Hindawi Publishing Corporation
Journal of Sensors
Volume 2016, Article ID 6156914, 15 pages
http://dx.doi.org/10.1155/2016/6156914

Research Article

Energy Optimization for Outdoor Activity Recognition

Mehdi Boukhechba,[1] Abdenour Bouzouane,[1] Bruno Bouchard,[1] Charles Gouin-Vallerand,[2] and Sylvain Giroux[3]

[1] *LIARA Laboratory, University of Quebec at Chicoutimi, Chicoutimi, QC, Canada G7H 2B1*
[2] *LICEF Research Center, Tele-Universite of Quebec, Quebec, QC, Canada G1K 9H6*
[3] *DOMUS Laboratory, University of Sherbrooke, Sherbrooke, QC, Canada J1K 2R1*

Correspondence should be addressed to Mehdi Boukhechba; m_boukhechba@esi.dz

Received 20 December 2015; Accepted 30 March 2016

Academic Editor: Chehri Abdellah

The mobile phone is no longer only a communication device, but also a powerful environmental sensing unit that can monitor a user's ambient context. Mobile users take their devices with them everywhere which increases the availability of persons' traces. Extracting and analyzing knowledge from these traces represent a strong support for several applications domains, ranging from traffic management to advertisement and social studies. However, the limited battery capacity of mobile devices represents a big hurdle for context detection, no matter how useful the service may be. We present a novel approach to online recognizing users' outdoor activities without depleting the mobile resources. We associate the places visited by individuals during their movements with meaningful human activities using a novel algorithm that clusters incrementally user's moves into different types of activities. To optimize the battery consumption, the algorithm behaves variably on the basis of users' behaviors and the remaining battery level. Studies using real GPS records from two big datasets demonstrate that the proposal is effective and is capable of inferring human activities without draining the phone resources.

1. Introduction

Human's activity recognition has represented an active topic of research for several decades. However, only in the recent years, with the increasing availability and facilities of collecting movement datasets from GSM or GPS equipped devices or even network wireless technologies like WI-FI [1] and RFID [2], have we had the possibility to study users' activities from their movement traces.

Mobile tracking devices, for example, phones and navigation systems, sense the movement of persons represented by positioning records that capture geolocation, time, and a number of other attributes. Sensing is based on a collection of information related to the achieved activity from raw sensor data (GPS, Wi-Fi, RFID, Bluetooth signals, microphone, camera, accelerometers, magnetometers, gyroscopes, barometers, proximity sensors, etc.) to extract a pertinent information about the current activity. In context awareness systems, the challenge relies on developing applications that sense and react to environmental changes to provide a value-added user experience.

As such, the mobile phone is no longer only a communication device, but also a powerful environmental sensing unit that can monitor a user's ambient context, both unobtrusively and in real time. Ambient sensing [3–5] has become a primary input for a new class of mobile services like activity recognition. In fact, real-time recognition of users' activities offers the possibility to understand what they are doing at the present moment and estimates their actions in the future. This context awareness property makes this field a major piece that provides services to a range of application domains such as real-time traffic monitoring [6, 7], social networking [8], and cognitive assistance.

Smartphones are ubiquitous and becoming more and more sophisticated, with ever-growing computing, networking, and sensing powers. This has been changing the landscape of individuals' daily life and has opened the doors for many interesting data mining applications like activity recognition. Sensors are the source for raw data collection in activity recognition [9]; however, the limited battery capacity of mobile devices represents a big hurdle for the quality and the continuity of the service. The embedded sensors in the mobile devices are major sources of power consumption. Hence, excessive power consumption may become a major obstacle to broader acceptance of context-aware mobile applications, no matter how useful the service may be.

Context information can be related to the user environment, but also to the device itself. Since smartphones are battery-powered, in an ideal scenario, the application will self-adapt and adjust its behavior according to the current battery status of the device. It is in this context that our research lies; we will try to propose a battery-aware activity recognition solution in order to preserve mobiles' life battery.

We bring a novelty to the activity recognition field via three points:

(1) We propose a novel self-adaptive clustering approach that adjusts the computational complexity of the algorithm on the basis of the remaining battery level. The goal is to prevent the massive draining of the mobile resources in order to capture users' movements for the longest time possible. Our mining method is based on a new version of online K-means, where we propose a temporal data window that is characterized by a variable size on the basis of a person's moving behavior and his phones' remaining resources.

(2) The majority of related works are based on the classification of historical records of people's trajectories using density based approach (see, e.g., [10]), where they try to identify the most visited place with post-treatment processes, for example, end of the day and every week. These methods fail in their ability to deal with the places less visited by people but important in their trajectories, for example, cemetery and airport, and cannot be used in fields like assistance where we need a real-time access to a person's activity. That is why we propose a new online solution which answers those difficulties.

(3) This work handles not only stationary behaviors but also moving activities like shopping. We introduce the speed and the variance of the orientation of people's trajectories as a new variable in our system for this purpose.

In this paper, we will demonstrate an innovative method to online switch raw GPS data to meaningful human activities. This method uses only a mobile device without network or historical record requirements, while consuming a minimum of mobile resources. The presented approach aims to enrich people's movements, represented in real time during their travel trajectories, with semantic information about the visited places. Our method is based on the real-time recognition of points of interest, "POI" (a place of interest is an (urban) georeferenced object where a person may carry out a specific activity), in users' trajectories. This service increases the use of this contribution, contrasting from economic uses like traffic management, public transportation, commercials, and advertising, to more serious uses like security, police, and risk management.

The following sections detail our contribution: Section 2 briefly reviews related work; Section 3 presents our approach in terms of three major components, that is, trajectory classification, spatial recognition, and activity discovery; Section 4 describes the experimentation by highlighting two dimensions: accuracy and power saving. Finally, conclusion and future focus, as well as the expected contributions, are summarized in Section 5.

2. Related Works

The emerging concept of activity recognition using mobile devices is a new topic for trajectory data analysis; however, research community's efforts are increasing day by day to carry clear definitions and common understandings [6]. We will highlight two parts of related works, works on activity recognition and semantic trajectories and battery-aware works.

2.1. Activity Recognition and Semantic Trajectory Works. Kang et al. in [1] utilized the access point MAC address of a WI-FI network to capture location data on a campus. They developed a time-based clustering algorithm to "extract places" taking advantage of the continuity of the WI-FI positioning. A new place is found when the distance of the new locations from the previous place is beyond a threshold and when the new locations span a significant time threshold. This algorithm is simple and works in an incremental way on mobile devices. However, the algorithm does not consider the reoccurrence of readings at the same location. More simply, each time it discovers a place, it is a "different" place. This also makes it difficult to discover places that are visited with high frequency but short-dwell time. Moreover, this method requires continuous location data collection with very fine intervals and, thus, large storage. Another shortage of this work is linked to the labeling of the discovered places; since it is made manually by authors, the work needs to be improved by providing an automatic way of labeling activities.

A clustering method called CB-SMOT (Clustering-Based Stops and Moves of Trajectories) is proposed in [10]; it is a clustering method based on the speed variation of the trajectory; it is used to infer semantic information from trajectories. Firstly, this method evaluates the trajectory sample points and generates clusters in places where the trajectory speed is lower than a given threshold for a minimal amount of time. In a second step, the method matches the clusters with a set of relevant geographic places defined by the user. However, the shortage in this solution is that data processing is nonincremental; it is not possible to analyze in real time users mobility using this solution. Moreover, as the capture of users' activities is based on stops, this solution fails to recognize moving activities, such as walking in a park.

CityVoyager, presented in [11], is a recommendation system designed for mobile devices, which recommends shops to users based on data analyzed from their past location history. Their system applies location data to the item-based collaborative filtering algorithm, an algorithm used in many online recommendation systems, by transforming location data history into a list that contains the names of each user's frequently visited shops and rating values indicating the user's preference for each shop. Authors track the visited shops by the loss of GPS signal. Nevertheless, it is known that GPS signals are frequently lost in urban areas even when the user is outdoors; these situations increase the possibilities of false detections. Furthermore, it is also known that it is possible to visit a shop without losing GPS signal, in a case where the shop does not include many obstacles hindering the diffusion of the GPS signal; this situation makes these shops unrecognizable by this work.

Authors claim to propose an approach designed for mobile phones. However, there is no adaptation noted to support this demanding environment. For instance, finding frequented shops requires heavy manipulation of the historical records of the users' visited shops. Authors seem to neglect the limited mobile's resources, since there is no support for limited battery life and there is no effort perceived to online detect and find the frequent shops.

