Battery-Aware Mobile Solution for Online Activity Recognition from Users' Movements

Conference Paper · July 2016

DOI: 10.1109/MobServ.2016.16

6 authors, including:

Abdenour Bouzouane
University of Québec in Chicoutimi
108 PUBLICATIONS **535** CITATIONS
SEE PROFILE

Charles Gouin-Vallerand
Télé-université
36 PUBLICATIONS **122** CITATIONS
SEE PROFILE

Sylvain Giroux
Université de Sherbrooke
168 PUBLICATIONS **934** CITATIONS
SEE PROFILE

Bruno Bouchard
124 PUBLICATIONS **546** CITATIONS
SEE PROFILE

Battery-Aware Mobile Solution for Online Activity Recognition from Users' Movements

Mehdi Boukhechba, Abdenour Bouzouane, Sebastien Gaboury
LIARA Laboratory
Université du Québec à Chicoutimi, Chicoutimi, G7H 2B1, Canada

Charles Gouin-Vallerand
LICEF research center
Télé-université du Québec
Québec, G1K 9H6, Canada

Sylvain Giroux
DOMUS Laboratory
Université de Sherbrooke
Sherbrooke, J1K 2R1, Canada

Bruno Bouchard
LIARA Laboratory
Université du Québec à Chicoutimi Chicoutimi, G7H 2B1, Canada

***Abstract*— One of the unique features of mobile applications is location awareness. 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 application domains, ranging from traffic management to advertisement and social studies. We present a novel approach to online recognition of users' outdoor activities without depleting mobile resources. We associate the places visited by individuals during their movements to meaningful human activities using a novel algorithm that incrementally clusters a user's moves into different types of activities. To optimize battery consumption, the algorithm behaves variably according to users' behaviours and the remaining battery level. Studies using real GPS records from a spatio-temporal region demonstrate that the proposal is effective and is capable of inferring human activities without draining the phone resources.**

Keywords—GPS Data; Human activities; Semantic enrichment; Trajectory analysis; Online clustering; Battery-awareness;

I. INTRODUCTION

Human's activity recognition has been an active topic of research for several decades. However, only in the recent years, thanks to the increasing availability and facilities of collecting movement datasets (from GSM or GPS-equipped devices, or even network wireless technologies such as WI-FI and RFID [6]), we have the possibility to study users' activities from their movement traces.

Mobile tracking devices (e.g.: phones and navigation systems) sense the movement of people who are represented by positioning records that capture geo-location, 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 pertinent information about an ongoing activity. In ubiquitous environments, 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 a communication device only, but also a powerful environmental sensing unit that can monitor a user's ambient context [2], both unobtrusively and in real time. Ambient sensing 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 future actions. 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 [8], social networking, and cognitive assistance. However, the limited-battery capacity of mobile devices represents a big hurdle for context detection [16]. The embedded sensors in mobile devices are major sources of power consumption. Hence, excessive power consumption may become a major obstacle to broader acceptance 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 behaviour according to the current battery status of the device. It is in this context that our research lies, we propose a battery-aware activity recognition solution in order to preserve mobiles' life battery.

Outdoor activity recognition field falls into two approaches; motional approaches that detect users' motion state (walking, taking a bus, running, etc.) and location approaches that determine the user's activity based on detecting the visited places (museum, cinema office, etc.). It is in that second field that our work fits. It allows determining the nature of the user's activity from a series of geographic positions. 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 in function 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 characterized by a variable size in function of a person's travel behaviour and his phones' remaining resources.
2. The majority of related works are based on the classification of historical records of people's trajectories using a density-based approach, by which they try to identify the most visited place with post-treatment processes (e.g.: end of the day, every week, etc.). These methods fail in their ability to deal with less visited places

by people, but important in their trajectories (e.g. cemetery, airport, etc.) and cannot be used to fields such as assistance in which we need a real-time access to people's activity. That's why we propose a new online solution which addresses these difficulties.

