i}}) = \frac{aggr\_p(SA_i, S_{t\sim t+L_{SA_i}})}{base\_p(SA_i)} \quad (2)$$

where $base\_p(SA_i)$ is the median probability value of $aggr\_p(SA_i, S_{t\sim t+L_{SA_i}})$ in the training data.

In addition to the ESP measurement, we model activity correlation (i.e., when an activity SA_j has been performed, the probability of another activity SA_i being performed) in the design of our sequential activity model. We use condition probability to model correlations between activities. We define the activity correlation probability of SA_i as $P(SA_i|SA_j)$, which is the conditional probability of SA_i given SA_j. The activity correlation probability for each sequential activity can be easily obtained from the training dataset.

4.3 Interleaved and Concurrent Activity Models

The design of complex activity models is crucial. A common practice is to train both interleaved and concurrent activity models from complex activity instances. We design our complex activity model based on our sequential model only, eliminating the need for training. To illustrate, we denote CA_i as both interleaved and concurrent activities. We denote CA_i as SA_a & SA_b) for an interleaved activity and $SA_a + SA_b$ for a concurrent activity, where two single sequential activities SA_a and SA_b are involved in. 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 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 ESP_{SA_a} and ESP_{SA_b} (i.e., some of the steps that belong to SA_a and SA_b respectively are interleaved or overlapped), the ESP measurement of CA_i can be computed as follows.

$$esp(CA_i, S_{t\sim t+L_{CA_i}}) = max[esp(SA_a, S_{t\sim t+L_{CA_i}}), esp(SA_b, S_{t\sim t+L_{CA_i}})] \quad (3)$$

The computation of activity correlation probability for interleaved and concurrent activities can be quite complex. There are three cases: 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 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 probability of CA_i, where $CA_j = SA_c$ & SA_d or $CA_j = SA_c + SA_d$, can be computed as follows.

$$P(CA_i|CA_j) = max(P(SA_a|SA_c), P(SA_a|SA_d), P(SA_b|SA_c), P(SA_b|SA_d)) \tag{4}$$

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

In summary, our activity model for sequential, interleaved and concurrent activities is defined as follows.

Definition 4. ***(Activity Model)*** *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 activity model of A_i is then defined as follows:*

$$model(A_i, A_j, S_{t\sim t+L_{A_i}}) = esp(A_i, S_{t\sim t+L_{A_i}}) * P(A_i|A_j)$$

4.4 The ESPar Algorithm

We are now ready to classify activities based on the above model. We first use a sliding window based algorithm to segment the sequence. For each possible activity, we obtain the test *feature vectors* using its corresponding sliding window (the length is the average duration of this activity), and compute the probability based on Definition 4. We then assign the activity label with the highest probability to the *feature vectors*. The process can be applied recursively to the entire sequence. However, since the sliding-window length of each activity is an approximation of its actual length, the segmentation may not be accurate. To overcome this drawback, we use a boundary detection algorithm in [8] to detect and adjust the boundary between two adjacent activities so that the next sliding window can be applied from the correct boundary.

5 Empirical Studies

We now move to evaluate our proposed algorithm. In this section, we first describe the trace we use, and then present and discuss the results obtained from a series of experiments.

5.1 Trace and Methodology

We use a real-wrold trace collected in [8]. The dataset consists of 26 sequential activities, 11 interleaved activities and 13 concurrent activities with a total number of 532 instances. We use *ten-fold cross-validation* (a common technique to evaluate predictive models) for our empirical studies. Note that the training process involves sequential instances only.