In [12], an algorithm is proposed to associate each stop in a user's trajectory to a list of possible visited places and each of these places is associated with a probability. Finally, depending on the kinds of activities associated with the identified place, the trajectory is classified into probable trajectory behavior. In this work, authors assume the moving object is a person that travels using transportation means associated with a traceable (GPS) device (car, bus, metro, or train). The person gets off the transportation means to walk to the final destination. During this time interval, the person is not traceable; accordingly, authors have used probabilities to find the visited place. This work uses numerous thresholds that are set manually as minimum duration of an activity. Nevertheless, since these parameters may depend on user profiles, this work may be ineffective on large datasets that contain several profiles.

While developing a rich body of work for managing moving objects, the research community has shown a little interest in the online recognition of POI in users' trajectories. The majority of related works are based on the classification of historical records excluding problems linked to mobile's performance, like battery life and low-computational capacities.

Moreover, nearly all approaches are based on the detection of stops within people's movements, neglecting activities with movements, and only a minority of these studies try to automatically identify the background geographic information, since generally we request a set of relevant geographic places defined manually by the user.

The majority of outdoor activity recognition approaches use a fixed activity's minimum duration threshold that represents the minimum time a user has to spend in the POI (place of interest) to be declared as visited place. This threshold prevents false activity detection, such as traffic jams. However, previously fixing this threshold will increase error probability because when set to a small value, it will increase the number of false activities, like passing by a POI; setting it to a high value will miss detection of some short-dwell activities, such as buying cigarettes at the convenience store. Thus, we propose a novel approach to online recognize users' visited places; our algorithm will be totally unsupervised, which operates without any beforehand fixed threshold.

2.2. Battery-Aware Works. The problem of power management on mobile devices has been well-explored. Viredaz et al. [13] surveyed many fundamental but effective methods for saving power on handheld devices. These methods concern a range of phone components such as processor, memory, display screen, audio system, and wireless networking. It has been suggested from the architectural point of view that the system hardware should be designed as a collection of interconnected building blocks that could function independently to enable independent power management.

In [14], authors proposed a dynamic frequency/voltage scaling (DVS) to reduce power consumption by configuring the processor based on the requirements of the executing applications. It is recognized as the basis of numerous energy management solutions [15]. DVS exploits the fact that dynamic power consumption is a strictly convex function of the CPU speed and attempts to save energy by reducing the supply voltage and frequency at runtime. Hence, the power-saving scheme should be fully customized for real-time power consumption situations and specific application requirements. However, these methods are more suitable for lower-level system design rather than application development.

Shih et al. [16] introduced a technique to increase battery lifetime of a personal digital assistant- (PDA-) based phone by reducing its idle power, the power a device consumes in a "standby" state. To do so, they essentially shut down the device and its wireless network card when the device is not being used, avoiding energy waste while not activated. The device is powered only when an incoming call is received or when the user needs to use the PDA for other purposes.

Authors in [17] have gone further by proposing a whole framework of Energy Efficient Mobile Sensing System (EEMSS) for automatic user-state recognition. The core component of EEMSS is a sensor management scheme for mobile devices that operates sensors hierarchically, by selectively turning on the minimum set of sensors to monitor a user state, and triggers new set of sensors if necessary to achieve state transition detection. Energy consumption is reduced by shutting down unnecessary sensors at any particular time.

As seen above, significant efforts have been undertaken for a wise use of mobile resources; however, the proposed techniques try to limit the calculation capacities to gain in battery life. Our perspective is different from this one. We propose a self-adaptive approach that changes dynamically the calculation capacities on the basis of battery life and user state. Our work will also stand out by the flexibility that it offers to users, since they have the freedom to choose the degree of austerity in the use of the battery.

3. Overview of the Approach

We assume the person is GPS traceable via a smartphone. Usually, a person's activities are divided into two

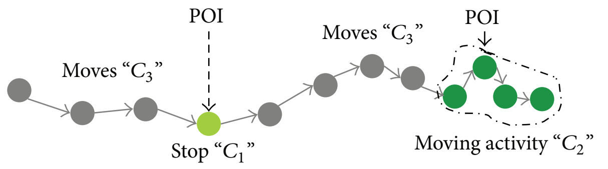


FIGURE 1: Relation between moves, stops, and activities with moving.

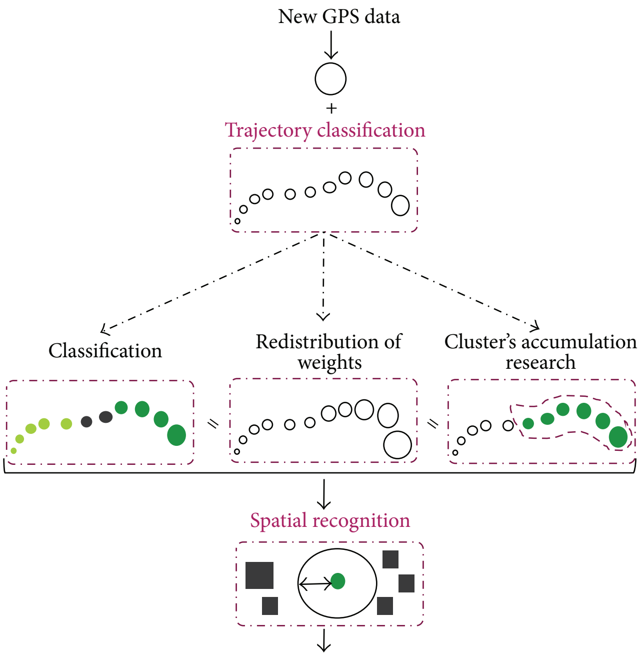

New GPS data
+
Trajectory classification
Classification
Redistribution of weights
Cluster's accumulation research
Spatial recognition

FIGURE 2: The overall process of our activity recognition approach.

behaviors: stationary and nonstationary, where the second one is also divided into two categories moving to reach a goal and moving to do a goal.

For example, working in the office is a stationary activity while going from the workplace to a shopping center is nonstationary activity; shopping itself is also a nonstationary activity, but the goal is to do shopping, so it is an activity with movements (see Figure 1). Based on these concepts we introduce 3 types of clusters:

(1) Stop concept, represented by "C_1", characterizes stationary activities.

(2) Activity with movements "C_2" is nonstationary activities that require movement over a time interval.

(3) Moves, represented by "C_3," are a set of actions that aim to move from a POI to another.

To deal with all these concepts, we present in Figure 2 the overall approach of our activity recognition mechanism.

In the first step, we introduced a real-time classification method based on K-means to classify every new position data according to the three families (stops, moves, and activity with movements). At the same time, we observe the accumulation of types of clusters, such that, after a certain threshold of the same cluster's accumulation, we conclude that the person is probably doing something interesting. For the second step, we summarize the accumulated clusters to a probable POI and we begin a geospatial research for the closest and the most meaningful geographical entity. If the research process succeeds, we determine this point as a POI. The third step is to assign the POI to an activity, like a museum to a tourism activity and gym to a sports activity.

3.1. *Step 1: Trajectory Classification.*

The aim of this step is to incrementally classify the continuous users' positions into different kinds of activities. The most recent sequence of GPS collection is stored in a temporal window called TW.

Definition 1. GPS collection is an assembly of GPS points $P = \{p_1, p_2, \ldots, p_n\}$. Each GPS point $p_i \in P$ contains a latitude (p_i.lat), a longitude (p_i.Lngt), a timestamp ($p_i.T$), a speed ($p_i.S$), and a bearing ($p_i.B$). We add to this information the variance of bearing ($p_i.V$) of the last L_{TW} GPS points where L_{TW} is the size of our temporal window TW and the weight ($p_i.W$) represents the importance of the point p_i according to the time generation.

Definition 2. Temporal window TW is a subgroup of a GPS collection with a variable length L_{TW}. In fact TW contains all p_i with not null weight $p_i.W$ (see Figure 3 that represents the relation between a GPS collection and TW; every point in the GPS collection is a record row from the database).

Once new GPS data is received, we achieve three parallel processes like below.

3.1.1. *Process 1: Classification.*

At the arrival of new GPS data, p_n is stored in TW. Classification process is not launched on every data arrival but after a threshold called $T_{\min}$ that will be exposed in process 3. We classify p_i in TW using online K-means; we consider two variables, speed $p_i.S$ and variance of bearing $p_i.V$.

The variance of bearing $p_i.V$ is calculated using this formula: $p_i.V = \sum(p_i - \overline{p_i})/l$, where $\overline{p_i} = \sum p_i/L_{\text{TW}}$; this calculation is made before recording the new p_i in TW and it represents the variance of user's orientation in the last L_{TW} p_i.

Stops behaviors are characterized by a very low $p_i.V$ and $p_i.S$; however, moves behaviors are characterized by high $p_i.S$ and low $p_i.V$ because a person's movements using transportation tend to be in a quick and straight manner. Moving activities are branded by a low $p_i.S$ and high $p_i.V$ (see Figure 4) since these activities are pedestrian actions which generally require a frequent shift of orientation like visiting a museum, shopping in a mall, or walking in a zoo. As said previously, these three types of clusters represent three families of activities and not three activities, each family containing a set of activities depending on the user's visited places.

