Accurate Recognition of the Current Activity in the Presence of Multiple Activities

Weihao Cheng, Sarah Erfani, Rui Zhang, and Ramamohanarao Kotagiri

The University of Melbourne, Parkville, Australia,
{weihaoc@student., sarah.erfani@, rui.zhang@, kotagiri@}unimelb.edu.au

Abstract. Sensor based activity recognition (AR) has gained extensive attention in recent years due to the ubiquitous presence of smart devices, such as smartphones and smartwatches. One of the major challenges posed by AR is to reliably recognize the current activity, when a given window of time series data contains several activities. Most of the traditional AR methods assume the entire window corresponds to a single activity, which may cause high error rate in activity recognition. To overcome this challenge, we propose Weighted Min-max Activity Recognition Model (WMARM), which reliably predicts the current activity by finding an optimal partition of the time series matching the occurred activities. WMARM can handle the time series containing an arbitrary number of activities, without having any prior knowledge about the number of activities. We devise an efficient dynamic programming algorithm that solves WMARM in $\mathcal{O}(n^2)$ time complexity, where n is the length of the window. Extensive experiments conducted on 5 real datasets demonstrate about 10%-30% improvement on accuracy of WMARM compared to the state-of-the-art methods.

1 Introduction

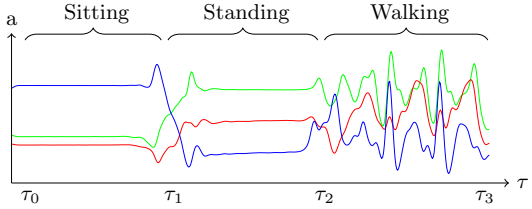
Sensor based activity recognition (AR) has become an important research topic in recent years due to the ubiquitous presence of the smart devices, such as smartphones and smartwatches. The main goal of AR is to identify a user's current activity, e.g., walking, running or being stationary, based on the sensor readings, e.g., acceleration. There are many applications of AR in our daily life [10], such as fitness tracking, safety monitoring, and context-aware behavior. Most of the existing AR methods [2,8,12] use segmented time series to train classifiers for recognition, where each sample represents a single activity. In practice, such AR systems utilize a window to capture the data stream of sensors in a fixed time duration, and supply the captured time series data to a trained classifier to predict the current activity. However, a window of the data stream may contain more than one activity causing transitions at arbitrary time positions, see Fig. 1 for an example. Simply using a time series containing multiple activities for classification that expects input containing single activity can lead to a poor recognition accuracy. A trivial approach is to use a small window, so that there is a high probability to capture the exact time series of the current activity

with a minimal chance of transition taking place. But the trade-off is that the fewer data points will result in lower recognition performance. Therefore, accurate recognition of the current activity in the presence of multiple activities is a challenging task.

There are a few related studies, which aim to minimize the effects induced by activity transitions [13], or recognize the transitions [7,14]. Rednic et al. [13] reported that activity transitions can cause rapid fluctuations in classifier output. They utilized filters to stabilize the prediction, but the approach is unable to identify the current activity. Some AR systems [7,14] learn a classifier to recognize the activity transitions in time series. As the transition can provide information of the activities sequence, this approach can be utilized to infer the current activity. However, it is unsuitable to handle time series containing several transitions, such as stand-walk-run, since there will be a factorial number of classes that should be trained. For example, if there are N different activities, and the system demands to handle at most m transitions, then the total number of required classes is $\sum_{r=1}^{m+1} P_r^N$, where P_r^N stands for the number of permutations selecting r ordered objects from N objects. As a consequence, learning transitions is not an efficient approach for current activity recognition.

To address this difficult problem, an idea is to divide the observed window of time series into segments matching activities transitions. Thereby, the clean time series of the current activity, which is represented by the last segment, can be obtained for recognition. However, the existing time series segmentation methods [1,3,6,4,9,17,15] have at least one of the following drawbacks: (1) optimal solution is not guaranteed; (2) requiring the input of an exact or a maximum number of transitions; (3) only focusing on segmentation without considering activity recognition performance. A detailed discussion is later provided in the Related Works. To address these drawbacks, we propose *Weighted Min-max Activity Recognition Model (WMARM)*, which reliably predicts the current activity by finding an optimal partition of the time series matching the occurred activities. WMARM calculates a set of segments that the maximum value of the recognition errors on those segments is minimized, and the current activity is recognized based on the last segment. WMARM can handle time series containing an arbitrary number of transitions without having any prior knowledge about the number of transitions. WMARM can also be extended by imposing weights on the segments to improve recognition accuracy. Since the search space size of WMARM is $\mathcal{O}(2^n)$, we provide an efficient algorithm using dynamic programming to solve the model in $\mathcal{O}(n^2)$ time complexity, where n is the length of the window. Moreover, we propose a computationally efficient implementation of WMARM that the time series is divided into frames for coarse-grained processing. We conduct extensive experiments on 5 real datasets. The results demonstrate the superior performance of WMARM compared to the state-of-the-art methods when handling time series that contains one or more activity transitions. We also measure the execution time of WMARM algorithm on a smartphone, and the results indicate that the model can be effectively used on such resource constrained devices.

