Markov Dynamic Subsequence Ensemble for Energy-Efficient Activity Recognition

Weihao Cheng
School of CIS, The University of Melbourne
weihaoc@student.unimelb.edu.au

Sarah Erfani
School of CIS, The University of Melbourne
sarah.erfani@unimelb.edu.au

Rui Zhang
School of CIS, The University of Melbourne
rui.zhang@unimelb.edu.au

Kotagiri Ramamohanarao
School of CIS, The University of Melbourne
kotagiri@unimelb.edu.au

ABSTRACT

Ubiquitous mobile computing technology provides opportunities for accurate Activity Recognition (AR). Recently, ensemble models using multiple feature representations based on time series subsequences have demonstrated excellent performance on recognition accuracy. However, these models can significantly increase the energy overhead and shorten battery lifespans of the mobile devices. We formalize a dynamic subsequence selection problem that minimizes the computational cost while persevering a high recognition accuracy. To solve the problem, we propose Markov Dynamic Subsequence Ensemble (MDSE), an algorithm for the selection of the subsequences via a Markov Decision Process (MDP), where a policy is learned for choosing the best subsequence given the state of prediction. Regarding MDSE, we derive an upper bound of the expected ensemble size, so that the energy consumption caused by the computations of the proposed method is guaranteed. Extensive experiments are conducted on 6 real AR datasets to evaluate the effectiveness of MDSE. Compared to the state-of-the-art methods, MDSE reduces 70.8% computational cost which is 3.42 times more energy efficient, and achieves a comparably high accuracy.

CCS CONCEPTS

• **Human-centered computing → Ubiquitous and mobile computing**;

KEYWORDS

Activity Recognition, Energy Efficiency, Markov Decision Process

ACM Reference format:
Weihao Cheng, Sarah Erfani, Rui Zhang, and Kotagiri Ramamohanarao. 2017. Markov Dynamic Subsequence Ensemble for Energy-Efficient Activity Recognition. In *Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, Melbourne, VIC, Australia, November 7–10, 2017 (MobiQuitous 2017),* 10 pages.
https://doi.org/10.1145/3144457.3144470

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
MobiQuitous 2017, November 7–10, 2017, Melbourne, VIC, Australia
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-5368-7/17/11…$15.00
https://doi.org/10.1145/3144457.3144470

1 INTRODUCTION

The rapid development of ubiquitous computing technology enables individuals using smartphones to monitor their daily activities, such as walking, running and cycling [7, 10, 15, 24]. The built-in sensors, e.g., accelerometer and gyroscope, provide valuable information regarding an individual's degree of functional ability and lifestyle. Accordingly, many applications of Activity Recognition (AR) including fitness tracking, safety monitoring, context-aware behavior, etc [19] have been developed. However, these applications mostly require continuously sensing in the background, which can consume a lot of energy. Existing works [6, 9, 11, 13, 14, 20, 25–28] handle this problem by scheduling the usage of a customized set of sensors. However, there is no guarantee of a specific hardware environment that all the required sensors can be used or configured freely. Therefore, it is an emergent challenge for developing an energy-efficient AR method given only sensor data.

The task of AR is to recognize the activity using sensor collected time series data. Recently, the models of using multiple feature representations from time series subsequences have demonstrated excellent performance on accuracy [2, 32]. For example, a Subwindow Ensemble Model (SWEM) [32] uses multiple sized windows to capture the time series for generating diverse feature representations, and infers activity with an ensemble of the predictions based on those features. Although SWEM delivers remarkable accuracy in practice, it is computationally expensive when compared to traditional methods. Both training and inference processes using multiple time series subsequences impose a high computational cost. For ordinary AR applications, the training cost is insignificant as the model can be computed offline, but the inference cost is crucial as this process is performed on the portable device in real time. Considering the limited battery life of smartphones and the energy overhead of continuously running AR applets in the background, the above state-of-the-art AR models [2, 32] are unsuitable for resource constrained mobile platforms.

In this paper, we aim to design a method that can reduce the computational cost while maintaining a competitive recognition accuracy. A drawback of SWEM is that it indiscriminately uses a fixed set of subsequences to predict every incoming time series instance. However, many instances can be accurately predicted with fewer subsequences thus saving a huge amount of energy, and only using many subsequences when predicts challenging instances, such as a fast walking instance, which can be confused with running. We formalize a problem of dynamically finding a set of subsequences that minimizes the computational cost with a desired accuracy of

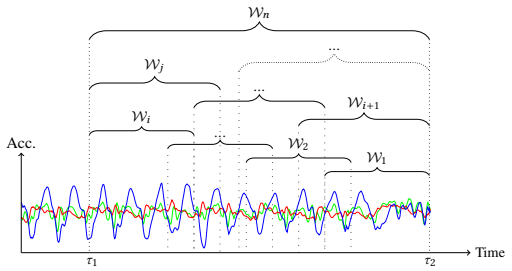


Figure 1: The time series shown in red, green and blue curves are 3-axis acceleration data. $\mathcal{W}_1,...,\mathcal{W}_n$ are n overlapping windows, which observes different ranges of the time series from timestamp τ_1 to τ_2. $\mathcal{W}_1,\mathcal{W}_2,...,\mathcal{W}_i$ share one size with 50% overlaps. Similarly, $\mathcal{W}_{i+1},...,\mathcal{W}_j$ share another size, and there are several different sizes of windows.

the ensemble prediction. We present two constraints for the problem which are proved to ensure a certain level of accuracy. To solve the minimization problem with the presented constraints, we propose **Markov Dynamic Subsequence Ensemble** (MDSE), an algorithm capable of dynamically choosing an optimal subsequence set for a given testing time series instance. The selection of the subsequences is modeled as a Markov Decision Process (MDP), where a policy is learned for choosing the best subsequence regarding the state of prediction. Due to the dynamic number of used subsequences (ensemble size), the computational cost can be extensively reduced. We further show the guaranteed computational efficiency of MDSE by deriving an upper bound of the expected ensemble size. Specifically, our main contributions are as follows:

- To improve energy efficiency for accurate AR, we formalize a problem that minimizes the computational cost with a desired accuracy of the ensemble prediction, by finding an optimal set of the subsequences.
- We present two constraints for the optimization problem and prove their effectiveness on ensuring a certain level of generalization accuracy.
- We propose MDSE to solve the constrained minimization problem. The selection of the subsequences is modeled as an MDP, where a policy is learned to choose the best subsequence during the dynamic process under the proposed constraints.
- We derive an upper bound of the expected ensemble size of MDSE, which indicates a guaranteed computational efficiency of MDSE.

We conduct extensive experiments on 6 real human activity datasets. The results of mathematical estimation show that MDSE can reduce average 74.41% of the original SWEM's computations on all datasets, and obtains similarly high accuracies which are only 1-2% less than SWEM. We also use a Google Nexus 5X to evaluate the energy expenditure of MDSE. The results show about 70.8% energy reduction, which demonstrates significant improvement in energy efficiency by our proposed method.

2 RELATED WORKS

Reducing energy cost for Activity Recognition (AR) on mobile devices has been studied for a decade. Many methods specifically manage the usage of sensors to reduce unnecessary energy expenditure. Lu *et al.* [20] propose Jigsaw system which controls the usage of high power GPS based on an accelerometer. Bhargava *et al.* [4] propose SenseMe system which uses linear accelerometer and rotation vector sensor as a means to suppress the usage of GPS. Bloch *et al.* [6] propose to use a low power cellular network information to detect user stationary/movement status which avoids battery-exhausting transportation mode detection. Some general sensor management methods have also been proposed. Kang *et al.* [11] propose SeeMon system which reduces the energy cost by iteratively selecting cost-efficient sensors in a greedy manner. Wang *et al.* [27] present a hierarchical sensor management scheme for power control. Gordon *et al.* [9] choose the sensors to use based on the estimated future activity and quantified activity-sensor dependencies. However, those methods generally rely on a set of sensors, which lose practicability on constrained hardware environments. Another type of methods reduces the energy cost by managing the sampling rate of a single sensor. Krause *et al.* [14] dynamically reduce sensor sampling rate to conserve energy. Wang *et al.* [26] obtain a Markov-optimal sensor sampling policy that minimizes user state estimation error under a given energy consumption budget. Yan *et al.* [28] choose the best sampling frequency based on empirical rules and thresholds. However, these methods fail to be applied in the situations that the sampling frequency cannot be adjusted freely. According to the shortcomings of the above methods, we need to propose an energy-efficient AR algorithm which is independent of sensor environments. In other words, we want to handle the energy issue from the classification point of view, that the algorithm can work in conjunction with the above methods to further reduce energy consumption.