Note that it is known that when a user stops somewhere, his positions during this stop may vary in the surroundings of the stop due to the positioning error of the tracking system. Consequently, in our algorithm, we automatically put

Location ID	Latitude	Longitude	Time	Speed	Bearing	Variance	Weight
601	48.4210453	−71.0572413	Mon Nov10..	0.74	76.5	49.7	0.61
602	48.4210453	−71.0572532	Mon Nov10..	0.79	76.5	88.36	0.73
603	48.4210453	−71.0572567	Mon Nov10..	0.24	76.5	115.7	1.0

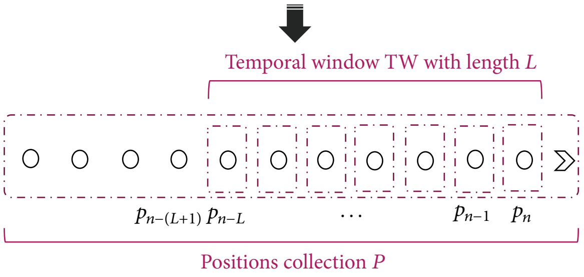


Figure 3: The relation between the GPS collection P and TW.

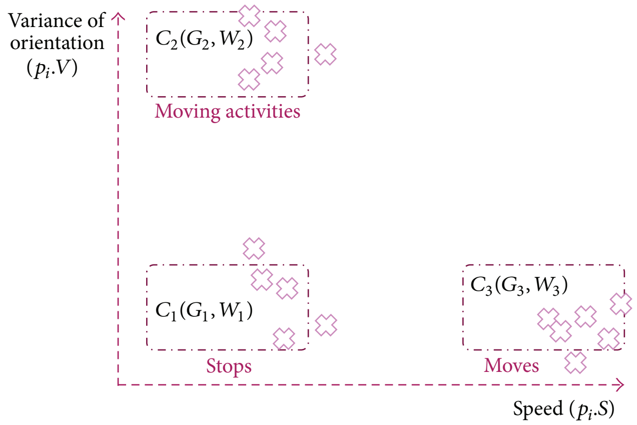


Figure 4: Inferring activity type using speed and the variance of orientation.

the bearing $p_i.B \rightarrow 0$ when we detect that the speed $p_i.S \rightarrow 0$ to avoid any error linked to this situation.

3.1.2. Process 2: Distribution of Weights $p_i.W$. Every p_i has a weight which determines the degree of resemblance of p_i's class to the current activity. Our methodology is inspired from the work in [18]; the weight of each data point decreases exponentially with the time t via a fading function $w_i = f(t) = 2^{-\lambda t}$, where $\lambda > 0$. The exponentially fading function is widely used in temporal applications where it is desirable to gradually discount the history of the past behaviors. The parameter λ is called exponential decay constant; the higher the value of λ, the lower the importance of the historical data compared to more recent data. The overall weight of the data stream is a constant $W = \sum_{t=0}^{t=t_c} 2^{-\lambda t} = 1/(1 - 2^{-\lambda})$ where t_c ($t_c \rightarrow \infty$) is the current time. λ can also be seen as the determining parameter of TW's length. When λ approaches 1, TW shrinks to its smallest size. Inversely, when λ approaches 0, TW spreads to its maximum size (see Figure 5).

Unlike [18], we chose a variable value of λ between 0 and 1 depending on two parameters, the remaining battery level "β" and the disorder of data "E" that already exists in TW (class of every p_i); the disorder is introduced in order to have an idea of the user instant state; for instance, if we have enough information on the user instant activity, there is no need to use a lot of data since the decision about the instant activity is already made; however, when the user state is difficult to be recognized, we will need more data to be able to detect the user activity; this ability to make a decision is called "disorder of data." Thus, these parameters are defined in the following definitions.

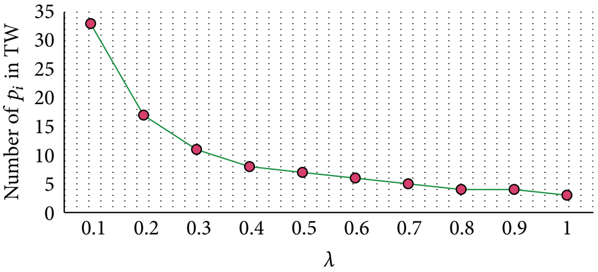


Figure 5: The impact of varying λ on TW's length.

Definition 3. The remaining battery level is represented by $\beta = \beta_r/\beta_{\text{th}}$, where β_r is the real remaining battery level. β_{th} represents a battery threshold specified by the user from which the algorithm starts to minimize calculation (see Figure 6). This user-defined parameter adds some flexibility to the application. For instance, if the user prefers to keep good precision even if it drains the mobile battery, β_{th} should take a small value (e.g., 10%). Conversely, when the user aims for a good preservation of the mobile battery, β_{th} should be given a higher value of around 90% or 100%. Again, when the user knows that he will not spend much time outside, he can adjust β_{th} to a small value to promote the precision and vice versa.

Definition 4. The disorder of data E represents the quality of TW's data that has a link with the ability to make a decision. When one is sure that an activity is performed, one needs less

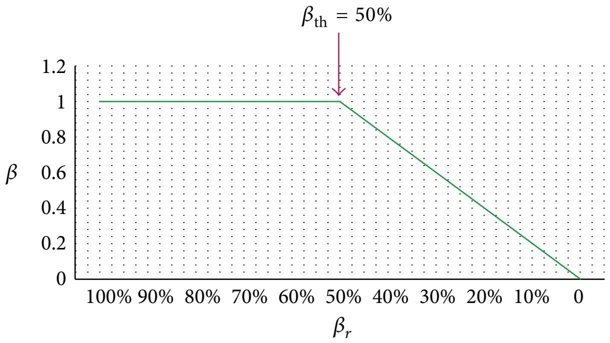


Figure 6: An example of β variation using a threshold $\beta_{\text{th}} = 50\%$.

data to make a decision so $E \rightarrow 1$, and $E \rightarrow 0$ when one has problems finding out what type of activity is executed. In other terms, when the user behavior is unpredictable (because the data in TW is heterogeneous) one says that there is a disorder in his data ($E \rightarrow 1$). Contrariwise, when one sees that the user behavior is stable, one says that there is a stability in the user data ($E \rightarrow 0$).

The disorder of data is often calculated using the entropy, which is a measure of unpredictability of information content, or, in other terms, it measures the homogeneity in a set of information. Consequently, E is calculated using the entropy of Shannon [19]; E of the new p_n is calculated using the entropy of the old data set in TW. The value of E is calculated as follows:

$$E = 1 - H_2\left(p_i\right) = 1 - \sum_{i=0}^{n-1} p_i \log_2 p_i, \qquad (1)$$

where $p_i = w_i/W$.

After introducing the remaining battery level β and the data disorder E, we propose a relation between the two parameters to calculate λ; we put

$$\lambda = 1 - \beta + E\beta. \qquad (2)$$

The equation demonstration is shown below.

Demonstration. Our reflection starts from the following logical rules:

(1) We shrink the TW's length to reduce the number of clustered points (in order to reduce battery consumption) when the battery is low or when we have some certitude about the user's activity, in other terms, stability in the user's behavior characterized by a low disorder of data.

(2) We spread the length of TW when we need numerous position points to make a decision (high disorder in the user's TW data) as long as we have enough battery level.

Table 1: Mathematical representation of the user, battery, and TW states.

Characteristic	Interpretation
Battery high	$\beta \rightarrow 100\%$
Battery low	$\beta \rightarrow 0\%$
High disorder	$E \rightarrow 0$
Low disorder	$E \rightarrow 1$
TW spreads	$\lambda \rightarrow 0$
TW shrinks	$\lambda \rightarrow 1$

Table 2: Parsing β, E, and λ to Boolean parameters.

Interpretation	Boolean parameters
$\beta \rightarrow 100\%$	β true
$\beta \rightarrow 0\%$	β false
$E \rightarrow 1$	E true
$E \rightarrow 0$	E false
$\lambda \rightarrow 1$	λ true
$\lambda \rightarrow 0$	λ false

Table 3: The truth table of β, E, and λ.

