Pervasive and Mobile Computing 38 (2017) 92–109

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

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

Information gain-based metric for recognizing transitions in human activities

Amin Sadri, Yongli Ren, Flora D. Salim *

School of Computer Science and Information Technology, RMIT University, Melbourne, Australia

ARTICLE INFO

Article history:
Received 20 April 2016
Received in revised form 23 November 2016
Accepted 4 January 2017
Available online 10 January 2017

Keywords:
Human activity recognition
Temporal segmentation
Information gain
Routine discovery

ABSTRACT

This paper aims to observe and recognize transition times, when human activities change. No generic method has been proposed for extracting transition times at different levels of activity granularity. Existing work in human behavior analysis and activity recognition has mainly used predefined sliding windows or fixed segments, either at low-level, such as standing or walking, or high-level, such as dining or commuting to work. We present an Information Gain-based Temporal Segmentation method (IGTS), an unsupervised segmentation technique, to find the transition times in human activities and daily routines, from heterogeneous sensor data. The proposed IGTS method is applicable for low-level activities, where each segment captures a single activity, such as walking, that is going to be recognized or predicted, and also for high-level activities. The heterogeneity of sensor data is dealt with a data transformation stage. The generic method has been thoroughly evaluated on a variety of labeled and unlabeled activity recognition and routine datasets from smartphones and device-free infrastructures. The experiment results demonstrate the robustness of the method, as all segments of low- and high-level activities can be captured from different datasets with minimum error and high computational efficiency.

1. Introduction

Understanding human behavior plays a key role in many context-aware applications such as targeted advertising and location-based services. Developing the science to analyze human activities and to describe how people move during their daily life promises to address a wide range of challenges. The work described in this paper is concerned with finding transition times when one changes his/her activity during a specific period of time. The activity could be a fine-grained activity (e.g., sitting, standing or running) or a coarse-grained activity, which is an aggregate of low-level activities and has a more complex semantic (e.g., shopping, working at the office or commuting). In this paper, we define fine-grained activities as low-level activities, and coarse-grained activities as high-level activities. A consequent key challenge is that these activities may be recorded in a variety of different data sources, e.g. sensor data, connectivity data, device-free data, and mobility data. To handle this, we present an information gain-based temporal segmentation method applicable to a wide range of pervasive data. Temporal segmentation approaches split time series into several homogeneous and non-overlapping intervals (segments). The output of applying temporal segmentation to the human activity data is a set of transition times (See Fig. 1).

Studying the transition times of human daily activities is important not only for understanding the mobility patterns but also for providing context-aware services in pervasive systems [1]. For example, a user may want to get the news

* Corresponding author.
E-mail addresses: amin.sadri@rmit.edu.au (A. Sadri), yongli.ren@rmit.edu.au (Y. Ren), flora.salim@rmit.edu.au (F.D. Salim).

http://dx.doi.org/10.1016/j.pmcj.2017.01.003
1574-1192/

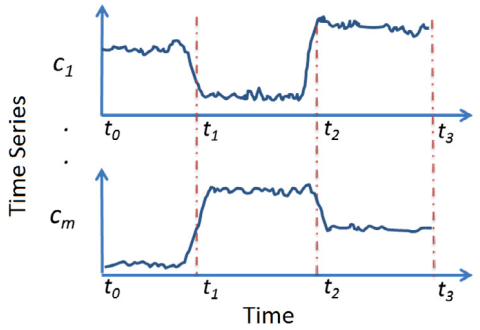

Fig. 1. Temporal Segmentation with m time series $[c_1 \ldots c_m]$ and 3 segments. t_1 and t_2 are the transition times and $[t_0 - t_1]$, $[t_1 - t_2]$ and $[t_2 - t_3]$ are the segments.

whenever he arrives home or before commuting to work. Furthermore, extracting transition times from daily activities is very useful in urban computing because it shows when people usually change their locations during a day. City planners use this information to improve transportation, urban planning, and environment [2].

Temporal segmentation is the key to processing low-level and high-level human activities. Regarding low-level activities, many techniques have been applied for activity recognition using various kinds of sensor data [3–6]. However, there is a need to segment the data prior to recognition or prediction tasks as the transition times between the activities are unknown. Regarding high-level activities (e.g. office work or dining), temporal segmentation helps us to understand humans' mobility patterns across the whole day. For example, assume one leaves home at 7 am, works at the office from 9 am to 5 pm, and goes outdoor until 12 am every day. With this interpretation, his daily routine can be expressed as home (7 am) → commuting (9 am) → office (5 pm) → outdoor (12 am) → home. [12 am–7 am], [7 am–9 am], [9 am–5 pm] and [5 pm–12 am] are the temporal segments in this example that convey semantic meaning about the routine. The temporal segmentation method proposed in this paper extracts transition times not only in low-level activities but also in high-level activities, namely at the routine level.

To deal with human behavior data, a generic and fast temporal segmentation method that is applicable to heterogeneous data is proposed in this paper. As the input data is not just a single time series, a multivariate method is required. Furthermore, the time series may come from heterogeneous channels with different ranges and different types (i.g., accelerometer, gyroscope, and thermometer). Our method is generic because it can handle multiple time series regardless of their heterogeneity and the varying correlation between multiple sensor channels (e.g. whether the data are positively or negatively correlated). A segmentation method ideally needs to be unsupervised opposed to supervised learning which requires data to be labeled. The computational cost of the algorithm should also be taken into account [7]. Methods analyzing high-level activity have been usually applied to small data including only a few tens of users [8,9]. However, in many real-world applications, data comes from large sources like smartphones in a city that requires fast approaches. Our approach is enough fast to process data from hundreds of users in few hours.

To introduce our approach, Information Gain-based Temporal Segmentation (IGTS), let us consider smartphone connectivity data. Connectivity data specifies that the smartphone is connected to which access point or cell tower at each time. In this case, the results illustrate when users usually change their locations during a day. We treat the mean values of the time series (i.e. connectivity to the Cell Tower IDs) in each segment as a random variable and thus each segment has an entropy. Based on information theory, entropy reflects uncertainty. Correspondingly, in our case, entropy refers to the predictability of user locations. The IGTS algorithm finds low-entropy segments that means the users' locations are 'predictable'. As information gain gives the expected reduction in entropy after the segmentation, it is used as an effective cost function for finding coherent segments. As a result, the best segmentation is the one with highest information gain value.

To find the best segmentation with the highest information gain, an optimization method should be deployed. We propose two approaches for optimization: TopDown and Dynamic Programming (DP). TopDown optimization is faster while it cannot guarantee to provide the global optima. On the other hand, DP optimization results in the global optima. In order to apply DP to our approach, the cost function is modified otherwise DP and the cost function are not compatible. Both TopDown and dynamic programming approaches are useful for specific applications due to the trade-off between the accuracy and running time.

The main contributions of this paper are as follows:

- We introduce information gain as the cost function for temporal segmentation problem. We argue that information gain, which gives the expected reduction in entropy after the segmentation, is an effective cost function to find coherent segments.
- A data transformation stage has been introduced to handle the data heterogeneity challenge for all information gain-based temporal segmentation problems in both low- and high-level activities.

- A heuristic method is proposed to find the number of segments, k. Proving that the information gain trend is monotonic, we propose a local knee point detection algorithm that indicates the best candidate for the number of segments.
- The proposed method is examined on several real-world datasets including activity recognition, movement detection, daily activity, and connectivity datasets.

In the next section, related work in both temporal segmentation and human mobility mining are discussed. We formally state the problem in Section 3. In Section 4, we present IGTS as a new temporal segmentation method that can be used for human activity data. The experiment results of our algorithm are reported in Section 5. Finally, Section 6 summarizes our work and presents possible applications for future works.

2. Related work

In this section, we briefly review the related work on temporal segmentation, the estimation of the number of segments, and transitions times in human activities.

2.1. Temporal segmentation

Time series segmentation approaches can be divided into three main groups: dynamic programming, heuristic, and probabilistic approaches [10]. Dynamic Programming (DP) research dates back to 1950s and it has been used in many different contexts such as waveforms [11,12], DNA sequence [13], and piecewise linear segmentation [14]. The main idea of dynamic programming is that a complex problem is divided into small problems that are solved first. The results of the subproblems are saved into a table of results and used to solve the main problem. In temporal segmentation problem, dynamic programming is used as an optimization method in conjunction with a cost function. The basic dynamic programming method for segmentation is called k-segmentation that minimizes the variance inside the segments [15,16]. Kehagias et al. used DP algorithm for minimization of Hubert's segmentation cost to segment hydrological and environmental time series [17]. Guo et al. deploy dynamic programming and Schwarz's Bayesian information criterion for segmentation [18]. However, not all types of the cost functions are compatible with dynamic programming [19]. The total complexity of k-segmentation is $O(kn^3)$ that consists of the complexity of the optimization approach (i.e. DP) and the cost function (i.e. variance). However, using some reprocessing steps, it is improved to $O(kn^2)$ [19]. Nevertheless, it is still impossible to use k-segmentation for large datasets.

Heuristic approaches can be divided into three groups: sliding window, TopDown, and BottomUp approaches [20,21]. In sliding window approaches, a window slides over the time series and a new segment is started when a specified error criterion is met. Online change-point detection approaches are similar to sliding window but the window size is not fixed. These methods only consider local boundaries and thus are not able to provide a global model. Specifically, the transition times are chosen based on the whole time series not just a window of the time series. Although online methods do not need to estimate k, other parameters such as thresholds should be set that indirectly affects k [22]. If the thresholds for the change is set too low, the number of segments becomes too many. On the other hand, a high threshold results in a small number of segments. TopDown approaches start with one segment. Then, the time series is recursively partitioned until a specified error criterion is met in each step [23,24]. Cheng et al. also apply TopDown algorithm as an optimization method and use mutual information as a cost function for co-clustering on a co-occurrence matrix. They consider the table as a joint probability distribution of two discrete random variables for clustering in text analysis domain [23]. In contrast to TopDown approaches, BottomUp approaches start with the maximum number of segments and the segments are merged until a specified error criterion is met.