Fig. 1: The time series shown in the curves are 3-axis acceleration signals. There are 3 activities captured in the time series with transition points τ_0, τ_1, τ_2, τ_3, where τ_0 and τ_3 are two end points. Sitting and standing are the previous activities, walking is the current activity that we expect to recognize.



2 Related Work

In this section, we provide a brief survey of the relevant contributions to activity recognition, activity transition processing, and time series segmentation. Traditional sensor based AR systems [2,8,12] train classifiers with segmented samples assuming that each of them contains only one activity. However, they usually fail to recognize the current activity when the input time series contains two or more activities. Rednic et al. [13] focused on reducing the fluctuations caused by activity transitions. They used the Exponentially Weighted Voting filter to avoid spurious prediction, but the method is unable to detect the current activity during transitions. Some works focused on learning and recognizing the transitions [7,14]. However, such approaches normally require training a factorial number of classes regarding the number of transitions are considered. Therefore, they are not efficient for current activity recognition.

Time series segmentation aims to divide a 1-dimension sequence into several homogeneous segments, and existing methods can be summarized into following categories: (1) Heuristic based methods [6] use top-down, bottom-up, sliding window or hybrid ways for time series dividing. The results of heuristic methods are not stable since the optimal solution cannot be guaranteed. (2) LASSO based methods [9] solve the segmentation problem via a least-square regression with a ℓ_1-penalty. The methods require the number of maximum transitions as input. (3) Clustering based methods [17] divide the subsequences in a time series into K-clusters by using K-mean approach. The methods require the number of patterns as input. (4) Dynamic programming based methods [1,3,4,15] obtain an optimal partition of the time series by revealing an optimal structure of the problem. There are two kinds of dynamic programming approaches. One is for handling K-segmentation problem [1,3,15], where the number of transitions is required. The other one [4] can handle an arbitrary number of transitions, but it does not take into account the recognition performance.

In this paper, we propose a current activity recognition model based on time series segmentation via dynamic programming. The model reliably recognizes the current activity while possessing efficient execution time.

3 Methodology

In this section, we first introduce the preliminary concepts of the methodology. We then propose Min-max Activity Recognition Model (MARM), which recognizes the current activity by optimally partitioning a given window of time series. We further improve the model by considering weights on the segments, and propose Weighted Min-max Activity Recognition Model (WMARM). Both of the models can be solved using dynamic programing in $\mathcal{O}(n^2)$ time complexity, where n is the length of the window. Finally, we propose an efficient implementation of WMARM for obtaining high performance on resource constraint devices.

3.1 Problem Statement

Let $X = \{x_1, x_2, ..., x_n\}$ be a time series observed by a window. We define $X_{i:j} = \{x_i, x_{i+1}, ..., x_{j-1}, x_j\}$ $(1 \leq i \leq j \leq n)$ as a subsequence of X containing data points from x_i to x_j. Suppose there is a set of m transition points $\boldsymbol{\tau} = \{\tau_1, \tau_2, ..., \tau_m\}$ in the time series X. We define $\tau_0 = 0$, $\tau_{m+1} = n$ and $0 = \tau_0 < \tau_1 < \tau_2 < ... < \tau_m < \tau_{m+1} = n$. Therefore, the transitions points divide the time series X into $m+1$ segments $\{X_{1:\tau_1}, X_{\tau_1+1:\tau_2}, ..., X_{\tau_m+1:n}\}$, where each segment $X_{\tau_i+1:\tau_{i+1}}$ represents a single activity that is different from its neighbors. For the reliable prediction of the current activity, we expect to locate those transition points that the observed time series can be well-divided into clean segments. As a consequence, the current activity can be exhibited by the last segment and identified accurately.

3.2 Min-Max Activity Recognition Model (MARM)

Suppose we have a hypothesis $P(y \mid Z)$, which outputs the probability of the activity y represented by the time series Z. We define an error function of Z as:

$$\mathcal{E}(Z) = 1 - \max_{y} P(y \mid Z). \tag{1}$$

The function $\mathcal{E}(Z)$ returns the probability error of the predicted activity $\hat{y}$, where $\hat{y}$ holds the highest probability and is represented as:

$$\hat{y} = \operatorname*{argmax}_{y} P(y \mid Z). \tag{2}$$

Thus, given a time series segment $X_{\tau_i+1:\tau_{i+1}}$, we can obtain the corresponding error $\mathcal{E}(X_{\tau_i+1:\tau_{i+1}})$ and the activity prediction $\hat{y} = \operatorname{argmax}_y P(y \mid X_{\tau_i+1:\tau_{i+1}})$. We propose a segmentation function $F(\boldsymbol{\tau})$ of the transition points $\boldsymbol{\tau}$ as follows:

$$F(\boldsymbol{\tau}) = \max_{\tau_i \in \boldsymbol{\tau} \bigcup \{\tau_0\}} \{\mathcal{E}(X_{\tau_i+1:\tau_{i+1}})\}. \tag{3}$$