3. This work not only handles stationary behaviours, but also moving activities such as 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 real-time switch raw users' movement data to meaningful human activities using only a mobile device without network or historical record requirements while consuming a minimum of mobile resources. The aim of the presented approach is to enrich peoples' movements, represented in real time during their travel trajectories, with semantic information about the visited places. Our method is based on the online recognition of points of interest "POI" (A Place of Interest is a (urban) geo-referenced 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 such as traffic management, public transportation, commercials, and advertising, to more serious uses, for instance: 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 two major components, i.e. trajectory classification and spatial recognition; Section 4 describes the experimentation by highlighting two dimensions: accuracy and power saving. Finally, conclusion and future focus, are summarized in Section 5.

II. 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 [8]. We will highlight two parts of related works, works on activity recognition and semantic trajectories, and battery-aware works.

A. Activity recognition and semantic trajectory works

Kang & al. in [6] 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 lack 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.

CityVoyager, presented in [14], is a recommendation system designed for mobile devices, which recommends shops to users based on data analyzed from their past location history. 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 outdoor, 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 doesn't 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 [13], an algorithm is proposed to associate each stop in a user's trajectory to a list of possible visited places. In this work, authors assume the moving object is a person that travels using transportation means associated to a traceable (GPS) device (car, bus, metro, train). The person gets off the transportation mean to walk to the final destination. During this time interval, the person is not traceable, consequently, authors have used probabilities to find the visited place. This work uses numerous thresholds that are set manually such as the minimum duration of an activity. Nevertheless, since these parameters may depend on user profiles, this work may be ineffective on large datasets that contains several profiles.

While developing a rich body of work for managing moving objects, the research community has shown a little interest to 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 tries 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 that operates without any beforehand fixed threshold.

B. Battery-aware works

The problem of power management on mobile devices has been well-explored. Viredaz & al. [5] 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 [16], authors proposed a dynamic frequency/voltage scaling (DVS) to reduce power consumption by configuring the processor based on the requirements of the executing applications. 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 run-time. 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.

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 in function 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.

III. OVERVIEW OF THE APPROACH

We assume the person is traceable via a smartphone, the type of users' traces is not important to us since the proposed approach works for any movements type (GPS, WIFI, Bluetooth or GSM triangulation, pedestrian dead-reckoning, etc.). Usually, a person's activities are divided into two behaviours: stationary and non-stationary, whereas 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 non-stationary activity; shopping itself is also a non-stationary activity, but the goal is to do shopping, so it's an activity with movements (see Fig. 1). Based on these concepts we introduce 3 types of clusters:

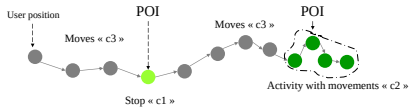

Fig. 1. Relation between moves, stops, and activities with movements.

1. Stop concept, represented by "c1", characterizes stationary activities.
2. Activity with movements "c2" is a non-stationary activities that require movement over a time interval.
3. Moves, represented by "c3", are a set of actions that aim to move from a POI to another.

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

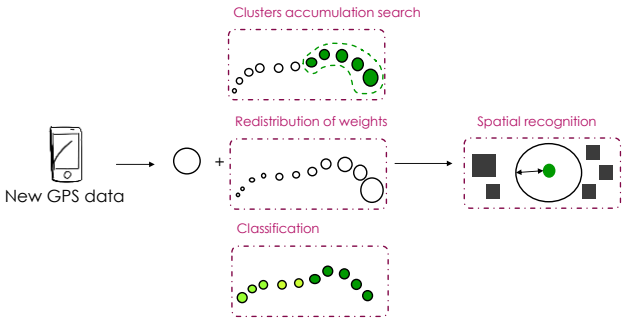

Fig. 2. The overall process of our activity recognition approach.

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.

A. 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 positions is stored in a temporal window called TW.

Definition 1: positions' collection P is an assembly of users' position points $P = \{p_1, p_2 \dots p_n\}$. Each 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} position points where L_{TW} is the size of our temporal window TW, and finally, the weight (p_i.W) that represents the importance of the point p_i according to the time generation.