β	E	λ
True	False	False
True	True	True
False	True	True
False	False	True

These conditions are sited as follows:

$$\begin{aligned} \text{Battery high} + \text{high disorder} &\longrightarrow \text{TW spreads}, \\ \text{Battery high} + \text{low disorder} &\longrightarrow \text{TW shrinks}, \\ \text{Battery low} + \text{high disorder} &\longrightarrow \text{TW shrinks}, \\ \text{Battery low} + \text{low disorder} &\longrightarrow \text{TW shrinks}. \end{aligned} \qquad (3)$$

If we parse the characteristics mentioned in (3) into a mathematical representation using the relations presented in Table 1, then, (3) will be written as follows:

$$\begin{aligned} \beta &\longrightarrow 100\% + E \longrightarrow 0 \Longrightarrow \lambda \longrightarrow 0, \\ \beta &\longrightarrow 100\% + E \longrightarrow 1 \Longrightarrow \lambda \longrightarrow 1, \\ \beta &\longrightarrow 0\% + E \longrightarrow 0 \Longrightarrow \lambda \longrightarrow 1, \\ \beta &\longrightarrow 0\% + E \longrightarrow 1 \Longrightarrow \lambda \longrightarrow 1. \end{aligned} \qquad (4)$$

Let β, E, and λ be Boolean parameters that take values as mentioned in Table 2.

The logical rules in (4) can be illustrated in the truth table, Table 3.

We notice that the truth table, Table 3, is that one of the Boolean implication function, where

$$\lambda \equiv F\left(\beta, E\right) \equiv \beta \wedge \neg E \equiv \beta \xrightarrow[\text{boolean}]{} E. \qquad (5)$$

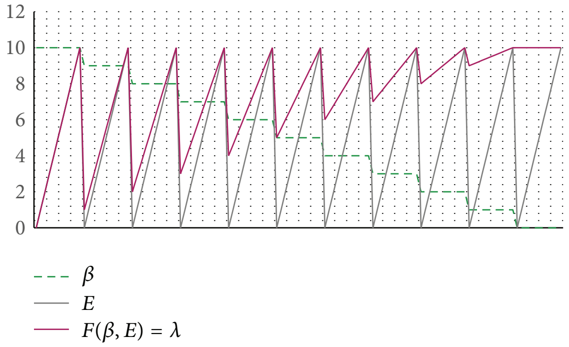


FIGURE 7: λ variation on the basis of β and E.

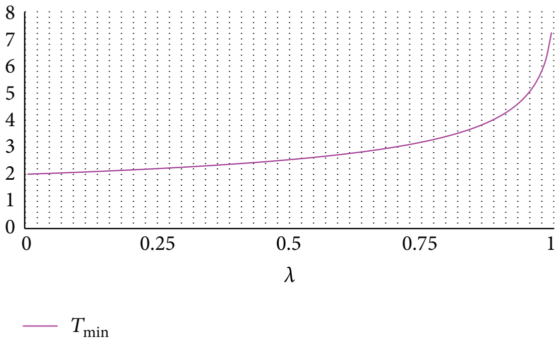


FIGURE 8: The relation between λ and $T_{\min}$.

In Boolean logic, the truth values of variables may only be 0 or 1; however, our parameters β, E, and λ take continuous values between 0 and 1. For instance, $\beta = 1$ means full battery and $\beta = 0$ means empty battery. Yet, in real life, there are many states between full and empty states. As such, the fuzzy logic was introduced [20] which is a form of many-valued logic in which the truth values of variables may be any real number between 0 and 1. Fuzzy logic has been extended to handle the concept of partial truth, where the truth value may range between completely true and completely false. In this context, our equation is called fuzzy implication:

$$\lambda \equiv F(\beta, E) \equiv \beta \xrightarrow[\text{fuzzy}]{} E. \tag{6}$$

Using the inequality of Reichenbach [21], the fuzzy implication can be written as

$$\lambda = F(\beta, E) = 1 - \beta + \beta E. \tag{7}$$

This equation has been tested in Figure 7 where we varied the two parameters β and E to observe the sensitivity of λ (the length of TW) to these changes.

In Figure 7, we notice that the sensitivity of TW's size (the value of λ) is linked to the battery level β and disorder E; however, when β begins to drop, the size of TW continuously ignores the disorder E until it reaches a total ignorance when $\beta = 0$ (see Figure 7 when $\beta = 0$).

After having distributed the weights of each p_i in TW and calculated the size of TW on the basis of the battery level β and disorder E, it is time to see if the user is performing some interesting activity.

3.1.3. Process 3: Cluster's Accumulation Search. The algorithm recognizes that someone is doing an activity if the weight of its cluster W_j exceeds a value μ, where $\mu = W/K$ with K representing the number of clusters (activities) used by K-means to classify TW; in our case $K = 3$ because we are trying to identify three families of clusters: stops; moves; and moving activities.

The most important question is "when do we search for a cluster accumulation?" To minimize the use of device resources, it is recommended to handle this step carefully. The research process is not launched on every data arrival T but after a time called $T_{\min}$ in which it is expected to have an activity.

Proposition 5. *$T_{\min}$ is the time from which $w_i = f(t) = 2^{-\lambda t}$ reaches μ; this is verified in this following condition $2^{-\lambda T_{\min}}\mu + 1 = \mu$; after development $T_{\min} = (1/\lambda)\log(\mu/(\mu - 1))$, where $\lambda = 1 - \beta + \beta E$.*

Consequently, on every $T_{\min}$ we check if there is any activity that its weight W_j exceeds μ; if found, we summarize the points p_i to one point $C_j(G_j, W_j)$, where $W_j = \sum_{i=0}^{n} w_{ij}$ and $G_j = \sum p_{ij}/n$, n is the number of points in this cluster, and p_{ij} represents the points p_i in the cluster j. After that we move to the spatial recognition of the summarized point C_j.

As said previously, λ is not static. It varies between 0 and 1 depending on the disorder of data in TW and the remaining battery level. From the relation between λ and $T_{\min}$ in Figure 8, we note that $T_{\min}$ is also affected by these parameters. When λ is near 1, TW will contain a minimum number of position points (see Figure 5) for the reason that the battery is low or there is some stability in the user's behavior (e.g., staying at home); in this case, there is no need to process the calculations on every step; otherwise, this useless calculation depletes the battery. Consequently, $T_{\min}$ will take a maximum value (see Figure 8).

Conversely, when λ is near 0, TW will contain a maximum number of position points (see Figure 5) for the reason that the battery is well charged and there is a big disorder in TW which troubles the users' activity identification. In this case the algorithm will process the calculations on every position-point arrival to quickly determine which type of activity the user is performing. Accordingly, $T_{\min}$ will take a small value (see Figure 8).

Algorithm 1 achieves two processes; the first is to store every GPS data p_i when it arrives; the second is performed every $T_{\min}$ to reduce the calculation. First step is to classify every point p_i in TW and then update the value of λ depending on the disorder of the activities types in TW and the remaining battery level. After calculating the center of gravity C_j of each cluster and the threshold μ, we start to search for an accumulation of a cluster that is verified by the condition $\max(W_j) > \mu$; we use the max of clusters' weight to avoid the case where two clusters exceed μ in the same time; if condition is verified we start to recognize the geographic environment of C_j. Finally, we update the value of $T_{\min}$ that will determine the next repetition of the process.

```
Input:
A GPS point $p_i$;
Output:
The activity of the person;
For each $T$
    Store $p_i$ in TW;
End for each
For each $T_{\min}$
    Classify every point in TW;
    Update $\lambda$;
    Update the centers of clusters $C_j(G_j, W_j)$;
    Calculate the threshold $\mu$;
    //Cluster's accumulation search
    If (max $(W_j) > \mu$) then
        POI = spatial recognition $(G_j)$;
        Activity = activity discovery (POI);
    End if
    Update $T_{\min}$;
    Return Activity
End for each
```

Algorithm 1: Trajectory classification.

After searching for a cluster's accumulation, we move to Step 2, spatial recognition of C_j.

3.2. Step 2: Spatial Recognition. This task aims further to understand the movement behavior of users, in terms of more semantically meaningful POI. In this step, we are going to transform a cluster of GPS points to an expressive human activity. First we are going to search for the nearest geographic entities to this cluster; thereafter, we are going to associate an activity type to the entity found.

We search for the closest and the most significant geographical feature compared to C_j; it is performed using a spatial query in a spatial database. Our database is powered by OSM (OpenStreetMap (OSM) is a community-driven project aimed at producing a high-quality detailed map of the world; OSM datasets are freely available under the Open Database license terms) and stored in the users' device for local use. This technique aims to discard networks use by offering offline services that will save a user's money and a phone's battery.

In fact, many mobile GIS solutions [22, 23] offer an offline version to avoid the constraints linked to the use of networks. For instance, Esri company [24] (one of the world's biggest GIS companies) has provided a whole runtime called "ArcGIS Runtime mobile SDK" [25] to support offline mapping; it includes map viewing, interaction, editing, and routing while fully disconnected from wireless.