Recently, Deep Learning (DL) has delivered remarkable classification accuracy in many areas [16]. A few works investigate using DL for the recognition of human activities [29, 30], where a Convolutional Neural Network (CNN) with convolutional layer is applied to learn features from time series subsequences in multiple temporal ranges. It is commonly known that the deep neural nets are computationally expensive even just for the inference, therefore, researchers propose special layers [5] for reducing computational and space complexity regarding resource constrained device. Although these studies reduce the computations of DL, they cannot dynamically minimize the cost for different instances.

The ensemble learning framework is based on solid mathematical foundation and can deliver competitive classification accuracy [33]. The ensemble models using multiple subsequence based feature representations demonstrated state-of-the-art performance in AR. Zheng *et al.* [32] find that a time series with fixed length cannot properly represent all the activities at the same time, because of the fact that activity patterns occur at different temporal scales. Therefore, they propose Subwindow Ensemble Model (SWEM) to overcome this weakness. SWEM uses multiple windows to capture a given time series for obtaining subsequences of different ranges, shown in Figure 1. SWEM generates feature representation for each subsequence, and predicts the activity with an ensemble of the classifiers for those

features, thereby the accuracy is significantly improved. However, SWEM is not computationally efficient, since the use of each subsequence incurs computations on feature extraction and prediction, then the total cost of SWEM equals the sum of all these subsequence costs, which is several times than traditional methods. Due to such an overhead, running of SWEM on mobile devices drains the battery quickly that affects the general daily use. Therefore, our goal is to improve the performance of SWEM by alleviating its energy consumption, without sacrificing the accuracy.

3 PROPOSED METHOD

In this section, we propose an energy-efficient Activity Recognition (AR) method with a high recognition accuracy. We first formalize a dynamic subsequence selection problem that minimizes the computational cost with our presented constraints for ensuring the recognition accuracy. We then propose Markov Dynamic Subsequence Ensemble (MDSE) which uses a Markov Decision Process (MDP) to solve the problem. Regarding the presented constraints, we provide theoretical analysis to verify their effectiveness. Finally, we show that the computational efficiency of MDSE is guaranteed by deriving an upper bound of the expected ensemble size of MDSE. We summarize the frequently used symbols in Table 1.

Symbol	Explanation
X	Time series data.
y	Activity label of X.
m	Total number of activities.
N	Number of subsequences regarding X.
X_i	The i-th subsequence of X.
H	Subsequence set.
C	Computational cost.
$\boldsymbol{d}$	Voting vector by a classifier.
$\boldsymbol{v}$	Voting proportion vector.
$\hat{v}$	The highest voting proportion.
β, η	Parameters of the proposed constraints.
$\mathcal{S}$	State space of MDP.
s	A state $s \in \mathcal{S}$.
$\mathcal{A}$	Action space of MDP.
a	An action $a \in \mathcal{A}$.
π	Policy of MDP.
$R(a)$	Penalty function regarding action a.
$V(s)$	Value function regarding state s.
Ω	Expected ensemble size of MDSE.

Table 1: Major expressions used in the paper

3.1 Markov Dynamic Subsequence Ensemble (MDSE)

Suppose there are m activities which are labeled from 1 to m. The task of AR is to recognize the current activity y based on a segment of time series data X collected from one or more sensors (e.g. accelerometer). Suppose (X, y) is drawn from a distribution $\mathcal{D}$, i.e. $(X, y) \sim \mathcal{D}$, where the length of X is L. We define a set of subsequences: $H_X = \{X_1, X_2, ..., X_N\}$, where $X_i \equiv X_{\tau_{i,1}:\tau_{i,2}}$ is a subsequence of X, ranging from timestamp $\tau_{i,1}$ to $\tau_{i,2}$, and $1 \le \tau_{i,1} < \tau_{i,2} \le L$. For each subsequence X_i, we can generate a feature representation $\phi(X_i)$, and then a classifier based on $\phi(X_i)$ can predict a voting vector $\boldsymbol{d}(X_i) \in \{0,1\}^m$:

$$\boldsymbol{d}(X_i) = (d_1, d_2, ..., d_m), \tag{1}$$

where $d_j = 1$ $(1 \le j \le m)$ implies that the activity j receives the vote from the classifier, and we have $d_1 + d_2 + ... + d_m = 1$. We denote the computational cost of utilizing X_i as $C(X_i)$. Given a subset $H \subseteq H_X$, the ensemble voting vector of using all the subsequences in H is defined as:

$$\boldsymbol{d}_H = \sum_{X_i \in H} \boldsymbol{d}(X_i), \tag{2}$$

and the ensemble voting proportion vector is defined as:

$$\boldsymbol{v}_H = \frac{1}{|H|} \boldsymbol{d}_H. \tag{3}$$

Thereby, we calculate the predicted activity as follows:

$$\hat{y}_H = \text{argmax}\{\boldsymbol{v}_H\}, \tag{4}$$

where $\hat{y}_H$ is the activity with the maximum number of votes. The total computational cost of using H is defined as:

$$C_H = \sum_{X_i \in H} C(X_i). \tag{5}$$

To improve computational efficiency for accurate AR, we define a problem that minimizes the computational cost while maintaining the accuracy at a certain level. Let ρ be a selection function which can dynamically choose a subset $H_X^\rho = \rho(H_X)$ of H_X based on a latent policy. Then, our problem can be formalized as minimizing the expected computational cost with a constraint of generalization accuracy:

$$\min_\rho \mathrm{E}_{(X,y)\sim\mathcal{D}} C_{H_X^\rho}, \tag{6}$$

$$s.t.\, P_{(X,y)\sim\mathcal{D}}(\hat{y}_{H_X^\rho} = y) \ge \alpha, \tag{7}$$

where $\alpha \in (0.5, 1)$ is the parameter bounding the generalization accuracy $P_{(X,y)\sim\mathcal{D}}(\hat{y}_{H_X^\rho} = y)$, i.e., the probability of $\hat{y}_{H_X^\rho} = y$. Since numerically ensuring a generalization accuracy is non-trivial, we present two alternative constraints to replace (7), and formalize an alternative problem as follows:

$$\min_\rho \mathrm{E}_{(X,y)\sim\mathcal{D}} C_{H_X^\rho}, \tag{8}$$

$$s.t.\ \max\{\boldsymbol{v}_{H_X^\rho}\} \ge \beta, \tag{9}$$

$$|H_X^\rho| \ge \eta, \tag{10}$$

where $\beta \in (0.5, 1)$ and $\eta \in (0, N]$ are predefined parameters. Later in section 3.2, we provide theoretical analysis to show that the constraints (9) and (10) can ensure a certain level of generalization accuracy regarding β and η.

To solve the above problem, we propose Markov Dynamic Subsequence Ensemble (MDSE) where the obtaining of H_X^ρ is modeled as a Markov Decision Process (MDP):

- $\mathcal{S}$ is a set of states. A state $s \in \mathcal{S}$ is defined as a tuple $s = (H, \max\{\boldsymbol{d}_H\})$.

- $\mathcal{A}$ is a set of actions, where $\mathcal{A} = \{1,2,...,N,N+1\}$. An action $a \in \mathcal{A}$ implies: (i) Using the subsequence X_a to make a prediction, if $1 \le a \le N$. (ii) Stopping the process, if $a = N+1$.
- $R(a)$: $\mathcal{A} \mapsto \mathbb{R}$ is the penalty function. It represents a valuable penalty by taking an action a. We define the penalty function as:

$$R(a) = \begin{cases} C(X_a) & 1 \le a \le N \\ 0 & a = N+1 \end{cases}. \tag{11}$$

We design a policy $\pi : \mathcal{S} \mapsto \mathcal{A}$, which returns the best action for each state $s \in \mathcal{S}$:

$$\pi(s) = \begin{cases} N+1 & \max\{\boldsymbol{v}_H\} \ge \beta \wedge |H| \ge \eta \\ M(s) & \text{otherwise} \end{cases}, \tag{12}$$

where $M(\boldsymbol{s})$ is a hash mapping from $\mathcal{S}$ to $\mathcal{A}$. Given a state $s = (H, \max\{\boldsymbol{d}_H\})$, we can obtain an action $a = \pi(s)$ for s, then the next state $s' = s'_a$ by taking the action a is obtained as:

$$s'_a = \begin{cases} (H \bigcap \{X_a\}, \max\{\boldsymbol{d}_{H \bigcap \{X_a\}}\}) & 1 \le a \le N \\ s & a = N+1 \text{ (stop action)} \end{cases}. \tag{13}$$

