TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ACRONYMS

ADL:	ACTIVITY OF DAILY LIVING
AI:	ARTIFICIAL INTELLIGENCE
BLE:	BLUETOOTH LOW ENERGY
BN:	BAYESIAN NETWORK
CB-SMoT:	CLUSTURING-BASED STOPS AND MOVES OF TRAJECTORIES
CRF:	CONDITIONAL RANDOM FIELD
DBN:	DYNAMIC BAYESIAN NETWORK
DBSCAN:	DENSITY-BASED SPATIAL CLUSTERING OF APPLICATIONS WITH NOISE
DT:	DECISION TREE
DVS:	DYNAMIC VOLTAGE SCALING
ESI:	EXPLORATION SATISFACTION INDICATOR
FRQNT:	FONDS DE RECHERCHE DU QUEBEC – NATURE ET TECHNOLOGIES
GATT:	GENERIC ATTRIBUTE
GDBSCAN:	GENERALIZED DBSCAN
GE:	GLOBAL ERROR
GIS:	GEOGRAPHICAL INFORMATION SYSTEM
GNSS:	GLOBAL NAVIGATION SATELLITE SYSTEM
GPS:	GLOBAL POSITIONING SYSTEM
GSM:	GLOBAL SYSTEM FOR MOBILE
HAR:	HUMAN ACTIVITY RECOGNITION
HCI:	HUMAN-COMPUTER INTERACTION
HE:	HABITS ERROR

HHMM:	Hierarchical Hidden Markov Model
HMM:	Hidden Markov Model
HT:	Habits’ tree
IoT:	Internet of Things
KC:	Knowledge Circle
KNN:	K-Nearest Neighbor
LE:	Learning Error
LIARA:	Laboratoire d’Intelligence Ambiante pour la Reconnaissance d’Activites
LIMVI:	Laboratoire de recherche en Informatique Mobile et Villes Intelligentes
MAS:	Multi-Agents Systems
OS:	Operating System
OSM:	Open Street Map
PDR:	Pedestrian Dead Reckoning
PHT:	Page-Hinckley Test
POI:	Point Of Interest
RAM:	Random Access Memory
RF:	Radio Frequency
RFID:	Radio Frequency Identification
RSSI:	Received Signal Strength Indication
SC:	Smart City
SPOI:	Smart Point Of Interest
STPM:	Semantic Trajectories Pre-processing Module
SVM:	Support Vector Machine

TW:	TEMPORAL WINDOW
UMTS:	UNIVERSAL MOBILE TELECOMMUNICATIONS SYSTEM
UQAC:	UNIVERSITE DU QUEBEC À CHICOUTIMI
VFDT:	VERY FAST DECISION TREES
WLAN:	WIRELESS LOCAL AREA NETWORK
XML:	EXTENDED MARKUP LANGUAGE

PART I

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1. THE ERA OF SMARTPHONES

The current era of smartphones can be considered as a golden age of mobile phones. In the modern world, these devices have become such an indispensable part of our lives that it is difficult to live without it, regardless of age, sex or social status.

Throughout its history, telephony has changed, new ways of communication have been devised, new models have been launched to meet the constantly changing demands of society, and at times, creating needs inconceivable up until then. The first telephone device was invented by Alexander Graham Bell in 1876 [1]. No one imagined that phones would evolve so remarkably and have such wide-reaching effects 140 years later.

Smartphone is a mobile phone with advanced features and functionality beyond traditional functionalities like making phone calls and sending text messages. They have been in a rapid expansion in the last decade, which was influenced by four important and continuing phases that led to major changes in our view of the world [2].

The first phase was purely meant for enterprises. In 1982 Bell Laboratories in the United States created the device now known as the first generation mobile phone (1G, analog voice) [3]. Second-generation phones (2G) appeared in 1990. They were smaller, lighter and cheaper, and based on GSM (Global System for Mobile Communications) providing digital cellular communication [1], which improved the quality and security of voice transmission. From that moment on, and following the expansion of computer use and the Internet, workstations connected to a local network were replaced with desktops and laptops with LAN (Local Area Network) and WAN (Wide Area Network) connections [4]. Society started to address the need for data transmission (Multimedia). And so began what is known as the third generation (3G) and the development of UMTS (Universal Mobile Telecommunications System) technology [2]. Mobile phones began to incorporate Internet connection, allowing the transmission of files. Among the functions or services offered, photo and video cameras and games gained great importance, adapting mobiles to the home and business environments.

Fourth generation (4G) devices appeared with faster voice and data transmission speeds [5]; they became vital social and leisure attributes [6]. The current tendency is clearly toward the integration of telephony with a large part of the services and applications that can be obtained using a computer. The challenge that various international technology companies were facing is the integration of a telephone, camera and laptop computer in the same device with access to numerous applications such as videoconferencing and social networks [7].

Recent technological advances and miniaturization have accelerated the convergence between mobile phones and powerful computers facilitating the development of the smartphone technology [6]. This is when the real revolution got under way. Smartphones computation and storage capabilities are ever-growing while integrating a set of sensors (accelerometer, microphone, GPS, Wi-Fi, digital compass, gyroscope, and, in the future, air quality and chemical sensors [8], [9]). 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.

This new generation of mobile phones has totally changed the way that people interact with mobile devices [6], so users can do more than just make calls. A smartphone can serve as a functional laptop or desktop that can fit in a pocket or small bag. Sensors enabled smartphones are set to become even more central to people's lives as they become intertwined with existing applications such as social networks and new emerging domains such as green applications [10], global environmental monitoring [11], personal and community healthcare [12], sensors augmented gaming [13], virtual reality [14], and smart transportation systems [15].

Consequently, a new paradigm is fast emerging: people are beginning to replace their personal computers with smartphones. The mobility and power afforded by smartphones allow users to interact more directly and continuously with their surroundings [16]. The set of services proposed by smartphones motivated people to take this powerful portable unit with them everywhere. This is how the smartphones entered its age of ubiquity.

1.2. THE UBIQUITY OF SMARTPHONES

Mark Weiser, who is known as the father of ubiquitous computing (ubicomp), initiated research work at Xerox Palo Alto Research Center (PARC) in 1988 [13]. He imagined ubiquitous computing as a world where computation and communication would be conveniently at hand and distributed throughout our everyday environment [17]. In contrast to the desktop computing, ubiquitous computing can occur using any device, in any location, and in any format.

The interactivity of ubiquitous computing devices to adapt to various situations necessitates different kinds of principles when building applications. In [18], Banavar provides us with different views of ubicomp elements: (1) A device is not a repository of custom software, but a portal into an application or data space. (2) An application is not a software to exploit a device's capabilities but is a means for a user to perform a task. (3) The computing environment is not a virtual space that exists to store and run software, rather it is a user's information enhancing surroundings [3]. Tasks need to be dynamic in ubicomp so that they can easily be transferred from device to device.

As mobile phones are rapidly becoming more powerful computationally, there has been significant progress toward Weiser's vision. Smartphones represent the first truly ubiquitous mobile computing device. Context aware computing and smart wearables [7], [17], wireless sensor networks, activity recognition using wearables [19], and Human-Computer Interaction (HCI) [20] are just examples of technologies that have spun off the ubiquitous computing idea.

The mobility and power afforded by smartphones allow users to get involved more directly and constantly with the external world more than ever before [6]. It helps both keeping in touch with others and managing everyday tasks while being everywhere. Technological trends result in ever more features packed into this small, convenient form factor. Smartphones can already see, hear and sense their environment.

The emerging capabilities of smartphones arise the use of mobile phones as input devices to various application fields. The ubiquity of mobile phones gives them a great potential to be the default physical interface for ubiquitous computing applications [2]. In fact, smartphones on board sensor data is interpreted through lightweight machine learning algorithms running on the smartphone itself. It is giving rise to a new area of research called smartphones sensing [21]. Consequently, by embedding these specialized sensors and pushing research in Artificial Intelligence (AI) [22] in the direction of telephony, it is now possible to continuously sense and infer people's physical activities, social interactions and surrounding context. Ultimately, it will be able to monitor their moves, predict their needs, offer suggestions, and even understand their moods [23]. As a result, phones get to work in a more autonomous way [24].

As such, 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. It has even opened doors for many interesting data applications. One of these promising fields is activity recognition, the subject of the present thesis.

1.3. ACTIVITY RECOGNITION

In our everyday lives, we often try to understand the activities of people around us and adjust our behavior accordingly. Our human nature is made to be sensitive to the environment. At home, we are curious to know what the rest of the family is doing: cooking, reading or watching TV; at work, we may need to know what a colleague is doing: is he in a meeting or on the phone; in the street, we keep an eye on the traffic light to be able to cross the road; while driving, we must be aware of the behavior of surrounded people. This ability to recognize activities seems so natural and simple for ordinary people, but it actually requires complicated functions of sensing, learning, and inference. We indeed all execute these three tasks without even aware that. Let us take the example of how we recognize a cooking activity. Maybe we happen to see someone at the dinner time in the kitchen, or we smell something is cooked, or we just find that the stove is on. From such evidences, we could infer the activity based on our past experiences [25]. All these functions of sensing environments, learning from past experience, and applying knowledge for inference are part of the amazing human intelligence; they are still great challenges for modern computers. The goal of activity recognition is to enable computers to have similar capabilities as humans in order to recognize people's activities [26]. If eventually we develop accurate computers that can reliably recognize people's various activities, we can drastically improve the way people interact with computers. We will have huge impacts on behavior, social and cognitive sciences, and we will be much closer to our dreams of developing robots that can offer help in our daily lives [27]. To achieve this goal, we must provide computers with artificial activity recognition functions that ordinary people possess [25].

Activity recognition is an important technology in pervasive computing [17], [28]. The main reason is that it can be applied to many real-life, human-centric problems. Activity recognition offers the possibility to understand what people are doing at a specific 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 [29], fitness monitoring [30], personal biometric signature [31], urban computing [32], social networking [33], marketing [34], police and security [35] and healthcare [36]. For instance, patients with diabetes, obesity, or heart disease are often required to follow a well-defined exercise routine as part of their treatment. Therefore, recognizing activities such as walking, running, or cycling becomes quite useful to provide feedback to the caregiver about the patient's behavior. Moreover, it can be used to automatically motivate the patient to be more active when no activity is detected [9]. Likewise, activity recognition can be used to assist people suffering from Alzheimer's disease in their daily outdoor tasks. Using some beforehand knowledge on people's destinations, activity recognition system can be used to detect every anomaly in their behaviors and launch assistance processes like reminders, suggesting a new destination or home back roads [37]. Furthermore, as one branch of human computer interaction, the activity recognition makes the computer even "smarter", it could provide the corresponding services based on what the user is doing [9]. For example, suppose that the user phone detects that the user is about to leave the room and its weather application indicates that it will rain later. A reminder will pop up with a message “Take an umbrella, it is going to rain with a high probability”. As smartphones become as essential as keys and the wallet are nowadays, the activity recognition using smartphones could help in achieving our daily tasks. It could help in the prevention of dangerous activities, such as elder people’s fall detection [38], youth Autism Spectrum Disorder detection in a classroom [39], or even suicide prevention [40].

The emerging concept of activity recognition is a new topic for data analysis. However, research community’s efforts are increasing day by day to carry clear definitions and common understandings. Diane Cook defined in her book [41] the activity recognition as the task of measuring the physical activity of a person via the use of objective technology. She presented in [42] the difference between the activity recognition and the activity discovery that aims to discover patterns in the data that does not belong to a predefined class. She demonstrates that activity discovery not only sheds light on behavioral patterns, but it can also boost the performance of recognition algorithms. Moreover, Lara and Labrador presented in [34] a survey on activity recognition using wearable sensors. In that work, a general architecture is first presented along with a description of the main components of any human activity recognition system. They also proposed a two-level taxonomy in accordance to the learning approach (either supervised or semi-supervised) and the response time (either offline or online). Then, the principal issues and challenges are discussed, as well as the main solutions to each one of them. Twenty-eight systems are qualitatively evaluated in terms of recognition performance, energy consumption, obtrusiveness, and flexibility, among others. Finally, they present some open problems and ideas that, due to their high relevance, should be addressed in future research.

As mentioned above, human activity recognition refers to the task of measuring the physical activity of a person via the use of objective technologies. This task is extremely challenging owing to the complexity and diversity of human activities. The volume and the complexity of data used in activity recognition systems exceeds the capacity of human computations [18]. To address this weakness, these complex tasks are delegated to the machines that have the ability to perform such tasks rapidly and efficiently. With this delegation, scientists have sought to create systems able to execute tasks in the place of humans. The goal is to give more control to software and to replace humans in critical and complex functions. Within this context, the topic of Artificial Intelligence (AI) [22] intervenes naturally. Artificial Intelligence has always been in the fiction dreams of human, and thus it has been an important trend in the creation of activity recognition systems [3].

1.3.1. DATA MINING

The last decade has been the mobile device technology era where the capture of the evolving position of moving objects has become ubiquitous. Mobile wearable tracking devices, e.g., phones and navigation systems collect the movements of all kinds of moving objects, generating huge volumes of mobility data. In combination with the Internet and social media, these trends have led to a new situation where the data grow exponentially [16]. The quantity of data, collected from moving users, challenges human ability to analyze the stream of input data. Therefore, new methods for online mining of moving object data are required. As such, activity recognition intervenes to propose intelligent systems to infer temporally contextualized knowledge regarding the state of the user on the basis of a set of heterogeneous sensor readings.

Data mining is the set of methods and algorithms allowing the exploration and analysis of databases [43]. It exploits tools from statistics and artificial intelligence. Data mining is used to find patterns, associations, rules or trends in datasets and usually to infer knowledge on the essential part of the information. It is often seen as a subtopic of machine learning. However, machine learning as defined by Arthur Samuel in 1959 [20], relates to the study, design and development of the algorithms that give computers the capability to learn without being explicitly programed. Contrary to data mining, machine learning is typically supervised, since the goal is to simulate the learning of already known properties from experience (training set) in an intelligent system [43]. Therefore, a human expert usually guides the machine in the learning phase. Within realistic situations, it is often not the case. While the two are similar in many ways, generally, in data mining the goal is to discover previously unknown knowledge that can then be exploited in activity recognition to make better decisions about the user performed activity.

This difference between machine learning and data mining is well described in [26] but under a different point of view. Authors divided human activity understanding into activity recognition and activity pattern discovery. The first focuses on accurate detection of the human activities based on a predefined activity model (machine learning). Therefore, an activity recognition researcher builds a high-level conceptual model first, and then implements the model by building a suitable pervasive system. On the other hand, activity pattern discovery is more about finding some unknown patterns directly from low-level sensor data without any predefined models or assumptions. Hence, the researcher of activity pattern discovery builds a pervasive system first and then analyzes the sensor data to discover activity patterns. Even though the two techniques are different, they both aim at improving human activity technology. Additionally, they are complementary to each other, the discovered activity patterns can be used to define the activities that will be recognized and tracked.

As described in [44], the complete process of data mining is illustrated in Figure 1. Before beginning the cycle, it is important to understand the context and the data related to the situation in which the data mining algorithm is applied. For example, we should know what the goal of the data mining is. What the consequences of errors are, whether they are insignificant (marketing) or critical (healthcare and police). Data consideration is also important but usually for the strategy design. First of all, it is important to know the types of attributes that are the most representative. Whether there is any strong correlation between two or more attributes. Those are examples of questions one should answer before even beginning the data mining cycle.

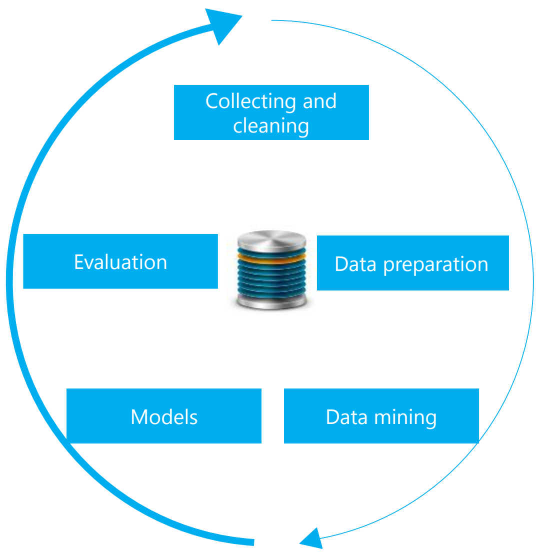


Figure 1: The overall data mining process

The first step is to collect and clean the data from potentially more than one source, which can be devices, sensors, software or even websites. The goal of this step is to create the data warehouse that will be exploited for the data mining. The second step consists in the preparation of the data in the format required by the data mining algorithm. Sometime in this step, the numerical values are bounded; other times, two or more attributes can be merged together or completely deleted. It is also at this step that high-level knowledge (temporal or spatial relationships, etc.) can be inferred for suitable algorithms. The next step is the data mining itself. It is important to choose or design an algorithm for the context and the data [43]. In Chapter 2, the main Activity recognition algorithms will be reviewed along with an assessment of their advantages and disadvantages depending on the application requirements. Finally, the data mining step should result in a set of models (decision trees, clusters, rules, etc.) that need to be evaluated. If the evaluation is not conclusive enough, the cycle can be repeated from one to many times. Indeed, data mining is a method that often does not give expected results the first time. Note that the collection and cleaning step is generally done only once regardless of the results.

Despite the massive use of data mining in activity recognition systems, there are still many issues that motivate the development of new techniques to improve the accuracy of data mining techniques for the activity recognition. Some of these challenges are (1) the selection of the attributes to be measured (2) the construction of a portable, unobtrusive, and inexpensive data acquisition system (3) the design of feature extraction and inference methods (4) the collection of data under realistic conditions (5) the flexibility to support new users without the need for retraining the system, and finally and the most important point, the subject of our thesis, the implementation on mobile devices meeting energy and processing requirements [6].

1.4. MOBILE-BASED ACTIVITY RECOGNITION

Traditional activity recognition systems are built on post treatment processes as described in Figure 1. First, researchers collect and prepare data, using smartphones for example, to sense the user environment. After that they use data mining algorithms to train the models that will be used to detect the activities performed by the user [41]. These algorithms may become complex and computationally expensive. As such, researchers tend to achieve these tasks on powerful machines (servers) following a pre-defined schedule, e.g. every day, every week…etc. [45]. For instance, authors of [23] proposed an approach to learn users' frequent patterns to answer the problem of activities models creation from a smart home sensor. The aim of this work is to learn users' patterns moving inside a smart home and use these patterns to predict the users' next activities or to detect abnormal activities. As user data come continuously inside a smart home, they have decided to launch the learning process using a distant server on predefined moments (every week). However processing data on predefined moments may become an obstacle for some applications that require an instantiate information about the user activity.

As presented above, smartphones computation and storage capabilities are ever-growing while integrating a suite of sensors that can be used to collect the data needed to feed the data mining models. As such, the scientific community started to figure out that it will be more convenient to implement the activity recognition model on smartphones rather than delegate it to distant servers. Consequently, smartphones are becoming not only sensing units, but a powerful machine that can handle an entire AI model.

In fact, achieving activity recognition tasks on mobile has several advantages: (1) the ubiquity of smartphones make this technique very efficient. As users take their phones with them everywhere, pushing activity recognition on smartphones let detecting users' behaviors anytime and everywhere. (2) Centralizing calculation on users smartphones facilitate the development of offline application to avoid the Internet dependency. (3) Naturally, using smartphones motivate to head toward the online learning that will offer instantiate information about the user state. Contrary to traditional systems, mobile-based activity recognition can detect the user achieved activity during it execution.