Consequently, offline processing of geographic data has become easier. However, the limit of this technique is the size of the stored data, since geographic data may cause a massive use of the storage capacity; for instance, the size of the OSM planet database is over 666 GB [26]. The usual solution to face this problem is to download a specific database to each user depending on the cities where they live (e.g., the size of the whole New York region is only 140 MB [27] including all geodata types such as roads, highways, buildings, and public parks); this can be done manually by the user, since he can define his area of interest while being online by choosing the area to download and panning/zooming to the area of capture. The second way is to download the geographic database automatically by the application by detecting the user's daily staying area; for instance, if the application detects that the user is living in Montreal city, it will wait until being online to download the OSM data of Montreal and its surroundings.

Many techniques to extract geographic data from OSM exist; the easiest way is to download and extract it from the OSM website. There are various web services that provide data extracts for a geographic area. For example, GeoFabrik is a company which specializes in working with OpenStreetMap. They provide a variety of free extracts in Shapefile (the Shapefile format is a geospatial vector data format for geographic information system (GIS) software) and raw OSM format on their download website (http://download.geofabrik.de/). After downloading raw OSM data and storing it in a spatial database, we use Algorithm 2 to search for the nearest spatial feature to C_j.

The querying process to search for the closest geographic entity is a famous technique based on identifying a proximity buffer; proximity analysis determines the relationship between selected geographic elements by identifying the locations of other elements within a specified distance (50 m in our case). Creating buffer zone regions is the most common method used in proximity analysis; for more details about the querying method please refer to the paper described in [28].

If the spatial recognition step finds a geographic entity in the neighborhood of the positions' cluster we can declare that the user has visited this place.

3.3. Step 3: Activity Discovery. Activity discovery is based on the exploitation of tags in OSM. OpenStreetMap data adhere to a simple XML schema with three basic elements: (i) nodes, that is, single geospatial points; (ii) ways, intended as ordered sequences of nodes; (iii) relations, grouping multiple nodes and/or ways. Each element includes a unique identification code, latitude and longitude coordinates, versioning information, and optional general-purpose informative tags. A tag is a key-value pair of Unicode strings of up to 255 characters. Each tag describes a specific feature of a spatial data element (see [29] for more details). This step permits having a human meaningful semantic information about the user visited place.

Tags are written in OSM documentation as key = value. The key describes a broad class of features (e.g., highways or names). The value details the specific feature that was generally classified by the key; for example, the geographic entity that contains a tag "building = apartment" represents a building arranged into individual dwellings, often on separate floors.

We assume that each tag represents an activity and each activity belongs to an activity family (see Figure 9).

Activities are organized in a taxonomy which generalizes the kinds of activities of interest for the movement analysis.

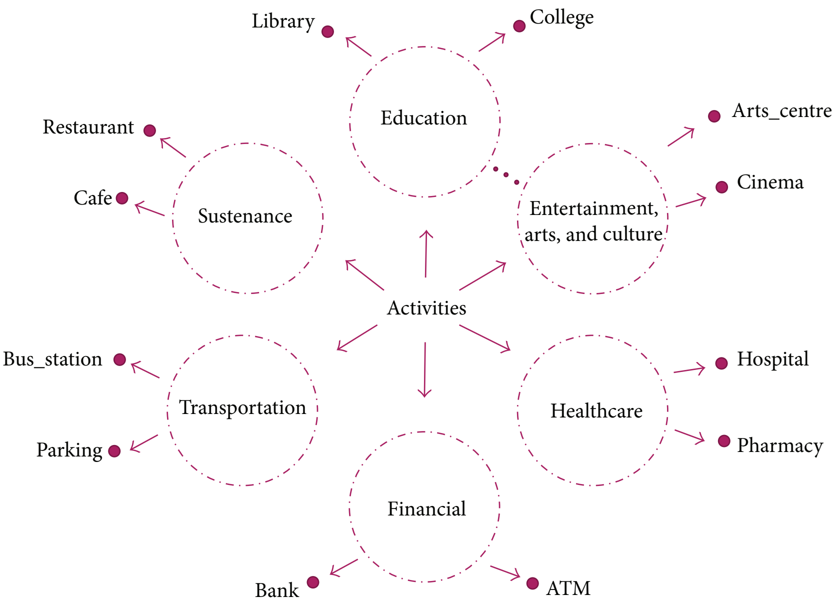


Figure 9: The taxonomy of activities.

```
Input:
A center of clusters $C_j$;
Output:
POI, Date;
If (there are some geographic entities in the neighborhood) then
    //search in the geographic database
    POI = the nearest geographic entity;
    Date = Get current date();
Else
    POI = null;
End else
Return POI, Date
```

Algorithm 2: Spatial recognition.

For example, the "going to college" activity can be specialized in "education." Subsequently, for example, if the nearest geographic entity to C_i has a tag "amenity = library" we deduce that the user is doing an educational activity in the library. This is performed using an algorithm (see Algorithm 3) that searches for the nearest geographic entity and extracts its tags to deduce the activity achieved. This taxonomy is also used to mine quickly users' past activities; for instance, if we search for the public transportation taken by the user, we just have to specify the activity type as "Transportation" in the SQL request and search for the transportation mode (bus station, taxi station, etc.).

Depending on the application requirements, the activity discovery process can extract further information about the visited POI, such as the address, the number of floors, the opening hours, and the building height. This semantic information permits enhancing the way users perceive and interact with their surroundings; for instance, authors in [30] proposed a system for a mobile navigation assistant fit for the purpose of moving within a complex built space. Their process of semantic-enhanced POI discovery is based on the exploitation of OSM tags, the same as our method; the results are used in an augmented indoor/outdoor navigation system.

4. Experimental Evaluation

In order to test our approach, we will divide the experimentation section into two parts: first, we will test the accuracy of our approach using the Family Coordination dataset [8] by comparing it with CB-SMOT method; then, we will test our approach's ability to save battery life by comparing our solution to LifeMap application described in [31].

4.1. Family Coordination Test

4.1.1. Dataset. We used the dataset described in [8] where researchers have conducted experiments in order to reveal the underlying causes of coordination breakdowns that a

```
Input:
POI;
Output:
Activity;
If (POI = null) then
    Activity = "no activity";
Else
    Tag = Get the tag of POI;
    Activity = Search for activity taxonomy (Tag);
End else
Return Activity
```

Algorithm 3: Activity discovery.

Figure 10: An example of our Spatialite database that contains the geographic entities of Pittsburgh.

routine learning system might be able to address. They tracked GPS locations of the members of six families during six months in Pittsburgh City in Pennsylvania, USA (see Figure 10); researchers made an effort to recruit a wide cross section, selecting families from a variety of ethnic and economic backgrounds, as well as expressing a variety of planning styles, child-rearing models, and transportation preferences. The GPS sampling for every family member including children (a total of 26 persons) was set at one-minute intervals which led to gathering more than two billion and half of GPS points. Moreover, every night during the study, a member of the research team would call the families and interview each parent about that day's management of their kids' activities. In preparation for the interviews, family members were asked to input their daily activities into a web-based survey. Researchers then used the survey to scaffold the phone interview, probing and documenting the overall family logistical plan at each point throughout the day. We used this information to compare the inferred activities to the real executed activities.

4.1.2. Test Scenario. We compared our solution to CB-SMOT algorithm [10] described in the related works; CB-SMOT is widely used to extract visited places from locations' trajectories; we have implemented this algorithm using Weka-STPM platform [32], an extension that adds trajectory processing tools to Weka (Waikato Environment for Knowledge Analysis) and includes CB-SMOT as a clustering algorithm; however, since Weka-STPM is a desktop solution, we have created a mobile version using the CB-SMOT class included in Weka-STPM. The input was the Family Coordination dataset parsed into a Postgresql/PostGIS database and the output was the inferred semantic trajectories (tracking visited places retrieved using Weka-STPM).

Our solution was implemented using Huawei P7 android phone where we have deployed our application that contains the online recognition algorithm, Family Coordination dataset, and a SQLite database that contains the geographic entities of Pittsburgh city needed in the spatial recognition process. Note that, as said previously, this first part of the experimentation is dedicated to testing our approach in terms of accuracy; consequently we will suppose that the phone's battery is fully charged during all the processes principally because we do not have access to the battery's life data in Family Coordination dataset. The ability to save the phone's battery and its impact on the accuracy will be presented in the second part.

4.1.3. Results. Results presented in Table 4 represent the comparison of our approach with three versions of CB-SMOT algorithm, by varying its MinTime parameter each time; it was set to 60 s in version 1, to 180 s in version 2, and to 500 s in version 3.

We have tested 10525 activities gathered from the activities of the 24 members of families.