When the action $a = N+1$ is obtained from π, we stop the MDP and return the final state as the output of the MDP. The subset H of the final state is considered as H_X^ρ. Given the policy π, an initial state s_0, and an X, the MDP will visit a series of states $s_1,...,s_K$ by taking actions $a_0, a_1, ..., a_{K-1}$, respectively. The total penalty of this process can be measured by a value function $V^\pi(s) : \mathcal{S} \mapsto \mathbb{R}$ as:

$$V^\pi(s) = R(a_0) + R(a_1) + ... + R(a_{K-1}) \mid s = s_0, \tag{14}$$

which can also be written as Bellman's equation:

$$V^\pi(s) = \begin{cases} R(\pi(s)) + V^\pi(s'_{\pi(s)}) & 1 \le \pi(s) \le N \\ R(\pi(s)) & \pi(s) = N+1 \end{cases}. \tag{15}$$

We can convert our problem into minimizing the value function regarding the policy π:

$$\min_\rho \mathrm{E}_{(X,y)\sim\mathcal{D}} C_{H_X^\rho},\ s.t.\ \max\{\boldsymbol{v}_{H_X^\rho}\} \ge \beta, |H_X^\rho| \ge \eta \tag{16}$$

$$= \min_\pi \mathrm{E}_{(X,y)\sim\mathcal{D}}[R(a_0) + R(a_1) + R(a_2) + ...] \tag{17}$$

$$= \min_\pi \mathrm{E}_{(X,y)\sim\mathcal{D}}[V^\pi(s_0) \mid s_0 = (\varnothing, 0)]. \tag{18}$$

Let $V_X^*(s)$ be the minimal value function for an X. Since $\mathcal{S}$ is a finite state space, $V_X^*(s)$ can be represented as follows:

$$V_X^*(s) = \begin{cases} 0 & \max\{\boldsymbol{v}_H\} \ge \beta \wedge |H| \ge \eta \\ \min\limits_{a \in \mathcal{A}, X_a \notin H} \{R(a) + V_X^*(s'_a)\} & \text{otherwise} \end{cases}, \tag{19}$$

where the restriction $X_a \notin H$ is to avoid the re-selection of a used subsequence which brings no diversity effect to the ensemble performance. Then, we initialize $V_X^*(s) = +\infty$ for each $s \in \mathcal{S}$, and use the value iteration algorithm [22] to repeatedly calculate $V_X^*(s)$ using (19) until convergence. Finally, the optimal policy $\pi^*(s)$ on the distribution $\mathcal{D}$ is learned as follows:

$$\pi^*(s) = \begin{cases} N+1 & \max\{\boldsymbol{v}_H\} \ge \beta \wedge |H| \ge \eta \\ \operatorname*{argmin}\limits_{a \in \mathcal{A}, X_a \notin H} \{R(a) + \mathrm{E}_{(X,y)\sim\mathcal{D}} V_X^*(s'_a)\} & \text{otherwise} \end{cases}. \tag{20}$$

After obtaining the policy $\pi = \pi^*$, MDSE can predict the activity of X by incorporating the MDP. The pseudocode of MDSE is presented in Algorithm 1.

Algorithm 1 Markov Dynamic Subsequence Ensemble (MDSE)

```
Input: A time series X
Output: Predicted activity ŷ
H ← ∅,  d_H ← 0
while |H| < N do
    s ← (H, max{d_H})
    a ← π(s)
    if a = N + 1 then
        break
    end if
    H ← H ⋂ {X_a}
    d_H ← d_H + d_{X_a}
end while
return ŷ = argmax{d_H}
```

3.2 Theoretical Analysis of the Accuracy Constraints in MDSE

We provide theoretical analysis to explain that the constraints (9) and (10) collaboratively ensure the solution of MDSE having a certain level of generalization accuracy regarding the parameters β and η. For the convenience of discussion, we use simplified notations in this section. We denote $H = H_X^\rho$, $\hat{y} = \hat{y}_H$, and let $\hat{v} \in [0,1]$ be the highest voting proportion, which is defined as:

$$\hat{v} = \max\{\boldsymbol{v}_H\}. \tag{21}$$

Thereby, the two constraints are expressed as: $\hat{v} \ge \beta$ and $|H| \ge \eta$.

3.2.1 Analysis of the 1st Constraint. We theoretically explain the effectiveness of (9) by showing that: given $\hat{v} \ge \beta$ and a fixed $|H|$, the generalized recognition accuracy increases monotonically with respect to $\beta \in (0.5, 1)$. Let $n = |H|$, then with the ensemble of n subsequences, the generalization accuracy of the ensemble prediction given $\hat{v} \ge \beta$ is written as the probability $P(\hat{y} = y \mid \hat{v} \ge \beta)$, which can be calculated based on the Bayes' rule:

$$P(\hat{y} = y \mid \hat{v} \ge \beta) = \frac{P(\hat{v} \ge \beta, \hat{y} = y)}{\sum_{j=1}^m P(\hat{v} \ge \beta, \hat{y} = j)} \tag{22}$$

$$= \frac{P(\hat{v} \ge \beta, \hat{y} = y)}{P(\hat{v} \ge \beta, \hat{y} = y) + \sum_{j \ne y} P(\hat{v} \ge \beta, \hat{y} = j)}.$$

Suppose the predictions of the n subsequences are independent with the identical generalization error rate ϵ. The chance of seeing exact k incorrect votes among the n classifiers is $\binom{n}{k}\epsilon^k(1-\epsilon)^{n-k}$. Let $\hat{n}$ be the number of votes that the predicted activity $\hat{y}$ received, i.e., $\hat{n} = n\hat{v}$. If $\hat{y} = y$, then those $\hat{n}$ votes are considered to be correct, and the other $n - \hat{n}$ votes are considered to be incorrect. When $\hat{v} \ge \beta$, then $\hat{n} \ge \lceil n\beta \rceil$ and correspondingly $n - \hat{n} \le n - \lceil n\beta \rceil$. Since $\beta > 0.5$, $\hat{y}$

is the only activity receives votes more than $n/2$, thus the probability $P(\hat{v} \geq \beta, \hat{y} = y)$ can be expressed as:

$$P(\hat{v} \geq \beta, \hat{y} = y) = \sum_{k=0}^{n-\lceil n\beta \rceil} \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k}. \tag{23}$$

If $\hat{y} = j$ for $j \neq y$, then those $\hat{n}$ votes are considered to be incorrect, and the correctnesses of the other $n - \hat{n}$ votes are not able to judge. For simplicity, we assume the error ϵ is equally shared on the other $m-1$ activities, then the chance of a classifier voting for activity $j\,(j \neq y)$ is:

$$\tilde{\epsilon} = \frac{\epsilon}{m-1}, \tag{24}$$

and the chance of a classifier voting for any activity other than j is $1 - \tilde{\epsilon}$. Similar to (23), the probability $P(\hat{v} \geq \beta, \hat{y} = j)$ for $j \neq y$ can be expressed as:

$$P(\hat{v} \geq \beta, \hat{y} = j) = \sum_{k=0}^{n-\lceil n\beta \rceil} \binom{n}{k} (1-\tilde{\epsilon})^k \tilde{\epsilon}^{n-k}. \tag{25}$$

We define $f(\beta) \equiv P(\hat{y} = y \,|\, \hat{v} \geq \beta)$ as a function of β, which can be written as:

$$f(\beta) = \tag{26}$$
$$\frac{\sum_{k=0}^{n-\lceil n\beta \rceil} \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k}}{\sum_{k=0}^{n-\lceil n\beta \rceil} \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k} + (m-1) \sum_{k=0}^{n-\lceil n\beta \rceil} \binom{n}{k} (1-\tilde{\epsilon})^k \tilde{\epsilon}^{n-k}}.$$

We assume $\epsilon < 0.5$ that the classifier outperforms random choice. The following theorem shows that $f(\beta)$ is a monotonically increasing function.