This set of advantages motivated the research community to invest in this direction. Smartphone applications with activity recognition techniques have been shown up in recent years as an alternative solution to traditional systems [46]. These applications usually have similar roles to track users' motion and activity logs such as jogging route, steps taken, sleeping time, daily visited places…etc. By mining the logged data, they may offer the user a summary on his life style and quality. For instance, authors of [47] presented *myHealthAssistant* where both daily activities as well as specific gym exercises and their counts are recognized and logged. This preventive healthcare application intends to motivate patients to increase their level of physical activity and to decrease the risk of disabling health conditions. The work of Song et al. [35] is another example of mobile-base activity recognition systems. They proposed a floor localization system to find 9-1-1 caller in buildings by inferring the current floor level using activity recognition techniques.

As seen so far, the smartphones have become so ubiquitous and powerful to a point that they have replaced computers. They have become a primary input for human activity recognition. Although the research on activity recognition is beneficial from the mobile sensors' unobtrusiveness, flexibility, and many other advances, it also faces challenges that are brought by them. In the next section, we review the major, common challenges for activity recognition using mobile phones.

1.5. Mobile-Based Activity Recognition Challenges

1.5.1. Complexity of the Activity Recognition

First of all, the accuracy of activity recognition, especially those based on the accelerometer data [48], is heavily affected by the subjects participated in training and testing stages. This is mainly due to the fact that different people have different habits and motion patterns. Even for the same subject, he may execute an activity differently at different times. In [48], the comparative experiments show that training and testing on the same subject achieves the highest accuracy. Training and testing on the same group of multiple subjects has the second-highest accuracy. The accuracy decreases when the test data is collected from the same subject but on different days. The lowest accuracy is in the setting where the training data is collected from one subject on one day and testing is conducted on another subject on a different day.

For the supervised systems, it is highly desirable that the training data must contain as many varieties of the subjects as possible. However, it is not easy to coordinate people of different ages and body shapes to collect data under a controlled lab environment, not to mention the varieties of the environment itself.

Another challenge is linked to the position of the phone when retrieving data from sensors. Due to the property of some sensors both in wearable sensors and smartphones (such as accelerometer), its raw reading heavily depends on the sensors' orientation and positions on the subject's body [24]. Moreover, these sensors are known to generate a lot of noise [46]. For example, when a user is walking while holding a phone in his hand, the moving data reading is quite different from the data reading if the phone is in his pocket [9].

The complexity of users' activities also brings an additional challenge to the recognition model. Users' behaviors when performing an activity may be too complex and too unpredictable that it may become very hard to recognize them. For example, the motion during the transition period between two activities is difficult for the underlying classification algorithm to recognize. People performing multiple tasks at the same time might also confuse the classifier which is trained under one activity-per-segment assumption [45]. Culture and individual difference might result in the variation in the way that people perform tasks. In addition, the change in user habits may become a major obstacle for pattern detection.

1.5.2. OUTDOOR VERSUS INDOOR ENVIRONMENTS

Many activity recognition solutions are designed for confined areas such as Smart Homes (SH). The main idea behind these indoor systems is to improve the existing environments by deploying a wide range of sensors in the user living area (e.g. accelerometers, loadcells, ultrasonic sensors, temperature sensors, flow switch, light sensors, pressure mats, RFID, IR motion sensors, electromagnetic contacts, smart power analyzer, microphones, video cameras…etc.). The data retrieved from these sensors are analyzed by centralized computers to detect the users' performed activities. For instance, we can use door contacts sensors to detect when a door in the user apartment is opened, we can use RFID tags to locate every moving objet inside the SH.

However, when we try to recognize users' activities in outdoor environments, we are confronted to another level of difficulty. The main reason is that reproducing the same technique as SHs may become technically and economically unfeasible in the city. As a solution, we have to replace all the sophisticated sensors of SH by one unit: the user smartphone. However, embedded sensors in smartphones are much fewer and less accurate than the external sensors used in SHs. Also, the phone resources are much more restricted than the powerful computers used in indoor environments. Obviously, this passage from SHs to smartphones is very challenging task.

Let us compare the city to SHs. As the city is much wider, the number of activities that can be executed there is much higher, which makes the activity recognition technique more challenging. Furthermore, as we will see in Chapter 3, outdoor activity recognition needs a background geographic data source to detect users visited places. Nevertheless, it is known that geographic data is characterized by heavy size and complex relation [49]. Thus, manipulating this type of data on mobile phones may become very challenging process.

1.5.3. CHANGEABILITY

Human behaviors are very complex and hard to predict, especially in open spaces like cities. One major problem met when trying to incrementally detect a user's activities is the changeability in his habits. For instance, a user that moves in the city, and that has the routine of going from home to work, can change this routine at any moment according to some unknown external pressures, e.g. an unexpected appointment, the user got sick, he unexpectedly drove his son to school…etc. As such, this change in a user habits may become a major obstacle for context detection and for prediction algorithms. We answer this issue by proposing, in Chapter 5, a new algorithm that let detecting these unexpected situations that may occur in a user routines.

1.5.4. DATA SIZE

A user that moves in the city generates a huge amount of data. For instance, suppose that we have an application that retrieves a user’s context data each 10 seconds. This sampling rate will generate about 21600 points per day for each sensor. Storing and processing incrementally all this data can quickly saturate the phone resources (RAM, storage, battery…etc.). As such, processing user data has to be done very carefully and wisely. We propose in this context, a new approach that estimates incrementally an appropriate time to process this data. In Chapter 3, we propose to not treat data all the time, but on calculated frequencies called T_{min} that represents an estimation of the probable next activity moments. Furthermore, in order to preserve the phone resources, we propose in the same chapter to work on few amount of data by introducing a dynamic temporal window TW. We also demonstrate that this technique preserves the phone resources while keeping a high accuracy rate.

1.5.5. ENERGY AND RESOURCE CONSTRAINTS

Generally, Activity recognition applications are supposed to operate 24 hours a day, 7 days a week, in particular for fields such as healthcare. As such, they require continuous sensing as well as online updating for the recognition model, both of which are energy consuming. For the online updating, it might also require significant computing resources due to the complex model that might be implemented [9]. Furthermore, one important challenge is the phone battery. In fact, 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 context-aware mobile applications, no matter how useful the proposed service may be.

Unfortunately, the limited battery capacity of mobile phones has not had the full attention of the research community. The majority of related works are basing their efficiency factors on the accuracy of their models while neglecting the model impact on the phone resources. As reported in [46], the most of the studies are missing resource consumption analysis, such as CPU, memory and battery usage. For the online activity recognition, such an analysis is an important factor in determining the feasibility of its implementation on a mobile phone. So far, comparative studies have been done offline, where different classification methods are compared in different simulation setups based on the accuracy only. We believe that it is not a fair comparison. In order to report a classification method or a set of features as most suitable for activity recognition on mobile phones, it is important to consider the trade-off between accuracy and resource consumption. As such, we focus this thesis on filling the gap between the accurate activity recognition service and the wise use of the mobile resources.

1.6. CONTRIBUTION OF THIS THESIS

The contribution of this thesis follows the footsteps of data mining and activity recognition approaches that have been developed during the last decades. 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 his visited places, e.g. museums, 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.

In this thesis we made a step forward in the context of online outdoor activity recognition using Smartphones. We propose an approach that operates completely on users' smartphones without any connectivity requirement. In particular, we explore the fundamental techniques to reduce the battery consumption of smartphones while recognizing incrementally users' activities.

Users' data are coming continuously. Storing and processing all data may become resource-hungry technique. As such, we answer during this work problems linked to streaming data, incremental learning and battery awareness. In fact, we succeed in making a link between the data mining algorithms and the mobile phone resources in order to improve the activity recognition power consumption on smartphones. Our main contributions during this thesis are summarized in the following points:

1. The majority of related works are based on the classification of historical records of people's trajectories using non-incremental data mining algorithms. These methods fail in their ability to instantly detect the user performed activities. As such, they cannot be used to fields such as assistance in which we need a real-time access to people's state. Research on offline activity recognition has been reviewed in several earlier studies in detail. However, work done on online activity recognition is still in its early stages and is yet to be reviewed. Our first contribution includes an online activity recognition service that let detecting incrementally users' activities without a massive use of users' historical records.
2. The second contribution continues in the direction of online learning where we propose a new version of online K-means that is designed for streaming services on mobile phones. We propose a novel self-adaptive clustering approach that adjusts the computational complexity of the algorithm according to 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 proposes a temporal data window characterized by a variable size according to a person's travel behavior and his phones' remaining resources.
3. Our third contribution concerns the type of the recognized activities. As described in Chapter 2, a significant part of related works has focused recognizing users' outdoor activities on detecting stops in their trajectories. Unfortunately, these works fail to detect activities that needs a movement to be executed such as shopping and running in a park. We demonstrate that the novel approach that we propose, succeeds to recognize both stationary and moving activities.
4. Activity recognition needs a spatial analysis of background geographic data in other to obtain semantic information about the performed activity. However, in

future cities (smart cities), the geographic data can change frequently. We propose a new spatial exploration technique in smart cities based on a distributed users' collaboration. This technique aims to avoid the use of the predefined geographic databases that can be very expensive to get and very difficult to update in smart cities; those future cities where information can change every hour and every day. The proposed service called NomaBlue is disconnected from the Internet, it can operate in any indoor/outdoor area and it doesn't require any pre-defined geographic databases.

5. After recognizing users' activities in smart cities, we propose an incremental approach to predict their next activities. Our fifth contribution includes a new algorithm for the online prediction of users' next visited locations that not only learns incrementally the users' habits, but also detects and supports the drifts in their patterns. At this stage, we propose a new algorithm of online association rules mining that supports the concept drift.
6. Our last contribution includes a novel hybrid approach that combines activity recognition and prediction algorithms. This combination is designed to online recognize users' outdoor activities while modulating the utilization of the mobile resources. Our approach minimizes activity computations by wisely reducing the search frequency of activities. We demonstrate that our approach is capable of reducing the battery consumption up to 60% while maintaining the same high accuracy rate.

After defining the major contributions of this thesis, we present next the research methodology that we follow to achieve these goals.

1.7. RESEARCH METHODOLOGY

The research presented in this thesis was carried out by following a research methodology divided into four key steps that were not necessarily done fully sequentially.

The first phase of the present work was to elaborate a state of the art of the research area related to the problem of activity recognition. In the first part, we the field of online activity recognition, particularly in an applicative context of mobile environment. It has helped to identify issues and specific needs of mobile phones. The second part of this phase aimed to achieve a state of the art on existing data mining approaches while focusing on the unsupervised streaming algorithms. The field of semantic trajectory analysis was also explored in order to acquire strong understandings on users' trajectories and to understand the different types of user mobility in the city. This part has allowed arriving to the proposed contributions of this thesis.

The second phase consisted in elaborating a complete solution of the activity recognition based on smartphones. This part was elaborated in the form of four layers as described in Figure 2. The first layer includes an online activity recognition model that recognizes users visited places incrementally and without using the Internet, a model that is aware of the user behavior and the limited phone resources. The second layer is to adapt the first layer to smart cities by adding a supplementary spatial exploration model that is designed to operate in future cities. In the third layer we propose an online algorithm that learns users' habits and predicts their next activities carrying the change that can occur in their habits. The last layer consolidates all the previous layers to propose a hybrid system, a system that switches between activity recognition and prediction in order to reduce the phone battery consumption efficiently.

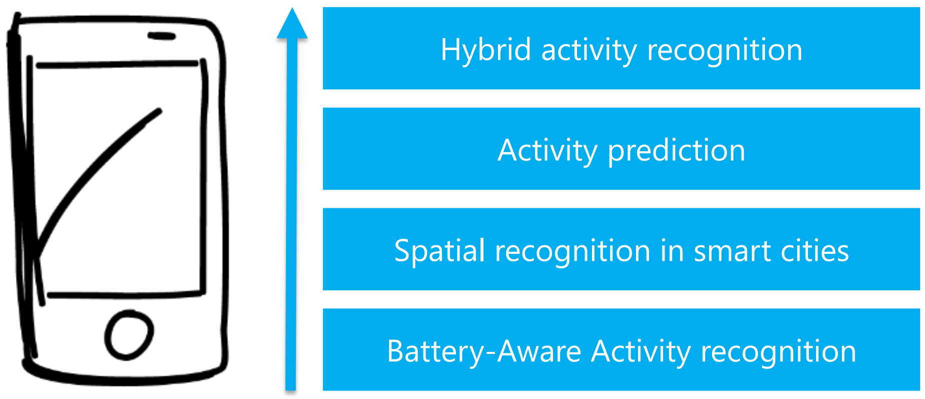

Hybrid activity recognition
Activity prediction
Spatial recognition in smart cities
Battery-Aware Activity recognition

Figure 2: The four layers of our mobile-based activity recognition approach

The third phase consisted into a software implementation of each of the four layers presented above. To do so, we have chosen to develop them using the Java programming language running on Android operating systems. The android application lets online collecting traces of users that are moving in the city, and to incrementally switch them into meaningful human activities. The application that we developed is battery-aware, i.e. it has the ability to self-adapt the computational complexity of the activity recognition algorithm according to 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.

The last phase of this research consisted in validating the new created and implemented model. In addition, it has the purpose of verifying the usefulness of the online approach and its impact on the phone resources. Tests were designed for each layer presented earlier. Then, a global experiment was done on the global solution to evaluate it.

It is worth noting that this research has led to multiple scientific publications:

- The advances in online activity recognition using mobile phones were published in the proceedings of the *8th ACM International Conference on Pervasive Technologies Related to Assistive Environments* [50], *the IEEE International Conference on Mobile Services (MS)* [51], *the 13th International Conference on*

Mobile Systems and Pervasive Computing (MobiSPC) [52], *The Journal of Sensors* [53] and under review for the *Pervasive and Mobile Computing Journal* [54].

- Our contributions on activity prediction were published in the proceedings of *the 6th international workshop on Human Behavior Understanding in conjunction with the ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)* [55] and under review for the *Journal of Ambient Intelligence and Humanized Computing* [56].
- Our contributions in the field of spatial recognition in smart cities are under review in the *IEEE International Conference on Pervasive Computing and Communication (PerCom 2017)* [57] and the *Expert Systems With Applications Journal* [58].

This encouraging recognition from scientists supports the conclusions of this thesis, the importance of the works realized by our team, and the obtained results.

1.8. THESIS ORGANIZATION

This thesis is divided into four parts separated among seven chapters. The first part is an introduction to the concepts that will be discussed throughout the thesis. It has provided a description of the ubiquity of Smartphones and how are used to bring more value to the activity recognition topic. It also presented the principal limits of mobile phones that we have to surpass during this thesis.

The second part of this thesis reviews the related work within Chapter 3. At first, we bring a discussion of the different types of activity recognition. We describe the process of tracking peoples' movements, how to model them, and how to define an outdoor activity. We also give details on the learning techniques that can be used to detect the users' activities. The second part of this chapter is devoted to related works on the activity recognition. We analyze some important works in both location and motional systems. In the third part, we present some important prediction works on learning users' patterns to estimate their next probable actions. Finally, we highlight works that propose mechanisms to reduce the phone battery consumption.

The third part of this thesis, which comprises Chapters 3, 4, 5 and 6, describes our contributions. Each chapter is the subject of one layer of our Activity recognition system.

In Chapter 3 we propose our first layer. We describe the overall approach of our battery-aware mobile activity recognition system. We detail our proposal in terms of three components: trajectory classification where we classify the ongoing user trajectory in order to detect his visited places. Next, we describe the traditional spatial recognition process and the activity discovery step that are needed to obtain semantic information about the performed activity. Experimental evaluations are also detailed at the end of this chapter where we describe the different datasets used in the experiments as well as the results of our approach.

In Chapter 4, we introduce the second layer NomaBlue. It is a new spatial recognition system that is designed for future cities. We justify during this chapter the need for such system. We present the different part of this collaborative spatial recognition. We give some examples of NomaBlue usages and we demonstrate, by using two datasets, that NomaBlue is efficient and technically feasible. We also show that NomaBlue is capable of creating an efficient dynamic flow of updated information in the smart cities. This updated information improves the user's urban explorations without using the Internet or predefined geographic database.

In Chapter 5, we introduce a new approach of predicting users' next destinations from irregular patterns, we show how we learn users' habits, how we proceed to track the new habits, and how the algorithm behaves when they occur.

A global hybrid approach is described in Chapter 6. An approach that combines both activity recognition and activity prediction in order to reduce the computational complexity of our approach, thus, the mobile battery consumption. The experimentation at the end of the chapter demonstrates how deeply our approach has been experimented to test its accuracy and ability to save batteries.

The fourth part of the thesis, composed of Chapter 7 and the appendix, concludes by presenting a detailed account of the research project highlighting the contributions of this work over previous works. This chapter also addresses the limitations of the proposed model and future works arising from this research. The chapter concludes with a more personal assessment of this experience of initiation into the world of scientific research. The appendix provides further details on some aspects related to the thesis that were not fitting in the text.

PART II

RELATED WORKS

CHAPTER 2

RELATED WORKS

2.1. TRACKING PEOPLES' MOVEMENTS

Human‘s activity recognition has been an active topic of research for several decades. However, only in 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 and RFID, 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 persons 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 the current activity.

Early GPS receivers could perform geo-location in outdoor environments with an accuracy varying between 300 m and 1 km [59]. However the current generation of smartphones have built-in GPS receivers that are accurate to within a few meters outdoors [60]. Outdoor trajectories are retrieved in the form of a GPS collection, each GPS contains some geographic information like; latitude, longitude, altitude, timestamp, speed, orientation, etc. Utilizing the device's GSM module to estimate the distance to cellular base stations [61], or the Wi-Fi module to estimate the distance to the known access points can increase the accuracy of GPS-based positional estimates and assists it where GPS signals are ineffective [62]. Although more power-intensive, these two approaches can be combined to improve positional estimates in outdoor environments.

This series of position points represents a trajectory that needs to be enriched by some semantic information to be more useful for human behavior understanding. The role of next sections is to present how to process to switch semantically the raw sensor data into meaningful human activities.

2.2. MODELING USERS' MOVEMENTS

While the ability to record continuous movement is the foundation of managing human outdoor activities, satisfying application requirements requires more than that. Namely many applications need a more structured recording of movement, i.e. as a temporal sequence of journeys, each one occupying a time interval in the object's lifespan and taking the object from a departure point to a destination point [63]. Daily trips of employees going from home to work and back, weekly journeys of individuals doing shopping at a mall, and movements of going to a restaurant or cinema are examples where the movement of persons is clearly perceived by the monitoring application as countable traveling units.

The emerging concept of semantic trajectory is a new topic for trajectory data analysis; however, the research community's efforts are increasing day by day to carry clear definitions on semantic trajectories. For instance, Spaccapietra et al. [63] propose a conceptual model for trajectories using two alternative approaches for trajectory modeling, based respectively on data types and design patterns.

2.2.1. TRAJECTORIES DEFINITION

Trajectories are defined as spatio-temporal functions that records the changing of the position of an object moving in a space during a given time interval [63]. Each trajectory includes a list of sample points that implement the function of the time-varying point; a list of stops; a list of moves, which lies between two consecutive stops, or between the beginning of the trajectory and the first stop, or between the last stop and the end of the trajectory, see Figure 3 .