We evaluate the performance of ESPar using time-slice accuracy which is a typical technique in time series analysis. It 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 the accuracy much. This time slice duration is short enough to provide precise measurements for applications. The metric of the time-slice accuracy is defined as follows.

$$Time\_slice = \frac{1}{N}\sum_{n=1}^{N}[inferred(n) = true(n)] \tag{5}$$

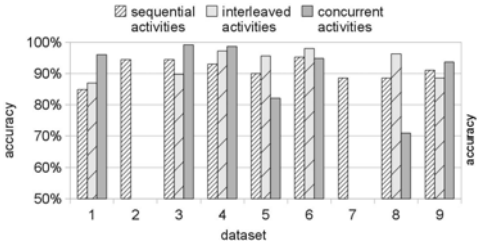

Fig. 2. Breakdown for type of activities

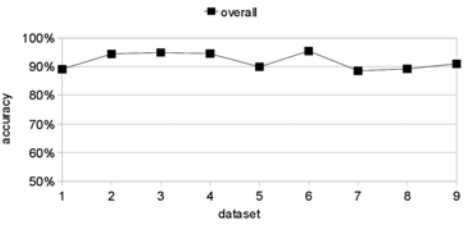

Fig. 3. Breakdown for overall accuracy

where $[inferred(n) = true(n)]$ produces 1 when true and 0 when false; $N = \frac{T}{\Delta t}$, i.e., the total number of time slices.

5.2 Accuracy

In this experiment, we evaluate the accuracy of ESPar. ESPar achieves an accuracy of 91.61% for sequential activity, 92.26% for interleaved activity, 92.32% for concurrent activity, respectively, and an overall accuracy of 91.89%. Figure 3 shows the breakdown for the overall accuracy, and Fig. 2 shows the breakdown for type of activities. One observation is that we obtain similar results for all the three types of activities. Most confusion takes place in the following three cases:

Case 1: A sequential activity is predicted as another sequential activity, e.g., for a sequential activity *washing face*, our result shows while 51.2% of its entire observation sequence is predicted correctly, 16.3% of them is predicted as *brushing teeth* and 14.1% of them is predicted as *brushing hair*.

Case 2: In a complex activity, only one of the sequential activities is detected and the other one is missed, e.g., for an interleaved activity *reading book/magazine & using phone*, while 76.1% of its entire observation sequence is predicted correctly, 19.1% of them is predicted as *reading book/magazine* (i.e., *using phone* is missed).

Case 3: A sequential activity is predicted as an interleaved or a concurrent activity, e.g., while 75.6% of the entire sequence of *reading book/magazine* is predicted correctly, 23.9% of them is predicted as an interleaved activity *reading book/magazine & using phone*.

To analyze the above results, we suggest a number of reasons for Case 1. Firstly, we observe many similarities among these activities such as similar hand movements and the same location. Secondly, RFID sensors may not work in some activities such as *washing face* since *face towel* is not tagged, or fail to detect tagged objects due to out of range. Thirdly, many tagged objects may be placed in close proximity. As a result, sensor noise may be introduced to the data stream. For Cases 2 and 3, the above possibilities may hold as well. In addition, for the two activities involved in a complex activity, ESPar seems to bias to the longer one. Possible solutions include adding more sensor modalities, which we leave for our future work.

Table 2. Comparison Results

Classifier	Accuracy				Remark
	Sequential Activity	Interleaved Activity	Concurrent Activity	Overall	
ESPar	91.61%	92.26%	92.32%	91.89%	pattern based model
HMM	70.99%	N.A.	N.A.	70.99%	temporal probabilistic model
CRF	86.93%	N.A.	N.A.	86.93%	temporal probabilistic model

5.3 Comparison Studies

In this experiment, we compare ESPar with temporal models (HMMs and CRFs). The results of the temporal models are based on sequential activities only since both models can not be directly applied to recognize complex activities. For a fair comparison, we use the same training and testing datasets for all the models. Table 2 summarizes the results.

For each dataset, an HMM was trained and the Viterbi algorithm was used to recover the state sequence; similarly for CRF. The result is shown in Table 2. ESPar outperforms both CRF (86.93%) and HMM (70.99%). This result can be probably explained as follows: First, the amount of training data is relatively small. In such circumstance, mining the differences between classes tends to be more effective for building a discriminative model. Although CRF is also a discriminative model, it focuses on mining the regularities in which a large amount of training data is typically required. This result reveals that ESPar 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 shortcomings such as over-fitting the training data and strong independence assumptions. ESPar also outperforms the Emerging Pattern based model [8], demonstrating that the ESP-based model is more efficient because it mines not only the differences between classes, but also the inherent ordering among activity sequences.