THEOREM 3.1. *Given $\epsilon \in (0, 0.5)$, $f(\beta)$ is a monotonically increasing function.*

PROOF. Let $t = n - \lceil n\beta \rceil$, and $0 \leq t \leq n$. We construct a new function $g(t)$ on $t \in \mathbb{Z}^+$ as:

$$g(t) = \frac{\sum_{k=0}^{t} \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k}}{\sum_{k=0}^{t} \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k} + (m-1) \sum_{k=0}^{t} \binom{n}{k} (1-\tilde{\epsilon})^k \tilde{\epsilon}^{n-k}}. \tag{27}$$

Let $\gamma = \lceil n\beta \rceil - n\beta$. Since $t = n - \lceil n\beta \rceil \Rightarrow \beta = (n - t - \gamma)/n$, then $g(t) = f(\frac{n-t-\gamma}{n})$, and we have $f(\frac{n-t-\gamma'}{n}) = f(\frac{n-t}{n})$ for any $\gamma' \in [0, 1)$. Consequently, if $g(t)$ is a monotonically decreasing function, then $f(\beta)$ is a monotonically increasing function. We rewrite $g(t)$ as:

$$g(t) = \frac{1}{1 + (m-1)u(t)}, \tag{28}$$

where $u(t)$ is:

$$u(t) = \frac{\sum_{k=0}^{t} \binom{n}{k} (1-\tilde{\epsilon})^k \tilde{\epsilon}^{n-k}}{\sum_{k=0}^{t} \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k}}. \tag{29}$$

Since $m - 1 > 0$, if $u(t)$ is proved to be a monotonically increasing function, then $g(t)$ is a monotonically decreasing function. Let $\theta_1(k) = \binom{n}{k}(1-\tilde{\epsilon})^k \tilde{\epsilon}^{n-k}$, and $\theta_2(k) = \binom{n}{k} \epsilon^k (1-\epsilon)^{n-k}$, then:

$$\frac{\theta_1(k)}{\theta_2(k)} = (\frac{\tilde{\epsilon}}{1-\epsilon})^n (\frac{1-\tilde{\epsilon}}{\tilde{\epsilon}})^k (\frac{1-\epsilon}{\epsilon})^k. \tag{30}$$

Since $0 < \tilde{\epsilon} \leq \epsilon < 0.5$, then $(1-\tilde{\epsilon})/\tilde{\epsilon} > 1$ and $(1-\epsilon)/\epsilon > 1$, thus we have:

$$\frac{\theta_1(k)}{\theta_2(k)} > \frac{\theta_1(k-1)}{\theta_2(k-1)}. \tag{31}$$

As a result, we can infer that:

$$u(t+1) - u(t) = \frac{\sum_{k=0}^{t} \theta_1(k) + \theta_1(t+1)}{\sum_{k=0}^{t} \theta_2(k) + \theta_2(t+1)} - \frac{\sum_{k=0}^{t} \theta_1(k)}{\sum_{k=0}^{t} \theta_2(k)} > 0 \tag{32}$$

According to the above derivation, we claim that $u(t)$ is a monotonically increasing function. Consequently, $f(\beta)$ is a monotonically increasing function. □

We prove the monotonic property of $P(\hat{y} = y \,|\, \hat{v} \geq \beta)$ on $\beta \in (0.5, 1)$. Therefore, we can increase $\beta \in (0.5, 1)$ to improve the recognition accuracy of MDSE.

3.2.2 Analysis of the 2nd Constraint. We theoretically explain the effectiveness of (10) by showing that: given $|H| \geq \eta$ and a fixed $\hat{v} \in (0.5, 1)$, the generalized recognition accuracy increases monotonically with respect to $\eta \in (0, N]$. Let $n = |H|$, the generalization accuracy of the ensemble prediction given $n \geq \eta$ is written as the probability $P(\hat{y} = y \,|\, n \geq \eta)$, which can be calculated based on the Bayes rule:

$$P(\hat{y} = y \,|\, n \geq \eta) = \frac{P(n \geq \eta, \hat{y} = y)}{P(n \geq \eta, \hat{y} = y) + \sum_{j \neq y} P(n \geq \eta, \hat{y} = j)}. \tag{33}$$

Similar to the analysis of the 1st constraint, we define $f(\eta) \equiv P(\hat{y} = y \,|\, n \geq \eta)$ as a function of η, which can be written as:

$$f(\eta) = \tag{34}$$
$$\frac{\sum_{n=\eta}^{N} \binom{n}{\lceil n\hat{v} \rceil} \epsilon^{n-n\hat{v}} (1-\epsilon)^{n\hat{v}}}{\sum_{n=\eta}^{N} \binom{n}{\lceil n\hat{v} \rceil} \epsilon^{n-n\hat{v}} (1-\epsilon)^{n\hat{v}} + (m-1) \sum_{n=\eta}^{N} \binom{n}{\lceil n\hat{v} \rceil} (1-\tilde{\epsilon})^{n-n\hat{v}} \tilde{\epsilon}^{n\hat{v}}}.$$

THEOREM 3.2. *Given $\epsilon \in (0, 0.5)$, $f(\eta)$ is a monotonically increasing function.*

PROOF. Similar to the proof of Theorem 3.1, we rewrite $f(\eta)$ as:

$$f(\eta) = \frac{1}{1 + (m-1)u(\eta)}, \tag{35}$$

where $u(\eta)$ is:

$$u(\eta) = \frac{\sum_{n=\eta}^{N} \binom{n}{\lceil n\hat{v} \rceil} (1-\tilde{\epsilon})^{n-n\hat{v}} \tilde{\epsilon}^{n\hat{v}}}{\sum_{n=\eta}^{N} \binom{n}{\lceil n\hat{v} \rceil} \epsilon^{n-n\hat{v}} (1-\epsilon)^{n\hat{v}}} \tag{36}$$

If we can prove $u(\eta)$ is a monotonically decreasing function, then $f(\eta)$ is a monotonically increasing function. Let $\theta_1(n) = \binom{n}{\lceil n\hat{v} \rceil}(1-\tilde{\epsilon})^{n-n\hat{v}} \tilde{\epsilon}^{n\hat{v}}$, and $\theta_2(n) = \binom{n}{\lceil n\hat{v} \rceil} \epsilon^{n-n\hat{v}} (1-\epsilon)^{n\hat{v}}$, then:

$$\frac{\theta_1(n)}{\theta_2(n)} = (\frac{1-\tilde{\epsilon}}{\epsilon})^{n-n\hat{v}} (\frac{\tilde{\epsilon}}{1-\epsilon})^{n\hat{v}} \tag{37}$$

$$= [(\frac{1-\tilde{\epsilon}}{\epsilon})^{1-2\hat{v}} (\frac{\tilde{\epsilon}(1-\tilde{\epsilon})}{\epsilon(1-\epsilon)})^{\hat{v}}]^n \tag{38}$$

Since $0 < \tilde{\epsilon} \leq \epsilon < 0.5$ and $\hat{v} > 0.5$, then we have $[(1-\tilde{\epsilon})/\epsilon]^{1-2\hat{v}} < 1$, and $[(\tilde{\epsilon}(1-\tilde{\epsilon}))/(\epsilon(1-\epsilon))]^{\hat{v}} < 1$. Therefore:

$$\frac{\theta_1(n)}{\theta_2(n)} < \frac{\theta_1(n-1)}{\theta_2(n-1)}. \tag{39}$$

As a result, we can infer that:

$$u(\eta+1)-u(\eta)=\frac{\sum_{n=\eta+1}^{N}\theta_1(n)}{\sum_{n=\eta+1}^{N}\theta_2(n)}-\frac{\sum_{n=\eta+1}^{N}\theta_1(n)+\theta_1(\eta)}{\sum_{n=\eta+1}^{N}\theta_2(n)+\theta_2(\eta)}<0. \tag{40}$$

According to the above derivation, we claim that $u(\eta)$ is a monotonically decreasing function. Consequently, $f(\eta)$ is a monotonically increasing function. □

We prove the monotonic property of $P(\hat{y}=y\,|\,n\geq\eta)$ on $\eta\in(0,N]$. Therefore, we can increase $\eta\in(0,N]$ to improve the recognition accuracy of MDSE. We generally set $1<\eta<N$ for obtaining a reasonable performance of MDSE. If we set $\eta=1$, MDSE will constantly make prediction based on only one subsequence. This is because that, when the first prediction is obtained, the algorithm finds $|H|=1\geq\eta$ as well as $\hat{v}=1\geq\beta$, and then stops the dynamic process immediately without taking other subsequences into consideration. If we set $\eta=N$, MDSE will constantly take all the subsequences into ensemble that degenerate into SWEM, since the constraint $|H|\geq\eta$ can only be fulfilled when $|H|=N$.