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

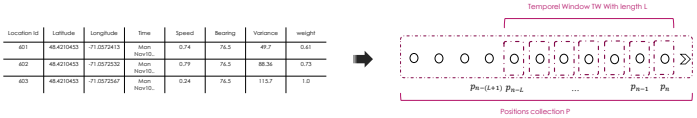

Fig. 3. The relation between the positions' collection P and TW

Once a new position data is received, we achieve three parallel processes like below:

1) Process 1: classification

At the arrival of a new position 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: $p_i.V = \sum(pi - \overline{p_i})/l$ where $\overline{p_i} = \sum p_i / L_{TW}$. 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 behaviours are characterized by a very low $p_i.V$ and $p_i.S$; however, moves behaviours 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 Fig. 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.

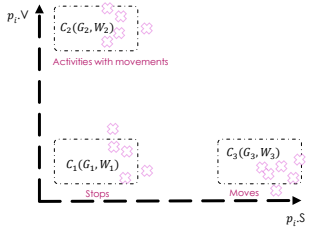

Fig. 4. Inferring activity type using speed and the variance of orientation.

Note that is 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 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.

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. 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 behaviours. The parameter λ is called exponential decay constant; the higher value of λ, the lower importance of the historical data compared to more recent data. The overall weight of the data stream is the following constant:

$$W = \sum_{t=0}^{t=t_c} 2^{-\lambda t} = \frac{1}{1 - 2^{-\lambda}}$$

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 it smallest size. Inversely, when λ approaches 0, TW spreads to it maximum size (see Fig. 5).

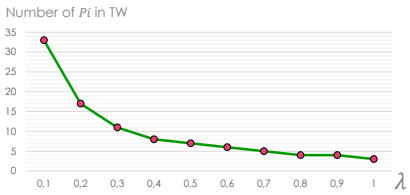

Fig. 5. The impact of varying λ on TW's length.

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). Thus, these parameters are defined in the following definitions:

Definition 3: The remaining battery level is represented by $\beta = \beta_r / \beta_{th}$ 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 Fig. 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.

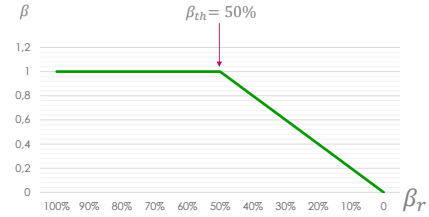

Fig. 6. Example of β variation using a threshold $\boldsymbol{\beta_{th}} = \mathbf{50}\%$.

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 we are sure that an activity is performed, we need less data to make a decision so $E \rightarrow 1$, and $E \rightarrow 0$ when we have problems finding out what type of activity is executed. Therefore, we need a maximum of points.

Based on these observations, E is calculated using the entropy of Shannon; the entropy is a measure of unpredictability of information content, or in other terms, it measures the disorder in a set of information. Consequently, 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:

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

Where $p_i = w_i / \mathrm{W}$. 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:

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

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 behaviour 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.

These conditions are sited as follows:

$$(2)\begin{cases}\text{Battery high} + \text{high disorder} \rightarrow \text{TW spreads}\\ \text{Battery high} + \text{low disorder} \rightarrow \text{TW shrinks}\\ \text{Battery low} + \text{high disorder} \rightarrow \text{TW shrinks}\\ \text{Battery low} + \text{low disorder} \rightarrow \text{TW shrinks}\end{cases}$$

If we parse the characteristics mentioned in (2) into a mathematical representation using the following relations:

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	$\lambda \rightarrow 1$

Then,(2) will be written as follows:

$$(3)\begin{cases}\beta \rightarrow 100\% \text{ and } E \rightarrow 0 \Longrightarrow \lambda \rightarrow 0\\ \beta \rightarrow 100\% \text{ and } E \rightarrow 1 \Longrightarrow \lambda \rightarrow 1\\ \beta \rightarrow 0\% \text{ and } E \rightarrow 0 \Longrightarrow \lambda \rightarrow 1\\ \beta \rightarrow 0\% \text{ and } E \rightarrow 1 \Longrightarrow \lambda \rightarrow 1\end{cases}$$