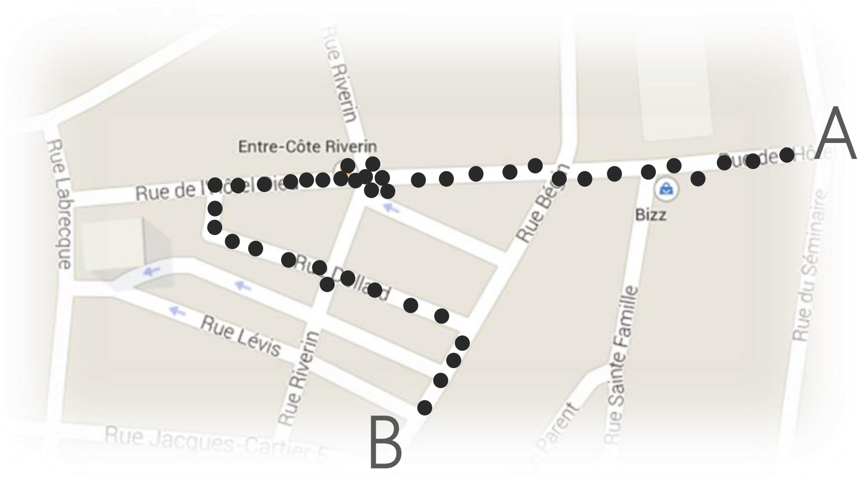


Figure 3: Example of a GPS trajectory of going from A to B

The original sense of the term trajectory denotes the changing position of an object in geographical space. It can be a 3D space (e.g. the trajectory of a plane or a person in a mountain) or a 2D space (e.g. the trajectory of a car on the highway). A trajectory is called spatio-temporal if spatial coordinates are used to express the position of the traveling object in the form of time stamped elements. Most frequently, the traveling object is geometrically represented as a point (e.g., a person, an animal, a car, a truck, a plane, a ship, a train). Yet the traveling object may have a surface or volume geometry (e.g. clouds, air pollution, oil spills, avalanches), in which case both change in position and change in shape may lead to a change in the trajectory.

2.2.2. THE COMPOSITION OF TRAJECTORIES

A trajectory has two facets: (i) a geometric facet which is represented by a spatio-temporal recording of the traveling point position. It is a delimited segment (i.e., a single continuous subset) of the spatio-temporal path covered by the object's position during the whole lifetime of the activity (ii) a semantic facet which is represented by the information that conveys the application-oriented meaning of the trajectory and its related characteristics.

The main idea is to associate semantic information to every trajectory component as follows:

Stops. Stops are a set of points where the user speed is too slow or null. For example, if trajectories are seen by the application as moves between cities, the database must be able to store and return in which city a stop is located. The same trajectories may be seen by another application as moves between countries, thus calling for a link between stops and countries [63]. For migrating birds, stops may be in geographical regions of interest to the birds or to the ornithologist.

Moves. Individuals need to move from a geographic area to another, this movement behavior is characterized by a set of points that share a high velocity depending on the type of the mode of transport (pedestrian, car, bus, train, etc.). For example, an application monitoring peoples' use of a train network may need to record which train has been used for which move, e.g. a trajectory of a traveling person may consist of a first move using the first train from City A to City B and a second move using the second train from City B to City C.

Begin and end of the trajectory. A trajectory is defined by start and end points. These points are important for trajectory analyses because they often represent an important event in user behaviors, e.g. start point is home and the end point is the work office. For instance, a company that monitors business trips done by its salespeople may restrict trajectory analyses to originate and end in the company's buildings.

2.3. Users' Outdoor Activities

It is necessary to have some common understandings on people's outdoor activities to be able to recognize them automatically. In the next point, we are going to see that depending on the nature of the studied trajectory part, we are able to learn the executed activity type.

An outdoor activity can be seen as a quality or state of being active or a vigorous or energetic action made in an environment other than home [42]. Activity thus encompasses both state and energy spent to achieve some outdoor task.

Activity recognition is defined in [45] as a technique capable of associating sensor readings and other inputs to a label taken from a set of distinct activities. The task therefore involves determining a set of activity labels and assigning sensor readings and other inputs to the appropriate activity labels.

2.3.1. Defining Activity Labels

Generally, in mobile and ubiquitous computing applications, the set of considered activity labels is constrained by the nature of sensors at hand and depends on the final application. A fitness monitoring system will define stationary, walking and running as activities whereas an automatic diary application will use just one traveling activity and divide the stationary state into several depending on the location of the user [45] such as staying at home and working in the office. Consequently, proposing a common understanding about labels is a challenging process since the level of activity detail depends on the use of the application. In many cases, human activity labels are assigned based on the context data and background knowledge. The context data includes sensor data, time and user inputs. Background knowledge is usually provided by GIS experts. The ideal scenario is to mine activity labels automatically which is a challenging task. We explore this weakness to bring novelties to the outdoor activity field. For example, we can extract the geographic information automatically from GIS platforms like OpenStreetMap [64]. Next, we can explore the ontology used in these platforms to extract contextual information about the concerned geographic entities [65], e.g. name: university; number of floors: 5; type: educational; entity type: building;…etc.

2.3.2. LEARNING TECHNIQUE

Globally, activity recognition models rely on matching the context data to activity labels. The activity learning techniques pass by three steps as described in Figure 5.

2.3.2.1. Sensing

Thanks to current sensor technologies, large scale capture of the evolving position of individual mobile objects has become technically and economically feasible. Mobile wearable tracking devices, e.g., phones and navigation systems, sense the movement of people represented by positioning records that capture geo-location, time, and a number of other attributes**.** Sensing step 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, humidity sensors, temperature sensors, ambient light sensors…etc.) [9] to extract pertinent information about the current activity. Some of the most important sensors are described in Table 1.

Table 1: A set of mobile phones sensors

Sensor	Description
GPS	Receives information from GPS satellites and then calculates the geographical location of the device
Accelerometer	Measures the acceleration force that applied to the device, including force of gravity
Ambient temperature	Measures the ambient room temperature
Gravity sensor	Measures the force of gravity that is applied to the device, in three axes (x; y; z)
Gyroscope	Measures the device’s rotation in three axes (x; y; z)
Light sensor	Measures the ambient light level (illumination)
Linear acceleration	Measures the acceleration force that applied to the device, force of gravity is excluded
Magnetometer	Measures the ambient geomagnetic field in three axes (x; y; z)
Barometer	Measures the ambient air pressure
Proximity sensor	Measures the proximity of an object relative to the view screen of a device.
Humidity sensor	Measures the humidity of ambient environment

As shown in Figure 4, three families of smartphone-based positioning solutions have been studied extensively: satellite-based solutions, sensor-based solutions, and RF (radio frequency) signal-based solutions [19]. Although the nature of sensors has become much more diversified, a good inference system is capable of accurately recognizing activities using the minimum number of sensors. We aspire to take this challenge in our work in order to recognize users' activities using a minimum number of sensors in order to preserve users' mobile resources.

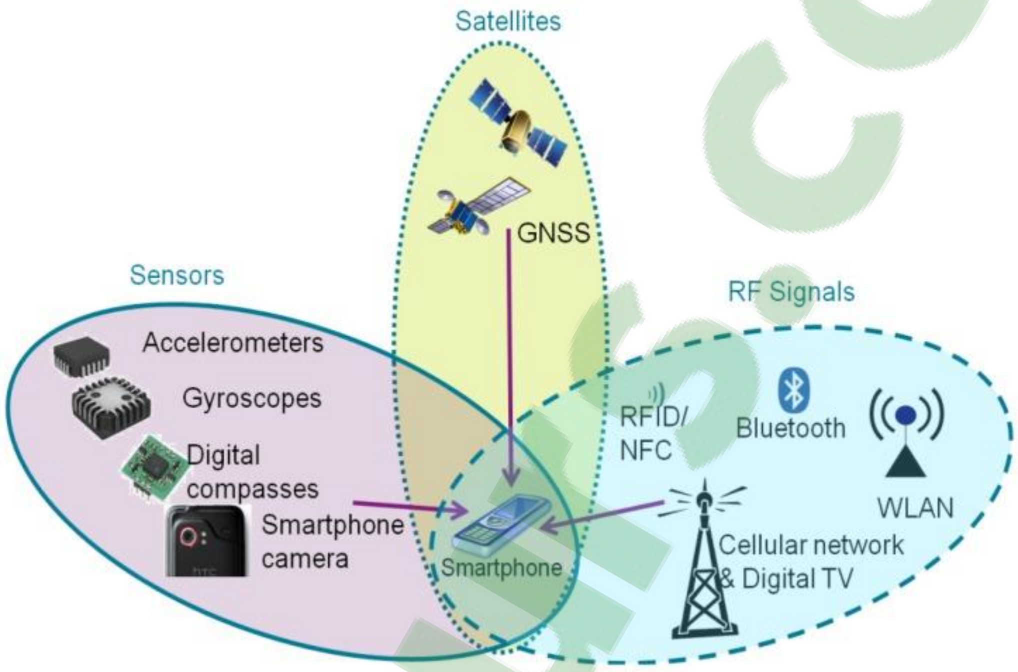


Figure 4: Three families of smartphone-based positioning solutions. Source:[45]

2.3.2.2. Pre-Processing

In this step, activity recognition systems transform raw data into a reduced representation called a feature vector [45]. The accuracy of the system depends strongly on how data is represented. Features should help differentiate between the studied activities. They can be low-level, such as mean and variance computed on a specific physical signal or high-level, like the user's abstracted location as estimated from cell tower signals.

2.3.2.3. Inference

The inference module is the core of an activity recognition system and the place where machine learning models are implemented. At this stage, the feature is transformed using some background knowledge into a meaningful human activity, e.g., staying at home, walking and driving, going to the restaurant…etc.

The treatment of these feature vectors varies from one solution to another. As the size of vectors supported in this inference process has an impact on the precision of algorithms (accuracy increases with the number of supported vectors), the majority of related works [66]–[69] has opted for a post treatment of these vectors'. They are processed on a specific time (end of the day, week, month, etc.) to make sure that the inference model has enough information to make a decision.

The post treatment of historical records usually uses massive databases of users feature vectors to guarantee a good accuracy of algorithms. It is made generally on desktop computers instead of on the data collection device (smartphone) to avoid problems linked to: (i) the reduced storage and calculation capacities and (ii) the complexity of performing an online inference process. This architecture is illustrated in Figure 5 in which sensing is limited to the phone's own sensors and pre-processing performed on the phone and the inference process on desktop computers. Unfortunately, these techniques are limited because they cannot be applied to fields where we need instant information about the user activity like police, assistance, emergency management…etc. Consequently, it is important to replace these offline solutions by online solutions in which the inference process will be made on smartphones. This online recognition of people's activities offers the possibility to understand what people are doing at the present moment, and estimates their actions in the future. It is in this context that our research fits, in which we exploit the increasing computational power of mobile phones in order to propose an absolute online solution that recognize users' activities during its execution while using a minimum of phone resources.

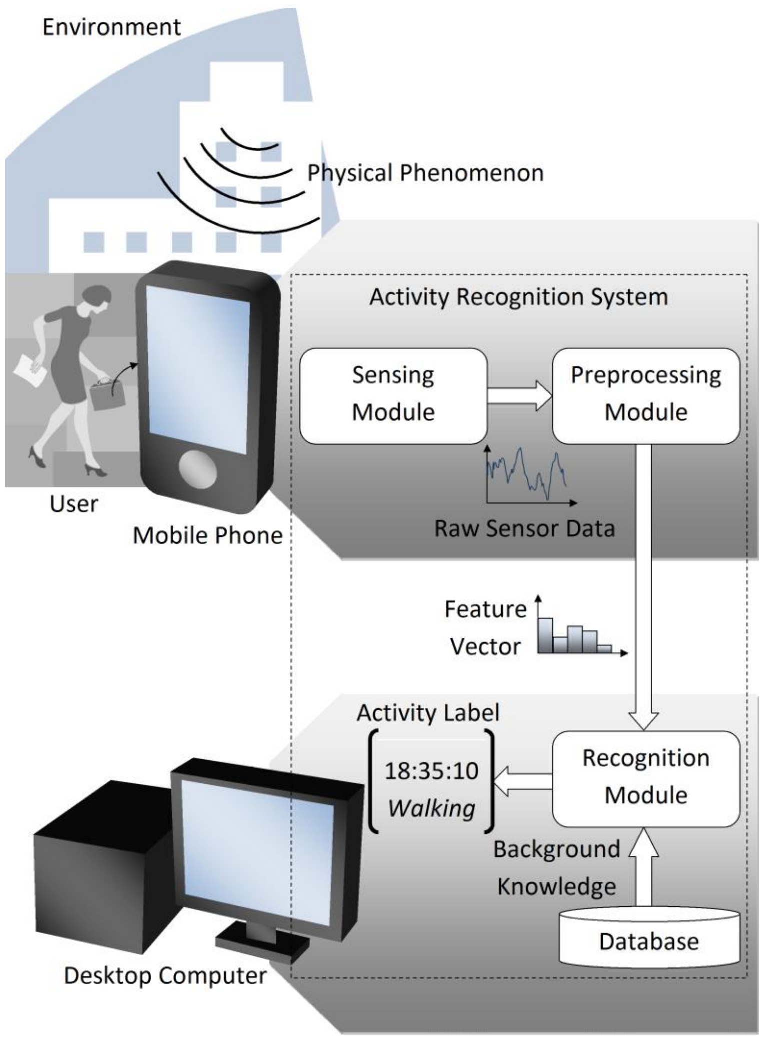


Figure 5 : Architecture of a mobile-phone-based activity recognition system. Source:[45] Sensing and pre-processing are performed directly on the phone while inference is made on a desktop computer with greater computational resources.

2.3.3. THE SEMANTIC OF TRAJECTORIES

People activities are divided into two behaviors: stationary and non-stationary, 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 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 Figure 6). Based on these concepts, users’ activities can be divided into three types:

1. Stationary activities that are characterized by stops.

2. Activity with movements are non-stationary activities that require movement over a time interval.

3. Moves are a set of behaviors that aim to move from a geographic area to another using transportations (car, bus, train, etc.).

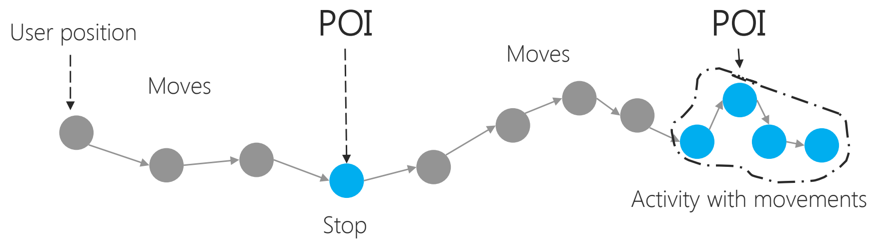


Figure 6: Relation between moves, stops and activities with movements

Recognizing the activities defined above vary in function of the type of sensors and the application field. Before studying the inference techniques, we are going to go further in the definition of activity types.

2.3.4. Location Versus Motional Activity Recognition

The best way to introduce the locational activity recognition may be to associate the performed activity to the better known and nearest geographic entity. In the latter, the objective is to estimate the position of a device anywhere as accurately as possible using techniques such as multilateration [70] or assisted GPS[71]. In contrast, locational activity recognition is interested in the meaning of the user's location rather than its coordinates. In other words, a locational activity recognition system distinguishes between locations only if it helps to determine what the user is doing [45]. Typical locations which reflect the user's activity are the workplace, home, restaurant, shopping center, gym, cinema…etc.

In motional activity recognition, the user's activity is abstracted as a motion state or a mode of transportation. Some motion states can be identified from cellular signals even without any knowledge about the location of observed cell towers [45]. For example, fluctuations in GSM signals have been shown to be sufficient to determine with reasonable accuracy whether a mobile phone carrier is walking, driving a motor car or staying stationary [72], [73]. The accelerometers embedded in recent smartphones can serve the same purpose and further distinguish between finer movements such as sitting, standing or running [74], [75], or even further, to detect stairs walking [76] or fall detection using gyroscope sensors [36].

As seen above, activity types differ in several ways. Consequently, the learning techniques of these activities vary too. In the next section, we are going to present the most important works and learning techniques that tried to recognize users' activities.

2.4. Activity Recognition Models

Activity recognition models vary on many axes. We will introduce some important data mining algorithms that have played an important role in pattern recognition field [72]–[75], [77], [78]. We are going to highlight three axes to be able to distinguish the existing works; the first axis is the manner of sample processing; secondly if the approach used is supervised or not and finally, if the algorithm used is incremental or not. We will end up with studying some existing works while developing a critical analysis.

2.4.1. Generative Versus Discriminative Models

Models implemented to recognize activity fall in one of two categories. Generative and discriminative models. Generative models [79] specify a joint probability distribution

$P_{X_1 X_2 \ldots X_n Y}$ over features $X_1, X_2, \ldots X_n$ and the inferred activity Y. They can either model data directly or be used to form a conditional distribution $P_{Y|X_1 X_2 \ldots X_n}$ through the use of Bayes's rule. Model parameters are usually estimated to maximize the likelihood of training data and new instances are classified to the most probable class given the features [45]. A generative probabilistic distribution is a principled way to model many machine learning problems and activity recognition problems, for instance, the Reality Mining project in [66] investigates how cell towers information and Bluetooth proximity data can complement each other to help infer important locations and social relationships using the Hidden Markov Model. Moreover, the Bayesian Network (BN) used in [80] and the Dynamic Bayesian Network (DBN) in [68] represent examples of other generative works.

The discriminative models provide a model only of activities conditional on features. This can either be done by specifying the conditional probability distribution $P_{Y|X_1 X_2 \ldots X_n}$, or by specifying decision boundaries. The discriminative algorithms adjust a possibly non-distributional model to data optimizing for a specific task, such as classification or prediction. For instance, in [69], Farrahi and Gatica-Perez consider a subset of 30 users and 121 consecutive days from the Reality Mining dataset [66] to compute both location and proximity features at two different time scales (a fine-grained one every 30 minutes and a coarse-grained one every 3-4 hours). The predictive power of those features is then evaluated for two different tasks. Features are tested alone and in pairs of one location and one proximity feature. Using a Support Vector Machine with a Gaussian kernel, the authors aim at (i) classifying a user's day as a weekday or weekend and (ii) classifying a user as a business or engineering student. Furthermore, the Artificial Neural Network (ANN) in [81] and C4.5 Decision Tree (DT) used in [74], [75] represent other examples of discriminative models. Although generative models usually require more training data and are sometimes outperformed by discriminative models.

They hold a major advantage over the latter in their ability to generate values of any variable in the model. Therefore, generative models can simulate data and provide a better understanding of the underlying processes [45].

2.4.2. SUPERVISED VERSUS UNSUPERVISED LEARNING

In addition to generative and discriminative models, learning techniques can be differentiated using another dimension based on the two ways of learning training (supervised and unsupervised). Most activity recognition systems rely on supervised learning. In this type of learning, training data is collected by the sensing module and transformed into a set of instances by the pre-processing module. Each training instance is then labeled for the user's true activity when data was collected. The supervised inference model generalizes from training data the activity labels of the unseen instances in the test situations, it needs a dataset consisting of both features and labels. The task is to construct an estimator which is able to predict the label of an object given the set of features.