Correct activities represent the number of activities recognized successfully, missed activities represent the number of activities that the users did but the algorithms have failed to recognize, and false activities represent the number of meaningless discovered activities like recognizing the stop of a car in a traffic jam as going to gas stations. The accuracy and the error are calculated as follows:

$$\begin{aligned} \text{Accuracy} &= \frac{\text{Correct}}{\text{number of tested activities}}, \\ \text{Error} &= \frac{\text{missed} + \text{false}}{\text{number of tested activities}}. \end{aligned} \tag{8}$$

Globally, our algorithm behaves better than CB-SMOT in terms of accuracy and error. We have recorded the accuracy of 78% in our solution and 61% for the average accuracy of the three versions of CB-SMOT. On the other hand, our error was around 22%, while the average error of the three versions of

Table 4: Comparison of our approach to CB-SMOT algorithm.

	Our approach	CB-SMOT V1 MinTime = 90 s	CB-SMOT V2 MinTime = 180 s	CB-SMOT V2 MinTime = 500 s
Tested activities	10525	10525	10525	10525
Correct	8313	4028	7157	8257
Missed	1812	6497	3368	2268
False	578	181	909	2789
Accuracy	78%	38%	68%	78%
Error	22%	63%	40%	48%

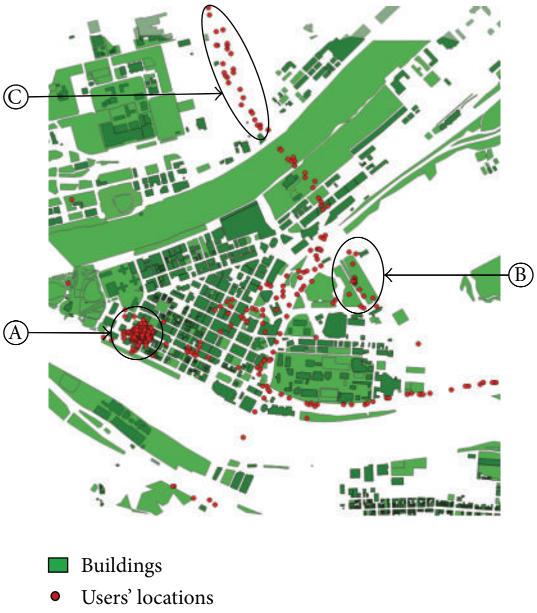


Figure 11: An example of one user's GPS locations projected on the geographic entities layer of Pittsburgh; (A) stationary activity, (B) moving activity, and (C) moves.

CB-SMOT was around 50%; these results are explained in the following points:

(i) Our algorithm succeeds in recognizing more activities because it supports the recognition of moving activities; contrariwise CB-SMOT recognizes only stop and move activities (see Figure 11).

(ii) CB-SMOT shows a higher error because of the MinTime threshold that is set manually (the time of staying from which the algorithm considers that the user has visited this place). Setting this threshold to a small value will increase the number of false alarms (see CB-SMOT V1 in Table 4), while setting it to high value will increase the number of missed activities (see CB-SMOT V2 in Table 4), and, in both cases, the error will be increased since it is calculated on the basis of the number of false and missed activities.

After having tested the accuracy of our approach, we will go further in the experimentation of our battery saving technique.

4.2. LifeMap Test

4.2.1. Dataset. Researchers in LifeMap project collected real traces from 68 persons over four weeks using HTC Hero, HTC Desire, and Samsung Galaxy S smartphones. The tracking application (called LifeMap) was running as a background service to automatically collect the user's mobility and to trace sensor usage time. To collect the ground truth, the participants explicitly labeled the place names and kept a diary of places they had visited with the entrance and departure times. Moreover, the advantage of using such dataset is the ability to compare the power consumption of our method to the authors' one, since authors tracked the battery status during all the experimentation process.

4.2.2. Test Scenario. In this step we will compare our approach to the LifeMap application used to recognize users' motility; the project can be found in [2], the LifeMap dataset can be found in [33], and the LifeMap mobile application can be found on android play store.

We used the LifeMap dataset to test our battery-aware approach; to do so, we developed an android application that is fed by LifeMap datasets; the main idea is to make it out as if the users have moved holding our application in their phones; the application recuperates the GPS coordinates one by one and processes each point using our online approach; after that we compare the battery consumption of our application with LifeMap application's one (see Figure 12).

LifeMap application uses a set of sensors to recognize users' activities; these sensors include GPS, accelerometer, digital compass, and Wi-Fi; the application uses a combination of these sensors to retrieve the activity performed by the user; it is clear that the use of all these sensors represents a major source of power consumption. Consequently, in order to bring an objective comparison between the LifeMap application and our application, it is logical that we have to include the same sensors even if we use only the GPS sensor. Indeed, we have enriched our application with the same sensors used in LifeMap application but the processing of this information is made under the rules defined by our battery-aware approach; for example, the processing of data is made on the estimated $T_{\min}$ that is calculated on the basis of the remaining battery level and the user's behavior.

Moreover, even if the GPS is activated, the application consumes more resources when it requests the users' position

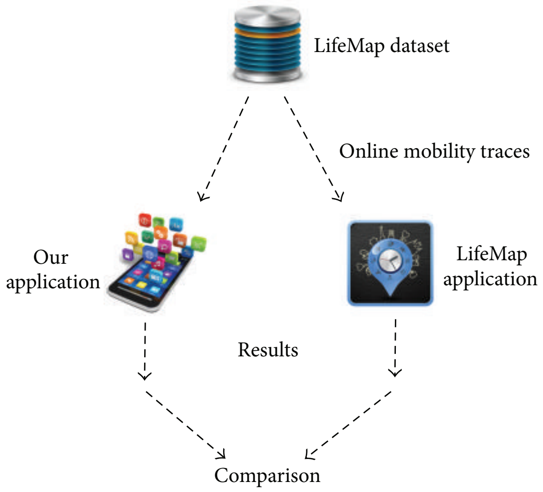


Figure 12: Test scenario to compare our solution with LifeMap application using LifeMap dataset.

from the GPS satellites (the case of LifeMap) than when it retrieves it from a database (the case of our application); consequently, in order to compare objectively the two approaches, we continue to obtain the user position from the database; however, on every point retrieved, we request the GPS satellite for a GPS position. Surely, this position is useless but it enables consuming the same resources used by LifeMap application to get a GPS position.

In order to automatically recognize users' activities, we have constructed a Spatialite geodatabase [34] where we stored the background geographic data needed in our spatial recognition (see Figure 13). Note that users' locations are recorded on different frequencies; authors in LifeMap dataset have linked the activity declared by the user to the sampling frequency in order to minimize the size of the database and the power consumption, which has sometimes led to low frequencies (e.g., every 10 minutes). In our approach, we try to recognize accurately users' activities while protecting their phones' battery; to test this service, we need to increase the sampling frequency to see if our work well behaves using a lot of location points. Consequently, we have implemented an interpolation algorithm that estimates the missing locations when the sampling frequency is high (more than one minute).

Our application was deployed on Samsung Galaxy S smartphone, one of the smartphones' models used by LifeMap application to recognize activity.

After having filled the necessities regarding the comparison between the two approaches, we will present the results of both of them.

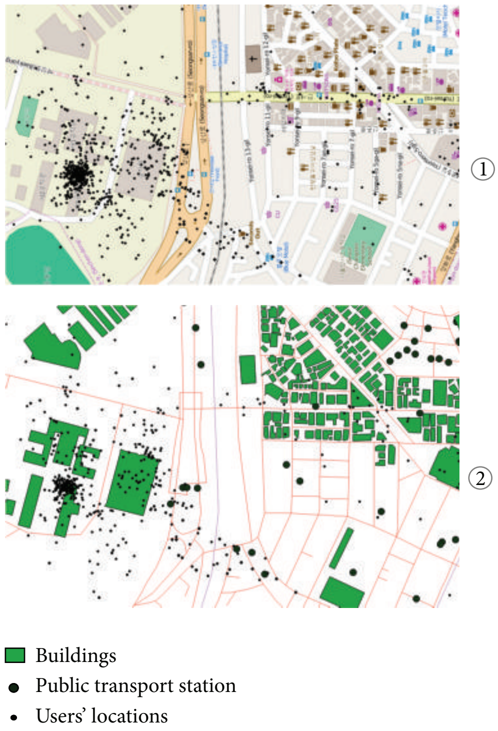


Figure 13: An example of one user's trajectory projected on (1) OpenStreetMap raster layer and (2) vector layer retrieved from our geodatabase.