Probabilistic-based segmentation algorithms consider the distribution of the data and transition times. For example, Bayesian techniques assume that the data originates from a Bayesian distribution and find the number and location of segments [25]. Another example of probabilistic-based approaches is Hidden Markov Models (HMM) that assigns each segment to a state in the HMM. A change-point is alarmed when switching from one state to another [26]. Kawahara et al. deploy the ratio of probability densities instead of the probability densities themselves [27]. Probabilistic methods can lead to acceptable results if the predefined model is correct. Matteson et al. do not make any assumptions about the distribution[28]. The non-parametric estimation of the number of change-points is based on the divisive hierarchical algorithm included in the *ecp* package in the R statistical software. This uses divergence measure to detect any distribution change within an independent sequence (see Section 2.1 in [28]). They deploy hierarchical significance testing to determine the statistical significance of change-points used as a stopping criterion for the iterative estimation procedure. Generally, Probabilistic-based methods require high computational time [10].

There have also been proposed other methods in addition to the three aforementioned main categories. For example, Keogh et al. use a hybrid approach called SWAB (Sliding Window And BottomUp) to combine the advantages of Sliding Window and BottomUp algorithms [29]. Some approaches combine gradient decent and TopDown algorithm to escape local optima [15]. The evolutionary method proposed by Chung et al. searches for a set of pattern templates which is determined by the user [30]. Yu et al. apply a cost function that depends on a covariance matrix using a low-complexity Pruned Exact Linear Time (PELT) method [31].

Similar to our method, E-Clustering introduced in [24] applies the concept of entropy and information gain for road traffic segmentation across a day to find when the traffic changes. However, our proposed method is different in several aspects.

First, the way we use entropy is different because we calculate the entropy considering the distribution of the average of the time series in each segment. On the other hand, in [24], the entropy is calculated considering the distribution of members of the clusters extracted from the previous stage (i.e. V-Clustering). Second, E-clustering is not applicable to continuous time series. Third, E-clustering is not able to handle positively correlated data. Specifically, if the clusters' members reduce by a ratio at the same time, the entropy does not change and thus it is not possible to use the entropy calculation proposed by E-Clustering for the segmentation of heterogeneous activity data. On the other hand, our method is modified to handle both negatively and positively correlated time series.

2.2. k estimation

Estimating the number of segments/clusters, k, is also a challenging problem in segmentation/clustering. Typically, five common approaches are: cross-validation [32,33], penalized likelihood estimation, permutation tests [34], re-sampling [35], and knee point detection in evaluation metric graph [36]. Probabilistic approaches use penalized likelihood estimation such as Bayesian Information Criterion (BIC) [37], Akaike Information Criterion (AIC), Minimum Message Length (MML) [38] and Minimum Description Length (MDL) [39]. When there is an evaluation metric for the number of segments, the knee point in "number of clusters vs. evaluation metric" graph reveals k. In our case, information gain of each split performs as the evaluation metric. Therefore, the knee point reveals the best k because with the larger k there is no sharp increase in information gain. Various methods are used to find the knee point of the graph such as the largest magnitude difference between two points, the first data point with the second derivative above some threshold value [40], and the data point with the largest second derivative [41]. While these methods consider local trends in the graph, other methods such as L-method [36] try to find a point on the curve that is farthest from lines fitted to the entire curve.

2.3. Transition times in human behavior

Finding the best segmentation is a key step before exploring the user activities. Regarding low-level activity recognition, activity transition times should be first identified from sensor data streams. Some works take advantage of user feedback for identification of initial and end points of the activities which is very obtrusive [42–44]. Yao et al. use slope variation-based segmentation before device-free posture recognition and show that the overall accuracy is improved with proper segmentation [45]. Sekine et al. identify changes between activities through the analysis of variations in the frequency characteristics [46]. Yoshizawa et al. proposed a heuristic method that differentiates between static and dynamic activities to improve the transition time detection [47].

The sliding window or windowing approach is the most common segmentation technique in segmentation of low-level activities. In this approach, the time series are split into windows of a fixed size from 0.1 s [48] to 12.8 s [49] or more [50]. Some studies allow a degree of overlap between windows [51]. Banos et al. evaluate the impact of the window size on Human activity recognition [22]. Most of the state-of-the-art methods usually focus on a specific data and use specific characteristics of that data. For example, methods proposed in [52,53] are designed for motion data where the data is highly correlated.

Compared to low-level activities, little work has focused on finding transition times in high-level activities. Bao et al. attempt to find transition times in the user's daily routine before identifying the status of the user. They use the variation of the homogeneous connectivity data to find the best segmentation among just 11 predefined segmentations [54]. Some studies skip this step and apply fixed-size time slots. For example, Farrahi et al. use a fixed segmentation for all users (i.e. 12–7 am, 7–9 am, 9–11 am, 11 am–2 pm, 2–5 pm, 5–7 pm, 7–9 pm, and 9 pm–12 am). However, this segmentation may not be the optimal choice and the best segmentation could be different from one user to another [55,9]. Wang et al. apply a simple fixed segmentation (i.e. 12 am–9 am, 9–6 pm, 6 pm–12 am) to analyze students lives [56]. Similarly, in [57], equal-length segments are used for prediction with Hidden Markov Model. With no doubt, the performance of the prediction approach is improved providing that the best segmentation is employed. Although topic models such as Latent Dirichlet Allocation (LDA) were initially designed for text mining, they act effectively in extracting human behavior patterns [9,58]. Sun et al. proposed a nonparametric method containing two costly stages [8]. In the first stage, low-level activities are extracted using Dirichlet Process Gaussian Mixture Model (DPGMM). To use DPGMM model, they reduce the dimension of the feature vector from 12 to 6. Then, 6-dimensional Gaussian distribution is used to model the input data that requires calculating mean and covariance matrix. In the second stage, high-level representation is inferred using Hierarchical Dirichlet Process (HDP).

2.4. Summary of the gaps

Most of the existing temporal segmentation methods are only applied to a single time series. Moreover, the temporal segmentation method should be able to deal with heterogeneous data from different types of sensors. Existing information gain-based methods are not able to handle positively correlated time series. Other state-of-the-art methods usually focus on a specific data and use specific characteristics of that data. For example, [53] is designed for motion data where the data is highly correlated. Yoshizawa et al. find the transitions considering the previous activity was a static activity or moving activity [47]. As the number of the transition times in human activity is unknown, parametric methods are not applicable here. Non-parametric versions of LDA method are costly because they focus on the clustering in addition to the segmentation.

3. Problem definition

The heterogeneous time series in this paper are sequences $\mathbf{X}$, which consist of n observations over various m-dimensional heterogeneous channels, $\mathbf{X} = \langle \mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_n \rangle$, where $\mathbf{x}_i \in \mathbb{R}^m$ denotes the ith observation over m different channels. From the matrix perspective, $\mathbf{X}$ denotes an $m \times n$ matrix where x_{ji} denotes the ith observation over jth time series ($x_{ji} > 0$). In this matrix, column i is x_i and presents ith observation while row j is c_j and presents all observations over the jth time series.

The segmentation $\langle \mathbf{s}_0, \ldots, \mathbf{s}_k \rangle$ of the dataset $\mathbf{X}$ consists of k non-overlapping segments $\mathbf{s}_j$ by k transition times $t_0 < \cdots < t_k < t_{k+1}$, where $t_0 = 1$ and $t_{k+1} = n$. Each observation belongs to exactly one continuous segment $\mathbf{s}_j = \langle \mathbf{x}_{t_j}, \ldots, \mathbf{x}_{t_{j+1}-1} \rangle$ and $\mathbf{T} = \langle \mathbf{t}_0, \ldots, \mathbf{t}_k \rangle$ is the set of transition times. $\mathcal{L} : N^k \rightarrow R$ is the cost function and the problem is finding a set of transition times that maximizes the cost function.

From the application point of view, we deploy temporal segmentation to detect transition times in human activities. In other words, temporal segmentation algorithms help us to capture human activity from multivariate time series. We aim to deal with the following problem:

Problem. Given time series data from multiple heterogeneous channels, how to detect times when the user changes his activities (e.g. from "walking" to "running" or from "dining" to "commuting").

4. Temporal segmentation based on information gain (IGTS)

In this section, we propose the Information Gain-based Temporal Segmentation (IGTS) method. After providing background on the concept of entropy, we introduce a novel information gain-based cost function. Then, we propose two optimization strategies to find the best set of transition times, and finally, we discuss a heuristic algorithm to find the best k.

4.1. Background: high/low entropy variables

In information theory, Shannon entropy (H) measures the variance of a Probability Distribution Function (PDF) to show the information about the quantity of interests. When the variance of a PDF is high, little information can be inferred about the quantity of interests.

$$H = -\sum_{i=1}^{m} p(i) \log p(i) \tag{1}$$

where $p(i)$ denotes the probability of ith outcome and H denotes the entropy of the probability distribution.

Many researchers have deployed the concept of entropy to measure the predictability in human's behavior [59,60]. Eagle et al. use the entropy of the cell towers (BTS antennas) connectivity distribution to measure the amount of predictable structure in an individual's life [61]. The predictability of the user's life can be quantified using entropy which is typically measured in bits. People who have entropic lives tend to be more variable and harder to predict. In contrast, low-entropy lives are characterized by strong patterns across all time scales.

The connectivity of the smartphone to the cell towers of the phone network has a PDF and entropy. Fig. 2 shows access duration distributions for four intervals for a specific user. The user has a low-entropy access duration between 7 am and 5 pm because the smartphone can be connected to all of the CIDs with the same probability. Therefore, it is hard to predict to which cell tower ID (CID) the user's phone is connected. In contrast, the access distribution to the CIDs has high entropy during midnight because the user is likely to connect to the dominant CID in this time period.

The main idea of the proposed method is that we try to find the segments with the lowest entropy so the user locations are predictable within each segment. Assume that the input time series are the CID connectivity data. In this case, each segment has an access distribution (as shown in Fig. 2). After calculating information gain for each segmentation, we find the best one with the highest information gain value. In fact, information gain performs as a cost function in this method. For the optimization method, we present two strategies that have their own pros and cons. Afterwards, we propose a formula to identify the best candidate for the number of segments, k. Finally, we show that dealing with heterogeneous data, the positive correlation between the time series should be removed first.