Constructing the training set may represent a real challenge and sometimes a scientific lock because of the difficulty encountered during this process. One way to acquire the ground truth is to prompt for activity labels on the fly as the user performs his daily activities. This method has the advantage that the information collected is fresh in the user's mind and therefore more accurate. However, providing labels using the interface of a mobile phone is not easy in practice. Also, not all activities can be labeled as they are performed. For instance, interrupting a meeting to input an activity label is not realistic. The authors of [82] proposed an online human activity recognition system on smartphones that classifies basic movements of a user, such as walking, running, sitting and standing. They developed a mobile application called activity logger in order to construct the training set, they demand for users to select the activity to be performed, put the phone into the packet and start to perform the related activity. For each activity, the application creates different training data files in which raw data from the 3-axes of the accelerometer is being logged.

An alternative approach is to label data after collecting data, e.g. at the end of the day, using a diary application or a paper notebook [83]. The issue with this method is obviously the low temporal accuracy and the limitations of human memory. In either case, activity labels provided by the user are subjective. For example, one user may interpret being at work as working while another user may only label as working time intervals during which he is actively engaged in his work. Secondly, training an activity recognition system in a supervised fashion requires considerable involvement from the part of the user who not only has to regularly charge his device and carry it with him for several weeks but also needs to spend time labeling his data [45].

Unsupervised learning aims to relieve the user from the burden of labeling, usually by clustering data based on some distance measure. With unsupervised learning it is possible to learn larger and more complex models than with supervised learning. This is justified by the fact that supervised learning tries to find the connection between two sets of observations. The difficulty of the learning task increases exponentially in the number of steps between the two sets and that is why supervised learning cannot, in practice, learn models with deep hierarchies.

One of the most common unsupervised algorithms is K-means, which tries to minimize the total intra-cluster variance [84]. K-means takes as a parameter the number of clusters, for example three clusters for stationary, walking and driving [73]. Moreover, our work in [50] succeeded in recognizing activities without any users' intervention. The approach combines an online classification method based on K-means with spatial techniques to retrieve users' activities automatically.

In many cases, the exact number of clusters depends on the user's personal habits. In particular, this is the case for locational activities. A child may have three significant locations: home, school and park while other users may have many more including workplace, restaurant, supermarket and library. More recent algorithms, such as DBSCAN have been used in that context [85]. DBSCAN finds the number of clusters automatically from the estimated density distribution of data points [86]. In either case, devising an appropriate distance measure is a difficult problem and can be quite subjective.

2.4.3. Incremental Versus Non-Incremental Learning

Generally, a classification problem is defined as follows: a set of N training examples of the form (x ,y) is given, where y is a discrete class label and x is a vector of *d* attributes (each of which may be symbolic or numeric). The goal is to produce from these examples a model $y = f(x)$ which will predict, with high accuracy, the classes y of the future examples *x* [80].

To solve this problem, traditional statistical analysis method (non-incremental methods) would load all training data into memory at once. However, compared to the explosive growth of today's information, the storage capacity is far from adequate. Moreover, when it comes to temporal series, traditional data mining algorithms have shown many limitations. As such, incremental learning algorithms are efficient methods to solve these problems. In incremental learning, the sample size increases throughout the training phase where it is supposed that data items arrive one after the other and that the whole dataset is not fully available at the beginning of the learning process.

One important problem met during the incremental learning is the change of the distribution of the observed variable during the learning process, this phenomenon is called the concept drift, it means that the statistical properties of the targeted variable, which the model is trying to predict, change over time in unforeseen ways [87]. This causes problems because the predictions become less accurate as time passes.

Incremental learning should be differentiated from online learning which conveys other meanings in the machine learning communities: in opposition to batch learning algorithms, online learning algorithms try to learn and update the current classifier using only the new available data without using any past observed examples [88] (historical data).

2.5. ACTIVITY RECOGNITION WORKS

We divide the activity recognition works into two types of approaches: location and motional approaches. Depending of the activity recognition application, systems tend to one specific type of systems, for instance, monitoring patient physical activities use motional systems in order to detect fatigue, however, a place recommender system will use location systems to propose an important place to visit.

2.5.1. LOCATIONS APPROACHES

The first main category of activity recognition approaches determines the user's activity in terms of location. We review below the major activity recognition systems that tried to recognize human activities basing on the spatial analysis of the users' context, i.e., the environment in which the user is located.

2.5.1.1. *Non-Mobile Approaches*

We mean by non-mobile approaches, a range of offline works that treat users' movements after collecting them and not at the same time. To remove any sort of confusion, note that these systems may use mobile devices to track peoples' traces but they are still non-mobile approaches since the inference process is made on desktops.

a) ***Palma et al.'s Work: CB-SMoT***

A clustering method is proposed in [89] called CB-SMoT (Clustering-Based Stops and Moves of Trajectories). This algorithms is designed to infer semantic information from trajectories. It is a clustering method based on the speed variation of the trajectory. This method first 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 (see Figure 7). The intuition of this method is that the parts of a trajectory in which the speed is lower than in other parts of the same trajectory, correspond to interesting places, i.e. places that the user has probably visited.

In a tourism application, for instance, a trajectory of a tourist that is visiting a new city would be something like: visit an important monument, visit a museum, go to his hotel, go to a night-club, and return to the hotel. Probably his/her trajectory has a lower speed around these places than it has in other parts of the trajectory where he was moving from one place to another. In a traffic management application, for instance, the speed of car trajectories will be lower in traffic jams, traffic lights, roundabouts, and electronic velocity controllers. Following this reasoning, they proposed a clustering-based algorithm to find low speed regions.

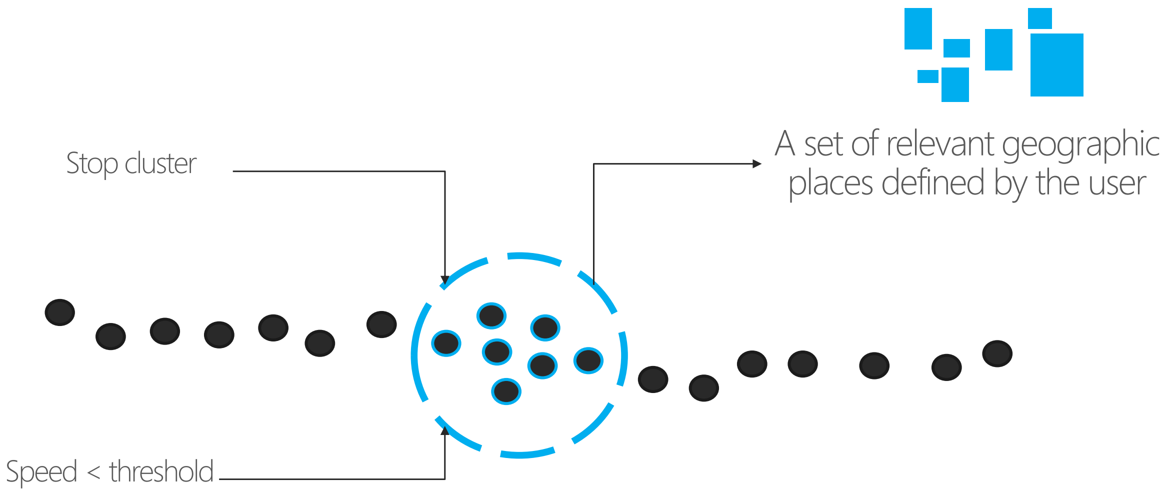


Figure 7: A descriptive diagram of CB-SMoT approach

They proposed a two-step algorithm to extract stops and moves. In the first step the slower parts of a trajectory, which they call potential stops, are identified using the variation of the DBSCAN [90] algorithm that considers one-dimensional line (trajectories) and speed. In the second step, the algorithm identifies where these potential stops (clusters) found in the first step are located, considering the geography behind the trajectories. CB-SMoT takes each potential stop (clusters) and tests both intersection and minimal stop duration with the candidate stops. In case that a potential stop does not intersect any of the candidate stops, it still can be an interesting place. Then, in order to provide this information to the user, the algorithm labels such places as unknown stops.

In [91], the same authors present a free software called Weka-STPM (Semantic Trajectories Pre-processing Module) that has been constructed into Weka [92] to preprocess raw trajectories in order to transform them into semantic trajectories. As a module of Weka, it allows the user to directly apply several mining algorithms available in Weka to mine semantic trajectories. Weka-STPM is the first tool for semantic trajectory reprocessing for data mining, it uses CB-SMoT in first order to generate the semantic trajectories.

The lack of this method is that data processing is non-incremental. It is not possible to real time analyze users' mobility using this solution. Moreover, as seen earlier, users' activities are divided into three families; stops, moves and moving activities, this solution detects only stops and moves and fails to recognize moving activities.

Introducing DBSCAN

DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) is an important density clustering method. It was developed by Ester [90] and it has influenced many other density-based clustering methods. The main advantage of DBSCAN is the capability to find clusters of many different shapes in a spatial dataset. Figure 8 shows some datasets where it is able to discover the intuitive visual clusters.

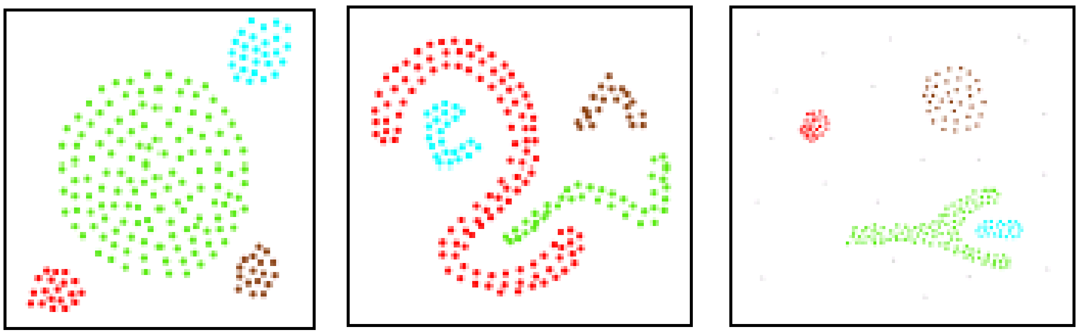


Figure 8: Three datasets with its respective clusters [90]

DBSCAN looks for some core cluster and tries to expand it by aggregating the nearest neighbors that met some conditions. In order to understand what these conditions are, it is necessary to present some related concepts. In fact, DBSCAN needs two parameters: *MinPts* and *Eps*. *MinPts* is a density measure that indicates the number of points needed in the neighborhood of a point in order to assign that point and its neighbors to a cluster. *Eps* is a distance used to delimit the neighborhood. Formally the neighborhood is defined as follows:

Eps-neighborhood of a point: The Eps-neighborhood of a point p, denoted by $N_{Eps}(p)$ is defined by $N_{Eps}(p) = \{q \in D | dist(p,q) \leq Eps\}$, where D is the whole dataset [89].

The Eps-neighborhood of p represents all points inside a circle centered in p and with a radius Eps. The $dist(p,q)$ is a function that returns the distance between the point p and the point q.

Noise: Let $C_1, C_2, \ldots, C_n$ be the clusters of the database D with respect to Eps and $MinPts$. It is defined as noise the set of points in the database D not belonging to any cluster C_i, this set can be defined as: $Noise = \{p \in D \mid \forall i : p \notin C_i\}$.

DBSCAN's advantage is that it has a good performance on large spatial databases, outperforming other density-based algorithms like CLARANS [93]. DBSCAN's complexity is $O(nlogn)$. It requires little knowledge about the spatial domain, since it needs only two parameters in order to be performed: $MinPts$ and Eps.

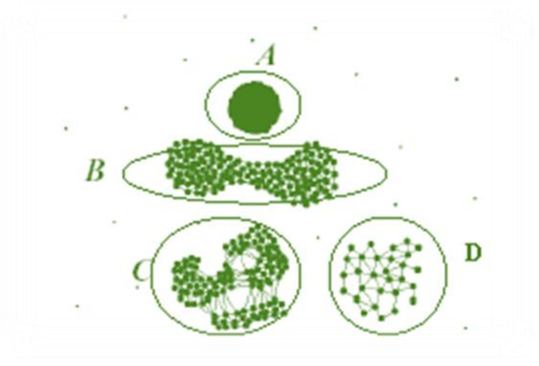


Figure 9: Clusters with different granularities [86]

The main disadvantage of DBSCAN is its difficulty to find clusters of different densities, like those shown in Figure 9. That figure shows 4 distinct clusters, where the cluster A has the greatest density among them and the cluster D has the lowest one. The clusters B and C have intermediate densities.

b) Huang et al.'s Work: STPA

The work of [94] infers activities from users trajectories. This paper presents an approach using spatial temporal attractiveness of a POI (place of interest) to identify activity locations and durations from raw GPS trajectory. The algorithm they propose finds the intersections of trajectories and spatial-temporal attractiveness prisms to indicate the potential possibilities for activities. Authors consider POIs as normal facilities where their attractiveness is decided by some intrinsic factors. For example, larger shopping malls usually attract more people if they go for purchases. They define this kind of factor as static, they are described as follows:

- The size of POI: generally the greater the size is the higher the attractiveness is under the same other conditions and the longer the duration is;
- The popularity of POI: generally the greater the popularity is the higher the attractiveness is;
- Categories of POI: those POIs that have close relationship to individuals' daily life normally have higher attractiveness, whereas functional POIs for special needs normally reflect lower attractiveness.

Because POIs discussed in their work are related to certain type of activities, the attractiveness should include the factor to describe this kind of relationship. One of the activity's basic components is starting time since a different time of the day implies different activities. Therefore POIs attractiveness will change as time flies forward. For example, restaurants are more attractive at midday when people normally choose to go for lunch. This kind of time varying factors are defined as dynamic factors named STPA. It is a function of the POI's category C, the time of day t, and the type of day D (weekday/holiday). This function is called dynamic function $f(C,t,D)$ and is written as follows:

$$STPA_p = F(S, \varepsilon_p, C, f(C, t, D)) \quad (1)$$

To implement STPA for practical activity identification, quantification method customizing STPA to an appropriate size is necessary. Since people will stay/move in an activity place, the area of this place can naturally describe the region that contains the activity. Authors use circle centered at POI P's location whose area equals POI P's size S to construct an attractiveness area in geographical plane. Then, the customized STPA of the POI P can be defined by Equation (2), where r is the standard radius of the circle; *w1* and *w2* are weighted coefficients of ε_p and C respectively, $r\,(t)$ is time varying radius.

$$\begin{gathered} r = \sqrt{S\,(w1\,\varepsilon_p + w2\,C)} \\ r(t) = f(C, t, D)\,r \\ STPA_p = F(r(t)) \\ w1 + w2 = 1 \end{gathered} \quad (2)$$

The experiments use one month of GPS trajectories provided by 10 volunteers. However, the small dataset used for experimentation appears unrepresentative for real users' daily routines, especially as the work seems very sensitive to the number of POIs since the accuracy drops with the increasing of POI's number which raises questions about the ability to use such work in big real datasets.

2.5.1.2. Mobile approaches

Mobile systems include activity recognition works that tried to achieve all the activity recognition steps (sensing, prepossessing and inference) on the mobile phone without the intervention of desktop computers.

a) Kang et al.' work: Time-based clustering

Kang et al. in [95] utilized the access point MAC address of a WI-FI network to capture location data on campus. They developed a time-based clustering algorithm to extract places taking advantage of the continuity of the WI-FI positioning. The developed application is called Place Lab. 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. The basic idea of their approach is to cluster locations along the time axis. As a new location measurement is reported, the new location is compared with previous locations. If the new location is moving away from previous locations, the new location is considered to belong to a different cluster than the one for the previous locations.

When a new location measurement event is generated by Place Lab, the cluster function in Algorithm 1 is invoked where d and t are the distance and time threshold parameters. The current cluster c_1 is the set of location measurements that belong to the current cluster. The pending location $ploc$ is used to eliminate outliers. Even if the new location is far away from the current cluster (distance is larger than the distance threshold d), the algorithm does not start a new cluster right away with the new location. Instead, the algorithm waits for the next location to determine if the user is really moving away from the cluster or the location reading was just a spurious outlier. The Places contain significant places where the user stays longer than the time threshold t.

When a new location measurement is generated from Place Lab, the algorithm compares the distance between the mean position of the current cluster and the new location with the distance threshold d. If the distance is less than d, the new location is added to the current cluster and the pending location is set to null (lines 2-3). If the distance is larger than d, the algorithm checks if there is a pending location (line 5). If there is a pending location, the algorithm closes the current cluster and checks the time duration of the current cluster (the difference between the oldest and newest locations in the cluster). If the time duration of the cluster is longer than the time threshold t, the cluster is added to the significant places (lines 6-7). Then, the algorithm starts a new cluster with the pending location and checks if the new location can be in the same cluster as the pending location (lines 8-14). If the distance between the new location and the current cluster is larger than d but there is no pending location, the algorithm set the pending location to the new location (line 16).

When a cluster is added to the set of significant places, the Algorithm 1 checks if the cluster is the same as one of the existing clusters (their centroids are within distance $d/3$ of each other). In order to identify more fine-grain places, authors use a smaller threshold ($d/3$) than the one used for forming clusters (d). The smaller threshold works because the difference between the averages of the location measurements over a period of time is likely to be much smaller than the difference between individual location measurements. If the newly added cluster is close enough to one of the clusters, then the two clusters are merged.

This algorithm is simple and works in a novel 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.

Algorithm 1: Time-based clustering algorithm

```
Cluster (loc)
Input:    measured location loc;
Output:  current cluster cl, pending location ploc, significant places
Places;
1:  if (distance($c_1$, loc) < d )
2:      add loc to cl
3:      ploc = null
4:  else
5:      if (ploc != null)
6:          if (duration(cl) > t)
7:              add cl to Places
8:          clear cl
9:          add ploc to cl
10:         if (distance(cl, loc) < d)
11:             add loc to c
12:             ploc = null
13:         else
14:             ploc = loc
15:     else
16:         ploc = loc
```

b) Spinsanti et al.'s Work: Where You Stop

In [96], 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 quantified with a probability. After that, depending on the kind of activities associated to the identified place, the trajectory is classified into a probable trajectory behavior. In this work they 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 out of the transportation means to reach the final destination walking. During this time interval the person is not traceable, which justifies the use of probability to find the visited place (see Figure 10).