The function $F(\boldsymbol{\tau})$ returns the maximum error of the segments corresponding to $\boldsymbol{\tau}$. Then, we propose MARM as:

$$\boldsymbol{\tau}^* = \operatorname*{argmin}_{\boldsymbol{\tau}} \{F(\boldsymbol{\tau})\}, \tag{4}$$

where we aim to find an optimal solution $\boldsymbol{\tau}^* = \{\tau_1^*, \tau_2^*, ..., \tau_m^*\}$ such that the maximum error of those segments is minimized. After obtaining $\boldsymbol{\tau}^*$, the current activity is represented by the last segment $X_{\tau_{m+1}^*:n}$, and is predicted as:

$$\hat{y}^* = \underset{y}{\operatorname{argmax}} P(y \mid X_{\tau_{m+1}^*:n}). \tag{5}$$

The intuitive explanation of solving MARM is to properly place the transition points by forcing down the upper bound of the recognition errors. Since the space size of valid $\boldsymbol{\tau}$ is $\mathcal{O}(2^n)$, exhaustive searching the solution is infeasible. However, we can employ dynamic programming to solve the problem in $\mathcal{O}(n^2)$ inspired from the work of [4]. We claim that the problem of optimizing our model exhibits optimal substructure, i.e., optimal solutions to a problem incorporate optimal solutions to related subproblems. Let X_k be the simplified notation of the time series $X_{1:k}$, and $X_0 = \emptyset$. Let $\boldsymbol{\tau}_k^*$ be an optimal solution on X_k, and $\boldsymbol{\tau}_0^* = \emptyset$. We propose the dynamic programming functional equation (DPFE) to solve MARM (Equation 4) as follows:

$$F_{X_l}(\boldsymbol{\tau}_l^*) = \min_{0 \le k < l} \{\max\{F_{X_k}(\boldsymbol{\tau}_k^*),\, \mathcal{E}(X_{k+1:l})\}\}\ (0 < l \le n), \tag{6}$$

where $\boldsymbol{\tau}_1^*, \boldsymbol{\tau}_2^*, ..., \boldsymbol{\tau}_{l-1}^*$ are the previous optimal solutions that have already been obtained. Then, $\boldsymbol{\tau}_l^*$ is calculated as follows:

$$\boldsymbol{\tau}_l^* = \boldsymbol{\tau}_p^* \bigcup \{p\}, \tag{7}$$

where p is the last transition point in $\boldsymbol{\tau}_l^*$ obtained by:

$$p = \underset{0 \le k < l}{\operatorname{argmin}} \{\max\{F_{X_k}(\boldsymbol{\tau}_k^*),\, \mathcal{E}(X_{k+1:l})\}\}. \tag{8}$$

The DPFE in Equation 6 indicates the optimal substructure that an optimal solution $\boldsymbol{\tau}_l^*$ to the problem regarding X_l is derived from the optimal solutions $\boldsymbol{\tau}_1^*, ..., \boldsymbol{\tau}_{l-1}^*$ to the subproblems regarding $X_1, ..., X_{l-1}$, which are the prefixes of the time series X. We show the correctness of the DPFE in Theorem 1.

Theorem 1. *$\boldsymbol{\tau}_l^*$ obtained by Equations 7 and 8 is an optimal solution to $F_{X_l}(\boldsymbol{\tau})$.*

Proof. We assume $\boldsymbol{\tau}_l^*$ is not an optimal solution of $F_{X_l}(\boldsymbol{\tau})$, and claim that $\boldsymbol{\tau}_l^+$ is an optimal solution. Suppose p is the last transition point of $\boldsymbol{\tau}_l^*$, then

$$F_{X_l}(\boldsymbol{\tau}_l^*) = \max\{F_{X_p}(\boldsymbol{\tau}_p^*),\, \mathcal{E}(X_{p+1:n})\}, \tag{9}$$

where $\boldsymbol{\tau}_p^* = \boldsymbol{\tau}_l^* - \{p\}$. Let q be the last transition point of $\boldsymbol{\tau}_l^+$, then

$$F_{X_l}(\boldsymbol{\tau}_l^+) = \max\{F_{X_q}(\boldsymbol{\tau}_q^+),\, \mathcal{E}(X_{q+1:n})\}, \tag{10}$$

where $\boldsymbol{\tau}_q^+ = \boldsymbol{\tau}_l^+ - \{q\}$. Since $\boldsymbol{\tau}_l^+$ is an optimal solution and $\boldsymbol{\tau}_l^*$ is not, hence $F_{X_l}(\boldsymbol{\tau}_l^+) < F_{X_l}(\boldsymbol{\tau}_l^*)$. But we have:

$$\begin{aligned} F_{X_l}(\boldsymbol{\tau}_l^+) &= \max\{F_{X_q}(\boldsymbol{\tau}_q^+),\, \mathcal{E}(X_{q+1:l})\} \\ &\ge \max\{F_{X_q}(\boldsymbol{\tau}_q^*),\, \mathcal{E}(X_{q+1:l})\} \\ &\ge \max\{F_{X_p}(\boldsymbol{\tau}_p^*),\, \mathcal{E}(X_{p+1:l})\} = F_{X_l}(\boldsymbol{\tau}_l^*), \end{aligned} \tag{11}$$

Algorithm 1 MARM Algorithm

Input: (1) The time series X of length n.
Output: (1) The set of transition points $\boldsymbol{\tau}^*$; (2) The predicted current activity $\hat{y}^*$.