4.2. Information gain-based cost function

Given a segmentation, we propose a cost function $\mathcal{L}$ by deploying the concept of Information Gain, which is the expected reduction in entropy caused by splitting the time series for a given segmentation [62]. Specifically, $\mathcal{L}$ is the expected reduction in the entropy caused by splitting $\mathbf{S}$ further, and is defined as follows:

$$\mathcal{L} = H(\mathbf{S}) - \sum_{i=0}^{k} \frac{|\mathbf{s}_i|}{|\mathbf{S}|} H(\mathbf{s}_i), \tag{2}$$

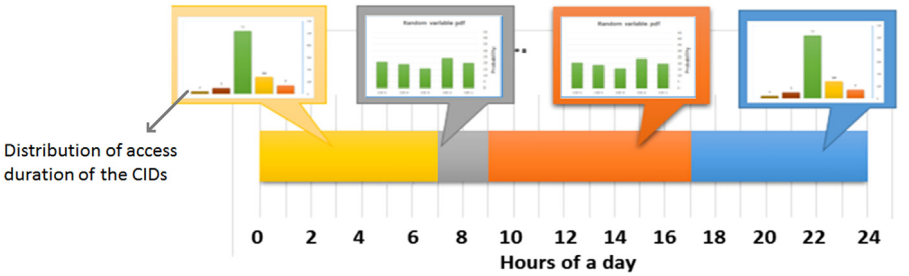

Fig. 2. Segmentation of a day based on the user connectivity data. In this segmentation, the time intervals are [12 am, 7 am], [7 am, 9 am], [9 am, 5 pm], and [5 pm, 12 am]. Between 7 am and 5 pm, the user's location is less predictable.

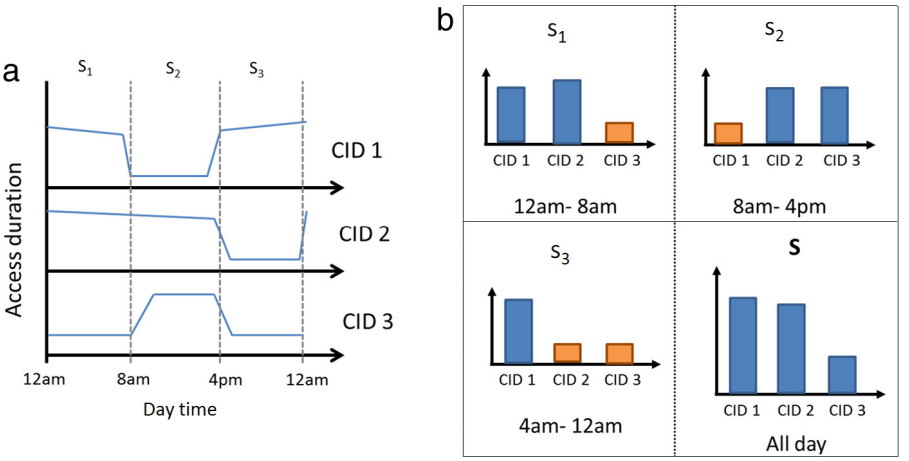

Fig. 3. (a) Average access duration to the CIDs for a sample user. (b) Each time interval has its own distribution and S presents the total distribution over a day.

where k is the number of segments and $|\mathbf{s}_i|$ is the length of the ith segment. $\mathbf{S}$ is the entire time series as a segment and $H(\mathbf{S})$ is the entropy of the whole time series. In our example, s_i is the CID access distribution over the ith segment (see Fig. 3).

To calculate the entropy of the segments, the CID access distributions are considered as random variables. Therefore, in Eq. (1), $p(i)$ is the probability that user's smartphone is connected to the ith CID. The following equation is used to calculate the entropy of the jth segment:

$$H(\mathbf{s}_j) = -\sum_{i=1}^{m} p_{ji} \log p_{ji}, \tag{3}$$

$$p_{ji} = \frac{\sum_{q=t_{j-1}}^{t_j} c_{i_q}}{\sum_{p=1}^{m} \sum_{q=t_{j-1}}^{t_j} c_{p_q}} \tag{4}$$

where m denotes the number of CIDs (or time series), p_{ij} is the probability that the smartphone is connected to the ith CID in the jth segment, and c_{i_q} denotes the qth observation of the ith channel (CID).

To speed up the algorithm, we take advantage of the cumulative sum (or integral function) of c_i. In this case, first, the cumulative sum of all the time series should be computed to be used in each entropy calculation.

$$p_{ji} = \frac{\sum_{q=t_{j-1}}^{t_j} c_{i_q}}{\sum_{p=1}^{m} \sum_{q=t_{j-1}}^{t_j} c_{p_q}} = \frac{F_i(t_j) - F_i(t_{j-1})}{\sum_{p=1}^{m} F_p(t_j) - F_p(t_{j-1})}, \tag{5}$$

$$F_i(t) = \sum_{j=1}^{t} c_{i_j}, \tag{6}$$

F_i is the cumulative sum of c_i. This modification makes the algorithm much faster because there is no need to sum up the observations for each entropy calculation. Specifically, the complexity of calculating cost function $\mathcal{L}$ is reduced from $O(mn)$ to $O(m)$.

4.3. Optimization method

In addition to defining a cost function, an optimization method is needed to find the best segmentation, which is NP-hard problem [63,64]. Considering the complexity of full search, it is clearly infeasible to be used for real-world data. We propose two optimization methods: TopDown and dynamic programming. TopDown optimization is faster while it cannot guarantee to provide the global optima. On the other hand, dynamic programming optimization is able to detect global optima but in a slower manner. As there is a trade-off between time and accuracy, either of the methods could be useful in different applications.

4.3.1. TopDown based optimization

A well-known greedy and fast optimization method is the TopDown or binary-split approach that runs in a hierarchical and greedy manner. This is initialized by treating all observations as one segment. Then, in each step, one transition time is added without making any changes in the transition times it has once set. The split that reduces the total cost most is chosen in each step. The pseudo code of the recursive implementation is shown in Algorithm 1. The IG-Seg (S, k) returns the transition times by adding one to the output of IG-Seg $(S, k-1)$. TopDown optimization is fast with the complexity of $O(nk)$. However, it may result in local optima.

FUNCTION: `IGTS-TopDown(S,k) /* TopDown optimization; recursive implementation */`
Input: S, k
$-S$:Set of m time series with the length of n
$-k$:Number of transition times
Output: T
$-T$:Set of transition times

if $k == 0$ **then**
  return ϕ; `/* terminating case of the recursive function */`
end
$T_{k-1} =$ `IGTS-TopDown(S,k-1)`;
$IG_{Max} = 0$; $t_k = 0$;
for $i = 1$ **to** n **do**
  if $IG(S, T_{k-1} \cup i) >= IG_{Max}$ **then**
    $IG_{Max} = IG(S, T_{k-1} \cup i)$; `/* IG(S,T) calculates the information gain of the segmentation */`
    $t_k = i$; `/* t_k is the kth transition time */`
  end
end
$T = T_{k-1} \cup t_k$;
return T ;

Algorithm 1: IGTS and TopDown optimization

4.3.2. Dynamic programming based optimization

Dynamic programming is a popular optimization strategy in the field of temporal segmentation [16,18]. The main idea of dynamic programming is to divide a complex problem into small problems that are solved first and their results are saved to solve the whole problem. However, information gain is not compatible with dynamic programming because it is not a separable cost function for each segment.

Here, we define a variant version of $\mathcal{L}$ to fit the dynamic programming framework:

$$Max(\mathcal{L}) = Max\left(H(\mathbf{S}) - \sum_{i=0}^{k} \frac{|\mathbf{s}_i|}{|\mathbf{S}|} H(\mathbf{s}_i)\right) = H(\mathbf{S}) - Min\left(\sum_{i=0}^{k} \frac{|\mathbf{s}_i|}{|\mathbf{S}|} H(\mathbf{s}_i)\right). \tag{7}$$

This means that instead of maximizing the information gain, we minimize $\sum_{i=0}^{k} \frac{|\mathbf{s}_i|}{|\mathbf{S}|} H(\mathbf{s}_i)$ called weighted entropy (W_H) because $H(\mathbf{S})$ is a constant and it is the same for all segmentation. Now, we can apply dynamic programming optimization to find a segmentation with minimum weighted entropy.

Dynamic programming-based IGTS is shown in Algorithm 2 and Fig. 4. $W_H(j, i, h)$ indicates minimum weighted entropy with h segments when the input time series is $\langle \mathbf{x}_j, \ldots, \mathbf{x}_i \rangle$. Clearly, we are looking for $W_H(1, n, k)$. In the first part of the algorithm, $W_H(j, i, 1)$ is calculated for all $1 \leqslant j < i \leqslant n$. In the second part, $W_H(1, i, h)$ is calculated recursively for all

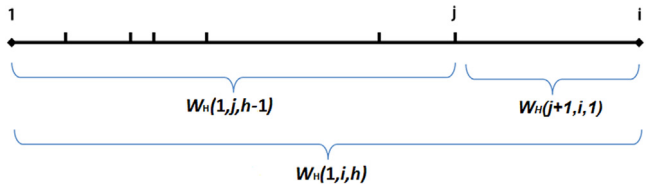

Fig. 4. To compute $W_H(1, i, h)$, the best possible point for the last change-point, j, is investigated. The best j is the one that maximizes $W_H(1, j, h-1) + W_H(j+1, i, 1)$.

$1 < i \leqslant n$ and $1 < h \leqslant k$. To find $W_H(1, i, h)$, all possible situations for the $(h-1)$th transition time are examined and the one that minimizes the weighted entropy is selected. The computational complexity of the optimization method based on dynamic programming is of order $O(kn^2)$.

FUNCTION: IGTS-DP(S,k)
Input: S, k
$-S$:Set of m time series with the length of n
$-k$: Number of transition times
Output: T
$-T$:Set of transition times

```
/* First part                                                              */
for i = 1 to n do
    for j = 1 to i do
        W_H(j, i, 1) = (|s_1| / |S|) H(s_1) = ((i-j)/n) H(s_1);
    end
end
/* Second part                                                             */
for h = 2 to k do
    for i = 1 to n do
        W_H(1, i, h) = W_H(1, i, h-1);
        for j = 1 to i-1 do
            if W_H(1, j, h-1) + W_H(j+1, i, 1) ⩽ W_H(1, i, h) then
                W_H(1, i, h) = W_H(1, j, h-1) + W_H(j+1, i, 1);
                P(i, h) = j;
                /* The position of the last transition time is stored in P(i, h).   */
            end
        end
    end
end
T = φ;
NextT = n;
for h=0 to k-2 do
    NextT = P(NextT, k-h);
    T = T ∪ NextT;
end
return T ;
```

Algorithm 2: IGTS and Dynamic Programming optimization

4.4. k estimation

Some methods ignore providing a framework for identifying k although k is usually unknown before segmentation in real-world applications. For example, in routine discovery problems, k could be different from one user to another or from one day to another and depends on the mobility pattern of the user. In this section, we discuss how to choose the best k among a given range.