3.3 Computational Efficiency of MDSE

A significant advantage of MDSE is that its computational efficiency is guaranteed, toward which we provide theoretical analysis in this section. Let l be an integer, and P_l be the probability that the ensemble size $|H|$ equals l when MDSE returns. The expected ensemble size Ω is calculated as:

$$\Omega=\sum_{l=\eta}^{N}lP_l. \tag{41}$$

Let $\hat{v}^{(l)}$ be the highest voting proportion by the first l chosen subsequences, then P_l can be expressed as:

$$P_l=\begin{cases}P(\hat{v}^{(\eta)}\geq\beta), & \text{if } l=\eta\\ P(\hat{v}^{(\eta)}<\beta,\ldots,\hat{v}^{(l-1)}<\beta,\hat{v}^{(l)}\geq\beta), & \text{if } \eta<l<N\\ P(\hat{v}^{(\eta)}<\beta,\ldots,\hat{v}^{(N-2)}<\beta,\hat{v}^{(N-1)}<\beta), & \text{if } l=N\end{cases} \tag{42}$$

Due to the non-triviality of P_l, the expected ensemble size Ω is hard to estimate. However, it is obvious that $\sum_{l=\eta}^{N}P_l=1$, then we can derive an upper bound for Ω:

$$\begin{aligned}\Omega&=\eta P_\eta+\sum_{l=\eta+1}^{N}lP_l\ \leq\ \eta P_\eta+N\sum_{l=\eta+1}^{N}P_l\\&=\eta P_\eta+N(1-P_\eta)=\eta+(N-\eta)(1-P_\eta)\\&=\eta+(N-\eta)P(\hat{v}^{(\eta)}<\beta).\end{aligned} \tag{43}$$

We present the probability $P(\hat{v}^{(\eta)}<\beta)$ using the representation of multinomial cumulative distribution function proposed in [18]. Let $\boldsymbol{v}^{(\eta)}=\{v_1^{(\eta)},v_2^{(\eta)},\ldots,v_m^{(\eta)}\}$ be the voting proportion vector, and $\hat{v}^{(\eta)}=\max\{v_1^{(\eta)},v_2^{(\eta)},\ldots,v_m^{(\eta)}\}$. Let $\eta_j=\eta v_j^{(\eta)}$ be the votes of the activity j. Then, $P(\hat{v}^{(\eta)}<\beta)$ can be written as:

$$\begin{aligned}P(\hat{v}^{(\eta)}<\beta)&=P(v_1^{(\eta)}<\beta,v_2^{(\eta)}<\beta,\ldots,v_m^{(\eta)}<\beta)\\&=P(\eta_1\leq\lceil\eta\beta\rceil-1,\eta_2\leq\lceil\eta\beta\rceil-1,\ldots,\eta_m\leq\lceil\eta\beta\rceil-1).\end{aligned} \tag{44}$$

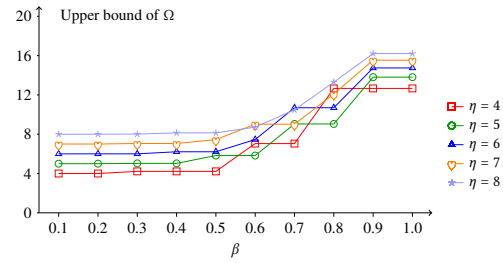


Figure 2: Given $N=20$ (subsequences), $m=6$ (activities) and $\epsilon=0.2$, the curves show the upper bound of Ω as function of β with different η.

Suppose each of the classifiers votes to activities $1,2,\ldots,m$ with probabilities $p_1,p_2,\ldots,p_m$, respectively. The multinomial cumulative distribution function can be presented as follows:

$$\begin{aligned}&P(\eta_1\leq\lceil\eta\beta\rceil-1,\eta_2\leq\lceil\eta\beta\rceil-1,\ldots,\eta_m\leq\lceil\eta\beta\rceil-1)\\&=\frac{\eta!}{\eta^\eta e^{-\eta}}\{\prod_{j=1}^{m}P(W_j\leq\lceil\eta\beta\rceil-1)\}P(\sum_{j=1}^{m}Z_j=\eta),\end{aligned} \tag{45}$$

where $W_j\sim\mathbf{P}(\eta p_j)$ = Poisson distribution, and $Z_j\sim T\mathbf{P}(\eta p_j)$ = truncated Poisson distribution with range $0,1,2,\ldots,\lceil\eta\beta\rceil-1$. Without loss of generality, we set p_1 as $1-\epsilon$ and set $p_2,\ldots,p_m$ as $\tilde{\epsilon}$ to obtain those distributions. Since the probability $P(\sum_{j=1}^{m}Z_j=\eta)$ is hard to calculate, we approximate it using normal distribution:

$$P(\sum_{j=1}^{m}Z_j=\eta)=\frac{1}{\sqrt{2\pi\sum_{j=1}^{m}\sigma_j^2}}\exp\{-\frac{1}{2}(\frac{\eta-\sum_{j=1}^{m}\mu_j}{\sqrt{\sum_{j=1}^{m}\sigma_j^2}})^2\}, \tag{46}$$

where $\mu_j=\mathrm{E}(Z_j)$ and $\sigma_j^2=\mathrm{Var}(Z_j)$. These two moments of Z_i can be calculated as:

$$\mu_j=\eta p_j(1-\frac{P(W_j=\lceil\eta\beta\rceil-1)}{P(W_j\leq\lceil\eta\beta\rceil-1)}), \tag{47}$$

$$\sigma_j^2=\mu_j-(\lceil\eta\beta\rceil-1-\mu_j)(\eta p_j-\mu_j). \tag{48}$$

A more precise estimation for $P(\sum_{j=1}^{m}Z_j=\eta)$ can be obtained by Edgeworth approximation [18], which additionally incorporates 2nd to 4th central moments. We omit the details here due to the page limitation. In summary, the expected ensemble size Ω of MDSE is bounded by:

$$\Omega\leq\eta+(N-\eta)\frac{\eta!}{\eta^\eta e^{-\eta}}\{\prod_{j=1}^{m}P(W_j\leq\lceil\eta\beta\rceil-1)\}P(\sum_{j=1}^{m}Z_j=\eta). \tag{49}$$

We intuitively show the upper bound of Ω in Figure 2, where the curves of the upper bounds are plotted with various settings of β and η. Let Ω_{ub} be the derived bound from (49), we can infer that the expected computational cost of MDSE is bounded by $\sum_{i=1}^{\lceil\Omega_{ub}\rceil}C(X_i)$ assuming $C(X_1)\geq C(X_2)\geq\ldots\geq C(X_N)$.

4 EMPIRICAL EVALUATIONS

In this section, we evaluate the performance of Markov Dynamic Subsequence Ensemble (MDSE) with respect to accuracy and computational cost, and compare it with 4 baseline methods. The experiments are conducted on both desktop and mobile platforms. For the experiments on the desktop platform, we implement the methods

with Python 2.7, and test the methods on a DELL PC with Intel (R) Core (TM) i7-4470 CPU 3.40 GHz, 16G RAM and 64-bit Ubuntu 14.04 LTS operating system. For the experiments on the mobile platform, we implement the methods with Java, and test the methods on a Google Nexus 5X with Android 6.0 system.

Datasets: We use the 3-axis acceleration data of 6 human activity datasets to evaluate the performance of MDSE: (1) Human Activity Sensing Consortium (HASC) 2011 [12]: The data of all 6 activities was collected with 100 readings per second by 7 subjects using iPhone/iPod. (2) Human Activity Recognition on Smartphones Dataset (HARSD) [1]: The data of all 6 activities was collected with 50 readings per second by 30 subjects using a Samsung Galaxy S II. (3) Actitracker dataset (ACTR) [17]: The data of all 6 activities was collected with 20 readings per second by 36 users using Android phones. (4) Daily Sport Activities dataset (DSA) [3]: The data of all 19 activities was collected with 25 readings per second by 8 subjects using an accelerometer placed on torso. (5) Opportunity Dataset (OPP) [23]: The data of selected 5 arm activities ('close', 'reach', 'open', 'release', 'move') was collected with 30 readings per second by 4 subjects using an accelerometer placed on left arm. (6) CHEST [8]: The data of selected 4 activities ('working at computer', 'standing', 'working', 'up/down stairs') was collected with 52 readings per second by 15 participants using an accelerometer placed on chest.

Data Preparations: Let $\mathcal{D}$ be a dataset, we randomly split $\mathcal{D}$ into three groups: $\mathcal{D}_1$, $\mathcal{D}_2$ and $\mathcal{D}_3$, where $|\mathcal{D}_1| = 0.5|\mathcal{D}|$, $|\mathcal{D}_2| = 0.25|\mathcal{D}|$ and $|\mathcal{D}_3| = 0.25|\mathcal{D}|$. For the time series in each group, we sequentially extract a number of 5 seconds segments as our instances, i.e, X. For $\mathcal{D}_1$ and $\mathcal{D}_3$, we perform the extraction for every 1.5 seconds. For $\mathcal{D}_2$, we perform the extraction for every 0.5 seconds. We use the instances of $\mathcal{D}_1$ to train the classifiers of MDSE and baselines. We use the $\mathcal{D}_2$ to learn the Markov Decision Process (MDP) for MDSE. We use the instances of $\mathcal{D}_3$ for testing.