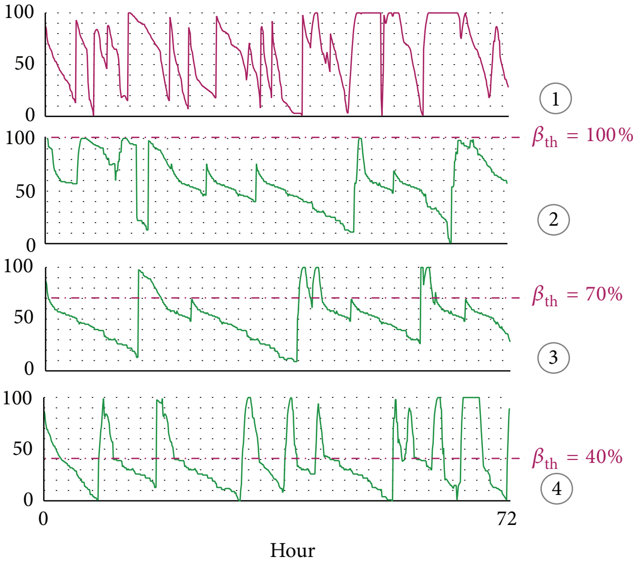


Figure 14: Results comparison between LifeMap and our solution for 72 hours of activity recognition. (1) LifeMap, (2) our solution using β_{th} = 100%, (3) our solution using β_{th} = 70%, and (4) our solution using β_{th} = 40%.

4.2.3. Test Results. The total number of recorded hours of battery status in LifeMap dataset is 48900 hours; in order to experiment on all the hours using one smartphone, we will need over five years of experimentations. Accordingly, we chose to use one random day of each user (rather than 30 days) using five smartphones; the total number of hours compared is 1632 hours. Due to insufficient space, we present in Figure 14 the tracking of one user's battery life for 72 hours using LifeMap and three versions of our approach where we vary each time the value of the threshold β_{th} from where the application starts to save battery life.

Table 5: The impact of varying β_{th} on the accuracy.

	$\beta_{\text{th}} = 100\%$	$\beta_{\text{th}} = 70\%$	$\beta_{\text{th}} = 40\%$	LifeMap
Accuracy	68.7%	77.1%	85.4%	78%

Our approach shows an interesting battery saving capacity. Globally, it consumes less resources than LifeMap even when β_{th} is set to a low level (40%).

We notice that, in our approach, battery consumption varies on the basis of β_{th}. When it is set to 100%, our algorithm starts to save the battery from the first moments, which explains the long battery life noticed in graph (2) of Figure 14. However, when β_{th} is set to 40%, we notice that the life of the battery (graph (4)) looks a bit like LifeMap's one (graph (1)); in this case, the algorithm consumption behaves like LifeMap between 100% and 40%, but when it falls under 40%, the algorithm starts saving batteries (see the horizontal dotted line in graph (4)).

Thus, β_{th} has an impact on the resource consumption, but what about the accuracy? Is it affected by the change of β_{th}? To answer these questions, we have tracked the accuracy of our algorithm when varying β_{th}; results are presented in Table 5.

The authors of LifeMap reported in [35] that the accuracy is around 78%. In our case, when β_{th} is set to a value less than 70%, the accuracy of our approach exceeds the LifeMap's one (see Table 5); this is justified by the fact that the LifeMap's technique for detecting important places is based on the time spent by the user around the POI. When a user stays at a given location for more than 10 minutes, the user state is considered stationary and the place is labeled as a POI. This technique fails to recognize moving activities that require a movement to be executed; furthermore, using a fixed time threshold leads to missing some short activities or falsely detecting some activities when the time threshold is set to a small value.

Varying β_{th} has an impact on the accuracy; the more the β_{th} is set to high value, the more the accuracy drops; this is justified by the fact that when β_{th} takes a high value, the temporal window TW shrinks to take less examples and the next processing time T_{min} will be set further. So if we try to save the battery, we will automatically lose a little bit of accuracy, but is there a compromise between β_{th} and the accuracy? Is it possible to find a β_{th} value that saves as much as possible the accuracy and the battery at the same time?

We have tracked the accuracy of one user's data from LifeMap dataset when varying slowly β_{th}; results are presented in Figure 15 where the accuracy and an estimation of battery life are presented on the basis of β_{th}. We notice that the higher the value of β_{th} is, the more we save resources and lose precision.

In order to find an optimal solution that enables saving both battery life and accuracy, we have to resolve the optimization problem that maximizes the battery life function $L(\beta_{\text{th}})$ and the accuracy function $A(\beta_{\text{th}})$ at the same time; this is a multiobjective optimization problem where we try to optimize two different functions; one of the existing solutions is the weighted sum method [36]; the solution is based on selecting a scalar weight s_i and maximizing the following composite objective function:

$$U = s_1 * L(\beta_{\text{th}}) + s_2 * A(\beta_{\text{th}}). \tag{9}$$

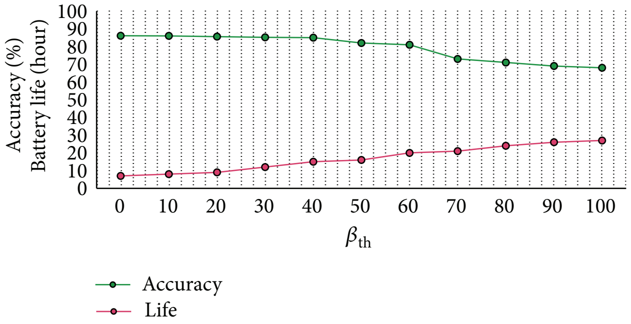


Figure 15: The impact of varying β_{th} on the accuracy.

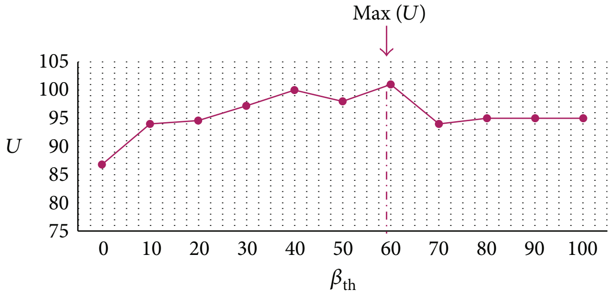


Figure 16: Optimization of β_{th} to maximize the accuracy and the battery life.

In our case the importance of the accuracy and the battery life are equal, so the weights of the two variables s_1 and s_2 are equal too ($s_1 = s_2 = 1$); after addition, the value of β_{th} that maximizes U is $\beta_{\text{th}} = 60\%$ (see Figure 16). Thus, if we need to maximize the accuracy and the battery life at the same time, β_{th} should be set around 60%.

The average precision of our approach when varying β_{th} from 0% to 100% is 79% (see Figure 15); for a system that proposes an activity recognition system keeping a long lifetime of the battery, we think that these results are promising.

The threshold β_{th} from which the system starts to save the phone's battery can be set automatically depending on the activity performed. Indeed, after having learned users' habits, we can link β_{th} to the predicted activity; for instance, when we predict that the user is going to spend a small time outdoors, we set β_{th} to promote the accuracy and vice versa.

5. Conclusions and Future Works

In this paper we proposed a new battery-aware technique for extracting semantically and incrementally important geographical locations from users' moves. We associate the places visited by individuals during their movements to meaningful human activities using an algorithm that clusters incrementally user's moves into different types of activities using two parameters, speed and the variance of the moves orientation. After detecting an important accumulation in a cluster's weight, the cluster is summarized into one point and another process will be launched that aims to search the most meaningful geographic entity near this point (POI); when found, we associate a semantic activity to it. Aiming to save phone batteries, our algorithm was implemented to change its computational complexity on the basis of the remaining battery level and the users' behaviors.

Our approach has been experimented in a real case study to test the accuracy of our recognition mechanism and to observe the impact of our technique on phone's resources. These tests demonstrate that, with a minimum of information, our proposals are capable of online recognizing a person's activities without depleting the phone resources.

Several promising directions for future works exist. First is the enhancement of spatial recognition process with the introduction of probability to assign a cluster to a geographic entity; this probability approach can take advantage of previously recognized activities; for example, a person doing tourist activities all day has more probability to finish his day in a restaurant or in a hotel than in other places. The second enhancement that can be applied to our inferring process is users' profiling. In fact move's pattern of individuals varies between young and old, healthy and sick, and male and female. The implementation of such reflections will improve the accuracy of our activity recognition process.

Competing Interests

The authors declare that there are no competing interests regarding the publication of this paper.

References

[1] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello, "Extracting places from traces of locations," in *Proceedings of the 2nd ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspot (WMASH '04)*, pp. 110–118, Philadelphia, Pa, USA, September 2004.

[2] *LifeMap: Mobility Monitoring Tool*, 2016, http://sourceforge.net/projects/lifemapandroid/.

[3] A. T. Campbell, S. B. Eisenman, K. Fodor et al., "Transforming the social networking experience with sensing presence from mobile phones," in *Proceedings of the 6th ACM Conference on Embedded Networked Sensor Systems (SenSys '08)*, pp. 367–368, ACM, New York, NY, USA, November 2008.