Estimating the number of the clusters/segments is still an open problem in clustering/segmentation. In our case, we use information gain as an evaluation metric and the knee point in "number of segments vs. evaluation metric" graph reveals k. The knee point, which is the point of maximum curvature of the graph, reveals the best k because with the larger k there is no remarkable increase in information gain.

Depending on the application, various methods can be used to find the knee point of the graph such as the largest magnitude difference between two points, the first data point with the second derivative above some threshold value [40], and the data point with the largest second derivative [41]. These methods consider local trends in the graph. On the other hand, non-local methods are more complex and analyze the entire graph. For example, L-method [36] finds a point on the curve that is farthest from lines fitted to the entire curve. Local methods work well for smooth and monotonically increasing or decreasing curves. Here, we first prove that "number of segments vs. information gain" is a monotonic and non-decreasing graph. Then, we propose a local method to determine the optimal k.

Lemma. *Information gain is non-negative.*

Proof. Proof is based on Jensen's inequality [65] and details can be found in [66]. ■

Theorem. *Maximum $\mathcal{L}$ is a monotonic nondecreasing function of k.*

$$\mathcal{L}_k \leqslant \mathcal{L}_{k+1} \tag{8}$$

where $\mathcal{L}_k$ denotes the maximum information gain with k segments.

Proof. Let $T_k = (t_1, t_2, \ldots, t_k), 0 < t_1 < \cdots < t_k < n$ be the transition times of the segmentation with maximum information gain consisting of $k+1$ segments and $\mathcal{L}_k$ be the information gain of this segmentation. We prove that $T_{k+1} = (t_1, t_2, \ldots, t_k, t_{k+1}), t_k < t_{k+1} < n$ results in higher information gain than T_k. Based on T_{k+1} and T_k, we have

$$S_i^k = \begin{cases} S_i^{k+1}, & 1 \leq i \leq k \\ S_i^{k+1} \cup S_{i+1}^{k+1}, & i = k+1 \end{cases} \tag{9}$$

where S_i^k is the ith segment when the transition times are T_k. In fact, we add one transition time and split the last segment into two segments.

In this proof, we show that the segmentation caused by T_{k+1} has more information gain than segmentation caused by T_k. Consequently, the best segmentation with $k+1$ transition times has more information gain than the best segmentation with k transition times.

$$\begin{aligned} \mathcal{L}_{k+1} - \mathcal{L}_k &\geq \left(H(S) - \sum_{i=1}^{k+2} \frac{|S_i^{k+1}|}{|S|} H(S_i^{k+1})\right) - \left(H(S) - \sum_{i=1}^{k+1} \frac{|S_i^k|}{|S|} H(S_i^k)\right) \\ &= -\frac{|S_{k+1}^{k+1}|}{|S|} H(S_{k+1}^{k+1}) - \frac{|S_{k+2}^{k+1}|}{|S|} H(S_{k+2}^{k+1}) + \frac{|S_{k+1}^k|}{|S|} H(S_{k+1}^k) \\ &= \frac{|S_{k+1}^k|}{|S|} \left(H(S_{k+1}^k) - \frac{|S_{k+1}^{k+1}|}{|S_{k+1}^k|} H(S_{k+1}^{k+1}) - \frac{|S_{k+2}^{k+1}|}{|S_{k+1}^k|} H(S_{k+2}^{k+1})\right) \\ &= \frac{|S_{k+1}^k|}{|S|} IG_{tk} \geq 0 \end{aligned} \tag{10}$$

where IG_{tk} is the information gain caused by adding t_{k+1}. This inequality results from the lemma and information gain formula. ■

The above theorem clarifies that there is an increase in information gain with each split. Therefore, number of segments vs. information gain is a monotonic, non-decreasing and local knee point detection approaches can be applied to our problem. In our study, we found that none of the previous knee point detection methods gives a satisfactory result with our algorithm. We define the ratio ρ between the current and next increase in $\mathcal{L}$ to reveal the relationship between the number of segments and $\mathcal{L}$:

$$\rho_i = \frac{\mathcal{L}_i - \mathcal{L}_{i-1}}{\mathcal{L}_{i+1} - \mathcal{L}_i}. \tag{11}$$

The best k among the given range is the one that has the largest ρ because the deviation of the trend has a sharp decrease. Fig. 5(b) shows an experiment on Human Activity Dataset (HAD), illustrating effectiveness of ρ. The maximum value of ρ shows the best k is 5, which is equivalent to ground truth. In the experiments, we show that this formula works significantly better than other knee point detection formulas.

The proposed method can be applied to high-level activities and low-level activities. Therefore, to extract transition times without any information about k, we should provide a range in the search space to inform the model which one we are looking for. For example, if we want to find transitions in one day of data, we should specify that we are interested in transitions in high-level activities or transitions in low-level activities. We apply IGTS-TopDown to a day of data from Daily Life Routine Dataset that contains both high-level and low-level activity labels. The range of k is set between 1 and 150.

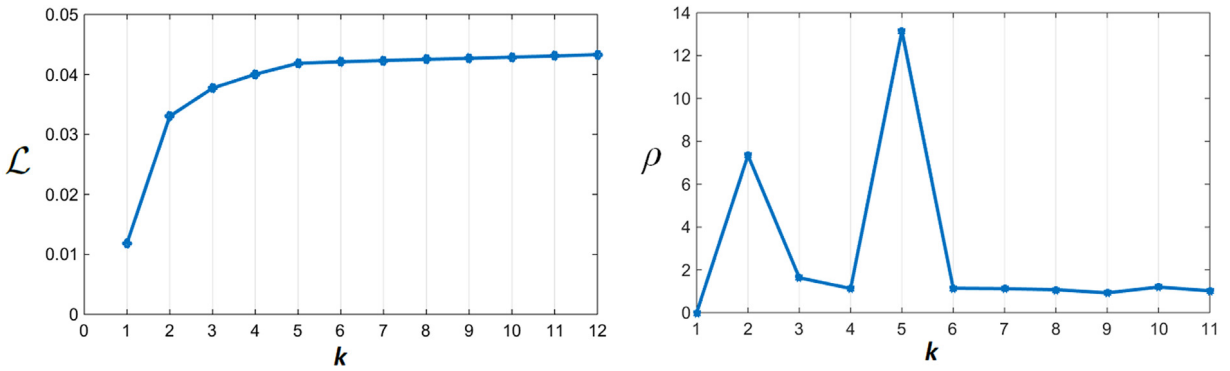

Fig. 5. (a) $\mathcal{L}$ from segmentation for different k. The actual k is 5 in this example. (b) The k with the maximum ρ is selected as the number of segments.

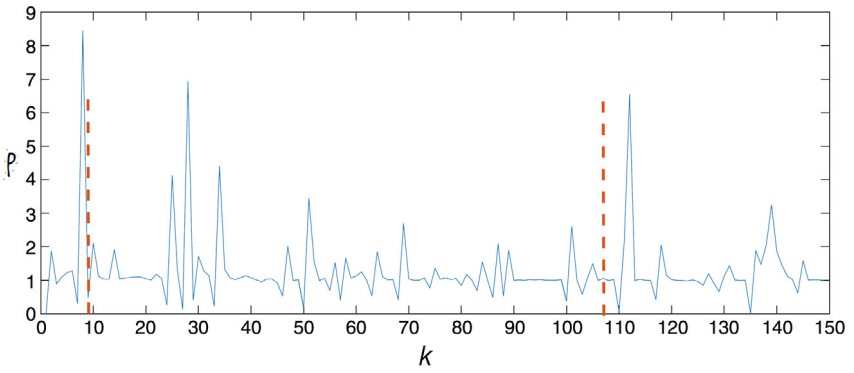

Fig. 6. There are two peaks around the two red lines that shows the number of high- and low- level activities. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Eq. (11) identifies the ρ. The high value of ρ shows a sharp increase in information gain that reveals the appropriate candidate for k. In this dataset, the number of high-level activities and low-level activities are 9 and 108 respectively. Fig. 6 shows the ρ for each k. The red lines are ground truth and knee point formula is calculated for each split. The proposed method for finding the number of segments works because there are two peaks around the actual number of segments for both high- and low- level activities.

4.5. Low-level activity and heterogeneous data

Here, we discuss how IGTS works with heterogeneous and positively correlated data sources, e.g. the time series from various sensor data. A positive correlation between two time series exists when by decreasing one of them, the other one also decreases and vice versa. The time series of CID access distributions do not have positive correlation because when the access to one CID increases in an interval, the access to the others decreases. On the other hand, the channels of sensor data, such as 3-axial linear acceleration and angular velocity, are highly correlated. Fig. 7(a) illustrates two time series with positive correlation, where the distribution is the same in $[t_0, t_1]$ and $[t_1, t_2]$. Simple information theory-based temporal segmentation methods are unable to detect the transition times in this case because the entropy is constant in all the segments. Thus, possible transition points could not be identified by using information gain as a cost function.

Assume X is an $m \times n$ matrix that presents the input time series where x_{ij} denotes the jth observation over ith time series (channel). c_i is the ith row and denotes all observations over the ith channel. To handle data heterogeneity problem, $\mathbf{X}$ is transformed as follows:

- Normalize $\mathbf{c}_i \in \mathbf{X}$ with $\sum_{j=1}^{n}(x_{ij} - \min_{\mathbf{c}_i})$:

$$\mathbf{c}_i \leftarrow \frac{\mathbf{c}_i - \min\limits_{\mathbf{c}_i}}{\sum\limits_{j=1}^{n}(x_{ij} - \min\limits_{\mathbf{c}_i})}, \tag{12}$$

where $\min_{\mathbf{c}_i}$ is the minimum of $\mathbf{c}_i$. This will remove the scale effects in heterogeneous data and give all the observations in different channels the equal priority. Without normalization, the algorithm relies only on the certain channels and leaves the other time series.