1: $\boldsymbol{\tau}_0^* = \emptyset$
2: $\hat{y}^* = Unkown$
3: $F_{X_0}(\boldsymbol{\tau}_0^*) = 0$
4: **while** $l = 1, 2, ..., n$ **do**
5: $\quad p = \underset{0 \le k < l}{\operatorname{argmin}}\{\max\{F_{X_k}(\boldsymbol{\tau}_k^*),\, \mathcal{E}(X_{k+1:l})\}\}$ ▷ Using Equation 8.
6: $\quad \boldsymbol{\tau}_l^* = \boldsymbol{\tau}_p^* \bigcup \{p\}$ ▷ Using Equation 7.
7: $\quad$ **if** $l == n$ **then**
8: $\quad\quad \hat{y}^* = \underset{y}{\operatorname{argmax}} P(y \mid X_{p+1:l})$ ▷ Predicting the current activity.
9: $\quad$ **end if**
10: **end while**
11: $\boldsymbol{\tau}^* = \boldsymbol{\tau}_n^*$
12: **return** $\boldsymbol{\tau}^*, \hat{y}^*$

which is a contradiction. Therefore, $\boldsymbol{\tau}_l^*$ is an optimal solution of $F_{X_l}(\boldsymbol{\tau})$ on the time series X_l.

Based on the proposed DPFE, we can use dynamic programming to obtain an optimal solution $\boldsymbol{\tau}_n^* \equiv \boldsymbol{\tau}^*$ that minimizes $F_{X_n}(\boldsymbol{\tau}) \equiv F(\boldsymbol{\tau})$. We present the algorithm of solving MARM in Algorithm 1. We explain and analyze the algorithm in terms of time complexity: In lines 4-10, we iteratively calculate $\boldsymbol{\tau}_l^*$ from $l = 1$ to n, and each $\boldsymbol{\tau}_l^*$ is calculated in lines 5-6 with $\mathcal{O}(n)$ time complexity. In summary, the final solution $\boldsymbol{\tau}^*$ can be found in $\mathcal{O}(n^2)$ time complexity. When calculating $\boldsymbol{\tau}_n^*$, the last segment $X_{p+1:n}$ is exhibited, and the current activity is predicted as $\hat{y}^*$, which is shown in line 8.

3.3 Weighted Min-Max Activity Recognition Model (WMARM)

MARM finds a set of optimal segments on the observed time series X, and obtains the prediction of the current activity represented by the last segment. Normally, we would like to have a more reliable prediction for the current activity. Therefore, we place emphasis on reducing the error for the last segment. To deliver a more accurate prediction, we propose a new segmentation function $F_{LA}(\boldsymbol{\tau})$ which imposes weights on the last segment and previous segments. Let $p \equiv \tau_m$ be the last transition point in $\boldsymbol{\tau}$, $F_{LA}(\boldsymbol{\tau})$ is defined as follows:

$$F_{LA}(\boldsymbol{\tau}) = \max\{(1-\mu) \cdot F_{X_p}(\boldsymbol{\tau} - \{p\}),\, \mu \cdot \mathcal{E}(X_{p+1:n})\}, \tag{12}$$

in which a weight parameter $\mu \in [0, 1]$ is multiplied to the error of the last segment $X_{p+1:n}$, and $1-\mu$ to the maximum error of the previous $m-1$ segments obtained by $F_{X_p}(\boldsymbol{\tau} - \{p\})$. We propose WMARM based on $F_{LA}(\boldsymbol{\tau})$ as follows:

$$\boldsymbol{\tau}^* = \underset{\boldsymbol{\tau}}{\operatorname{argmin}}\{F_{LA}(\boldsymbol{\tau})\}. \tag{13}$$

By setting μ properly, the accuracy of prediction can be improved. If μ is set to 0.5, the model is equivalent to the original MARM without weight. WMARM (Equation 13) can be solved with the following theorem:

Theorem 2. *Given $\boldsymbol{\tau}_1^*, \boldsymbol{\tau}_2^*, ..., \boldsymbol{\tau}_{n-1}^*$, which are the optimal solutions of $F_{X_1}(\boldsymbol{\tau})$, $F_{X_2}(\boldsymbol{\tau})$, ..., $F_{X_{n-1}}(\boldsymbol{\tau})$, respectively. An optimal solution $\boldsymbol{\tau}^*$ of $F_{LA}(\boldsymbol{\tau})$ can be calculated as:*

$$\boldsymbol{\tau}^* = \boldsymbol{\tau}_p^* \bigcup \{p\}, \tag{14}$$

where p is the last transition point of $\boldsymbol{\tau}^*$ obtained by:

$$p = \underset{0 \le k < n}{\operatorname{argmin}} \{\max\{(1-\mu) \cdot F_{X_k}(\boldsymbol{\tau}_k^*),\, \mu \cdot \mathcal{E}(X_{k+1:n})\}\}. \tag{15}$$

Proof. We assume $\boldsymbol{\tau}^*$ is not an optimal solution, and claim that $\boldsymbol{\tau}^+$ is an optimal solution, then

$$F_{LA}(\boldsymbol{\tau}^*) = \max\{(1-\mu) \cdot F_{X_p}(\boldsymbol{\tau}_p^*),\, \mu \cdot \mathcal{E}(X_{p+1:n})\}, \tag{16}$$

where $\boldsymbol{\tau}_p^* = \boldsymbol{\tau}^* - \{p\}$. Let q be the last transition point of $\boldsymbol{\tau}_l^+$, then

$$F_{LA}(\boldsymbol{\tau}^+) = \max\{(1-\mu) \cdot F_{X_q}(\boldsymbol{\tau}_q^+),\, \mu \cdot \mathcal{E}(X_{q+1:n})\}, \tag{17}$$

where $\boldsymbol{\tau}_q^+ = \boldsymbol{\tau}^+ - \{q\}$. Since $\boldsymbol{\tau}^+$ is an optimal solution and $\boldsymbol{\tau}^+$ is not, then $F_{LA}(\boldsymbol{\tau}^+) < F_{LA}(\boldsymbol{\tau}^*)$. But we have:

$$\begin{aligned} F_{LA}(\boldsymbol{\tau}^+) &= \max\{(1-\mu) \cdot F_{X_q}(\boldsymbol{\tau}_q^+),\, \mu \cdot \mathcal{E}(X_{q+1:n})\} \\ &\ge \max\{(1-\mu) \cdot F_{X_q}(\boldsymbol{\tau}_q^*),\, \mu \cdot \mathcal{E}(X_{q+1:n})\} \\ &\ge \max\{(1-\mu) \cdot F_{X_p}(\boldsymbol{\tau}_p^*),\, \mu \cdot \mathcal{E}(X_{p+1:n})\} \\ &= F_{LA}(\boldsymbol{\tau}^*), \end{aligned} \tag{18}$$

which is a contradiction. Therefore, $\boldsymbol{\tau}^*$ is an optimal solution of $F_{LA}(\boldsymbol{\tau})$ on the time series X.

To find the exact solution of WMARM, we present a dynamic programming algorithm in Algorithm 2. Similar to Algorithm 1, Algorithm 2 computes $\boldsymbol{\tau}_1^*$, $\boldsymbol{\tau}_2^*$, ..., $\boldsymbol{\tau}_{n-1}^*$ in $\mathcal{O}(n^2)$, as shown in lines 4-7. Calculating the last transition point p needs to iteratively examine the optimal value of $F_{X_k}(\boldsymbol{\tau}_k^*)$ from $k = 0$ to $n-1$, as shown in line 8, which needs $\mathcal{O}(n)$ time complexity. Then, the final solution $\boldsymbol{\tau}^*$ is obtained by combining p into $\boldsymbol{\tau}_p^*$, shown in line 9. In summary, the total time complexity of Algorithm 2 is $\mathcal{O}(n^2)$. Finally, the last segment $X_{p+1:n}$ is exhibited when calculating $\boldsymbol{\tau}^*$, and the current activity is predicted as $\hat{y}^*$, which is shown in line 10.

3.4 Efficient Implementation of WMARM

WMARM partitions the time series on data point level, which results in $\mathcal{O}(n^2)$ time complexity. However, the input time series for activity recognition may

Algorithm 2 WMARM Algorithm

Input: (1) The time series X of length n.
Output: (1) The set of transition points $\boldsymbol{\tau}^*$; (2) The predicted current activity $\hat{y}^*$.