[4] N. J. Yuan, Y. Zheng, X. Xie, Y. Wang, K. Zheng, and H. Xiong, "Discovering urban functional zones using latent activity trajectories," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 3, pp. 712–725, 2015.

[5] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: a recommender system for finding passengers and vacant taxis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 10, pp. 2390–2403, 2013.

[6] Y. Zheng, "Trajectory data mining: an overview," *ACM Transactions on Intelligent Systems and Technology*, vol. 6, no. 3, article 29, 2015.

[7] Y. Li, Y. Zheng, H. Zhang, and L. Chen, "Traffic prediction in a bike-sharing system," in *Proceedings of the 23rd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '15)*, 2015.

[8] S. Davidoff, J. Zimmerman, and A. K. Dey, "How routine learners can support family coordination," in *Proceedings of the 28th Annual CHI Conference on Human Factors in Computing Systems (CHI '10)*, pp. 2461–2470, Atlanta, Ga, USA, April 2010.

[9] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 140–150, 2010.

[10] A. Tietbohl, V. Bogorny, B. Kuijpers, and L. O. Alvares, "A clustering-based approach for discovering interesting places in trajectories," in *Proceedings of the 23rd Annual ACM Symposium on Applied Computing (SAC '08)*, pp. 863–868, ACM Press, Ceará, Brazil, March 2008.

[11] Y. Takeuchi and M. Sugimoto, "CityVoyager: an outdoor recommendation system based on user location history," in *Ubiquitous Intelligence and Computing: Third International Conference, UIC 2006, Wuhan, China, September 3–6, 2006. Proceedings*, vol. 4159 of *Lecture Notes in Computer Science*, pp. 625–636, Springer, Berlin, Germany, 2006.

[12] L. Spinsanti, F. Celli, and C. Renso, "Where you stop is who you are: understanding peoples activities by places visited," in *Proceedings of the Behaviour Monitoring and Interpretation Workshop (BMI '10)*, 2010.

[13] M. A. Viredaz, L. S. Brakmo, and W. R. Hamburgen, "Energy management on handheld devices," *ACM Queue*, vol. 1, no. 7, pp. 44–52, 2003.

[14] J. P. Pillai and K. G. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems," *ACM SIGOPS Operating Systems Review*, vol. 35, no. 5, pp. 89–102, 2001.

[15] B. Zhao, H. Aydin, and D. Zhu, "Reliability-aware dynamic voltage scaling for energy-constrained real-time embedded systems," in *Proceedings of the 26th IEEE International Conference on Computer Design (ICCD '08)*, pp. 633–639, Lake Tahoe, Calif, USA, October 2008.

[16] E. Shih, P. Bahl, and M. J. Sinclair, "Wake on wireless: an event driven energy saving strategy for battery operated devices," in *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking (MobiCom '02)*, pp. 160–171, Atlanta, Ga, USA, September 2002.

[17] Y. Wang, J. Lin, M. Annavaram et al., "A framework of energy efficient mobile sensing for automatic user state recognition," in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*, pp. 179–192, ACM, June 2009.

[18] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in *Proceedings of the 6th SIAM International Conference on Data Mining*, pp. 328–339, Bethesda, Md, USA, April 2006.

[19] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[20] V. Novák, I. Perfilieva, and J. Močkoř, *Mathematical principles of fuzzy logic*, Kluwer Academic, Dodrecht, The Netherlands, 1999.

[21] H. Reichenbach, *Elements of Symbolic Logic*, Macmillan, New York, NY, USA, 1947.

[22] GIS2GO, Mobile offline GIS app, 2016, http://www.gis2go.com/home-en.html.

[23] GISCloud, Mobile offline GIS app, 2016, http://www.giscloud.com/blog/offline-maps-are-out/.

[24] Esri, Company's History, http://www.esri.com/about-esri#who-we-are.

[25] ArcGIS Runtime SDKs, 2016, https://developers.arcgis.com/arcgis-runtime/.

[26] OpenStreetMap, Planet OSM, 2016, http://wiki.openstreetmap.org/wiki/Planet.osm.

[27] OpenStreetMap extracts, New York region, 2016, http://download.geofabrik.de/north-america/us/new-york.html.

[28] D. O'Sullivan and D. J. Unwin, *Geographic Information Analysis*, John Wiley & Sons, Hoboken, NJ, USA, 2003.

[29] OpenStreetMap, 2016, http://wiki.openstreetmap.org/.

[30] M. Ruta, F. Scioscia, S. Ieva, G. Loseto, and E. Di Sciascio, "Semantic annotation of OpenStreetMap points of interest for mobile discovery and navigation," in *Proceedings of the IEEE 1st International Conference on Mobile Services (MS '12)*, pp. 33–39, Honolulu, Hawaii, USA, June 2012.

[31] J. Chon and H. Cha, "LifeMap: a smartphone-based context provider for location-based services," *IEEE Pervasive Computing*, vol. 10, no. 2, pp. 58–67, 2011.

[32] L. O. Alvares, A. Palma, G. Oliveira, and V. Bogorny, "Weka-STPM: from trajectory samples to semantic trajectories," in *Proceedings of the 11th Workshop de Software Livre (WSL '10)*, pp. 164–169, Porto Alegre, Brazil, 2010.

[33] *LifeMap Dataset*, Yonsei University, 2016, http://www.crawdad.org/yonsei/lifemap/20120103/.

[34] Spatialite Tools, https://www.gaia-gis.it/fossil/spatialite-tools/wiki?name=OSM+tools.

[35] Y. Chon, E. Talipov, H. Shin, and H. Cha, "Mobility prediction-based smartphone energy optimization for everyday location monitoring," in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems (SenSys '11)*, pp. 82–95, ACM, Seattle, Wash, USA, November 2011.

[36] J. Koski, "Multicriteria truss optimization," in *Multicriteria Optimization in Engineering and in the Sciences*, Stadler, Ed., vol. 37 of *Mathematical Concepts and Methods in Science and Engineering*, pp. 263–307, Plenum Press, New York, NY, USA, 1988.

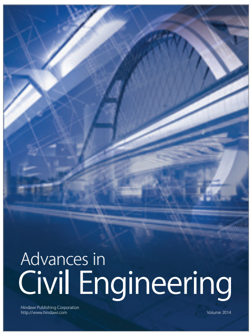
Journal of Engineering

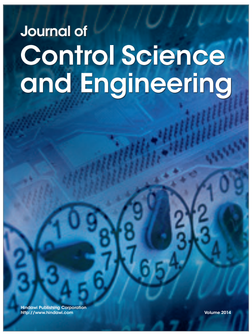
The Scientific World Journal

International Journal of Rotating Machinery

Journal of Sensors

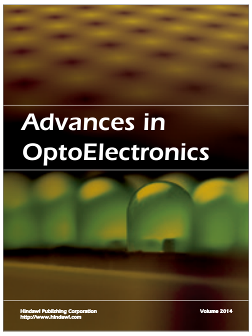
International Journal of Distributed Sensor Networks


Advances in Civil Engineering


Journal of Control Science and Engineering

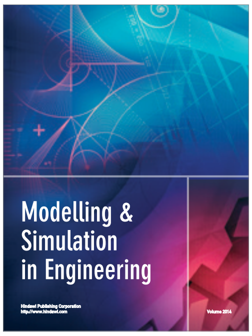
Journal of Robotics

Journal of Electrical and Computer Engineering

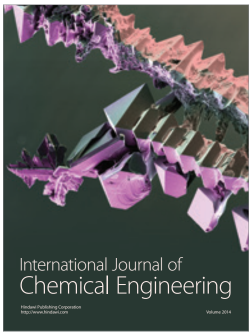

Advances in OptoElectronics

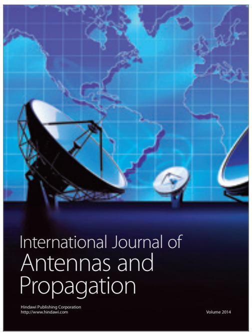
VLSI Design

International Journal of Navigation and Observation

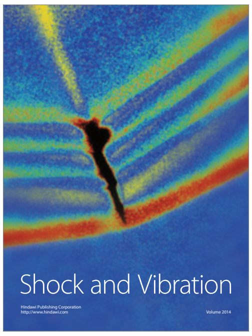

Modelling & Simulation in Engineering

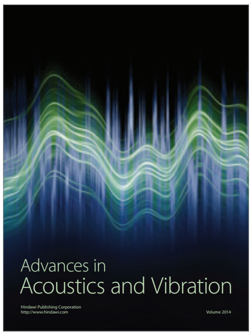
International Journal of Aerospace Engineering


International Journal of Chemical Engineering


International Journal of Antennas and Propagation

Active and Passive Electronic Components


Shock and Vibration


Advances in Acoustics and Vibration