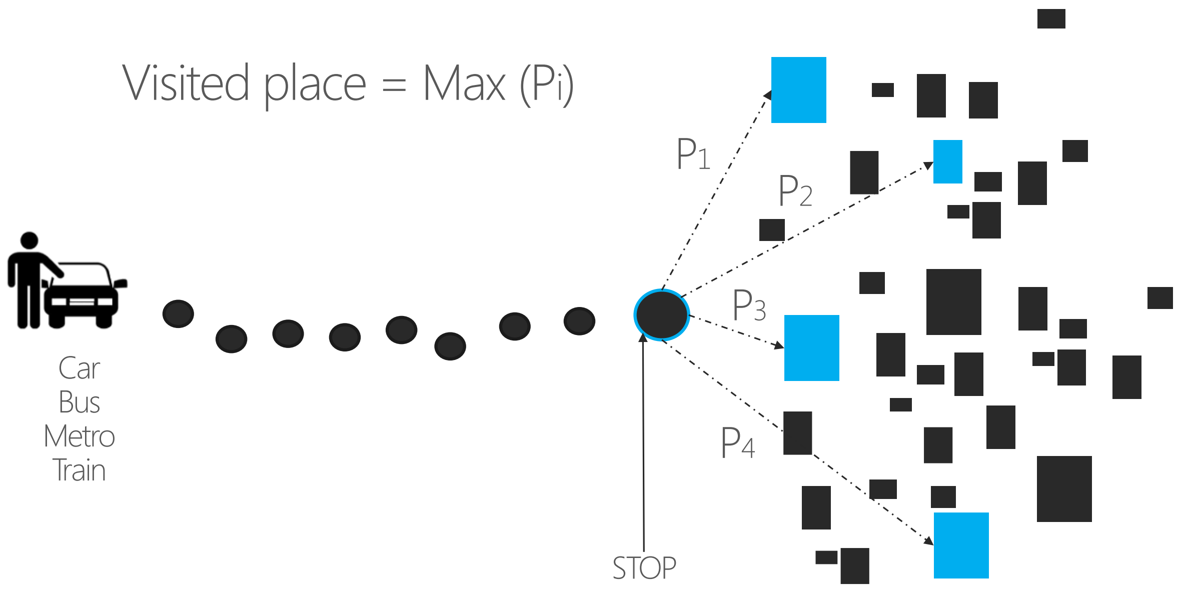


Figure 10: A descriptive diagram of Spinsanti's work

Movement data is collected from tracking devices, and trajectories are reconstructed from them. Given the trajectories, the first step is to identify the stops of the trajectories. The stops are the portion of trajectories where the movement stops for a given time duration and where they assume the user is performing some activity. For each stop of a trajectory, they compute the set of POIs that can be associated to it. Two conditions are taken into account: (i) having enough time to go and come back from a stop to POI and (ii) having enough time to visit the POI. This means that the amount of time a person could spend in a place is not the complete stop duration, but the time needed to cover the distance between the POI and the stop must be taken into account. So, during this step they disregard all the stops that cannot be associated to "interesting" places, such as the stops with a very short time duration that are typical of the movement itself, such as a traffic light. The component "Visited POI" takes as input the list of stops and the POIs and computes a ranked list – based on probabilities – of possible points of interests visited by the user. This probability is calculated in function of a spatial probability $SpatP$ that is linked to distance, and the temporal probability $TempP$ that is linked to time spent. The formula that computes the probability related to the distance between the $POI\ P_i$ and its associated stop is illustrated in Equation (3), $d_{i,x}$ is the distance between the $POI\ P_i$ and the stop Sx.

$$SpatP_{i,x} = \frac{\frac{\sum_k d_{i,k}}{d_{i,x}}}{\sum_k \frac{\sum_k d_{i,k}}{d_{i,k}}} \tag{3}$$

The closer is the *POI* to the stop, the higher is the value returned by this formula.

On the other hand, the formula that takes into account the average visit time of the $POI\ P_i$ is illustrated in Equation (4). TM_i is the average visit time of the POI P_i and $TE_{i,x}$ is the maximum time that a person has to spend to visit the $POI\ P_i$, that is to say the difference between the duration of the stop Sx and the time needed to reach P_i and to come back to the stop.

$$TempP_{i,x} = \frac{\frac{TM_x}{TE_{i,x}}}{\sum_x \frac{TM_x}{TE_{i,x}}} \tag{4}$$

The total probability is calculated as described in Equation (5) where α is a weight used to give more or less importance to the distance criterion.

$$P_{i,x} = \frac{TempP_{i,x} + \alpha\, SpatP_{i,x}}{\alpha + 1} \tag{5}$$

The lack of this method is that users' activities are broader to be categorized into one type based on stop behavior. Indeed, this work fails to recognize activities that require a movement during its execution like walking in a park or shopping. Moreover, 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 contain several profiles.

c) Takeuchi et al.'s Work: CityVoyager

CityVoyager presented in [97] is a recommendation system designed for mobile devices. It recommends shops to users based on data analyzed from their past location history. This approach applies location data to an item-based collaborative filtering algorithm. This algorithm is used in many online recommendation systems. It transforms location data history into a list that contains the names of each user's frequently visited shops and rating values which indicate how fond the user is of each shop. This list can be directly used as input to the filtering algorithm to make recommendations in the exact same manner as conventional recommendation systems. The transformation of data is done using their place learning algorithm, which can find users frequented places completely with their proper names (e.g. "The Ueno Royal Museum"). No explicit user manipulation is required in the process. In addition, authors claim that their system is able to further narrow down shops based on prediction of user movement and geographical conditions of the city, such as the layouts of streets, resulting in more timely recommendations.

In the place learning phase, raw location data from GPS is reconstructed into a list of each user's frequently visited shops. This process can be further divided into two sub-phases: detecting visits to shops, and finding frequented shops.

1. In the visited shops detection step, authors use the unavailability of GPS signals as evidence that the user has gone indoors. GPS signals cannot penetrate through most building walls, so visits can be detected fairly accurately using this method. The system judges that the user has visited a shop when GPS signals are continuously unavailable for a period of time longer than a threshold. The system then records the location of the visit, and also records the length of time that signals were lost, as the approximate duration of the visit.

2. In the second step, the authors try to find the frequented shops. Following the above procedure, locations of a user's past visits are plotted. By analyzing these plots of past visits, it is possible to reveal which shops the user has frequently visited. Unfortunately, this task is more difficult than it seems, since GPS is known to produce errors, of around 10 meters in clear-sky conditions and significantly larger in urban areas. The proposed place learning algorithm of past visits implies the existence of one or more frequented shops in the nearby area, but it is not possible to know exactly which shop(s) the user has frequently visited. The best that can be done is to estimate the frequented shops by the proposed place learning algorithm.

Authors have evaluated the performance of their system at Daikanyama, one of Tokyo's most popular shopping districts where they show the overall effectiveness of the approach. Although some aspects of the system still need further evaluation to be fully validated like the fail reported by the authors during the experimentation step when a shop which only sells fashion items for women was recommended to a male user.

Authors track the visited shops by the loss of GPS signals, though, it is known that GPS signals frequently become 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 the case when the shop doesn't include many obstacles that hinder the diffusion of the GPS signal, these shops are surely unrecognized by this type of work.

They 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 a heavy manipulation of the historical records of users' visited shops. Authors seem to neglect the limited mobile's resources since there is no support for the limited battery life and there is no effort perceived to online detect and find frequent shops.

Besides, as perceived in the CityVoyager system architecture presented in Figure 11, the user app needs to be continuously connected to the server in order to receive recommendation data. Unfortunately, this constant connection with the server can generate both excessive energy consumption and heavy use of bandwidth.

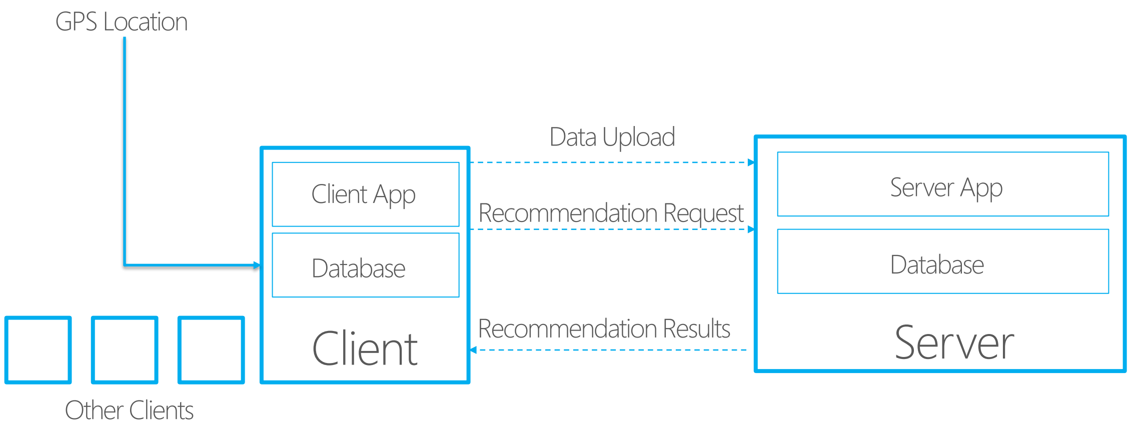


Figure 11: The CityVoyager system architecture

2.5.2. Motional Approaches

The second main category of activity recognition approaches outputs the user's activity in terms of motion state. We review below the major activity recognition systems for mobile phones which fall into that category. As previously, each system is named after the project in which it was developed followed by a short description.

a) Miluzzo et al.'s Work: CenceMe

The CenceMe system [74], [75] combines inference of activity, travel, conversation and presence of individuals using a Nokia N95 mobile phone. CenceMe implements a split-level classification scheme whereby inference is run in part on the phone and in part on a back-end server to improve scalability. Activity recognition is performed directly on the phone using on-board accelerometer data to determine whether the user is sitting, standing, walking or running. The accelerometer sensor and event detector are Symbian C++ modules that act as daemons producing data for corresponding JME client (Java Micro Edition Embedded Client) methods. A reprocessing module fetches raw accelerometer data from the local storage component and extracts lightweight features including the mean, standard deviation and number of peaks of the accelerometer readings along the three axes of the accelerometer. CenceMe's inference module is based on a C4.5 Decision Tree and evaluated in a small-scale supervised experiment involving eight users, student and faculty from Dartmouth College. These users annotated their actions over a one-week period at intervals of approximately 15 to 30 minutes. With an average accuracy of 78.89% reported, figures are up to 20% lower than those reported using custom hardware [98]. In particular, the system has difficulties differentiating between sitting and standing, and between walking and running.

The position of the phone was found to impact recognition accuracy. Specifically, holding the phone in a trouser pocket or at the belt produces similar results but having it at a lanyard position yields poor accuracy when classifying sitting and a slightly lower accuracy for running. The length of the lanyard cord and its type were also found to affect the results.

The CenceMe system is one of the existing implementations which actually runs on a mobile phone. However, the techniques implemented are relatively rudimentary and therefore achieve low accuracy even on a custom small-scale controlled experiment.

Introducing C4.5

The C4.5 algorithm [99] is an extension of the ID3 algorithm [100]. ID3 is an algorithm used to generate a decision tree from the top to down without backtracking. To select the most useful attribute for classification, a criterion named the information gain based on the information theory is exploited [44]. The information gain of a given attribute X with respect to the class attribute Y is the reduction in uncertainty about the value of Y when we know the value of X. In order to calculate the information gain we need to know the information entropy. If $E(S)$ is the information entropy of the set S and n is the number of different values of the attribute in S, and $f_S(i)$ is the frequency of the value i in the set S, then the information entropy is calculated according to the following equation :

$$E(S) = -\sum_{i=1}^{i=n} f_s(i) log_2 (f_s(i)) \tag{6}$$

The entropy is always a number comprised between 0 and 1 inclusively. If all the examples are in the same class, the entropy of the population is null. If there is the same number of positives and negative examples in binary classification, the entropy is maxed. The best attribute is selected based on the information gain factor that is given by the following equation:

$$G(S, A) = E(S) - \sum_{i=1}^{m} f_s(A_i)\, E(S_{A_i}) \tag{7}$$

$G(S, A)$ is the gain of the set S after a split over the A attribute; m refers to the number of different values of the attribute A in S; $f_S(A_i)$ is the frequency of the items possessing A_i as i^{th} value of A in S; and S_{A_i} is a subset of S. There are three requirements for the training data of ID3 algorithm. The first one is that all of the training data objects must have common attributes, and these attributes should be previously defined. The second requirement is that the attributes' values should be clearly indicated and a value indicating a special attribute should indicate no more than one state. The third requirement is that there must be enough test cases to distinguish valid patterns from chance occurrences.

The most important improvement in C4.5 is the capability to handle both continuous and discrete attributes. To do that a threshold is created and the list of value is split into those which are above the threshold and those that are equal or below. Another improvement comes from its capacity to handle training data with missing attributes. In that case, the values of a missing attribute are simply not used in the gain and entropy calculations. Finally, C4.5 can backtrack on the tree to perform what is called the pruning. That important phase reduces the risk of over-fitting data by replacing the branches that do not help for a leaf. Due to that particular step, the C4.5 computational complexity is higher than its predecessor [44].

b) Kose et al.'s Work: OHARSP

Authors in [101] propose an online human activity recognition system on smartphones. They focus on activity recognition using the embedded accelerometers on smartphones in order to classify basic movements of a user, such as walking, running, sitting and standing. The authors evaluated the performance of two classification algorithms: Naïve Bayes and Clustered KNN, which is an improvement of minimum Distance and k-nearest neighbor (KNN) classification algorithms that works in real-time.

The clustered KNN and the Naïve Bayes classifiers are implemented on Android phones to detect four main activities; which are walking, running, standing and sitting. For this purpose, the process is divided into two phases.

At first stage, training data is collected for each activity separately. To do so, authors developed an application called Activity Logger. In this application, the user selects the activity to be performed, puts the phone into the packet and starts to perform the related activity. For each activity, the application creates different training data files in which raw data from the 3-axes of the accelerometer is being logged. Low pass filter is applied to raw data for noise removal. Before starting the activity recognition tests, a few minutes of training data for each activity is collected by each subject.

In the second stage, activity recognition is performed using the selected classifier. First, the application extracts necessary features of training sets for each activity according to the classification method being used. Depending on the size of the training set and the processor performance of the phone, this step may take a few minutes. The main screen of the application allows the user to select the system parameters, such as the sensor sampling rate and window size.

The performance of these classifiers is tested on five different subjects. Test results show that the clustered KNN approach outperforms other classifiers in terms of accuracy and execution time. Authors also analyzed the impact of sensor sampling rate and window size (that is used for segmenting the data) on the performance of the activity recognition. They reported that the CPU and memory usage of their system never exceeded 42% and 22 MB respectively. Applications using minimum distance classifiers and clustered KNN consume nearly the same amount of resources. On the other hand, Naïve Bayes has considerably higher CPU usage.

c) *Zheng et al.'s Work: GeoLife Transportation Mode*

The work reported in [102] is a part of a well-known research project called GeoLife, which is a GPS-log-driven application over Web Map. It focuses on lively visualization, effective organization, fast retrieval and deep understanding of GPS track logs for both personal and public use. The solution is designed for geographic and mobile applications on the Web, authors propose an approach using raw GPS data that is based on supervised learning to automatically learn the transportation modes including walking, taking a bus, riding a bike and driving (see Figure 12).

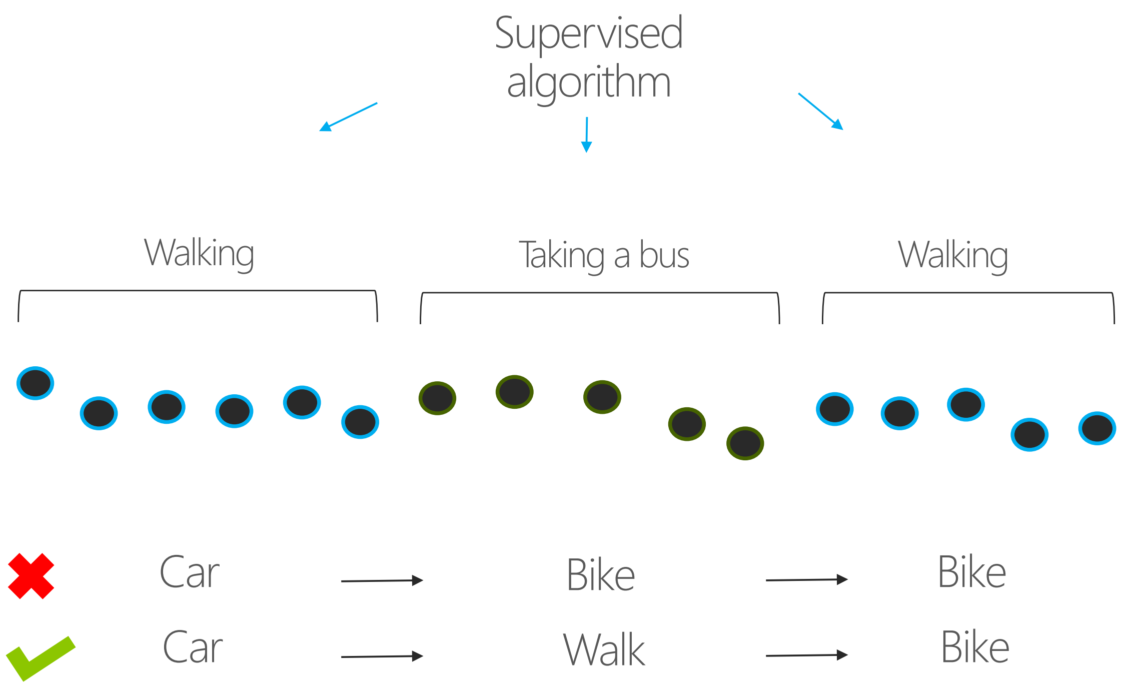


Figure 12: A descriptive diagram of Zheng's work

When a GPS log file comes, first, the inference algorithm divides the GPS track into trips and then partition each trip into segments by change points. Then, it extracts features from each segment and sends these features to the inference model.

Figure 13 depicts how authors calculate the features that will be used in the inference model. Given two consecutive GPS points, for example, p_1 and p_2, they calculate the spatial distance L_1, temporal interval T_1 and heading direction ($p_1.head$) between them.

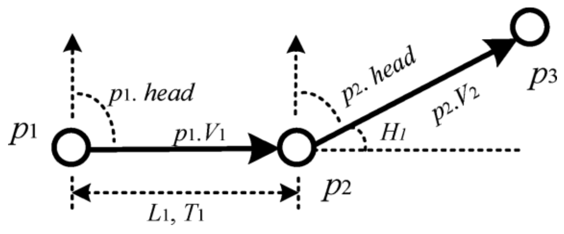


Figure 13: Feature calculation based on GPS logs.

Subsequently, the velocity of p_1 can be computed as Equation (8).

$$p_1.V_1 = L_1/T_1. \tag{8}$$

Then, the heading change, such as H_1 of three consecutive points like p_1, p_2 and p_3 can be calculated as Equation (9).

$$H_1 = |p_1.head - p_2.head| \tag{9}$$

Further, more features, such as acceleration and expectation of velocity, can be calculated in this manner.

Two alternative ways are considered when it attempts to learn a user's transportation mode (see Figure 14). In one way, it regards the segments of GPS tracks as independent instances. General classifiers like Decision Tree are employed to perform inference. After the inference, a post-processing, which takes the transition probability between different transportation modes into account, is implemented to improve the prediction accuracy. For instance, if the prediction is $Car \rightarrow Bike \rightarrow Bike$ while the ground truth is $Car \rightarrow Walk \rightarrow Bike$ the algorithm assumes that a prediction error has occurred and corrects the prediction basing on the transition probability between different transportation modes. It is more probable to walk after leaving a car than to take a bike directly.

In the other way, GPS data are deemed as a kind of sequential data. Conditional random field (CRF) [103], a framework for building probabilistic models to segment and label sequence data, is leveraged to perform the inference. Since the conditional probabilities between different transportation modes have been considered in the CRF graphical model, in this way, it is not needed to pass by a post-processing step. In the inference, the mode of transportation can take four different values including Bike, Bus, Car and Walk.