6 Conclusions

In this paper, we study the problem of recognizing sequential, interleaved and concurrent activities using a BSN. We propose a novel Emerging Sequential Pattern, and demonstrate that, leveraging on ESP, both simple and complex activities can be effectively recognized in a unified framework.

For our future work, we will integrate more sensors such as acoustic sensor and gyro sensor into our BSN. The more sensor modalities we use, and the stronger ESPs we obtain. We will also explore the concept of ESPs to other classification tasks.

Acknowledgement. This work was supported by the Danish Council for Independent Research, Natural Science under Grant 09-073281, National 973 program of China under Grant 2009CB320702, National 863 program of China under Grant 2009AA01Z117, Natural Science Foundation of China under Grants

60736015 and 60721002, Project for Core High Technology of the Ministry of Science and Technology of China under Grant 2009ZX01043-001-06, and Jiangsu Climbing Program under Grant BK2008017.

References

1. Bao, L., Intille, S.S.: Activity recognition from user-annotated acceleration data. In: Ferscha, A., Mattern, F. (eds.) PERVASIVE 2004. LNCS, vol. 3001, pp. 1–17. Springer, Heidelberg (2004)
2. Logan, B., Healey, J., Philipose, M., Tapia, E.M., Intille, S.: A long-term evaluation of sensing modalities for activity recognition. In: Krumm, J., Abowd, G.D., Seneviratne, A., Strang, T. (eds.) UbiComp 2007. LNCS, vol. 4717, pp. 483–500. Springer, Heidelberg (2007)
3. Patterson, D., Fox, D., Kautz, H., Philipose, M.: Fine-grained activity recognition by aggregating abstract object usage. In: Proc. IEEE Int'l Symp. Wearable Computers, Osaka (October 2005)
4. Vail, D.L., Veloso, M.M., Lafferty, J.D.: Conditional random fields for activity recognition. In: Proc. Int'l Conf. Autonomous Agents and Multi-agent Systems, AAMAS (2007)
5. van Kasteren, T.L.M., Noulas, A.K., Englebienne, G., Kröse, B.J.A.: Accurate activity recognition in a home setting. In: Proc. Int'l Conf. Ubicomp, Seoul, Korea (September 2008)
6. Modayil, J., Bai, T.X., Kautz, H.: Improving the recognition of interleaved activities. In: Proc. Int'l Conf. Ubicomp, Seoul, South Korea (September 2008)
7. Wu, T.Y., Lian, C.C., Hsu, J.Y.: Joint recognition of multiple concurrent activities using factorial conditional random fields. In: Proc. AAAI Workshop Plan, Activity, and Intent Recognition, California (July 2007)
8. Gu, T., Wu, Z., Tao, X., Pung, H.K., Lu, J.: epSICAR: An Emerging Patterns based Approach to Sequential, Interleaved and Concurrent Activity Recognition. In: Proc. IEEE Int'l Conf. on Pervasive Computing and Communications (Percom 2009), Galveston, Texas (March 2009)
9. Lombriser, C., Bharatula, N.B., Roggen, D., Tröster, G.: On-body activity recognition in a dynamic sensor network. In: Proc. Int'l Conf. Body Area Networks, BodyNets (2007)
10. Fayyad, U., Irani, K.: Multi-interval discretization of continuous-valued attributes for classification learning. In: Proc. Int'l Joint Conf. on Artificial Intelligence, San Francisco (1993)
11. Dong, G.Z., Li, J.Y.: Efficient mining of emerging patterns: discovering trends and differences. In: Proc. ACM Int'l Conf. on Knowledge Discovery and Data Mining, San Diego, CA, USA, pp. 43–52 (August 1999)