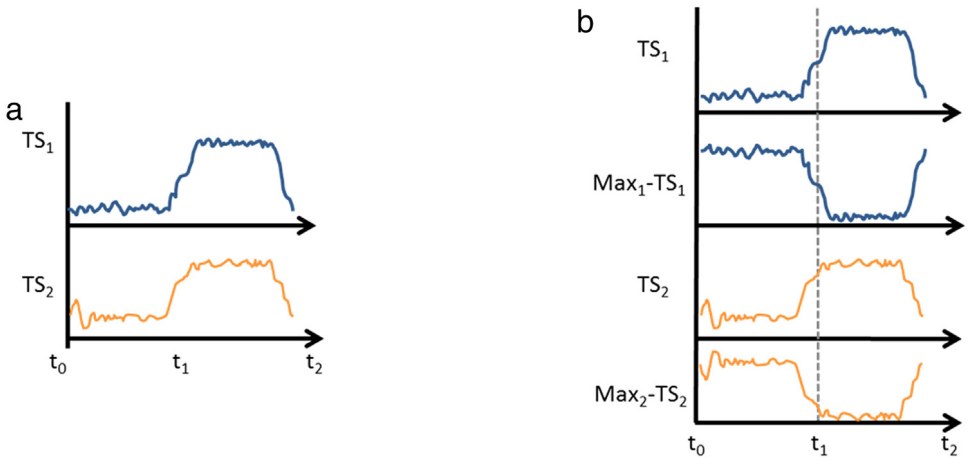

Fig. 7. (a) Two time series with positive correlation. Transition times cannot be detected by simply applying information gain as a cost function because $[t_0, t_2]$, $[t_0, t_1]$, and $[t_1, t_2]$ have the same distribution. (b) Two complement time series are added. Thus, the time series are not positively correlated any more.

- Remove positive correlation by adding the negative value to the maximum value of each time series. Eq. (13) shows the new matrix.

$$\begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & & \vdots \\ x_{m1} & \cdots & x_{mn} \\ \max\limits_{c1} - x_{11} & \cdots & \max\limits_{c1} - x_{1n} \\ \vdots & & \vdots \\ \max\limits_{cm} - x_{m1} & \cdots & \max\limits_{cm} - x_{mn} \end{bmatrix} \tag{13}$$

  where $\max_{\mathbf{c}_i}$ denotes the maximum of $\mathbf{c}_i$. After this stage, the number of rows is doubled and the input matrix becomes a $2m \times n$ matrix. The time series are not positively correlated anymore because when one time series falls, the corresponding negative one rises. Consequently, information theory metrics, e.g. information gain, can effectively distinguish the transition times.

Fig. 7(b) illustrates $\mathbf{X}$ transformed from time series in Fig. 7(a). Experiment results show that considering the negative time series not only makes the method more generic but also improves the accuracy. Furthermore, without this transformation, it is not possible to analyze a single time series because the entropy is always zero for one channel.

5. Experiments and evaluation

In this section, we examine the proposed IGTS method on a variety of different data sources, including sensor data [67], device-free RFID data [45], connectivity data [68], and mobility data [58]. The proposed method has no limitation on activities. In the experiment, our main focus is to demonstrate the generality of the method on various datasets collected with different types of devices and for various purposes. Table 1 shows the semantics of each dataset. We compare IGTS with different segmentation methods. All the datasets contain multivariate time series either from correlated or from uncorrelated channels. USC-HAD [67] and Device Analyzer [68] are heterogeneous. USC-HAD consists of time series from accelerometer and gyroscope, and device analyzer consists of different connectivity data, such as Cell Tower IDs and Wi-Fi access points. We apply the proposed method to an unlabeled connectivity dataset and measure the inner cluster distance. Finally, we show the performance of the proposed method for finding the actual k.

5.1. Datasets

Synthetic data (Syn): It contains 2–4 time series with the length of 420. The transition times are selected randomly with the uniform distribution. The value of the time series in each segment is set to a random integer between 1 and 6. To make the data more realistic, we add white Gaussian noise with the signal to noise ratio (SNR) of 10 dB. In this case, the transition times can hardly be recognized by eye. Fig. 8(b) illustrates a sample synthetic time series.

Device-free posture recognition by RFID (RFID): The data is collected from an array of 9 passive RFID tags to recognize 12 different human postures [45]. Each posture is performed by 6 subjects for 60 s. The data collection is device-free. The tags

Table 1
Semantic of the datasets.

Dataset	Meaning of a transition time	Meaning of a segment	Example of segments
Device-Free posture recognition [45]	Change in the posture	A posture is performed	Standing freely, standing straightly, sitting, sitting leaning back, sitting leaning left, sitting leaning right, sitting leaning forward, lying in bed, etc.
USC Human Activity Dataset [67]	Change in the low-level activity	A low-level activity is performed	Walking forward, walking upstairs, walking downstairs, running forward, jumping, sitting, standing, sleeping, etc.
Movement detection by Bluetooth iBeacons	Movement of the device collecting Bluetooth RSSI	The device is still	Different locations of the device
Daily Life Routine Dataset [58]	Change in the high-level activity	A high-level activity is performed	Commuting, office work, lunch, dinner
Device analyzer connectivity data [68]	Usual change in location during a day	The smartphone is connected to a certain set of CIDs	Working hours when the smartphone is connected to the CIDs near the workplace

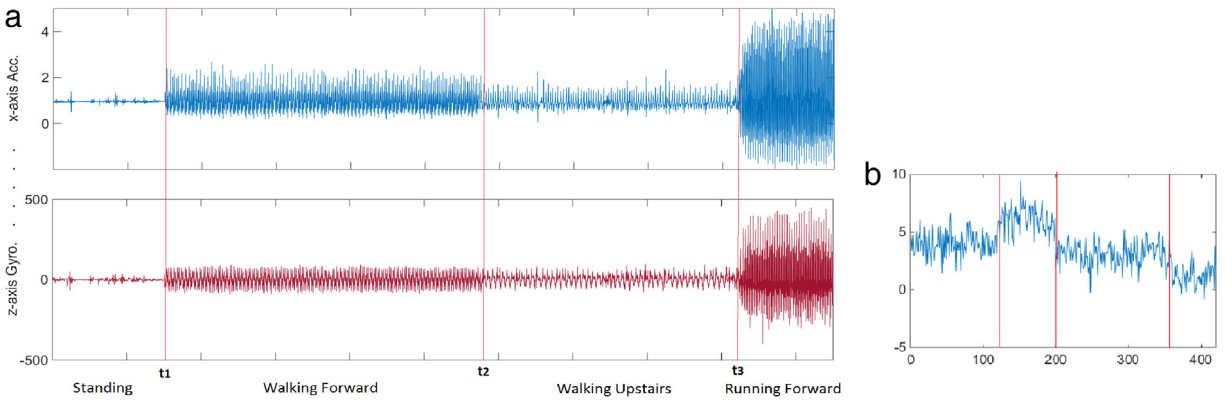

Fig. 8. (a) Sample multivariate time series from USC Human Activity dataset. (b) Sample synthetic time series.

are placed on a wall and the subject performs predefined postures between the wall and the RFID antenna. The corresponding sequence of RSSI is collected at the sampling rate of 2 Hz.

USC Human Activity Dataset (HAD): The dataset includes the most basic and common low-level human activities in daily life from a diverse group of human subjects [67]. In total, it contains 12 activities collected from 14 subjects while each consists of 1000 to 10 000 samples. The 12 human activities include: walk forward, walk left, walk right, go upstairs, go downstairs, run forward, jump up and down, sit and fidget, stand, sleep, elevator up, and elevator down. To make the data more real, each subject was asked to perform 5 trials for each activity on different days. The data is captured by a 3-axis accelerometer and a 3-axis gyroscope, sampled at 100 Hz (see Fig. 8(b)).

Movement detection by Bluetooth iBeacons (iBeacon): iBeacon hardware transmitters are a class of Bluetooth low energy devices that broadcast their locations to nearby portable electronic devices such as smartphones and tablets. We use a smartphone to collect the data receiving signal strength indicator (RSSI) of four Bluetooth iBeacons. During the data collection, a smartphone with fixed position starts collecting RSSI of the iBeacons. After 1–2 min, the position of the smartphone is changed within the experiment area and the change time is recorded. Over the experiment, 5326 RSSI samples are collected from iBeacons at the frequency of 2.5 Hz. We apply temporal segmentation to this dataset to find the transition times when the position of the smartphone is changed.

Daily Life Routine Dataset: This dataset contains not only 34 daily low-level activity labels but also 4 high-level routine class annotations such as commuting, lunch, office work, and dinner [58]. Two wearable accelerometer-based sensors were placed at the dominant wrist and in the right pocket of a single male subject as he was performing his everyday activities. The accelerometer sampling rate data is 100 Hz, and the features (i.e., mean and variance of acceleration of each axis) are computed over a 0.4-s window.

Smartphone logs from Device Analyzer: Device Analyzer gathers data about running background processes, wireless connectivity, GSM communication, and some system status and parameters. For privacy purposes, MAC addresses, Wifi SSIDs, CIDs, and other forms of identification are hashed. Therefore, there is no ground truth or information about the subjects and semantic of the location [68]. In our experiments, the time series are CIDs access distributions. In our experiment, the proposed algorithm is applied to 27 789 days of CID connectivity data that is collected from 271 devices. Each device has at least two weeks of data.

5.2. Labeled low-level activity datasets

We evaluate different versions of our algorithm on low-level activity datasets while k is known. Furthermore, we examine the proposed k estimation method.

5.2.1. Evaluation metrics and experiment setting

The detected transition times are evaluated from two aspects: (i) FN (False Negative) is the number of missed transition times (ii) closeness of the detected transition times to the actual transition times. For the first aspect, the precision of the detection is reported. A detected transition time is considered as true positive if the closest actual transition time is closer than 10% of the length of the total time series. For the second aspect, Root Mean Square Error (RMSE) is calculated among the true positive detected transition times. Thus, it is always below 10%.

For k estimation method, the output of our proposed method is compared with the actual k. For each k, we run the algorithm 40 times on different synthetic time series. The results include confusion matrix as well as the accuracy over 320 runs.

The experiments run on a desktop PC with the configuration of Intel(R) Core i7, 3.4 GHz and 8G RAM.

5.2.2. Baseline and proposed methods

For comparison, we use k-segmentation, which is a foundation of segmentation algorithms in many works [16]. It is based on dynamic programming and finds the segmentation with the minimum variance inside the segments. The second baseline is a state-of-the-art method proposed by Matteson et al. in which the positions of the change-points are estimated based on hierarchical clustering and measuring the differences in multivariate distributions [28]. This method is included in the *ecp* package in the R statistical software.