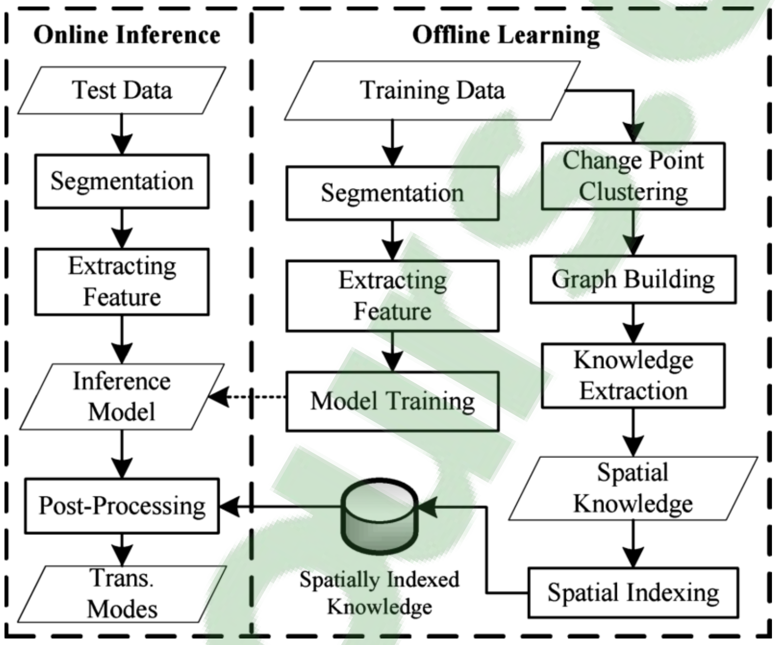


Figure 14: Architecture of GeoLife transportation approach

Four different inference models including Decision Tree, Bayesian Net, Support Vector Machine (SVM) [104] and Conditional Random Field (CRF) are studied in the experiments. Authors evaluated the approach using the GPS data collected by 45 users over six-month period. As a result, beyond two other segmentation methods, the achieved method showed a higher degree of accuracy in predicting transportation modes and detecting transitions between them.

This method starts by a segmentation process, a step where the trajectory is divided into a set of the same velocity parts, authors claim that the solution is usable for mobile phones, however, this technique seems not very friendly to the limited resources environments. For instance, applying segmentation process on every trajectory is penalizing for the computational complexity, why do authors repeat the same process even if the trajectory is the same? (e.g. there is no need to repeat the same calculation every time the user goes from home to work as it is very probable that he will take the same transportation mode), consequently, it will be better to process the segmentation wisely in order to prevent a massive draining of mobile's battery.

2.6. Location Prediction Works

Significant research effort has been undertaken in both the mobile computing and the spatial data mining domains [63], [77]. Many advances in tracking users' movements have emerged resulting in several proposals for predicting future users' locations. The main approach proposed is to learn the user's patterns from his historical locations and try to predict the next location via different techniques.

a) Morzy's Work

In [78], Morzy introduces a new method for predicting the location of a moving object. He extracts the association rules from the moving object database using a modified version of Apriori [105] and uses the rules extracted when a trajectory is given via matching functions. He selects the best association rule that matches this trajectory, and then uses it for the prediction.

Apriori is an association rule algorithm. An association rule is a rule of the form *condition => consequence* that aims to find relations between the data. For example, let's say that we have a dataset comprised of transactions made at Walmart by customers. We could discover a rule such as *if Sunday and Diapers => Beers*. That rule would mean that very often, when it is Sunday and someone buy diapers he will also buy beers. Association rules mining algorithms define the terms *very often* with two attributes named the support and the confidence. The first one defines the minimum frequency of both the left and right part of the rule. For example, supposes we have the item set {{A}, {B}, {AB}, {BA}, {B}, {AB}, {AB}}, the support of AB would be $Support(\{AB\}) = \frac{3}{7} \approx 43\%$. The second, the confidence, is the probability threshold of the right part being true if the left part is validated:

$$Confidence\ (X => Y) = p(Y|X) = \frac{p(X \cup Y)}{p(X)} = \frac{Support(\{AB\})}{Support(\{A\})} \tag{10}$$

Apriori relies upon two principles. The first one is the search for the frequent k-itemsets whose support is higher than a fixed minimum support. The second phase consists to build the association rules from the found frequent k-itemsets. A rule is retained only if its confidence is higher than a fixed minimum confidence [44]. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

Apriori uses breadth-first search and a Hash tree structure to count candidate item sets efficiently. It generates candidate item sets of length k from item sets of length $k - 1$. Then it prunes the candidates which have an infrequent sub pattern. The candidate set contains all frequent $k - length$ item sets. After that, it scans the transaction database to determine frequent item sets among the candidates. For more details about Apriori algorithm please refer to the paper described in [105].

Unfortunately, the work of Morzy does not allow incremental training of the models, since it is based on a posteriori learning. Secondly, the fact that authors propose matching functions in the form of strategies (simple, polynomial, logarithmic and aggregation strategies) can create computational complexities and difficulties to choose and set the right parameters for the right strategy.

b) Gambs's Work

Gambs et al. extend a previously proposed mobility model called the Mobility Markov Chain (n-MMC) in order to keep track of the n previous visited locations [106]. Authors introduce Mobility Markov Chain (MMC) as a model that represents the mobility behavior of an individual as a discrete stochastic process in which the probability of moving to a state (i.e. POI) depends only on the previous visited states and the probability distribution of the transitions between states. More precisely, a MMC is composed of:

• A set of states $P = \{p_1,\dots p_k\}$, in which each state corresponds to a frequent POI (ranked by decreasing order of importance). These states generally have an intrinsic semantic meaning and therefore semantic labels such as "home" or "work" can often be attached to them. The semantics of some states can sometimes be deduced automatically from the structure of the MMC.

• A set of transitions, such as $t_{i,j}$, which represents the probability of moving from state p_i to state p_j . A transition from one state to itself can occur if the individual has a probability of moving from one state to an occasional location before coming back to this state. For instance, an individual can leave his "home" to go to the pharmacy before coming back to "home".

Thereafter, authors describe in Algorithm 2 an algorithm for learning a n-MMC out of the trail of mobility traces D of an individual, which is decomposed in two steps. During the first step, a clustering algorithm called Density-Joinable cluster (DJ-Cluster) [107] is used to discover the POIs. Afterwards, during the second step, the transitions between those POIs are computed. DJ-Cluster takes as input three parameters: (i) $MinPts$: the minimal number of points necessary to form a cluster, (ii) ε: the maximum radius of the cluster and (iii) d_{mer}: a merging distance for the clusters.

Algorithm 2: Construction of a n-MMC

```
Cluster (loc)
Input: $D, n, MinPts, \varepsilon, d_{mer}$
Output: the Mobility Markov chain
1:  Run DJ-Cluster ( $MinPts, \varepsilon$) on $D$
2:  Merge the clusters that share at least a common point
3:  Merge the clusters that are within $d_{mer}$ distance of each other
4:  Let $listPOIs$ be the list of all constructed clusters
5:  for each cluster C in $listPOIs$
6:      Compute the $time\_interval$, $radius$ and $density$ of C
7:  end for
8:  Sort the clusters in $listPOIs$ by decreasing order according
    to their densities
9:  for each cluster $C_i$ in $listPOIs$
10:     Create the corresponding state $p_i$ in the mobility Markov
    chain
11: end for
12: for each mobility trace $m$ in $D$
13:     if the distance $(m, C_i) <$ radius $s_i$ then
14:     Update the $n-1$ locations and current position with $C_i$
15:     Label the $m$, the $n - 1$ previous locations, $C_i$
16:     else
17:     Label the $m$ with the value "unknown"
18:     end if
19: end for
20: Delete all traces that are "unknown"
21: Squash all the successive mobility traces sharing the same
    label into a single occurrence
22: Compute all the transition probabilities between each pair of
    states of the Markov chain
23: return the Mobility Markov chain computed
```

Authors show that while the accuracy of the prediction grows with n, choosing n > 2 does not seem to bring an important improvement to the cost in terms of space. However, like the previous works, this one has a lack of computational complexity and the incremental support. In addition, the three datasets used in the authors' experiments were collected in a controlled environment where data was gathered from specific participants who were aware of the experiments.

c) *Asahara's Work*

Asahara et al. proposed in [108] a method for predicting pedestrian movement on the basis of a Mixed Markov chain Model (MMM) [109]. In their prediction model, they integrate some complex parameters such as pedestrian's personality merged with his previous status. MMM is an intermediate model between individual and generic models.

The prediction of the next location is based on a Markov model belonging to a group of individuals with similar mobility behavior. This approach clusters individuals into groups based on their mobility traces and then generates a specific Markov model for each group. The prediction of the next location works by first identifying the group a particular individual belongs to and then inferring the next location based on this group model (see Figure 15).

The authors tested their solution in a major shopping mall and report an accuracy of 74.4% for the MMM method and, in a comparison over the same dataset, they reported that methods based on Markov-chain models, or based on Hidden Markov Models, achieve lower prediction rates of about 45% and 2%, respectively.

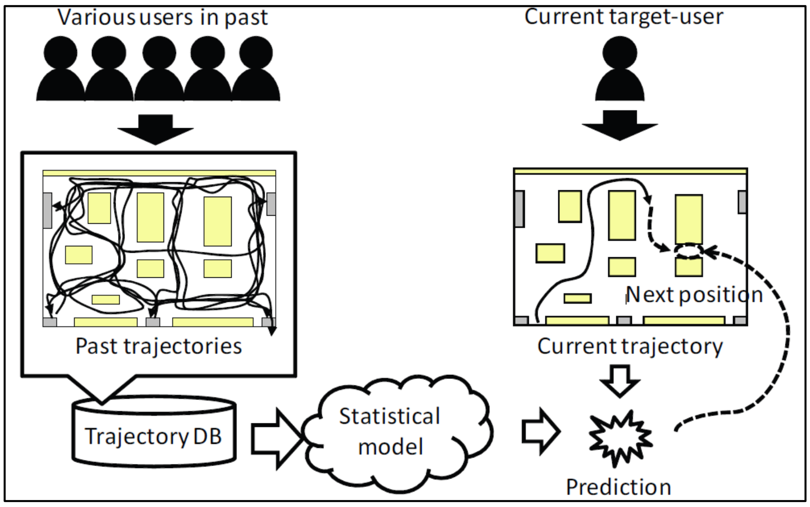


Figure 15: Pedestrian-movement prediction [108]

d) NextPlace: A Spatio-Temporal Prediction Framework

In [110], authors proposed a spatio-temporal user location prediction approach based on nonlinear analysis of the time series of start times and duration times of POIs visits. In order to obtain an estimation of the future behavior, the history of a user's visits to each of its significant locations is considered. Then, for each location they try to predict when the next visits will take place and for how long they will last. After this estimation, the predictions obtained for different locations is analyzed, in order to produce a unique prediction of where the user will be at a given future instant of time.

For each user, the proposed algorithm keeps track of all previous visits to a set of locations. I.e., for each visit it considers the instant when it started and how long it lasted. The algorithm predicts the next visits to a given location by means of the previous history of visits $((t_1; d_1), (t_2; d_2), \dots, (t_n; d_n))$:

1. Two time series are created from the sequence of previous visits: the time series of the visit daily start times C and the time series of the visit durations D defined as follows: $C = (c_1, c_2 \dots, c_n), D = (d_1, d_2, \dots, d_n)$. Where c_i is the time of the day in seconds corresponding to the time instant t_i (i.e. c_i is in the range [0; 86400]);

2. They search in the time series C sequences of m consecutive values $(C_{i-m+1}, \dots, C_i)$ that are closely similar to the last m values $(C_{n-m+1}, \dots, C_n)$;

3. The next value of time series C is estimated by averaging all the values C_{i+1} that follow each found sequence;

4. At the same time, in time series D the corresponding sequences $(d_{i-m+1}, \dots, d_i)$ are selected; the sequences have to be located exactly at the same indexes as those in C;

5. The next value of time series D is then estimated by averaging all the values d_{i+1} that follow these sequences.

As an example, if the last three visits of a user to a location are Monday at 6:30pm, Monday at 10:00pm and Tuesday at 8:15am, authors analyze the history of visits in order to find sequences that are numerically close to (6:30pm, 10:00pm, 8:15am), i.e. (6:10pm, 9:50pm, 8:35am) and (6:35pm, 10:10pm, 8:00am): then, assuming that the next visits that follow these subsequences start at 1:10pm and 12:40pm and last for 40 and 30 minutes respectively, they estimate the next visit at 12:55pm for 35 minutes, averaging both arrival times and duration times.

The main idea behind this algorithm is the assumption that human behavior is strongly determined by daily patterns: the sequence of visit start times is therefore mapped to a 24-hour time interval, focusing only on the start time of each visit. The choice of the value m has an impact on the accuracy of the prediction: in fact, this can be improved by taking into account more visits in order to identify particular patterns that may be present only in certain intervals of time such as specific days.

Authors have evaluated NextPlace by comparing it with a version based on a linear predictor and a probabilistic technique based on spatio-temporal Markov predictors over four different datasets. They have reported an overall prediction precision up to 90% and a performance increase of at least 50% over the state of the art.

The lack of this approach is that the intuition behind NextPlace is built on the fact that the sequence of important locations that an individual visits every day is more or less fixed, with only minor variations that are also usually deterministically defined. As an example, if a woman periodically goes to the gym on Mondays and Thursdays, she may change her routine for those days, but the changed routine will be more or less the same over different weeks. As such, the authors excluded the fact that a user can change, for some reason, his routine (concept drift), for the example cited earlier, the woman can definitely stop going to the gym.

e) DB-tree Algorithm

Authors in [111] present two new algorithms that use the frequent patterns tree (FP-tree) structure to reduce the required number of database scans. One of the proposed algorithms is the DB-tree algorithm, which stores all the database information in an FP-tree structure and requires no re-scan of the original database for all update cases, the algorithm stores in descending order of support all items and counts all items in all transactions in the database in its branches.

FP-Tree preprocesses the transaction database as follows: in an initial scan the frequencies of the items (support of single element item sets) are determined. All infrequent items (that is, all items that appear in fewer transactions than a user-specified minimum number) are discarded from the transactions, since, obviously, they can never be part of a frequent item set. In addition, the items in each transaction are sorted, so that they are in descending order depending on their frequency in the database [112]. This pre-processing step is demonstrated in Figure 16, which shows an example transaction database on the left. The frequencies of the items in this database, sorted descendingly, are shown in the middle of this table. If we are given a user specified minimal support of 3 transactions, items f and g can be discarded. After doing so and sorting the items in each transaction descendingly depending on their frequencies, we obtain the reduced database shown in Figure 16 on the right.

a d f	d a
a c d e	d c a e
b d	d b
b c d	d b c
b c	b c
a b d	d b a
b d e	d b e
b c e g	b c e
c d f	d c
a b d	d b a

d	8
b	7
c	5
a	4
e	3
f	2
g	1

Figure 16: Transaction database (left), item frequencies (middle), and reduced transaction database with items in transactions sorted descendingly depending on their frequency (right)

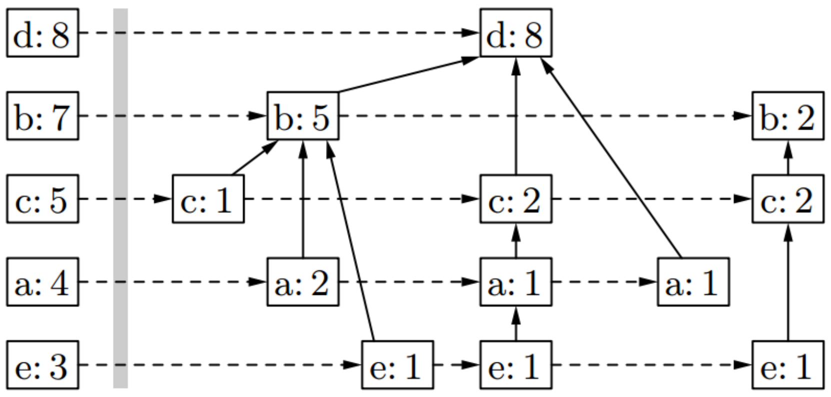


Figure 17: FP-tree for the (reduced) transaction database shown in Figure 16.

The DB-Tree is constructed in the same way as done in FP-Tree (see Figure 17) except that it includes all the items instead of only the frequent 1-items. The second algorithm is the PotFp-tree (Potential frequent pattern) algorithm, which uses a prediction of future possible frequent item sets to reduce the number of times the original database needs to be scanned when previous small item sets become large after database update.

The first disadvantage of the three algorithms, with all respect to the authors, is the non-support of the concept drift, since none of them support the changes in the sequences behavior. The second problem is the restructuring of the tree to store the node in descending order of support. This technique not only increases the computational complexity, but represents likewise an inadequate solution to fields where the order of items is important like peoples' habits.

While developing a rich body of work for mining moving object data, the research community has shown very little interest to the online mining of these objects since the mainstream of related works relies on a post-hoc analysis of a massive set of data to learn and predict locations.

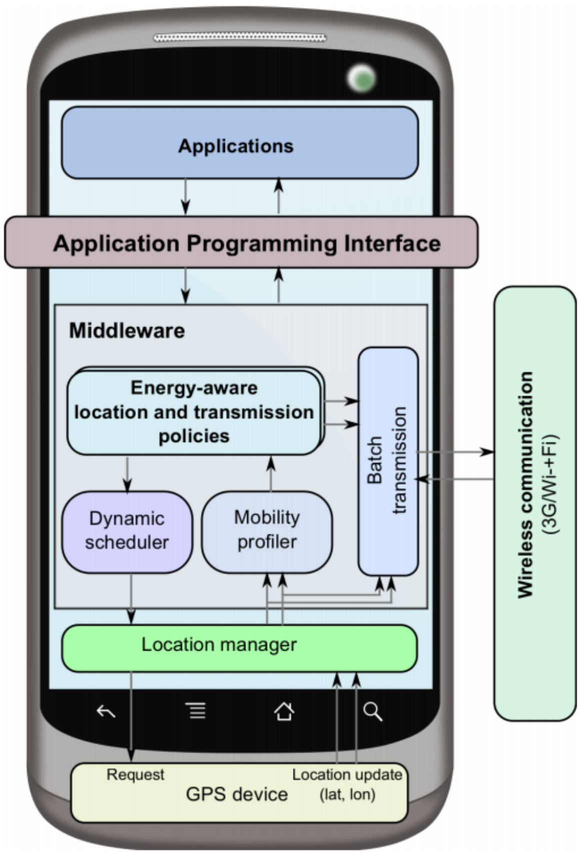
Moreover, one of the big issues that can easily shatter the most robust next location predictive model is the habits' drift. In fact, from the time when the data begins to behave in a non-regular manner; the predictive models will face difficulties to acheive their work. As such, in Chapter 5, we propose an online predictive model that supports the data drift.

2.7. Battery-Aware Works

The problem of power management on mobile devices has been well-explored. We will detail above some important works in the field of phone energy saving.

a) Galeana-Zapién's Work

In [113], authors propose a modular middleware architecture and runtime environment to directly interact with location-based application programming interfaces (APIs) and embedded sensors. The main reason of that is to manage the duty cycle process based on energy and context aspects. Authors propose an approach to adapt sensing rate and energy-aware transmission within a green context-aware middleware. The solution builds upon the introduction of three modular components: dynamic scheduler, the mobility profiler and the batch transmission module that, in cooperation with the mobile operating system, interact with the mobile sensors for energy control purposes, see **Figure** 18.



Applications
Application Programming Interface
Middleware
Energy-aware location and transmission policies
Batch transmission
Dynamic scheduler
Mobility profiler
Location manager
Wireless communication (3G/Wi-+Fi)
Request
Location update (lat, lon)
GPS device

Figure 18: Middleware architecture for context-aware and energy-efficient sampling and transmission of data sensor streams. [113]

The dynamic scheduler component of the middleware allows to simplify the sensor reading processes in the mobile device. Sensor access is simplified by providing a single software layer that enables agile development of mobile applications. This framework is designed for applications that require to specify parameters to low-level rate-adaptive duty-cycling for GPS-based positioning, i.e., dynamic adaptation of sensing parameters. The periodic sensing interval of the dynamic scheduler can be selected to be flexible based on the granularity and energy consumption requirements given by the location sensing application. Under different scenarios, this layer can be used as a low-level enabler for further energy savings. Developers are able to request controlled location updates according to established location resolution and energy trade-offs.