Let β, E and λ be Boolean parameters that take values as follows:

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

The logical rules in (3) can be illustrated in the following truth table:

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

We notice that this truth table is that one of the Boolean implication function where:

$$\lambda \equiv F(\beta, E) \equiv \beta \wedge \neg E \equiv \beta \xrightarrow[boolean]{} E$$

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 where β = 1 means full battery and β = 0 means empty battery. Yet, in real life there are many states between full and empty states. As such, the fuzzy logic was introduced [15] 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:

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

Using the inequality of Reichenbach [9], the fuzzy implication can be written as:

$$\lambda = F(\beta, E) = 1 - \beta + \beta E$$

Consequently, (1) is proven; this equation has been tested in Fig. 7 where we varied the two parameters β and E to observe the sensitivity of λ (the length of TW) to these changes.

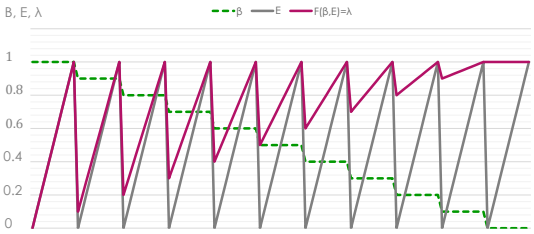

Fig. 7. λ variation in function of β and E.

In Fig.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 starts to ignore disorder E until it reaches a total ignorance when β=0 (see Fig. 7 when β=0). After having distributed the weights of each p_i in TW and calculated the size of TW in function of the battery level β and disorder E, it is time to see if the user is performing some interesting activity.

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 $\mu = \frac{W}{k}$ with k representing the number of clusters (activity families) used by K-means to classify TW, in our case k=3 because we are trying to identify three families of clusters: Stops; moves; 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 1: 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 $T_{min} = \frac{1}{\lambda}\log\frac{\mu}{\mu-1}$, where λ = 1- β + β E. Consequently, on every T_{min} we check if there is any activity whose 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. Then, we move to the spatial recognition of the summarized point C_j.

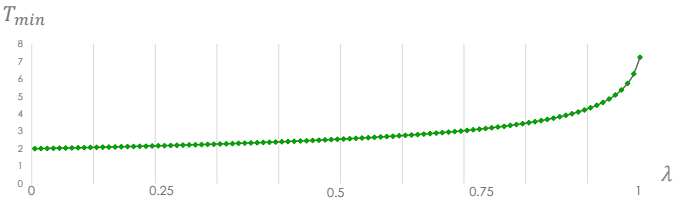

Fig. 8. The relation between $\boldsymbol{\lambda}$ and $\boldsymbol{T_{min}}$

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 Fig. 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 Fig. 5) for the reason that the battery is low or there is some stability in the user's behaviour (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 Fig. 8).

Conversely, when λ is near 0, TW will contain a maximum number of position points (see Fig. 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 (Fig. 8).

Algorithm 1: trajectory classification

Input: A position point p_i;
Output: The activity of the person;
1: **For each** T
2: Store p_i in *TW ;*
3: **End for each**
4: **For each** T_{min}
5: Classify every point in *TW* ;
6: Update λ;
7: Update the centers of clusters C_j (G_j, W_j) ;
8: Calculate the threshold μ;
9: **If** (max(W_j)>μ) **then** POI = spatial recognition (G_j); **End**
10: Update T_{min};
11: **Return** Activity
12: **End for each**

Algorithm 1 achieves two processes. The first is to store every position p_i when it arrives; the second performed every T_{min} to reduce the calculation. First step is to classify every point p_i in TW, 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 begin the 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 at the same time. If the condition is verified, we begin the geographic environment recognition of C_j. Finally, we update the value of T_{min} that will determine the next process repetition.