We evaluate the three versions of our algorithm. IGTS-TopDown and IGTS-DP are presented in Algorithm 1 and Algorithm 2, respectively. We also evaluate IG-Based, which is a simple version of our algorithm without normalization, transformation, and speed-up stages. The difference between IGTS-TopDown and IG-based is that in IGTS-TopDown, the time series are doubled and normalized. Furthermore, cumulative sum or integral of the time series is used to speed up the algorithm. IG-Based is similar to E-Clustering [24] in terms of the cost function and optimization method. However, E-Clustering is not applicable to continue and multivariate time series because it takes advantage of information gain in a different way.

For k estimation method, we find 10 transition times and the corresponding information gains. Then, we compare our method with 4 knee point detection formulas [40], including:

- $\mathcal{L}_i - \mathcal{L}_{i-1}$: difference between magnitudes;
- $2\mathcal{L}_i - \mathcal{L}_{i-1} - \mathcal{L}_{i+1}$: second derivative;
- $\frac{\mathcal{L}_i}{\mathcal{L}_{i-1}} - \frac{\mathcal{L}_{i+1}}{\mathcal{L}_i}$: deference between the ratios;
- $\frac{\mathcal{L}_i - \mathcal{L}_{i-1}}{\mathcal{L}_{i-1}}$: relative difference between magnitudes.

5.2.3. Results

Table 2 shows the experiment results for 4 datasets including one synthetic and three real-world datasets. The dataset name, the number of segments, the number of time series, and the average length of the time series are specified in each row. The performance and the running time are the average values over 20 runs. The only exception is k-segmentation on USC-HAD in which the number of runs was reduced to four because it was too time-consuming. In experiments with the precision value less than %50, the MSE is not mentioned because it is non-meaningful.

Four issues can be inferred from the results. First, IGTS methods are faster than the baselines. k-segmentation is very slow specially for large n. The method proposed in [28] is slow on USC-HAD dataset because it takes a long time to estimate the distribution of the heterogeneous time series. Second, IG-based works as well as IGTS-DP and IGTS-TopDown on the iBeacons and RFID datasets because the time series in these datasets have negative correlations. Increasing the RSSIs of some iBeacons leads to decreasing the other RSSI because when the smartphone gets closer to some iBeacons, it becomes farther from others. Third, neither IG-based nor k-segmentation can capture the transition times in USC-HAD that contains heterogeneous data from accelerometer and gyroscope. Fourth, IGTS-DP is slower than IGTS-TopDown but it provides better results. The output of IGTS-DP and IGTS-TopDown are different if TopDown approach results in local optima. Otherwise, both methods find the global optima and the results are the same. The probability that TopDown approach ends up in local optima or global optima depends on the size and type of the time series. HAD time series are long and IGTS-TopDown cannot usually find the global optima, which is the best answer. That is why IGTS-DP performs better than IGTS-TopDown on HAD dataset.

Fig. 9 shows the confusion matrix of the proposed k estimation formula. Each row of the confusion matrix represents the instances in a predicted class while each column represents the instances in an actual class. The entry in row i and column j denotes how many times our approach detected k as i while the actual number is j. As we run the experiment 50 times for each k, the sum of each column is 50. The results show that the proposed approach can find the correct k.

The accuracies of our k estimation method and other knee point detection formulas are reported in Table 3. Obviously, for IGTS, the proposed formula works significantly better than other knee point detection formulas.

Table 2
Experiment results for low-level activity.

Input				Matteson [28]			*k*-segmentation			IG-Based TopDown			IGTS TopDown			IGTS DP		
Data	*k*	*m*	*n*	P[a]	T[a]	R[a]	P	T	R	P	T	R	P	T	R	P	T	R
Syn	6	2	420	100	0.1	0.1	100	2.4	0.1	76	1.0	5.9	100	0.1	0.1	100	1.4	0.1
	7	2	420	100	0.1	0.1	100	2.7	0.1	76	1.1	8.5	100	0.1	0.1	100	1.4	0.1
	6	3	420	100	0.1	0.1	100	2.6	0.1	80	1.2	8.4	100	0.2	0.1	100	1.5	0.1
	5	5	420	100	0.1	0.1	100	2.8	0.1	67	1.2	8.0	100	0.2	0.1	100	1.5	0.1
RFID	3	9	748	100	0.1	1.1	100	10.2	1.1	100	0.1	1.3	100	0.1	1.1	100	3.4	1.1
	7	9	1 496	100	0.8	1.1	100	80.8	1.2	100	0.9	1.0	100	1.0	1.0	100	17.6	1.0
	11	9	2 244	100	2.4	1.0	100	430.7	0.9	100	3.4	1.1	100	3.7	1.1	100	38.1	1.1
iBeacon	3	4	790	100	0.1	1.1	100	0.1	1.7	100	0.1	1.5	100	0.1	1.6	100	0.1	0.9
	4	4	988	100	0.1	0.8	97	0.1	2.2	100	0.1	2.0	100	0.1	1.8	100	0.1	1.4
	5	4	1 185	100	0.1	1.7	97	0.2	1.7	100	0.1	2.1	100	0.1	1.8	100	0.1	1.3
HAD	2	6	12 600	95	21.8	0.2	25	804.6	–	19	130.0	–	95	0.3	0.5	98	4.2	0.1
	3	6	16 800	93	132.8	0.4	50	2766.8	–	16	539.5	–	98	0.3	0.6	100	19.1	0.1
	4	6	21 000	93	782.9	0.3	33	4498.2	–	17	811.5	–	98	0.4	0.5	98	35.9	0.1

[a] P (%): Precision, T (s): Running Time, R(%): Root Mean Square Error (RMSE).

Table 3
Evaluation on different knee point detection approaches to find *k*.

Method	$\mathcal{L}_i - \mathcal{L}_{i-1}$	$2\mathcal{L}_i - \mathcal{L}_{i-1} - \mathcal{L}_{i+1}$	$\frac{\mathcal{L}_i}{\mathcal{L}_{i-1}} - \frac{\mathcal{L}_{i+1}}{\mathcal{L}_i}$	$\frac{\mathcal{L}_i - \mathcal{L}_{i-1}}{\mathcal{L}_{i-1}}$	$\frac{\mathcal{L}_i - \mathcal{L}_{i-1}}{\mathcal{L}_{i+1} - \mathcal{L}_i}$
Description	Magnitude difference	Second derivative	Ratio difference	Relative magnitude difference	Proposed formula
Accuracy	10%	32%	25%	22%	**71**%

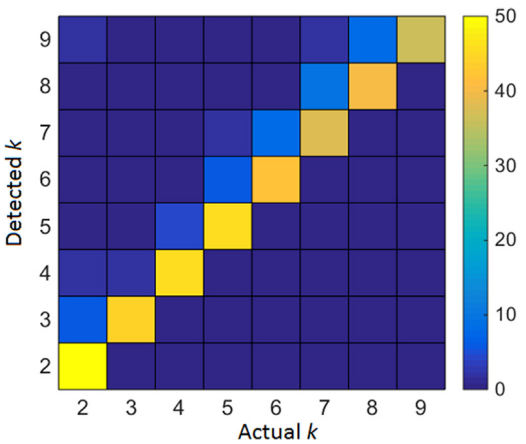

Fig. 9. Confusion matrix of *k* estimation approach.

5.3. Labeled high-level activity datasets

In this experiment, we apply IGTS-TopDown and IGTS-DP to Daily Life Routine dataset to extract transition times in high-level activities. The high-level activities in this dataset are commuting, lunch, office work, and dinner.

5.3.1. Baseline method

For comparison, we choose the method proposed in [8]. They use Latent Dirichlet Allocation (LDA) to infer high-level activities and report qualitative evaluation.

5.3.2. Results

IGTS-TopDown and IGTS-DP have the same results on Daily Life Routine dataset. The qualitative evaluation of our method as well as LDA is shown in Fig. 10.

It is obvious that our approach has less false positive errors. Using the proposed heuristic approach, detected *k* is 7 while LDA approach finds 15 transition times. The two false positives of our approach are around 6 pm. By investigating the ground truth labels, we found that "walking freely" is frequent between 5 pm and 6 pm. Furthermore, LDA is costly because it has

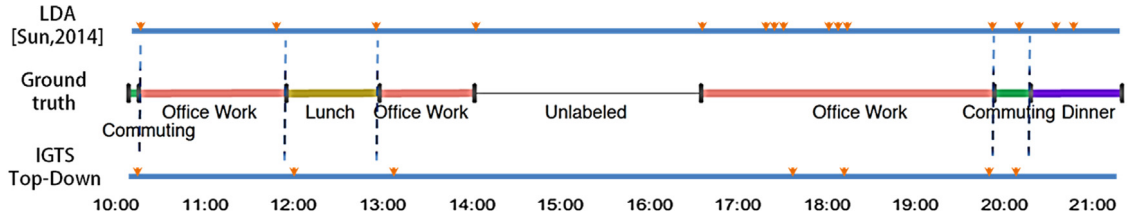

Fig. 10. Qualitative comparison between the proposed method and Latent Dirichlet Allocation (LDA) [8].