For continuous location monitoring, periodic duty-cycling of GPS is not appropriate, as this embedded sensor could deplete the battery in a few hours. Thus, an energy-efficient usage of GPS is required, taking advantage of individual mobility patterns. The mobility profiler's main goal is to estimate/predict the system state and user mobility from GPS data streams to dynamically schedule position updates to minimize power consumption with some tolerance in position accuracy.

The context states, e.g., static or moving fast, can be used as a basis to define high-level policies to further reduce energy consumption. By knowing the user mobility state and transitions, GPS updates can be rescheduled at specific times and trigger location readings according to some policies. For instance, sampling at a coarse grain when the user is rather stationary and raising the sampling rate gradually according to the estimated user speed. The mobility profiler in the middleware provides good hints of user mobility from GPS logs, so as to adaptively change the location sampling rate according to a policy.

The batch transmission module is the key enabler of energy-awareness. It reduces the communication energy overheads introduced by the transmission of GPS data to a remote system through the wireless media. This module caches position fixes locally, and then, it selectively transmits packed subsets of GPS location streams driven by an adaptive duty cycle set by the mobility profiler and an energy budget. Moreover, this is the element that allows finding trade-offs between the relevance of the GPS data and the energy cost of transmitting such information through wireless access.

b) SenseLess

The authors in [114] present the SenseLess application to perform energy-efficient mobile sensing. The proposed application makes use of less power-hungry sensors (e.g., accelerometer) as a means to augment location change detection. SenseLess is able to detect when a user is not moving, and then, it stops sensing the GPS position to save energy. It makes use of a GPS sensor, and when no GPS signal is detected, it resorts to Wi-Fi technology to acquire location information. The authors argue that compared to a GPS-only approach, the SenseLess application is able to reduce energy consumption by more than 58% when determining the user's location and maintaining the accuracy of the sensed data. This approach, however, does not consider the resolution of locations, which is of paramount importance for location-based systems, as it is directly related to the quality of information.

c) EnTracked

This work proposes EnTracked [115], a system that, based on the estimation and prediction of system conditions and mobility, schedules position updates to both minimize energy consumption and optimize robustness. The realized appraoch tracks pedestrian targets equipped with GPS-enabled devices. The system is configurable to realize different trade-offs between energy consumption and robustness (see Figure 19).

The proposed device model consists of two parts: a power model that describes the power usage of the phone; and a delay model that, for instance, describes the delays when requesting a phone feature, e.g., the time it takes for the GPS to return a position. In both models authors consider the following phone features: accelerometer (a); GPS (g); radio idle (r); radio active (s); idle (i).

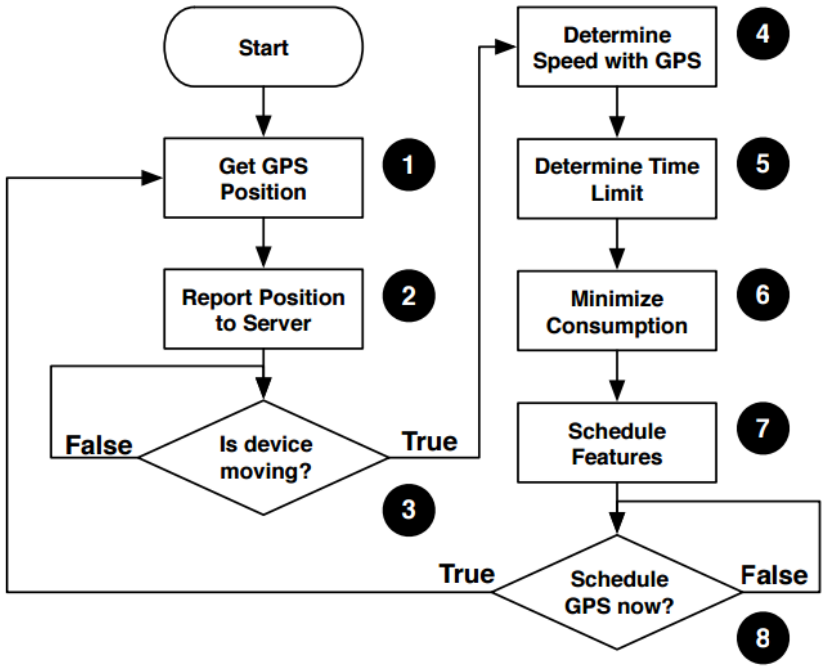


Figure 19: Entracker client logic [115]

These are the features that authors find relevant for phone tracking, ’idle’ is not strictly a feature, but is included in the power model for completeness. For interactive user applications on the device, one would also need to take into account the power usage of features such as the computations of the application logic, the key strokes, camera use, and screen use.

The power model consists of two functions defined in Equation (11): the power function power and the consumption function $c_{d,p}$ where d is a feature’s power-off delay and p its power consumption.

$$power(T) = \sum_{t=1}^{T} i_p + c_{a_d,a_p}(a_t) + c_{g_d,g_p}(g_t) + c_{r_d,r_p}(r_t) + c_{s_d,s_p}(s_t)$$
$$c_{d,p}(x) = \begin{cases} p, & \text{if } x \leq d \\ 0, & \text{if } x > d \end{cases} \tag{11}$$

The equation uses the variables a_t, g_t, r_t, s_t for the different features listed in the feature list above. Each variable denotes, at time step t, the number of seconds since the feature was last powered off (a variable is zero if the feature is in use in the current time step t). Since the idle power consumption is constant no variable is introduced.

Furthermore the parameters i_p, a_p, g_p, r_p, s_p denote the power consumption of a feature, e.g., 0.324 watts for internal GPS. The parameters a_d, g_d, r_d, s_d denote the number of seconds that a feature takes to power off after last use, e.g., 30 seconds for internal GPS.

Authors provide extensive experimental results by profiling how devices consume power, by emulation on collected data and by validation in several real-world deployments. Results from this profiling show how a device consumes power while tracking its position. Results from the emulation indicate that the system can estimate and predict system conditions and mobility.

d) EEMSS

Authors in [116] have gone further to propose a whole framework of Energy Efficient Mobile Sensing System (EEMSS) for automatic users state recognition. The core component of EEMSS is a sensor management scheme for mobile devices that operates hierarchically. It selectively turns on the minimum set of sensors to monitor users' state and trigger new set of sensors if necessary to achieve state transition detection. Energy consumption is reduced by shutting down unnecessary sensors at any particular time.

The overall architecture of EEMSS is described in Figure 20 where: (1) System reads in the XML state descriptor which contains the sensor management scheme. (2) Management module determines the sensors to be monitored based on current users' state which is specified by the sensor management scheme. (3) Management module instructs the sensor control interface to turn on/off sensors. (4) Sensor control interface operates individual sensors. (5) Sensor interface reports readings to the classification module. (6) Classification module determines the user state. (7) Classification module forwards the intermediate classification result to the management module. (8) The user's state is updated and recorded in real-time. (9) The relevant information is also displayed on the smartphone screen.

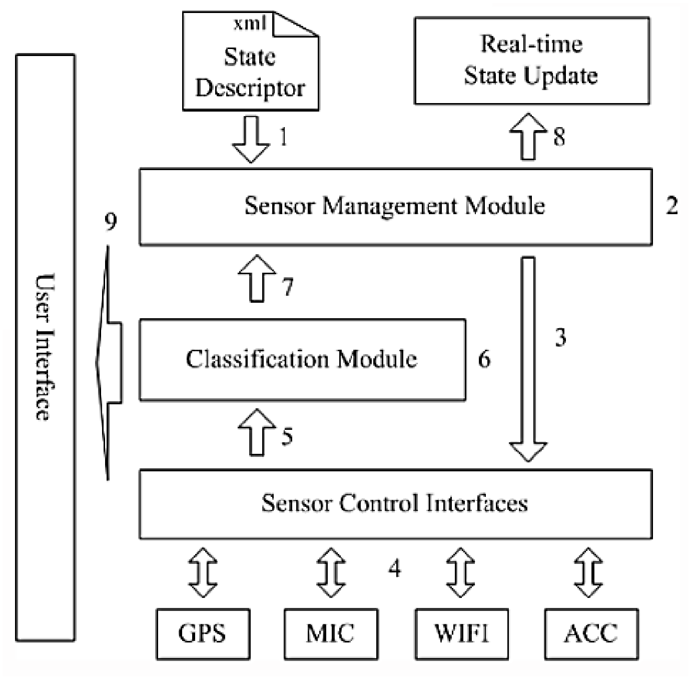


Figure 20: System architecture of EEMSS [116]

e) Other Works

Viredaz et al. [117] surveye many fundamental but effective methods for saving power on handheld devices. These methods concern a range of phone components like processor, memory, display screen, audio system and wireless networking. It has been suggested from the architecture 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 [118], authors propose 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 [119]. DVS exploits the fact that the 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. In other words, the power-saving scheme should be fully customized for real-time power consumption situation and the specific application requirements. However, these methods are more suitable for lower-level systems design rather than application development.

Authors of [120] introduce a technique to increase the battery lifetime of a personal digital assistant (PDA)-based phone by reducing its idle power, i.e., the power that 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 to avoid energy waste. The device is powered only when an incoming call is received or when the user needs to use the PDA for other purposes.

As seen above, significant efforts have been undertaken for a wise use of mobile resources. However, almost all 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 according to the battery life and the user state. Our work will also stand out by the flexibility that it offers to users, since we leave some freedom to them to choose the degree of austerity in the use of the battery.

Contrariwise other solutions, our approach takes the remaining battery level as a parameter to adjust the battery consumption. The consumption of our battery-aware approach varies depending on the battery state. We demonstrate in Chapter 3 that our approach succeeds in managing the battery consumption efficiently without depreciating the accuracy of our algorithm.

2.8. Chapter Conclusion

In this chapter, we reviewed the most important approaches regarding human outdoor activity recognition. We have discussed the advantages and disadvantages of each model in the optic of discovering what would be needed for this thesis. We have brought some common understandings concerning the principal notions of outdoor activity recognition such as trajectories, semantic enrichment of trajectories, sensing techniques, inference techniques...etc. We categorized the learning techniques in function of three axis; the first axis is the manner of data sampling, secondly if the approach used is supervised or not and, thirdly, if the concerned algorithm is incremental or not. In order to have a wide view on the existing outdoor activity works, we have divided the related works into two types; location systems that determine the user's activity in terms of location and motional systems that recognize activities in terms of motion state. As our work fits more in the category of location systems, we have developed this section into two types: non-mobile approaches that require the use of computers for the activity recognition and mobile approaches that try to achieve all the recognition processes on the mobile phone.

We have also explored the existing works that predict users' next destinations from their historic patterns. Besides, we have highlighted some important battery-aware works since we aim proposing battery-friendly solutions throughout all the models that we develop either for the recognition or the predictive systems.

In the next chapter, we will present our first part of contributions that focuses on proposing an online battery-friendly outdoor activity recognition system.

PART III

OUR CONTRIBUTIONS

CHAPTER 3

BATTERY-AWARE ACTIVITY RECOGNITION

3.1. CHAPTER INTRODUCTION

One of the unique features of mobile applications is the location awareness. Mobile users take their devices with them everywhere which increases the availability of users' traces. Extracting and analyzing knowledge from these traces represent a strong support for several application domains, ranging from traffic management to marketing and social studies. However, the limited-battery capacity of mobile devices represents a big hurdle for context detection [9]. 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.

The user context 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 presented in this chapter takes place. 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 determines the nature of the user's activity from a series of geographic positions.

As seen in chapter 2, the research community has shown 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. Additionally, only a minority of these studies tried 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 inside a place to be declared as a visited place. This threshold prevents false activity detection, such as traffic jams.

However, fixing the activity's minimum duration threshold in advance will increase the error probability. The reason is that when it is 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 in [53] a novel approach that online recognizes users visited places. Our algorithm is totally unsupervised and operates without any beforehand fixed threshold.

We answer these shortcomings by proposing an approach that brings a novelty 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 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 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 behaviors, but also activities with movements 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 chapter, 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 (see Figure 21).

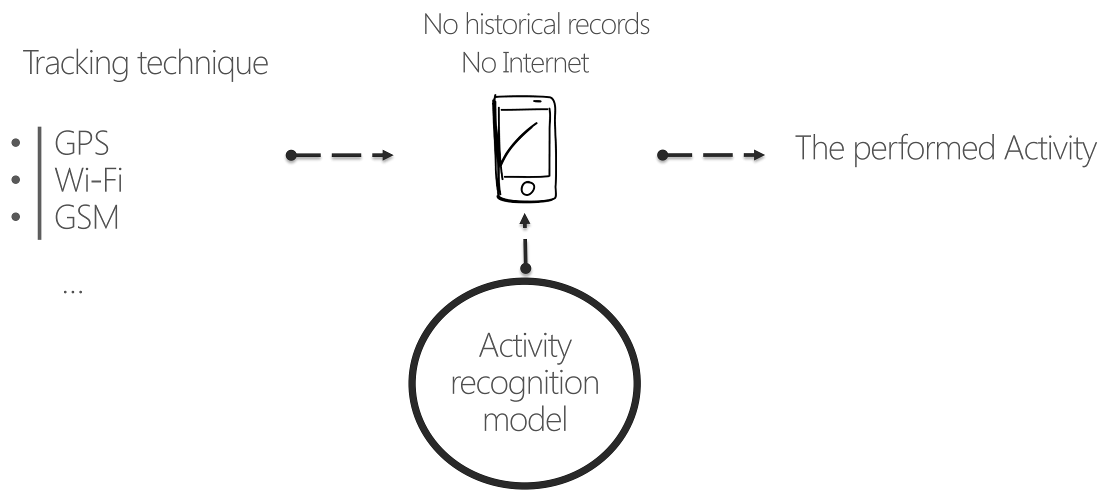


Figure 21: Online activity recognition system without Internet or historical records

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" in users' trajectories. A point of interest is an urban geo-referenced object where a person may carry out a specific activity. 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.

3.2. 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 behaviors: 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 Figure 22). Based on these concepts we introduce 3 types of clusters as follows:

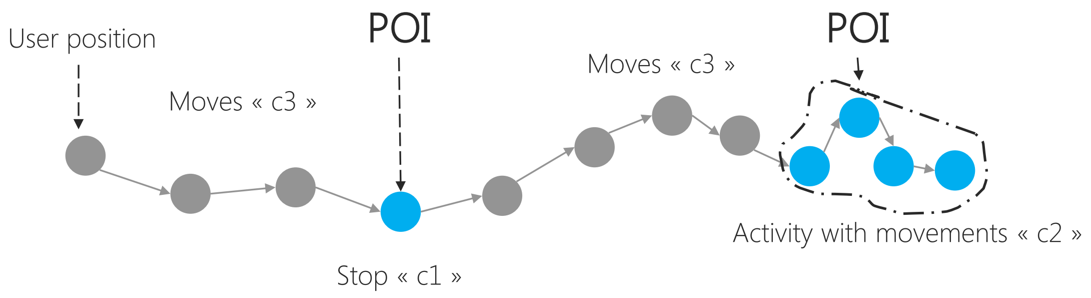


Figure 22: Relation between moves, stops, and activities with movements.

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 Figure 23 the overall approach of our activity recognition mechanism.

In the first step, we introduce 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 search process succeeds, we determine this point as a POI. The third step is to assign the POI to an activity, for instance we assign a museum to tourism activity and gym to a sports activity.

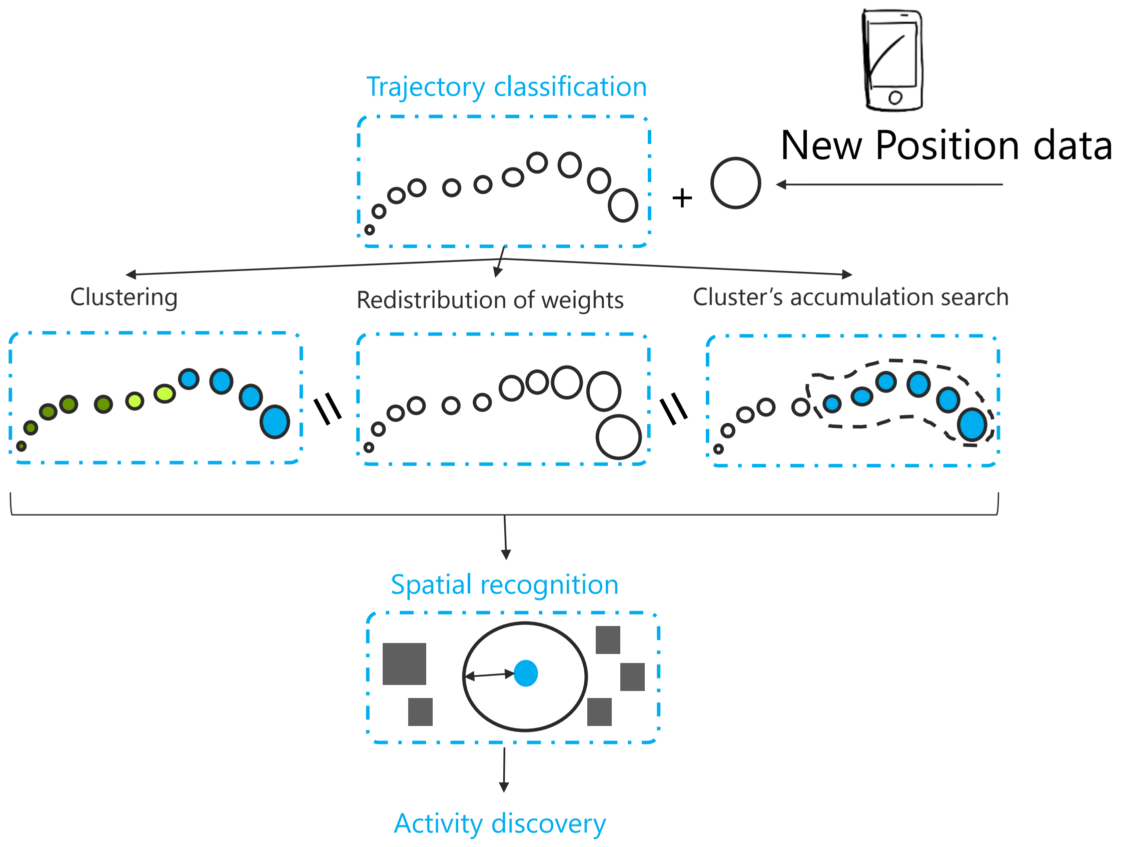

Trajectory classification
New Position data
+
Clustering
Redistribution of weights
Cluster's accumulation search
Spatial recognition
Activity discovery

Figure 23: The overall process of our activity recognition approach.

3.2.1. 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 is an assembly of users' position points P = $\{p_1, p_2 \ldots p_n\}$. 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 Figure 24 that represents the relation between positions' collection and TW. Every point in the positions' collection is a record row from the database.