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

B. Step 2: spatial recognition

This task aims at further understanding the movement behaviour of users, in terms of more semantically meaningful POI. In this step, we are going to transform a cluster of position points to an expressive human activity. First, we are going to look for the nearest geographic entities to this cluster; then, we will 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 our spatial database. Our database is powered by OSM [10] and stored in the users' device for local use. This technique aims at discarding networks use by offering offline services that will save a user's money and a phone's battery.

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 OSM. They provide a variety of free extracts in Shapefile and raw OSM format on their download website. After downloading raw OSM Data, and storing it in a spatial database, we use A querying algorithm to search for the nearest spatial feature to C_j. The querying process 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 (50m). 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 [3]. 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.

Before proceeding to the experimentation part of our research, we need to highlight an important point; the spatial recognition presented in this paper was performed using a spatial database to recognize outdoor visited places such as cinema, restaurant, etc. However, it can be easily applied to detect indoor visited places by using an indoor positioning technique (like PDR, Wi-Fi, or Bluetooth triangulation) and an indoor spatial database that contains the set of the indoor geographic entities that the user may possibly visit. For instance, if the user moves within the university, the detected visited places (POI) can either be the class rooms, the university restaurant, teacher office, meeting room …etc.

IV. EXPERIMENTAL EVALUATION

We test our approach's ability to save battery life by comparing our solution to LifeMap application described in [2]. Researchers in LifeMap project collected real traces from 68 people over four weeks. To collect the ground truth, the participants explicitly labeled the place names and kept a diary of places they had visited with 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 throughout the duration of the experimentation process. The LifeMap project can be found in [7]. The LifeMap dataset in [1] and the LifeMap mobile application can be found on android play store.

We developed an android application that is fed from LifeMap dataset. The main idea is to make it out as if the users had moved holding our application in their phones; the application recuperates the GPS coordinates one by one and processes each point using our online approach. We then compare the battery consumption of our application with LifeMap application's one (see Fig. 9).

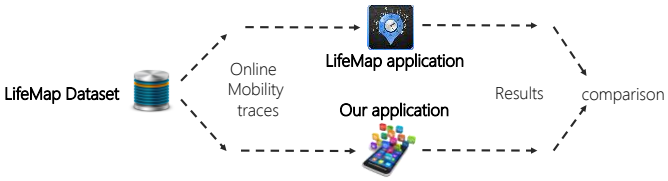

Fig. 9. Test scenario to compare our solution with LifeMap application using LifeMap Dataset.

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 include the same sensors as lifeMap. Indeed, we have enriched our application with all sensors type used in LifeMap, 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 according to the remaining battery level and the user's behaviour.

Moreover, even if the sensors are activated, we request each used sensor to obtain a data not necessarily used in our approach (such as Wi-FI and accelometer). For instance, the application consumes more resources when it requests the users' position 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 consumes the same resources used by LifeMap application to get a GPS position.

In order to automatically recognize users' activities, we have constructed a Spatialite geo-database [12] in which we have stored the background geographic data needed in our spatial recognition (see Fig. 10). 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 accurately recognize 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 various location points. Consequently, we have implemented an interpolation algorithm that estimates the missing locations when the sampling frequency is high (more than one minute).

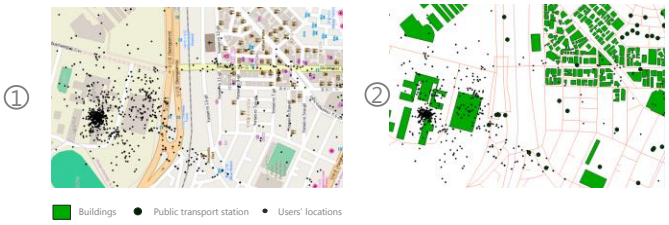

Fig. 10. Example of one user's trajectory projected on: (1) OpenStreetMap raster layer and (2) Vector layer retrieved from our geo-database.

We have chosen to deploy our application on the Samsung Galaxy S smartphone, the same smartphone model used by LifeMap application for activity recognition. After having filled the necessities regarding the comparison between the two approaches, we will present the results for both of them.