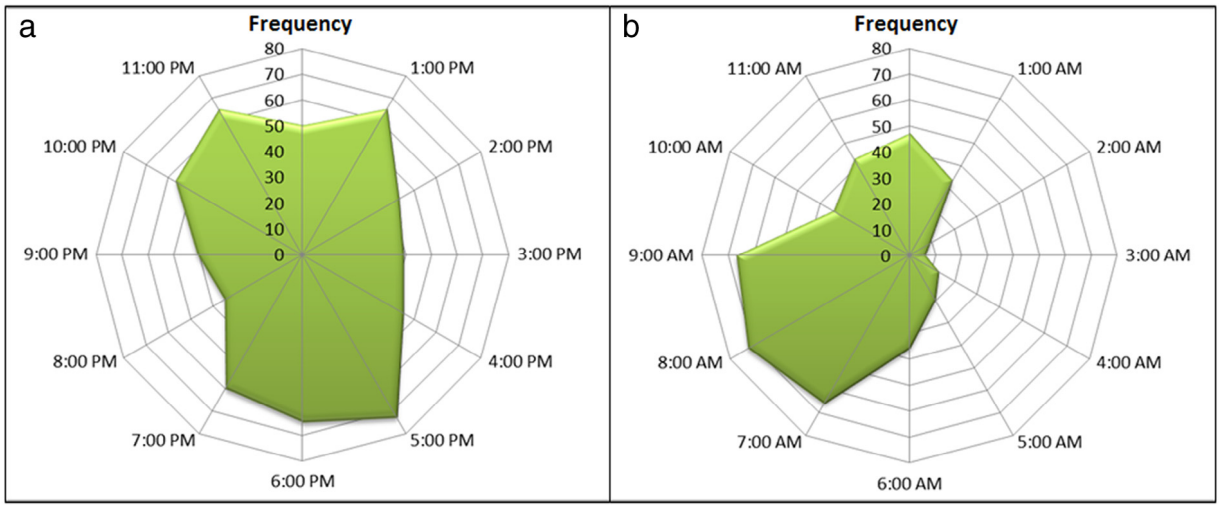

Fig. 11. (a) Transition times in midnight and morning. (b) Transition times in afternoon and night. The number of transition times for all users is reported hourly. It can be inferred that users mainly change their places during 8 am and 5 pm.

two complex stages (see related work). On the other hand, the total running time for segmentation and finding the best k is only 10 s for our algorithm.

If we continue the segmentation, next transition times show the transitions in low-level activities. IGTS algorithm, first, extracts the transitions in the coarse-grained activities, and then finds the transitions in the fine-grained activities. We define coarse-grained activities as high-level activities, and fine-grained activities as low-level activities. Specifically, the earlier transition times denote the transition times in high-level activities. By continuing segmentation, the segments become shorter and later transition times denote transitions in low-level activities. As discussed, by choosing the range of k, we can specify whether we are interested in low-level or high-level activities. For example, in Fig. 6, if the range is between 2 and 10, the best k is 8, which indicates the number of segments for high-level activities.

5.4. Segmentation at routine level from unlabeled dataset

Device Analyzer dataset provides the connectivity data for each user and identifies the connected CID at each time. By processing connectivity data, we segment 24 h of a day into several parts. The output of segmentation shows when the user usually changes his location during a day. In this experiment, the accuracy is 15 min which means the transition times have only four possible formats: hh:00, hh:15, hh:30, and hh:45, where hh indicates the hour of the day. IGTS-TopDown processes the whole data which is 33 Gb in just 4 h.

5.4.1. Results

Fig. 11 shows the frequency distribution of the transition times for all users per hour. For example, 65 users usually change their locations between 1 pm and 2 pm. According to the graphs, users are less likely to change their locations during night. In contrast, 5 pm in the evening and 8 am in the morning are the times when the users have the most movements. This information can help city planners to improve transportation, urban planning, and environment [2].

5.4.2. Validation

Because of the lack of ground truth in Device Analyzer dataset, we examine our method by computing inner cluster distance. We show that our algorithm tends to find coherent segments that means the connected CID does not change so much within the segments. There have been several ways for measuring coherence of the segments. When a segmentation is

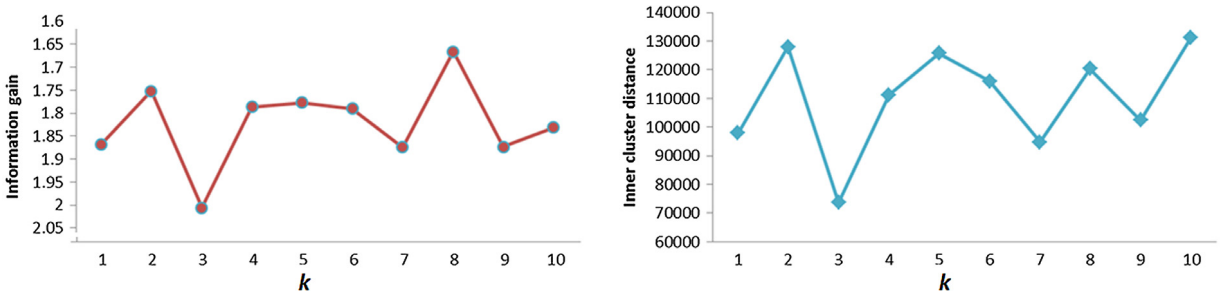

Fig. 12. Comparison between proposed approach and inner cluster distance.

evaluated based on the segments themselves, it is called inner cluster distance. The experiment illustrates that segmentation with the higher information gain value results in lower inner cluster value.

To calculate inner cluster distance, the distance between each couple of instances is calculated for each segment. In the CID example, each instance is a 15-min slot and the distance between two instances is calculated based on the difference in access distributions. For example, [10:00, 10:15] and [11:45, 12:00] have a lower distance value if their user's smartphone connects to similar CID during these periods. Finally, the average over all such distances indicates the inner cluster distance. To compare the outcome of our algorithm with the inner cluster distances, first 10 different segmentations and 10 users are chosen randomly. Then, information gain and inner cluster distance are calculated for each segmentation and each smartphone data. The average of both information gain and inner cluster distance shown in Fig. 12. Considering the reverse order of information gain axis, it is clear that the segmentation with high information gain value has less inner cluster distance which means access durations to the CIDs have a little change within each interval. It should be noticed that measuring information gain is much faster than measuring the inner cluster distances because it does need to compare every two instances. In this experiment, it takes 478.9 ms to calculate the inner cluster distances while our method accomplishes in just 3.7 ms.

6. Conclusion

In this paper, we present a new temporal segmentation method to extract transition times from human activity time series. For each segment, we consider the distribution of the average values of the time series as random variables. Then, the entropy of the distribution is calculated for each segment and information gain, which is the average reduction in entropy, is obtained for the segmentation. We argue that the best segmentation is the one with the highest information gain. To the best of our knowledge, none of the previous works has applied information gain in this way. The way that we use entropy is new and it is different from the common calculation of signal entropy. In the common way of calculating of the entropy of a time series, first, PDF of the time series is estimated then based on the PDF, entropy is calculated. In this case, p denotes the probability of the values of the time series. On the other hand, in our method, first, we compute the integral of each time series, then the entropy is computed among the results. Here, p denotes the integral of a time series. In our method, entropy is used among the features from multiple time series in each segment. Two processing stages are introduced to enable the proposed IGTS method to handle heterogeneous data. We also show that working with the cumulative sum (or integral function) of the time series instead of the original time series decreases the computational cost leading to a faster algorithm.

The proposed cost function is used in conjunction with two optimization methods: TopDown and Dynamic Programming. The original concept of information gain is not applicable to dynamic programming, and we have introduced a modified cost function to be fit with dynamic programming optimization. Furthermore, we prove that information gain increases with each split and proposed a local knee point detection formula to find the number of segments. Experiments show that the proposed formula performs much better than the existing ones. Finally, IGTS is examined on a range of datasets including activity recognition data, connectivity data, and movement data from smartphones, wireless infrastructure, and device-free environment. The results reveal the effectiveness and robustness of the proposed approach. The proposed method focuses on applications that need to consider the whole time series. It is currently not applicable for online applications. In the future, we plan to investigate how to utilize the proposed algorithm in an incremental mode for online processing.

Acknowledgment

This project is funded by RMIT Sustainable Urban Precinct Project (SUPP) "Online Infrastructure for iCO2mmunity".

References

[1] N. Roy, T. Gu, S.K. Das, Supporting pervasive computing applications with active context fusion and semantic context delivery, Pervasive Mob. Comput. 6 (1) (2010) 21–42.