Settings of MDSE: Let τ be the number of readings in one second, and $X_{k_1\tau:k_2\tau}$ be the subsequence of X containing readings from the k_1-th second to τ_2-th second. We generate the subsequence set H_X as $\{X_{0:2\tau}, X_{1\tau:3\tau}, X_{2\tau:4\tau}, X_{3\tau:5\tau}, X_{0:2.5\tau}, X_{1.25\tau:3.75\tau}, X_{2.5\tau:5\tau}, X_{0:3\tau}, X_{\tau:4\tau}, X_{2\tau:5\tau}, X_{0:3.5\tau}, X_{1.5\tau:5\tau}, X_{0:4\tau}, X_{\tau:5\tau}, X_{0:4.5\tau}, X_{0.5\tau:5\tau}, X_{0:5\tau}\}$, where $|H_X| = 17$. For each $X_i \in H_X$, we perform Fast Fourier Transform (FFT) on each axis channel of the time series, and combine the Fourier coefficients of the 3 channels as the feature $\phi(X_i)$ [21]. The cost $C(X_i)$ of using X_i is set as $C(X_i) = L_i \log L_i$ where L_i is the length of X_i. Since the subsequences can be grouped by 6 different lengths: 2τ, 2.5τ, 3τ, 3.5τ, 4τ, 4.5τ, and 5τ, we build base classifiers one for each length. The choosing of classification model for the base classifier is varied, where the details are described in the latter section.

Settings of Baselines: We test 4 baseline methods in comparison with MDSE: (i) Single Base Classifier (SBC): We classify X using a single base classifier. (ii) Subwindow Ensemble Model (SWEM): We classify X by combing all the base classifiers which are trained based on the subsequences of H_X in the same manner of MDSE. (iii) Convolutional Neural Network (CNN): We classify X by a CNN, whose structure is referenced from [30]. The CNN consists of a convolutional layer of 20 feature mappings with 1s length filter striding 0.5s, a max-pooling layer with 0.5s length filter, a hidden layer with 1024 output units, a hidden layer with 30 output units, and a softmax layer. We set learning rate to 0.001 and batch size to 10 for training the CNN. (iv) Recurrent Neural Network (RNN): We classify X by a RNN which consists of a LSTM layer and a Softmax layer. We divide X into a sequence of 0.5 seconds chunks, which are used as input of the RNN. We set learning rate to 0.001 and batch size to 10 for training the RNN. We examine Decision Tree (DT), Logistic Regression (LR) and K-Nearest Neighbor (KNN) as the base classifier for SBC, SWEM and MDSE. The settings of DT, LR and KNN are: (i) DT: We use Classification and Regression Tree (CART) as the Decision Tree algorithm. (ii) LR: We set the regularization weight parameter to 1. (iii) KNN: We set the neighbor number to 1.

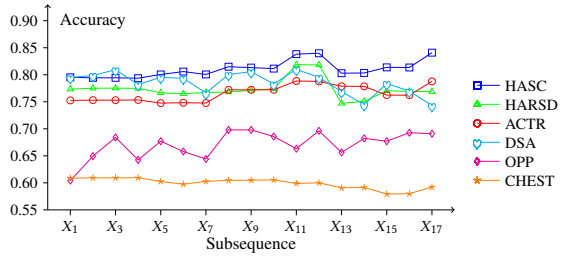
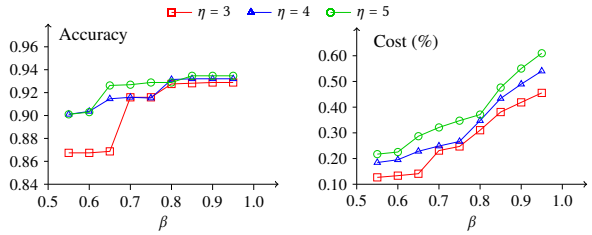


Figure 3: Accuracies of Decision Trees (DTs) using the subsequences X_1, X_2, ..., X_{17} on different datasets.



(a) Accuracy **(b) Percentage of Cost**

Figure 4: Accuracy and Cost (%) of Markov Dynamic Subsequence Ensemble (MDSE) as functions of β with different η.

4.1 Performances of Different Subsequences

We conduct experiments to investigate the accuracy of DT for each single subsequence $X_i \in H_X$. Figure 3 shows the accuracies of using the subsequences from X_1 to X_{17}, on the 6 datasets. As can be seen in the figure, all the subsequences have different accuracies. A smaller subsequence usually result in low accuracy, since only a few data points are captured which many not be a good representative of the activity. A larger subsequences capture more data points which generally deliver a higher accuracy. However, this is not exactly correct, since a subsequence with more data points may not represents the activity properly. Therefore, we perform AR using a number of feature representations based on multiple time series subsequences.

4.2 Markov Dynamic Subsequence Ensemble

We study the performance of the proposed MDSE regarding accuracy and computational efficiency. We compare the evaluation results of

	HASC					HARSD					ACTR				
Method	Accuracy	#Sub	Cost	Cost (%)	Time	Accuracy	#Sub	Cost	Cost (%)	Time	Accuracy	#Sub	Cost	Cost (%)	Time
SBC + DT	83.18%	1	4482.89	10.02%	0.224	80.60%	1	1991.45	10.11%	0.149	75.66%	1	664.39	10.28%	0.090
SWEM + DT	93.47%	17	44741.59	100%	2.780	85.86%	17	19695.79	100%	1.996	87.35%	17	6463.85	100%	1.346
MDSE + DT	91.59%	5.17	10339.73	23.11%	0.769	84.62%	4.02	3142.80	15.96%	0.428	86.54%	5.32	1550.86	23.99%	0.420
SBC + LR	90.04%	1	4482.89	10.02%	0.249	83.92%	1	1991.45	10.11%	0.155	89.38%	1	664.39	10.28%	0.102
SWEM + LR	94.83%	17	44741.59	100%	3.141	85.17%	17	19695.79	100%	2.047	90.93%	17	6463.85	100%	1.485
MDSE + LR	93.66%	5.10	10335.02	23.10%	0.890	85.31%	6.30	6208.77	31.52%	0.760	90.89%	5.82	1804.69	27.92%	0.523
SBC + KNN	88.94%	1	4482.89	10.02%	2.591	83.65%	1	1991.45	10.11%	1.013	83.25%	1	664.39	10.28%	2.787
SWEM + KNN	91.85%	17	44741.59	100%	34.144	84.55%	17	19695.79	100%	14.768	85.28%	17	6463.85	100%	35.983
MDSE + KNN	92.43%	3.89	6872.99	15.36%	6.213	84.13%	3.54	2611.47	13.26%	2.861	85.39%	4.21	1087.78	16.83%	7.055
	DSA					OPP					CHEST				
Method	Accuracy	#Sub	Cost	Cost (%)	Time	Accuracy	#Sub	Cost	Cost (%)	Time	Accuracy	#Sub	Cost	Cost (%)	Time
SBC + DT	73.95%	1	870.72	10.20%	0.104	68.49%	1	1084.32	10.20%	0.124	59.90%	1	1991.45	10.11%	0.169
SWEM + DT	94.12%	17	8537.68	100%	1.566	76.71%	17	10634.65	100%	1.648	73.31%	17	19695.79	100%	2.165
MDSE + DT	93.60%	5.46	2096.86	24.56%	0.483	75.34%	6.94	3651.18	34.33%	0.704	72.13%	6.55	6198.73	31.47%	0.791
SBC + LR	85.04%	1	870.72	10.20%	0.109	76.71%	1	1084.32	10.20%	0.123	69.31%	1	1991.45	10.11%	0.177
SWEM + LR	89.47%	17	8537.68	100%	1.607	77.05%	17	10634.65	100%	1.665	71.08%	17	19695.79	100%	2.386
MDSE + LR	89.47%	8.80	4019.88	47.08%	0.886	77.40%	7.41	4055.15	38.13%	0.768	71.21%	10.68	11719.38	59.50%	1.492
SBC + KNN	94.34%	1	870.72	10.20%	0.952	72.95%	1	1084.32	10.20%	0.719	64.13%	1	1991.45	10.11%	1.536
SWEM + KNN	95.88%	17	8537.68	100%	17.549	78.77%	17	10634.65	100%	11.687	67.15%	17	19695.79	100%	21.378
MDSE + KNN	95.66%	3.66	1165.30	13.65%	3.327	79.45%	6.05	3000.77	28.22%	4.252	67.32%	5.23	4657.79	23.65%	6.166

Table 2: Comparing MDSE with SBC and SWEM using base classifiers: DT, LR and KNN.