The total number of recorded hours of battery status in LifeMap dataset is 48900 hours. In order to experiment all the hours using one smart phone, 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 Fig. 11 the tracking of one user's battery life for 72 hours using LifeMap and three versions of our approach, in which we vary each time the value of the threshold β_{th} from where the application starts to save battery life. Then, we will present the accuracy of both approaches for the entire tested hours (1632).

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%).

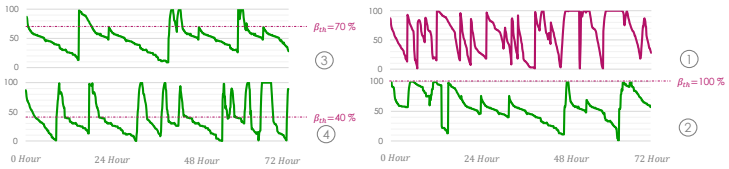

Fig. 11. Results comparison between LifeMap and our solution for 72 hours of activity recognition. (1) LifeMap, (2) our solution using $\boldsymbol{\beta_{th}}$ = 100%, (3) our solution using $\boldsymbol{\beta_{th}}$ = 70%, (4) our solution using $\boldsymbol{\beta_{th}}$ = 40%.

We notice that in our approach, battery consumption varies in function 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 the Graph 2 of Fig. 11. However, when β_{th} is set to 40%, we notice that the life of the battery (Graph 4) shows slight similarity to LifeMap's one (Graph 1). In that case, the algorithm consumption behaves like LifeMap between 100% and 40%; but, when it falls under 40%, the algorithm starts saving the battery (see the horizontal dotted line in Graph 4). Thus, β_{th} has an impact on the resource consumption. But what about accuracy? Is it affected by the change of β_{th}? To answer these questions, we have tracked our algorithm's accuracy using the following formula:

$$Accuracy = \frac{Corrects}{number\ of\ tested\ activities}$$

TABLE 4: The impact of varying β_{th} on the accuracy

	$\boldsymbol{\beta_{th}}$ **= 100%**	$\boldsymbol{\beta_{th}}$ **= 70%**	$\boldsymbol{\beta_{th}}$ **= 40%**	**LifeMap**
Accuracy	68,7 %	77,1 %	85,4 %	78%

Results presented in Table 4 represent the accuracy evolution when varying β_{th} during the 1632 hours of experiment. The authors of LifeMap reported in [4] accuracy of 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 (see Table 4); this is justified by the fact that 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 false detecting some activities when the time threshold is set to a small value.

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

We have tracked one user's data accuracy from LifeMap dataset when varying slowly β_{th}; results are presented in Fig. 12 in which accuracy and an estimation of battery life are presented in function of β_{th} . We notice that the higher the value of β_{th} is, the more we save resources and lose accuracy.

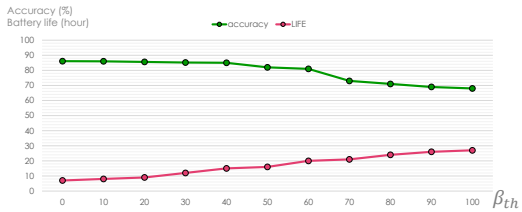

Fig. 12. The impact of varying β_{th} on the accuracy.

In order to find an optimal solution that lets saving both battery life and accuracy, we have to resolve the optimization problem maximizing the battery-life function L (β_{th}) and accuracy function A (β_{th}) at the same time. This is a multi-objective optimization problem in which we try to optimize two different functions. One of the existing solutions is the weighted sum method [11]; the solution is based on selecting a scalar weight s_i and maximizing the following composite objective function:

$$U = s_1 * \mathrm{L}(\beta_{th}) + s_2 * \mathrm{A}(\beta_{th})$$

In our case, the importance of accuracy and battery life is equal. So, the weight of the two variables s_1 and s_2 are equals too ($s_1 = s_2 = 1$). After addition, the value of β_{th} that maximizes U is $\beta_{th} = 60\%$ (see Fig. 13). Thus, if we need to maximize accuracy and battery life at the same time, β_{th} should be set around 60%.