$\boldsymbol{\tau}_0^* = \emptyset$
$\hat{y}^* = Unkonwn$
$F_{X_0}(\boldsymbol{\tau}_0^*) = 0$
while $l = 1, 2, ..., n-1$ **do**
  $p = \underset{0 \le k < l}{\operatorname{argmin}}\{\max\{F_{X_k}(\boldsymbol{\tau}_k^*), \mathcal{E}(X_{k+1:l})\}\}$ ▷ Using Equation 8.
  $\boldsymbol{\tau}_l^* = \boldsymbol{\tau}_p^* \bigcup \{p\}$ ▷ Using Equation 7.
end while
$p = \underset{0 \le k < n}{\operatorname{argmin}}\{\max\{(1-\mu) \cdot F_{X_k}(\boldsymbol{\tau}_k^*), \mu \cdot \mathcal{E}(X_{k+1:n})\}\}$ ▷ Using Equation 15.
$\boldsymbol{\tau}^* = \boldsymbol{\tau}_p^* \bigcup \{p\}$ ▷ Using Equation 14.
$\hat{y}^* = \underset{y}{\operatorname{argmax}} P(y \mid X_{p+1:n})$ ▷ Predicting the current activity.
return $\boldsymbol{\tau}^*$, $\hat{y}^*$

contain hundreds of data points, which produces a large amount of computation regarding the quadratic complexity. In practice, we can divide the time series into several frames of size h, and we treat the frame as the basic element in the time series. Therefore, running WMARM on the frame level reduces the amount of computation by a factor of h^2. Moreover, the computations of obtaining the hypothesis $P(y \mid Z)$ in the training phase is also reduced as a result of coarse-grained time series.

4 Empirical Evaluation

In this section, we evaluate the performance of WMARM in terms of accuracy on a desktop platform. The experiment scripts are written in Python 2.7 on 64-bit Ubuntu 14.04 LTS operating system. We also evaluate the execution time of the proposed WMARM algorithm on an iPhone 6 with iOS 9.0 system. The source code is written in Objective-C and C++.

Datasets: The experiments are conducted on 5 datasets: (1) Human Activity Sensing Consortium (HASC) 2011 [5]; (2) Human Activity Recognition on Smartphones Dataset (HARSD)[1]; (3) Actitracker dataset (ACTR) [8]; (4) Daily Sport Activities dataset (DSA)[1]; (5) Smartphone-Based Recognition of Human Activities and Postural Transitions Data Set (HAPT)[1]. We use the acceleration data of those datasets [11].

Experimental Settings: We use 4-fold cross validation for one round of evaluation, and repeat the evaluation for 5 rounds (the datasets are randomly partitioned each round). We use 3/4 data to generate clean samples (contain only one activity) for training the classifiers, and 1/4 data to generate several 5 seconds time series samples where each sample is randomly formed by $K+1$ segments of different activities with K transitions ($K = 0, 1, 2, 3$). The length of each activity

[1] https://archive.ics.uci.edu/ml/datasets.html

segment is randomly selected with no less than 0.5s. We extract the 1/3 lowest frequency Fourier coefficients of given time series as features. We set $h = n/10$ for the efficient implementation of WMARM. We train 10 classifiers on time series of sizes from 0.5s to 5.0s (every 0.5s), respectively, to form the hypothesis $P(y \mid Z)$. Each classifier is a random forest with 10 estimators.
Baselines: We use 5 baseline methods in comparison with WMARM: (1) Naive: We simply use the entire time series for predicting without segmentation. (2) GIR: We use Global Iterative Replacement (GIR) algorithm [3] to segment the time series, where the last segment is used for predicting. (3) OPM: We use Optimal Partitioning Method (OPM) [4] to segment the time series. (4) RNN: We use Recurrent Neural Network (RNN) [16] with LSTM + Softmax layers to predict the current activity. We extract features on every 0.5s time series frames to form the sequential input of RNN. (5) MSG: We manually obtain the true segment of the current activity for predicting, and we denote this method as 'ManualSegment' (MSG).

4.1 Measuring the Accuracy of Current Activity Recognition

To evaluate the accuracy of WMARM for current activity recognition, we compare WMARM with the baselines using datasets: HASC, HAR, ACTR, and DSA. In this experiment, we set the weight $\mu = 0.7$ for WMARM since we experimentally show later that this value of μ obtains the best accuracy among the cross-validation. According to the results shown in Table 1, WMARM outperforms Naive, GIR, OPM, and RNN in all cases, except for $K = 0$ on ACTR dataset, since WMARM always find the optimal segments for recognizing the current activity. Generally, it should be expected that no algorithm can obtain a better accuracy than MSG. However, WMARM outperforms MSG when $K = 0$ on HASC, HAR, and DSA datasets. This is because that WMARM finds the best fitted segment for recognition instead of the true segment, thereby a part of noise can be excluded. To statistically compare the performance of WMARM with the baselines, we conduct the Wilcoxon signed-rank test on their results (80 pairs for each test). The returned p-values represent the lowest level of significance of a hypothesis that results in rejection. This value allows one to determine whether two methods have significantly different performance. We set the significance level $\alpha = 0.05$ for the comparison. For $K = 1, 2, 3$, the returned p-values (7.747e-15 to 1.199e-13), reject the null hypothesis for the comparisons: WMARM vs. all the baselines except for MSG, which indicates superior performance of WMARM against those methods. For $K = 0$, only the p-value of WMARM vs. GIR (2.477e-06) rejects the null hypothesis, which indicates the similar performances of the two methods.