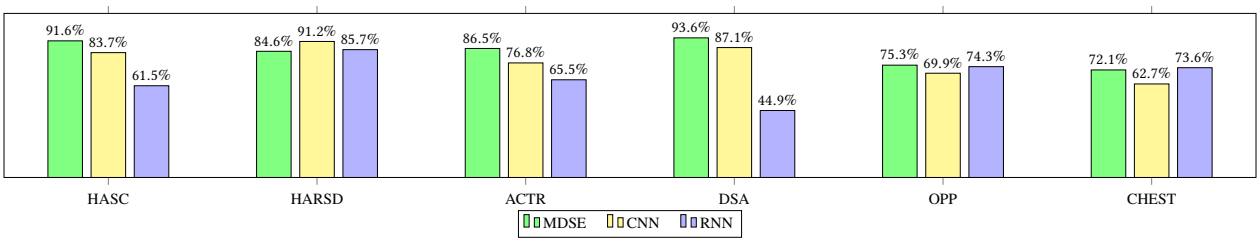


Figure 6: Comparing accuracy of MDSE with CNN and RNN on different datasets.

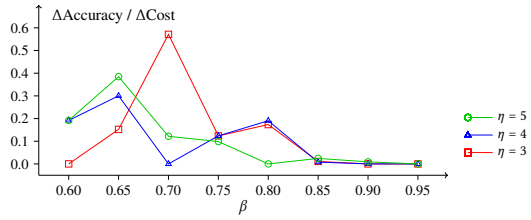


Figure 5: ΔAccuracy/ΔCost of MDSE as a function of β with different η.

MDSE to SBC and SWEM, where DT, LR and KNN are used as base classifier separately. The parameters of MDSE are set as $\beta = 0.7$, $\eta = 3$. We use 5 indicators for the comparison including: (1) 'Accuracy'; (2) '#Sub': the number of used subsequences; (3) 'Cost': the average cost for one instance; (4) 'Cost (%)': the percentage of 'Cost', where the 'Cost (%)' of SWEM is set to 100%, and the others are represented as proportional rates to the 'Cost' of SWEM; (5) 'Time': the average execution time for one instance (millisecond). The experimental results on 6 datasets are shown in Table 2. For accuracy comparison, MDSE performs only approximately 1-2% less than SWEM. However, for computational cost comparison, MDSE reduces average 74.41% of the SWEM's computational cost. Specifically, MDSE reduces average 74.32% with DT, 67.41% with LR, and 81.51% with KNN. To explore the statistical significance of the performances of MDSE and SWEM, we conduct Wilcoxon signed-rank test on their results (18 pairs). 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 following statistical comparisons. For the accuracy comparison between MDSE and SWEM, the returned p-value $= 0.102434 > \alpha$ fails to reject the null hypothesis of the comparison, implying a similar accuracy performance of the two methods. For the computational cost comparison between MDSE

	HASC				HARSD				ACTR			
Method	Accuracy	#Sub	Cost	Cost (%)	Accuracy	#Sub	Cost	Cost (%)	Accuracy	#Sub	Cost	Cost (%)
Random $K-2$	89.20%	4	10424.912	23.30%	83.99%	3	3433.619	17.43%	84.18%	4	1518.817	23.50%
Random $K-1$	90.17%	5	13054.256	29.18%	84.48%	4	4593.724	23.32%	84.37%	5	1896.888	29.35%
MDSE	**91.59%**	5.17	10339.734	**23.11%**	**84.62%**	4.02	3142.803	**15.96%**	**86.54%**	5.32	1550.862	**23.99%**
Random K	90.88%	6	15732.360	35.16%	85.17%	5	5752.307	29.21%	85.42%	6	2277.220	35.23%
Random $K+1$	91.72%	7	18366.606	41.05%	85.03%	6	6928.949	35.18%	85.46%	7	2657.417	41.11%
Random All (SWEM)	93.47%	17	44741.588	100.00%	85.86%	17	19695.794	100.00%	87.35%	17	6463.855	100.00%
	DSA				OPP				CHEST			
Method	Accuracy	#Sub	Cost	Cost (%)	Accuracy	#Sub	Cost	Cost (%)	Accuracy	#Sub	Cost	Cost (%)
Random $K-2$	88.16%	4	1994.436	23.36%	71.92%	5	3107.821	29.22%	69.42%	5	5774.697	29.32%
Random $K-1$	89.96%	5	2494.316	29.22%	73.63%	6	3758.417	35.34%	70.12%	6	6937.692	35.22%
MDSE	**93.60%**	5.46	2096.855	**24.56%**	**75.34%**	6.94	3651.182	**34.33%**	**72.13%**	6.55	6198.734	**31.47%**
Random K	90.61%	6	3001.907	35.16%	75.34%	7	4371.330	41.10%	70.38%	7	8102.068	41.14%
Random $K+1$	91.80%	7	3505.834	41.06%	77.05%	8	5000.964	47.03%	71.01%	8	9265.011	47.04%
Random All (SWEM)	94.12%	17	8537.676	100.00%	76.71%	17	10634.647	100.00%	73.31%	17	19695.794	100.00%

Table 3: Comparing MDSE with Random K methods (DT as base classifier). Due to the page limitation, we only show the results of Random $K-2$, $K-1$, K, $K+1$ and All, where K is set as the smallest integer that greater than the obtained ensemble size of MDSE.

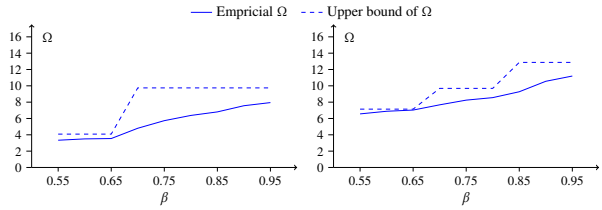
and SWEM, the returned p-value $= 0.000196 < \alpha$ rejects the null hypothesis, implying a significant cost efficiency of MDSE against SWEM. We also compare the accuracy of MDSE to the CNN and RNN. The results on 6 datasets are shown in Figure 6. MDSE obtains better performances than both the CNN and RNN on 4 out of 6 datasets. The CNN only shows a better result on HARSD dataset, and the RNN only shows better results on HARSD and CHEST datasets. Designing widely applicable deep neural nets for AR still need more investigations which can be our future work.

We further evaluate the effectiveness of MDSE by comparing it with a method called Random K which randomly chooses K subsequences for ensemble prediction. We test Random K from $K = 1$ to $|H_X|$. According to the comparison results shown in Table 3, when MDSE uses a similar number of subsequences to the Random K, MDSE achieves a higher or similar accuracy, and spends less computational cost.

4.3 The Accuracy Constraints

The recognition accuracy can be ensured by the presented constraints (9) and (10), since we can prove that the accuracy is monotonically increasing with respect to the parameters β and η. Therefore, we conduct experiments to study the effects of increasing β and η to accuracy and computational cost. We test MDSE on HASC dataset with β varying from 0.55 to 0.95, and $\eta \in \{3, 4, 5\}$. The 'Accuracy' and 'Cost (%)' results are shown in Figure 4a and Figure 4b, respectively. The accuracy improves by increasing β and η, which verifies our analysis. The computational cost is also increased, since that the more restrictive constraints will cause MDSE take more subsequences into ensemble, which results in more computations.

A critical problem is how to choose β and η. Given a fixed η, we can use ΔAccuracy/ΔCost as an indicator to express the ratio of accuracy increment and cost increment by increasing β. We test ΔAccuracy/ΔCost with respect to β and η, where $\mathcal{D}_2$ is used as testing data. The curves of ΔAccuracy/ΔCost are plotted in Figure 5.



(a) $\eta = 3$ **(b)** $\eta = 6$

Figure 7: The dashed curve shows the upper bound of Ω with respect to β, and the solid curve shows the empirical Ω.

We can observe a significant peak when $\beta = 0.7$ and $\eta = 3$. Therefore, using $\beta = 0.7$ and $\eta = 3$ is an empirically optimal choice for MDSE.

4.4 The Computational Efficiency of MDSE

The computational efficiency of MDSE is guaranteed, since we can estimate an upper bound of the expected ensemble size Ω. We evaluate the correctness of the upper bound which is presented in (49). We conduct experiments on HASC dataset to obtain the empirical Ω (average ensemble size) of MDSE. We estimate the upper bound of Ω using (49), where the generalization error ϵ is set as the largest empirical error of the base classifiers. According to the results shown in Figure 7, the curve of empirical Ω is consistently below the curve of the estimated upper bound, which verifies the correctness of our derived bound for Ω.