[2] Y. Zheng, L. Capra, O. Wolfson, H. Yang, Urban computing: concepts, methodologies, and applications, ACM Trans. Intell. Syst. Technol. (TIST) 5 (3) (2014) 38.
[3] Ó.D. Lara, A.J. Pérez, M.A. Labrador, J.D. Posada, Centinela: A human activity recognition system based on acceleration and vital sign data, Pervasive Mob. Comput. 8 (5) (2012) 717–729.
[4] J. Cheng, M. Sundholm, B. Zhou, M. Hirsch, P. Lukowicz, Smart-surface: Large scale textile pressure sensors arrays for activity recognition, Pervasive Mob. Comput. (2016).
[5] N.C. Krishnan, D.J. Cook, Activity recognition on streaming sensor data, Pervasive Mob. Comput. 10 (Part B) (2014) 138–154.
[6] H. Qian, Y. Mao, W. Xiang, Z. Wang, Recognition of human activities using svm multi-class classifier, Pattern Recognit. Lett. 31 (2) (2010) 100–111.
[7] D.J. Cook, S.K. Das, Pervasive computing at scale: Transforming the state of the art, Pervasive Mob. Comput. 8 (1) (2012) 22–35.
[8] F.-T. Sun, Y.-T. Yeh, H.-T. Cheng, C.-C. Kuo, M. Griss, Nonparametric discovery of human routines from sensor data, in: 2014 IEEE International Conference on Pervasive Computing and Communications, PerCom, IEEE, 2014, pp. 11–19.
[9] K. Farrahi, D. Gatica-Perez, Discovering routines from large-scale human locations using probabilistic topic models, ACM Trans. Intell. Syst. Technol. 2 (1) (2011) 3:1–3:27.
[10] V. Panagiotou, Blind segmentation of timeseries: A two-level approach (Thesis), 2015.
[11] B. Jackson, J.D. Scargle, D. Barnes, S. Arabhi, A. Alt, P. Gioumousis, E. Gwin, P. Sangtrakulcharoen, L. Tan, T.T. Tsai, An algorithm for optimal partitioning of data on an interval, IEEE Signal Process. Lett. 12 (2) (2005) 105–108.
[12] T. Pavlidis, S.L. Horowitz, Segmentation of plane curves, IEEE Trans. Comput. (8) (1974) 860–870.
[13] J.V. Braun, R. Braun, H.-G. Müller, Multiple changepoint fitting via quasilikelihood, with application to dna sequence segmentation, Biometrika 87 (2) (2000) 301–314.
[14] R. Bellman, On the approximation of curves by line segments using dynamic programming, Commun. ACM 4 (6) (1961) 284.
[15] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmaki, H.T.T. Toivonen, Time series segmentation for context recognition in mobile devices, in: Proceedings IEEE International Conference on Data Mining, 2001. ICDM 2001, pp. 203–210.
[16] H. Hiisila, Segmentation of time series and sequences using basis representations (Thesis), 2007.
[17] A. Kehagias, E. Nidelkou, V. Petridis, A dynamic programming segmentation procedure for hydrological and environmental time series, Stoch. Environ. Res. Risk Assess. 20 (1–2) (2006) 77–94.
[18] H. Guo, X. Liu, L. Song, Dynamic programming approach for segmentation of multivariate time series, Stoch. Environ. Res. Risk Assess. 29 (1) (2015) 265–273.
[19] A. Gionis, H. Mannila, Segmentation algorithms for time series and sequence data, in: Tutorial at 5th SIAM International Conference on Data Mining, Vol. 2005.
[20] E. Keogh, S. Kasetty, On the need for time series data mining benchmarks: a survey and empirical demonstration, Data Min. Knowl. Discov. 7 (4) (2003) 349–371.
[21] A. Gensler, T. Gruber, B. Sick, Blazing fast time series segmentation based on update techniques for polynomial approximations, in: 2013 IEEE 13th International Conference on Data Mining Workshops, ICDMW, Dec 2013, pp. 1002–1011.
[22] O. Banos, J.-M. Galvez, M. Damas, H. Pomares, I. Rojas, Window size impact in human activity recognition, Sensors 14 (4) (2014) 6474–6499.
[23] W. Cheng, X. Zhang, F. Pan, W. Wang, Hicc: an entropy splitting-based framework for hierarchical co-clustering, Knowl. Inf. Syst. (2015) 1–25.
[24] J. Yuan, Y. Zheng, X. Xie, G. Sun, T-drive: enhancing driving directions with taxi drivers' intelligence. Vol. 25, 2013, pp. 220–232.
[25] R.P. Adams, D.J. MacKay, Bayesian online changepoint detection, 2007. arXiv preprint arXiv:0710.3742.
[26] T. Mori, Y. Nejigane, M. Shimosaka, Y. Segawa, T. Harada, T. Sato, Online recognition and segmentation for time-series motion with hmm and conceptual relation of actions, in: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005, (IROS 2005), IEEE, 2005, pp. 3864–3870.
[27] Y. Kawahara, M. Sugiyama, Sequential change-point detection based on direct density-ratio estimation, Stat. Anal. Data Min. 5 (2) (2012) 114–127.
[28] D.S. Matteson, N.A. James, A nonparametric approach for multiple change point analysis of multivariate data, J. Amer. Statist. Assoc. 109 (505) (2014) 334–345.
[29] E. Keogh, S. Chu, D. Hart, M. Pazzani, An online algorithm for segmenting time series, in: Proceedings IEEE International Conference on Data Mining, 2001. ICDM 2001, 2001, pp. 289–296.
[30] F.-l. Chung, T.-C. Fu, R. Luk, V. Ng, Evolutionary time series segmentation for stock data mining, in: 2002 IEEE International Conference on Data Mining, 2002. ICDM 2003. Proceedings, IEEE, 2002, pp. 83–90.
[31] H. Yu, C. Li, J. Dauwels, Network inference and change point detection for piecewise-stationary time series, in: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP), IEEE, 2014, pp. 4498–4502.
[32] P. Smyth, Model selection for probabilistic clustering using cross-validated likelihood, Stat. Comput. 10 (1) (2000) 63–72.
[33] P. Smyth, Clustering using monte carlo cross-validation, in: KDD, 1996, pp. 126–133.
[34] K.T. Vasko, H.T. Toivonen, Estimating the number of segments in time series data using permutation tests, in: 2002 IEEE International Conference on Data Mining, 2002. ICDM 2003. Proceedings, IEEE, 2002, pp. 466–473.
[35] V. Roth, T. Lange, M. Braun, J. Buhmann, A resampling approach to cluster validation, in: Compstat, Springer, 2002, pp. 123–128.
[36] S. Salvador, P. Chan, Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms, in: 16th IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2004, IEEE, 2004, pp. 576–584.
[37] C. Fraley, A.E. Raftery, How many clusters? which clustering method? answers via model-based cluster analysis, Comput. J. 41 (8) (1998) 578–588.
[38] R.A. Baxter, J.J. Oliver, The kindest cut: minimum message length segmentation, in: Algorithmic Learning Theory, Springer, 1996, pp. 83–90.
[39] M.H. Hansen, B. Yu, Model selection and the principle of minimum description length, J. Amer. Statist. Assoc. 96 (454) (2001) 746–774.
[40] A. Foss, O.R. ZaÃŕane, A parameterless method for efficiently discovering clusters of arbitrary shape in large datasets, in: Proceedings, 2002 IEEE International Conference on Data Mining, ICDM 2003, IEEE, 2002, pp. 179–186.
[41] R. Scott Harris, D.R. Hess, J.G. Venegas, An objective analysis of the pressure-volume curve in the acute respiratory distress syndrome, Am. J. Respir. Crit. Care. Med. 161 (2) (2000) 432–439.
[42] J. Lester, T. Choudhury, G. Borriello, A practical approach to recognizing physical activities, in: Pervasive Computing, Springer, 2006, pp. 1–16.
[43] S. Dernbach, B. Das, N.C. Krishnan, B.L. Thomas, D.J. Cook, Simple and complex activity recognition through smart phones, in: 2012 8th International Conference on Intelligent Environments, (IE), IEEE, 2012, pp. 214–221.
[44] Z. He, L. Jin, Activity recognition from acceleration data based on discrete consine transform and svm, in: IEEE International Conference on Systems, Man and Cybernetics, 2009, SMC 2009, IEEE, 2009, pp. 5041–5044.
[45] L. Yao, Q.Z. Sheng, W. Ruan, X. Li, S. Wang, Z. Yang, Unobtrusive posture recognition via online learning of multi-dimensional rfid received signal strength, in: 2013 International Conference on Parallel and Distributed Systems, (ICPADS), IEEE, 2015.
[46] M. Sekine, T. Tamura, T. Togawa, Y. Fukui, Classification of waist-acceleration signals in a continuous walking record, Med. Eng. Phys. 22 (4) (2000) 285–291.
[47] M. Yoshizawa, W. Takasaki, R. Ohmura, Parameter exploration for response time reduction in accelerometer-based activity recognition, in: Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication, ACM, 2013, pp. 653–664.
[48] R. Marx, Ad-hoc accelerometer activity recognition in the iball, in: Proceedings of the 2012 ACM Conference on Ubiquitous Computing, Pittsburgh, PA, USA, Vol. 58, 2012.
[49] A. Mannini, S.S. Intille, M. Rosenberger, A.M. Sabatini, W. Haskell, Activity recognition using a single accelerometer placed at the wrist or ankle, Med. Sci. Sports Exerc. 45 (11) (2013) 2193.
[50] M. Stikic, T. Huynh, K.V. Laerhoven, B. Schiele, Adl recognition based on the combination of rfid and accelerometer sensing, in: Second International Conference on Pervasive Computing Technologies for Healthcare, 2008, PervasiveHealth 2008, IEEE, 2008, pp. 258–263.
[51] L. Bao, S.S. Intille, Activity recognition from user-annotated acceleration data, in: Pervasive Computing, Springer, 2004, pp. 1–17.

[52] A. VÃűgele, B. KrÃijger, R. Klein, Efficient unsupervised temporal segmentation of human motion, in: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, Eurographics Association, 2014, pp. 167–176.
[53] X. Weiwei, W. Weiqiang, B. Peng, S. Liya, T. Leiming, A novel method for automated human behavior segmentation, Comput. Anim. Virtual Worlds (2016).
[54] X. Bao, N.Z. Gong, B. Hu, Y. Shen, H. Jin, Connect the dots by understanding user status and transitions, in: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication, ACM, 2014, pp. 361–366.
[55] K. Farrahi, D. Gatica-Perez, What did you do today?: discovering daily routines from large-scale mobile data, 2008.
[56] R. Wang, F. Chen, Z. Chen, T. Li, G. Harari, S. Tignor, X. Zhou, D. Ben-Zeev, A.T. Campbell, Studentlife: Assessing mental health, academic performance and behavioral trends of college students using smartphones, in: Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp'14, ACM, New York, NY, USA, 2014, pp. 3–14.
[57] W. Mathew, R. Raposo, B. Martins, Predicting future locations with hidden markov models, 2012.
[58] T. Huynh, M. Fritz, B. Schiele, Discovery of activity patterns using topic models, in: Proceedings of the 10th International Conference on Ubiquitous Computing, UbiComp'08, ACM, New York, NY, USA, 2008, pp. 10–19.
[59] H. Pham, C. Shahabi, Y. Liu, Ebm: an entropy-based model to infer social strength from spatiotemporal data, in: Proceedings of the 2013 International Conference on Management of Data, ACM, 2013, pp. 265–276.
[60] J. McInerney, S. Stein, A. Rogers, N.R. Jennings, Breaking the habit: Measuring and predicting departures from routine in individual human mobility, Pervasive Mob. Comput. 9 (6) (2013) 808–822.
[61] N. Eagle, A. Pentland, Reality mining: sensing complex social systems, Pers. Ubiquitous Comput. 10 (4) (2006) 255–268.
[62] C.E. Shannon, W. Weaver, The Mathematical Theory of Communication, University of Illinois Press, 2015.
[63] J. Kleinberg, C. Papadimitriou, P. Raghavan, Segmentation problems, J. ACM 51 (2) (2004) 263–280.
[64] A. Gionis, H. Mannila, E. Terzi, Clustered segmentations, in: Workshop on Mining Temporal and Sequential Data, 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (KDDâĂŹ04), Citeseer, 2004.
[65] J. Peajcariaac, Y. Tong, Convex Functions, Partial Orderings, and Statistical Applications, Elsevier Science, 1992.
[66] https://www.cs.cmu.edu/~ggordon/780-fall07/fall06/homework/15780f06-hw4sol.pdf.
[67] M. Zhang, A.A. Sawchuk, Usc-had: A daily activity dataset for ubiquitous activity recognition using wearable sensors, in: ACM International Conference on Ubiquitous Computing (Ubicomp) Workshop on Situation, Activity and Goal Awareness (SAGAware), Pittsburgh, Pennsylvania, USA, September 2012.
[68] D.T. Wagner, A. Rice, A.R. Beresford, Device analyzer: Large-scale mobile data collection, ACM SIGMETRICS Perform. Eval. Rev. 41 (4) (2014) 53–56.