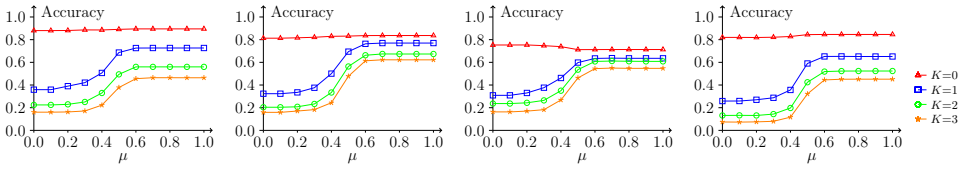
4.2 Evaluating the Impact of μ on Accuracy

We conduct experiments to evaluate the performance of WMARM with different settings of μ. According to the results shown in Fig. 2, when there is no transition in the data, i.e., $K = 0$, the accuracy is slightly affected by the weight μ since the

Table 1: Results of accuracy on datasets: HASC, HARSD, ACTR, and DSA. K is the number of transitions in time series.

	HASC						HARSD					
K	Naive	GIR	OPM	RNN	WMARM	MSG	Naive	GIR	OPM	RNN	WMARM	MSG
0	88.00%	85.00%	88.55%	17.05%	**89.50%**	88.00%	81.30%	81.00%	82.85%	20.95%	**83.65%**	81.30%
1	35.90%	59.35%	46.25%	37.35%	**72.65%**	83.45%	32.35%	53.60%	45.25%	46.90%	**77.00%**	80.60%
2	22.45%	41.65%	28.45%	40.70%	**56.00%**	73.30%	20.40%	34.10%	29.20%	53.75%	**67.40%**	79.05%
3	16.05%	28.20%	19.85%	43.50%	**46.45%**	69.25%	15.90%	23.85%	22.85%	58.95%	**62.25%**	77.80%
	ACTR						DSA					
K	Naive	GIR	OPM	RNN	WMARM	MSG	Naive	GIR	OPM	RNN	WMARM	MSG
0	75.30%	73.25%	**75.50%**	18.65%	71.30%	75.30%	82.00%	76.00%	82.15%	7.00%	**84.60%**	82.00%
1	30.90%	51.60%	46.15%	43.55%	**63.70%**	72.85%	25.85%	63.75%	32.75%	19.25%	**65.25%**	77.30%
2	23.65%	37.55%	32.20%	50.15%	**61.15%**	70.45%	13.15%	47.80%	18.85%	22.30%	**52.30%**	73.90%
3	16.30%	25.65%	23.10%	50.00%	**54.95%**	67.90%	7.40%	31.85%	12.80%	28.50%	**45.10%**	70.05%

Fig. 2: Accuracy of WMARM with respect to μ on datasets: HASC, HARSD, ACTR, and DSA. For $K = 0$, the accuracy slightly improves on HASC, HARSD, and DSA, and slightly drops on ACTR. For $K = 1, 2, 3$, the accuracy reaches maximum around $\mu = 0.7(\pm 0.1)$, and then slightly decreases by less than 1% or becomes stable.



(a) HASC (b) HARSD (c) ACTR (d) DSA

optimal result should be only one segment. However, when there are transitions in the data, i.e., $K > 0$, the accuracy of WMARM significantly improves with respect to the increase of μ when $\mu < 0.7$, since the model emphasis more on the last segment, i.e., the current activity. The accuracy normally reaches the maximum around $\mu = 0.7(\pm 0.1)$, then slightly decreases by less than 1%, or becomes stable in a few cases. Since over emphasizing the weight on the last segment may impair the segmentation results on the previous segments, so that the prediction accuracy on the last segment is affected by the previous segments.

4.3 Measuring the Accuracy on Actual Transitions

In the previous experiments, we use splicing testing samples in order to study the performance of WMARM. In this experiment, we evaluate WMARM on actual transitions resulting from user's changing activities, for example changing naturally from running to walking. The samples are extracted from HAPT dataset [14] which provides several long time series containing a protocol of activities. We randomly select 70% of the data for training and the rest for testing, and

repeat the experiment 10 times. The training samples are extracted during the activities, and the testing samples are extracted between transitions. WMARM obtains 75.41% accuracy, which outperforms the baselines: Naive (30.34%), GIR (49.44%), OPM (39.97%), and RNN (55.82%), except for MSG (76.69%). To explore the statistical significance of the performances of the methods on handling actual transitions, we conduct the Wilcoxon signed-rank test on their results (10 pairs). We set the significance level $\alpha = 0.05$ for the comparison. The p-values of WMARM vs. Naive/GIR/OPM/RNN (0.003346 for all), reject the null hypothesis for the accuracy measurements, implying a significant improvement of WMARM over those methods.

4.4 Evaluating the Execution Time on Smartphone

To evaluate the execution time of the WMARM algorithm, we develop an iOS App on iPhone 6 using Objective-C, and implement the WMARM algorithm as an internal function using C++. The App captures the 3-axis acceleration data with 100 samples per second, which is supplied to WMARM algorithm for processing. We observe a total execution time for 500 runs, and calculate the average time. WMARM algorithm only costs averagely 0.0153 seconds for one execution, which is not expensive for running AR systems on smartphones. Naive method costs averagely 0.0012 seconds for one execution, but its accuracy is much lower than WMARM.