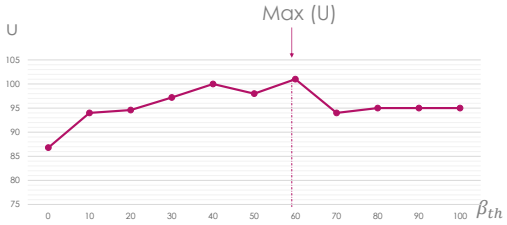

Fig. 13. Optimization of $\boldsymbol{\beta_{th}}$ to maximize the accuracy and the battery life.

The average accuracy of our approach when varying $\boldsymbol{\beta_{th}}$ from 0% to 100% is 79% (see Fig. 12), for a system that proposes an activity recognition system keeping a long lifetime of the battery. We believe 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 short time while outdoor, we decrease β_{th} to promote accuracy and vice versa.

V. CONCLUSIONS AND FUTURE WORKS

In this paper, we proposed a new battery-aware technique for extracting semantically and incrementally important geographical locations from users' movements. We associated the places visited by individuals during their movements to meaningful human activities using an algorithm that cluster incrementally a user's moves into different types of activities using two parameters, speed and the variance of the moves orientation, our algorithm was implemented to change its computational complexity in function of the remaining battery level and the users' behaviours.

Several promising directions for future works exist. Such as 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 to improve the accuracy.

REFERENCES

[1] LifeMap Dataset, (2016): http://crawdad.org/yonsei/lifemap/20120103/

[2] J. Chon, H. Cha, "LifeMap: A Smartphone-Based Context Provider for Location-Based Services", IEEE Pervasive Computing, vol.10, no. 2, pp. 58-67, April-June 2011, doi:10.1109/MPRV.2011.13

[3] D. O'Sullivan and D. Unwin: Geographic Information Analysis. John Wiley & Sons, Hoboken, NJ (2003)

[4] Y. Chon, E. Talipov, H. Shin, H. Cha, "Mobility Prediction based Smartphone Energy Optimization for Everyday Location Monitoring," in Proceeding of 9th ACM Conference on Embedded Networked Sensor Systems (SenSys'11), 2011, ACM, Seattle, WA, USA.

[5] M. A. Viredaz, L. S. Brakmo, and W. R. Hamburgen. Energy management on handheld devices. ACM Queue, 1:44-52, 2003.

[6] J. H. Kang, W. Welbourne, B. Stewart and G. Borriello, Extracting places from traces of locations. In *Proc. WMASH*, pages 110–118, 2004.

[7] LifeMap,(2016) :http://sourceforge.net/projects/lifemapandroid/

[8] Y. Zheng. Trajectory Data Mining: An Overview. ACM Transactions on Intelligent Systems and Technology (ACM TIST). 2015, vol. 6, issue 3.

[9] Reichenbach, H., Elements of Symbolic Logic, Macmillan, NY, 1947.

[10] OpenStreetMap, (2016): http://wiki.openstreetmap.org/ .

[11] J. Koski, "Multicriteria truss optimization", Multicriteria Optimization in Engineering and in the Sciences, edited by Stadler, New York, Plenum Press, 1988.

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

[13] L. Spinsanti, F. Celli, C. Renso, where you stop is who you are: understanding people's activities by places visited. In the proceedings of Behaviour Monitoring and Interpretation (BMI) workshop, 2010.

[14] Takeuchi, Y. et al. CityVoyager: An Outdoor Recommendation System Based on User Location History. In *Proceedings of UIC'2006*, (Berlin, 2006), Springer Press: 625-636.

[15] V. Novák, I. Perfilieva, and J. Močkoř, (1999) Mathematical principles of fuzzy logic Dodrecht: Kluwer Academic. ISBN 0-7923-8595-0

[16] JP. Pillai and K. G. Shin. Real-time dynamic voltage scaling for low-power embedded operating systems. SIGOPS Oper. Syst. Rev., 35(5):89-102, 2001.