4.5 Evaluation on Smartphone

We study the performance of MDSE on smartphone regarding accuracy, energy cost and the size of classification model. We compare the evaluation results of MDSE to SBC and SWEM, where DT is used as base classifier. The methods are implemented using Java

with Weka Machine Learning Library [1], and tested on a Google Nexus 5X. In order to measure the pure algorithm running cost excluding the impact from sensors, we import the HASC dataset to the smartphone where the data is prepared following our previous Data Preparation steps. We run the methods on the data of testing group (939 instances), and use PowerTutor [31] to measure a total power consumption. We run each method for 3 times to calculate the average results. According to Table 4, MDSE obtains an accuracy about 1.4% less than the SWEM, but reduces about 70.8% energy of the SWEM. Hence, the model size of MDSE is not large, which demonstrates the feasibility of applying MDSE on mobile devices.

Method	Accuracy	Energy Cost	Model Size
SBC	82.22%	0.95 J	87 KB
SWEM	92.65%	10.03 J	1367 KB
MDSE	91.27%	2.93 J	2958 KB

Table 4: Performances on a Google Nexus 5X.

5 CONCLUSION

State-of-the-art models using feature representations based on multiple time series subsequences have shown to be effective in Activity Recognition (AR). However, they have a large energy cost when they are deployed on mobile devices. In this article, we address this major issue by formalizing a dynamic subsequence selection problem that minimizes the computational cost with constraints for ensuring the recognition accuracy. We theoretically show that our presented constraints can guarantee the generalization accuracy at a certain level. To solve the problem, we propose Markov Dynamic Subsequence Ensemble (MDSE) which learns a policy that chooses the best subsequence given a state of prediction. We then derive an upper bound of the expected ensemble size of MDSE, thus the computational efficiency is guaranteed. We conduct experiments on 6 real human activity datasets to evaluate the performance of MDSE. Comparing to the state-of-the-arts, MDSE demonstrates 70.8% energy reduction that is 3.42 times more energy efficiency, and achieves a similarly high accuracy.

REFERENCES

[1] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L Reyes-Ortiz. 2013. A public domain dataset for human activity recognition using smartphones. In *Proceedings of ESANN*.

[2] O. Banos, J.-M. Galvez, M. Damas, A. Guillen, L.-J. Herrera, H. Pomares, I. Rojas, C. Villalonga, C. S. Hong, and S. Lee. 2015. *Multiwindow Fusion for Wearable Activity Recognition*. 290–297.

[3] B. Barshan and M. C. Yüksek. 2014. Recognizing daily and sports activities in two open source machine learning environments using body-worn sensor units. *Comput. J.* (2014), bxt075.

[4] P. Bhargava, N. Gramsky, and A. Agrawala. 2014. SenseMe: A System for Continuous, On-device, and Multi-dimensional Context and Activity Recognition. In *Proceedings of MobiQuitous*. 40–49.

[5] S. Bhattacharya and N. D. Lane. 2016. Sparsification and Separation of Deep Learning Layers for Constrained Resource Inference on Wearables. In *Proceedings of SenSys*. 176–189.

[6] A. Bloch, R. Erdin, S. Meyer, T. Keller, and A. d. Spindler. 2015. Battery-Efficient Transportation Mode Detection on Mobile Devices. In *Proceedings of MDM*, Vol. 1. 185–190.

[7] A. Bulling, U. Blanke, and B. Schiele. 2014. A Tutorial on Human Activity Recognition Using Body-worn Inertial Sensors. *ACM Comput. Surv.* 46, 3, Article 33 (2014), 33 pages.

[8] P. Casale, O. Pujol, and P. Radeva. 2012. Personalization and User Verification in Wearable Systems Using Biometric Walking Patterns. *Personal Ubiquitous Comput.* 16, 5 (2012), 563–580.

[9] D. Gordon, J. Czerny, T. Miyaki, and M. Beigl. 2012. Energy-Efficient Activity Recognition Using Prediction. In *Proceedings of ISWC*. 29–36.

[10] S. Hemminki, P. Nurmi, and S. Tarkoma. 2013. Accelerometer-based Transportation Mode Detection on Smartphones. In *Proceedings of SenSys*. 13:1–13:14.

[11] S. Kang, J. Lee, H. Jang, H. Lee, Y. Lee, S. Park, T. Park, and J. Song. 2008. SeeMon: Scalable and Energy-efficient Context Monitoring Framework for Sensor-rich Mobile Environments. In *Proceedings of MobiSys*. 267–280.

[12] N. Kawaguchi, Y. Yang, T. Yang, N. Ogawa, Y. Iwasaki, K. Kaji, T. Terada, K. Murao, S. Inoue, Y. Kawahara, Y. Sumi, and N. Nishio. 2011. HASC2011Corpus: Towards the Common Ground of Human Activity Recognition. In *Proceedings of UbiComp*. 571–572.

[13] S. Khalifa, M. Hassan, and A. Seneviratne. 2015. Pervasive self-powered human activity recognition without the accelerometer. In *Proceedings of PerCom*. 79–86.

[14] A. Krause, M. Ihmig, E. Rankin, D. Leong, Smriti Gupta, D. Siewiorek, A. Smailagic, M. Deisher, and U. Sengupta. 2005. Trading off prediction accuracy and power consumption for context-aware wearable computing. In *Proceedings of ISWC*. 20–26.

[15] N. C Krishnan and D. J Cook. 2014. Activity recognition on streaming sensor data. *Pervasive and mobile computing* 10 (2014), 138–154.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of NIPS*. 1097–1105.

[17] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. 2011. Activity Recognition Using Cell Phone Accelerometers. *SIGKDD Explor. Newsl.* 12, 2 (2011), 74–82.

[18] B. Levin et al. 1981. A representation for multinomial cumulative distribution functions. *The Annals of Statistics* 9, 5 (1981).

[19] J. W. Lockhart, T. Pulickal, and G. M. Weiss. 2012. Applications of Mobile Activity Recognition. In *Proceedings of UbiComp*. 1054–1058.

[20] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell. 2010. The Jigsaw Continuous Sensing Engine for Mobile Phone Applications. In *Proceedings of SenSys*. 71–84.

[21] T. Plötz, N. Y. Hammerla, and P. Olivier. 2011. Feature Learning for Activity Recognition in Ubiquitous Computing. In *Proceedings of IJCAI*. 1729–1734.

[22] M. L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*.

[23] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, J. Doppler, C. Holzmann, M. Kurz, G. Holl, R. Chavarriaga, H. Sagha, H. Bayati, M. Creatura, and J. d. R. Millàn. 2010. Collecting complex activity datasets in highly rich networked sensor environments. In *Proceedings of INSS*. 233–240.

[24] M. Shoaib, S. Bosch, O. D. Incel, H. Scholten, and P. J.M. Havinga. 2015. A Survey of Online Activity Recognition Using Mobile Phones. *Sensors* 15, 1 (2015), 2059–2085.

[25] Y. Tang and C. Ono. 2016. Detecting Activities of Daily Living from Low Frequency Power Consumption Data. In *Proceedings of MobiQuitous*. 38–46.

[26] Y. Wang, B. Krishnamachari, Q. Zhao, and M. Annavaram. 2010. Markov-optimal Sensing Policy for User State Estimation in Mobile Devices. In *Proceedings of IPSN*. 268–278.

[27] Y. Wang, J. Lin, M. Annavaram, Q. A. Jacobson, J. Hong, B. Krishnamachari, and N. Sadeh. 2009. A Framework of Energy Efficient Mobile Sensing for Automatic User State Recognition. In *Proceedings of MobiSys*. 179–192.

[28] Z. Yan, V. Subbaraju, D. Chakraborty, A. Misra, and K. Aberer. 2012. Energy-Efficient Continuous Activity Recognition on Mobile Phones: An Activity-Adaptive Approach. In *Proceedings of ISWC*. 17–24.

[29] J. B. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy. 2015. Deep Convolutional Neural Networks on Multichannel Time Series for Human Activity Recognition. In *Proceedings of IJCAI*. 3995–4001.

[30] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang. 2014. Convolutional Neural Networks for human activity recognition using mobile sensors. In *Proceedings of MobiCASE*. 197–205.

[31] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang. 2010. Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones. In *Proceedings of CODES/ISSS*. 105–114.

[32] Y. Zheng, W.-K. Wong, X. Guan, and S. Trost. 2013. Physical Activity Recognition from Accelerometer Data Using a Multi-Scale Ensemble Method.. In *Proceedings of IAAI*. 1575–1581.

[33] Z.-H. Zhou. 2012. *Ensemble methods: foundations and algorithms*. CRC press.

[1] http://www.cs.waikato.ac.nz/ml/weka/