5 Conclusions

In this paper, we highlight a problem normally presented in activity recognition (AR) that traditional methods usually fail to recognize the current activity in the presence of multiple activities. To solve this problem, we devise Weighted Min-max Activity Recognition Model (WMARM), which predicts the current activity by optimally partitioning the observed window of time series matching the activities presented. WMARM considers weights on the partitioned segments to obtain reliable recognition accuracy. WMARM can also effectively process the time series containing an arbitrary number of transitions without any prior knowledge about the number of transitions. Instead of exhaustively searching the optimal solution of WMARM in exponential space, we propose an efficient dynamic programming algorithm that computes the model in $\mathcal{O}(n^2)$ time complexity, where n is the length of the window. Moreover, we present an efficient implementation of WMARM that the computation cost can be further reduced. Extensive experiments on 5 real datasets have demonstrated the superior performance of WMARM on handling time series with one or more activity transitions. The results show about 10%-30% improvement on the accuracy of current activity recognition compared to state-of-the-art methods. The experiment on iPhone 6 shows the prominent computational efficiency of WMARM.

References

1. Bellman, R.: On the approximation of curves by line segments using dynamic programming. Communications of the ACM 4(6), 284 (1961)
2. Hemminki, S., Nurmi, P., Tarkoma, S.: Accelerometer-based transportation mode detection on smartphones. In: Proceedings of Conference on Embedded Networked Sensor Systems. p. 13 (2013)
3. Himberg, J., Korpiaho, K., Mannila, H., Tikanmaki, J., Toivonen, H.T.: Time series segmentation for context recognition in mobile devices. In: Proceedings of International Conference on Data Mining. pp. 203–210 (2001)
4. Jackson, B., Scargle, J.D., Barnes, D., Arabhi, S., Alt, A., Gioumousis, P., Gwin, E., Sangtrakulcharoen, P., Tan, L., Tsai, T.T.: An algorithm for optimal partitioning of data on an interval. Signal Processing Letters 12(2), 105–108 (2005)
5. Kawaguchi, N., Ogawa, N., Iwasaki, Y., Kaji, K., Terada, T., Murao, K., Inoue, S., Kawahara, Y., Sumi, Y., Nishio, N.: Hasc challenge: gathering large scale human activity corpus for the real-world activity understandings. In: Proceedings of Augmented Human International Conference. p. 27 (2011)
6. Keogh, E., Chu, S., Hart, D., Pazzani, M.: Segmenting time series: A survey and novel approach. Data mining in Time Series Databases 57, 1–22 (2004)
7. Khan, A.M., Lee, Y.K., Lee, S.Y., Kim, T.S.: A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer. Transactions on Information Technology in Biomedicine 14(5), 1166–1172 (2010)
8. Kwapisz, J.R., Weiss, G.M., Moore, S.A.: Activity recognition using cell phone accelerometers. SigKDD Explorations Newsletter 12(2), 74–82 (2011)
9. Levy-leduc, C., Harchaoui, Z.: Catching change-points with lasso. In: Proceedings of Advances in Neural Information Processing Systems. pp. 617–624 (2008)
10. Lockhart, J.W., Pulickal, T., Weiss, G.M.: Applications of mobile activity recognition. In: Proceedings of Conference on Ubiquitous Computing. pp. 1054–1058 (2012)
11. Nguyen, T., Gupta, S.K., Venkatesh, S., Phung, D.Q.: A bayesian nonparametric framework for activity recognition using accelerometer data. In: Proceedings of International Conference on Pattern Recognition. pp. 2017–2022 (2014)
12. Reddy, S., Mun, M., Burke, J., Estrin, D., Hansen, M., Srivastava, M.: Using mobile phones to determine transportation modes. Transactions on Sensor Networks (TOSN) 6(2), 13 (2010)
13. Rednic, R., Gaura, E., Kemp, J., Brusey, J.: Fielded autonomous posture classification systems: design and realistic evaluation. In: Proceedings of ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing. pp. 635–640 (2013)
14. Reyes-Ortiz, J.L., Oneto, L., Samà, A., Parra, X., Anguita, D.: Transition-aware human activity recognition using smartphones. Neurocomputing 171, 754–767 (2016)
15. Rosman, G., Volkov, M., Feldman, D., Fisher III, J.W., Rus, D.: Coresets for k-segmentation of streaming data. In: Proceedings of Advances in Neural Information Processing Systems. pp. 559–567 (2014)
16. Sak, H., Senior, A.W., Beaufays, F.: Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In: Proceedings of Interspeech. pp. 338–342 (2014)
17. Tseng, V.S., Chen, C.H., Huang, P.C., Hong, T.P.: Cluster-based genetic segmentation of time series with dwt. Pattern Recognition Letters 30(13), 1190–1197 (2009)