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

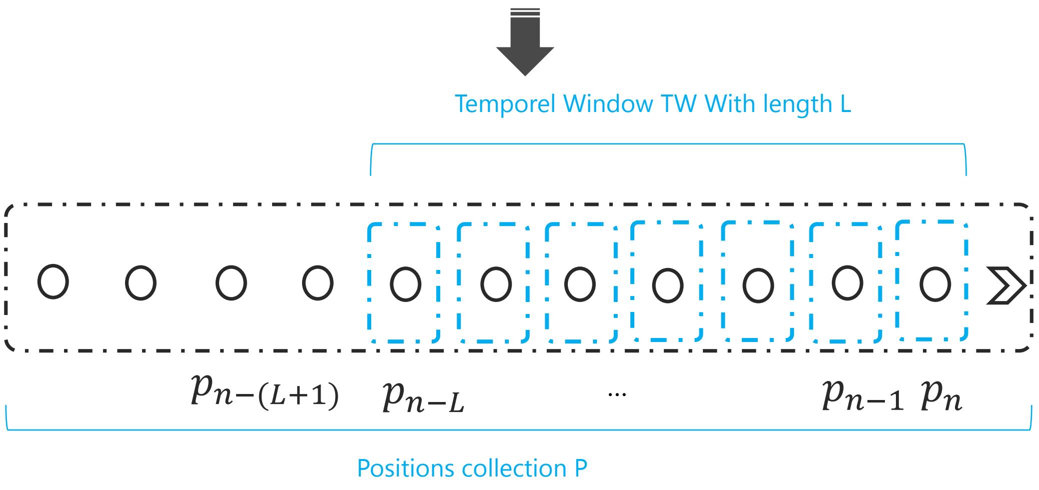


Figure 24: The relation between a position collection P and TW

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

3.2.1.1. Process 1: Classification

At the arrival of a new position data, p_n is stored in TW. The 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: $p_i\,.V = \sum(pi - \overline{p_\iota})/l$ where $\overline{p_\iota} = \sum p_i/L_{TW}$. 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.

Stop behaviors are characterized by a very low $p_i.V$ and $p_i.S$; however, move 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 25) 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.

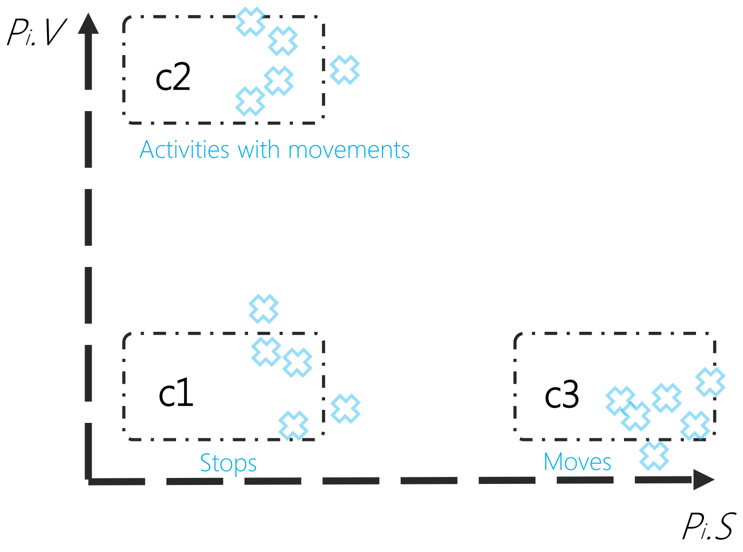


Figure 25: Inferring activity types using the speed and the variance of orientation

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.

3.2.1.2. Process 2: Distribution of weights

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 according to 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 value of λ, the lower importance of the historical data compared to more recent data. The overall weight of the data stream is the constant presented in equation (12) Where t_c ($t_c \rightarrow \infty$) is the current time.

$$W = \sum_{t=0}^{t=t_c} 2^{-\lambda t} = \frac{1}{1 - 2^{-\lambda}} \tag{12}$$

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 26).

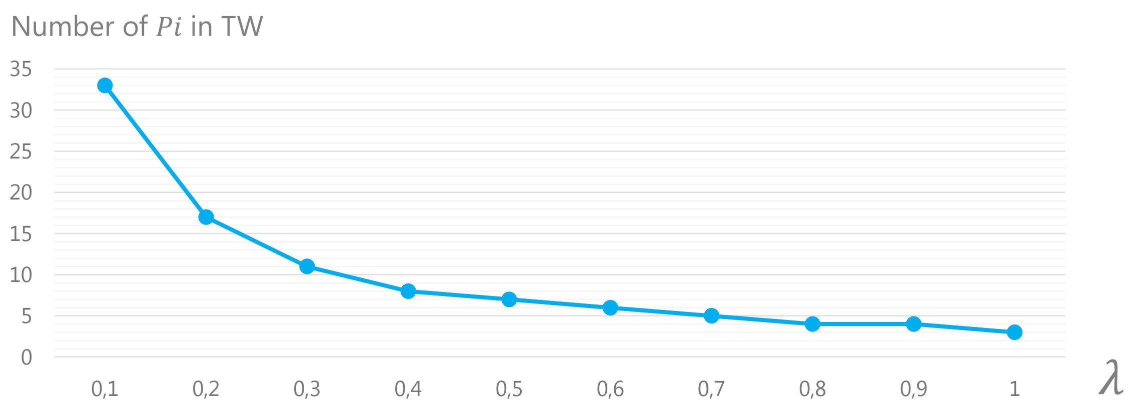


Figure 26: The impact of varying λ on TW's length.

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 Figure 27).

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.

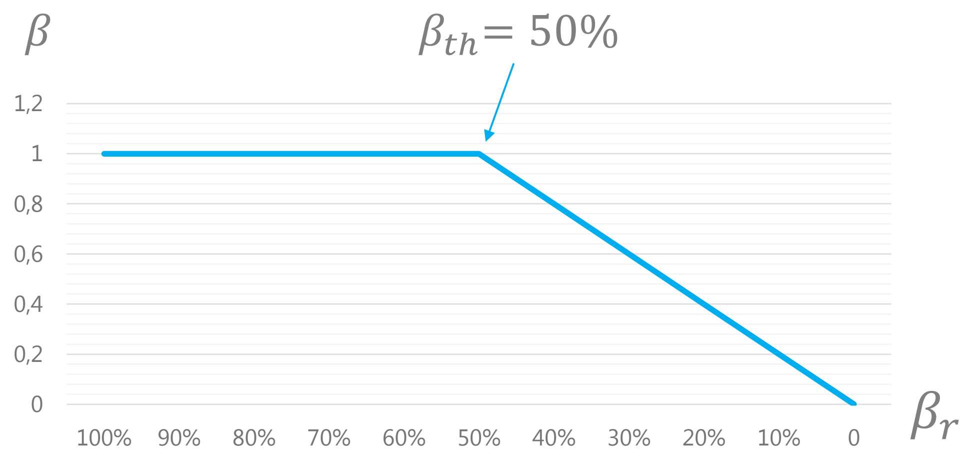


Figure 27: Example of β variation using a threshold β_{th} = 50%.

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 → 1, and E → 0 when we have 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, we say that there is a high disorder in his data (E→1). Contrariwise, when we see that the user behavior is stable, we say that there is a stability is the user data (E → 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 [121]; E of the new position point P_n is calculated using the entropy of the old dataset in TW. The value of E is calculated as follows:

$$E = 1 - H_2(p_i) = 1 - \sum_{i=0}^{n-1} p_i \, log_2 \, p_i \qquad (13)$$

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:

$$\lambda = 1 - \beta + E\beta \qquad (14)$$

The equation demonstration is shown below.

Demonstration: Our proceeding 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.

These two conditions are sited as follows:

$$\begin{cases} \text{Battery high } + \text{ high disorder } \rightarrow \text{ TW spreads} \\ \text{Battery high } + \text{ low disorder } \rightarrow \text{ TW shrinks} \\ \text{Battery low } + \text{ high disorder } \rightarrow \text{ TW shrinks} \\ \text{Battery low } + \text{ low disorder } \rightarrow \text{ TW shrinks} \end{cases} \qquad (15)$$

If we parse the characteristics mentioned in Equation (15) into a mathematical representation using the following relations:

Table 2: 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, Equation (15) will be written as follows:

$$\begin{cases} \beta \rightarrow 100\% \text{ and } E \rightarrow 0 \Longrightarrow \lambda \rightarrow 0 \\ \beta \rightarrow 100\% \text{ and } E \rightarrow 1 \Longrightarrow \lambda \rightarrow 1 \\ \beta \rightarrow 0\% \text{ and } E \rightarrow 0 \Longrightarrow \lambda \rightarrow 1 \\ \beta \rightarrow 0\% \text{ and } E \rightarrow 1 \Longrightarrow \lambda \rightarrow 1 \end{cases} \tag{16}$$

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

Table 3: 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 Equation (16) can be illustrated in the following truth table:

Table 4: The truth table of β, E and λ.

$\boldsymbol{\beta}$	$\boldsymbol{E}$	$\boldsymbol{\lambda}$
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:

$$\lambda \equiv F(\beta, E) \equiv \neg\beta \lor E \equiv \beta \xrightarrow[boolean]{} E \tag{17}$$

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 [122] 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, the Equation (17) is called fuzzy implication and is written as follows:

$$\lambda \equiv F(\beta, E) \equiv \beta \xrightarrow[fuzzy]{} E \tag{18}$$

Using the inequality of Reichenbach [123], the fuzzy implication in equation (18) can be written as follows:

$$\lambda = F(\beta, E) = 1 - \beta + \beta E \tag{19}$$

We notice that the Equation (19) and the Equation (14) are the same. Consequently Equation (14) is proven. This equation has been tested in Figure 28 where we varied the two parameters β and E to observe the sensitivity of λ (the length of TW) to these changes.

In Figure 28, 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 Figure 28 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.

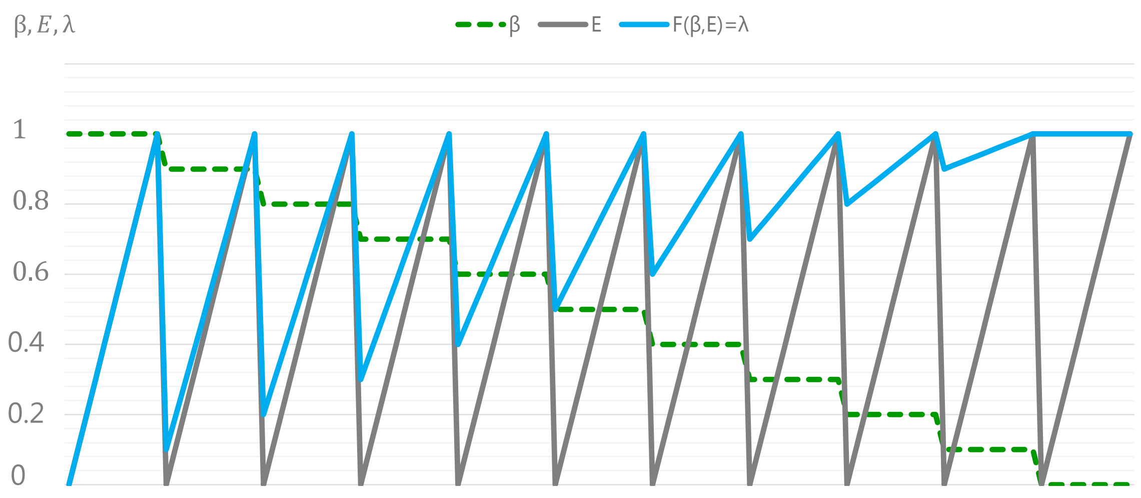


Figure 28: λ variation in function of β and E.

3.2.1.3. Process 3: Clusters accumulation search

The algorithm recognizes that someone is doing an activity if the weight of its cluster W 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; activities with movements.

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, the equation is written as Equation (20) where λ = 1- β + β E.

$$T_{min} = \frac{1}{\lambda}\log\frac{\mu}{\mu - 1} \tag{20}$$

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).

$W_j = \sum_{i=0}^{n} w_{ij}$ 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.

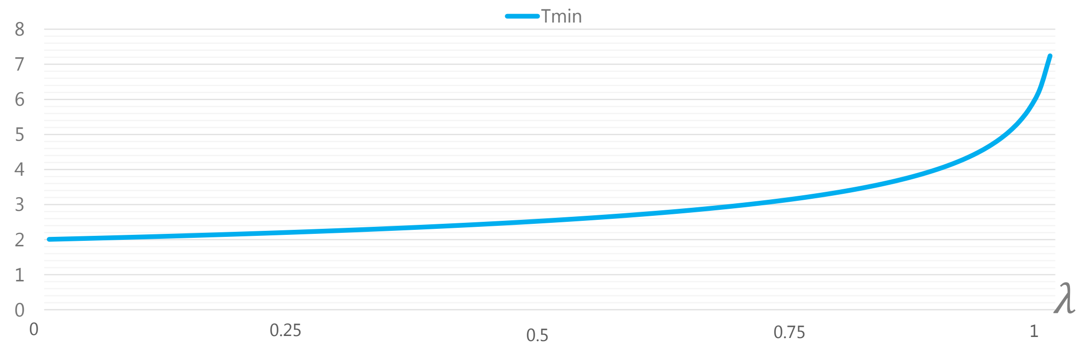


Figure 29: The relation between λ and T_{min}

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 Figure 29, 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 26) 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 29).

Conversely, when λ is near 0, TW will contain a maximum number of position points (see Figure 26) 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 (Figure 29).

Algorithm 3: Trajectory classification

```
Input:    A position point $p_i$;
Output:  The activity of the person;
    24: For each T
    25:     Store $p_i$ in $TW$;
    26: End for each
    27: For each $T_{min}$
    28:     Classify every point in $TW$;
    29:     Update $\lambda$;
    30:     Update the centers of clusters  $C_j$ $(G_j, W_j)$  ;
    31:     Calculate the threshold $\mu$;
    32: If (max($W_j$)>$\mu$) then  POI = spatial recognition ($G_j$);
    33: End
    34: Update $T_{min}$;
    35: Return Activity
    36: End for each
```

Algorithm 3 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.

3.2.2. SPATIAL RECOGNITION

This task aims at 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 found entity.

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. We propose for this step two ways to execute spatial analysis requests in the city. The first solution is to store a geographic database of the city in the mobile phone in order to avoid using the Internet. The second solution is designed for smart cities and is presented in Chapter 4.

Our first solution is designed to work for the actual situation where cities are not "too smart" yet. Our database is powered by OSM [64] 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 [124] offer an offline version to avoid the constraints linked to the use of networks. For instance, Esri company [125] (one of the world's biggest GIS companies) have provided a whole runtime called "ArcGIS Runtime mobile SDK" [126] 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 storage capacity, for instance, the size of the OSM planet database is over 666 GB [64]. The usual solution to face this problem is to download a specific database to each user depending on the cities where they live, for instance the size of the whole New

York region is only 140 MB [127] including all geo-data types such as roads, highways, buildings, public parks… etc. This can be done manually by the user, since he can define his area of interest while being connected 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 connected 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 [128] is a company which specializes in working with OpenStreetMap. They provide a variety of free extracts in Shapefile [49] and raw OSM format on their download website [129]. After downloading raw OSM Data, and storing it in a spatial database, we use Algorithm 4 to search for the nearest spatial feature to C_j.

Algorithm 4: Spatial recognition

```
Input: A center of clusters $C_j$;
Output: POI, Date;
    1:  If ( there are some geographic entities in the neighborhood )
    2:      //search in the geographic database
    3:      POI = the nearest geographic entity;
    4:      Date= Get current date();
    5:  Else
    6:      POI = null ;
    7:  End
    8:  Return POI, Date
```

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 (50m in our case). Creating buffer zone regions is the most common method used in proximity analysis [130].

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.

3.2.3. ACTIVITY DISCOVERY

Activity discovery is based on the exploitation of tags in OSM. OpenStreetMap data adheres to a simple XML schema with three basic elements: (i) nodes, i.e., 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 [65]. This step allows to have a meaningful human semantic information about the user visited place.

Tags are written in OSM documentation as key=value. The key describes a broad class of features (for example, 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 30).

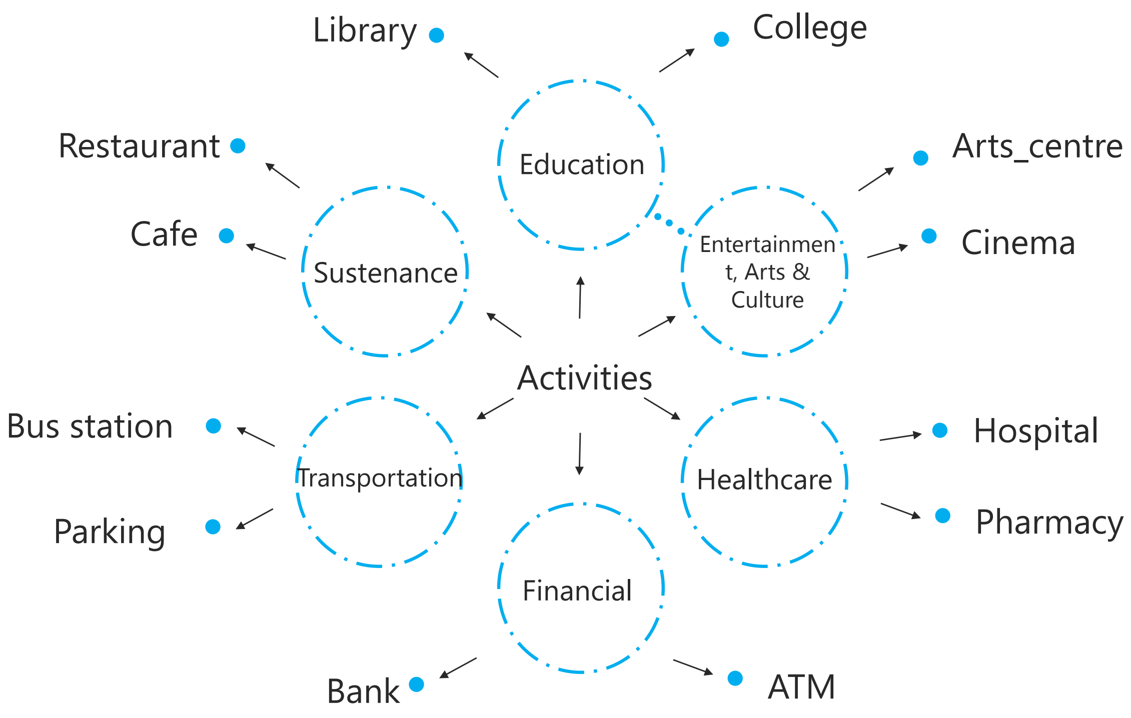


Figure 30: The taxonomy of activities.

Activities are organized in a taxonomy which generalizes the kinds of activities of interest for the movement analysis. For example, the "going to college" activity can be specialized in "education", and so on. 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 5) that searches for the nearest geographic entity, and extracts the tags related to it in order to deduce the activity achieved by the user.

This taxonomy is also used to mine quickly users' past activities, for instance, if we search for the public transportation token 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.).

Algorithm 5: Activity discovery

```
Input: POI;
Output: Activity;
  1:  If (POI = null ) then
  2:      Activity = 'no activity';
  3:  Else
  4:      Tag = Get the tag of POI ;
  5:      Activity = Search for activity taxonomy (Tag) ;
  6:  End else
  7:  Return Activity
```

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 allows to enhance the way users perceive and interact with their surroundings. For instance, authors in [65] 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.

3.3. 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 [131] 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 [132].

3.3.1. Family Coordination Dataset

We used the dataset described in [131] where researchers have conducted experiments in order to reveal the underlying causes of coordination breakdowns that a 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 31).

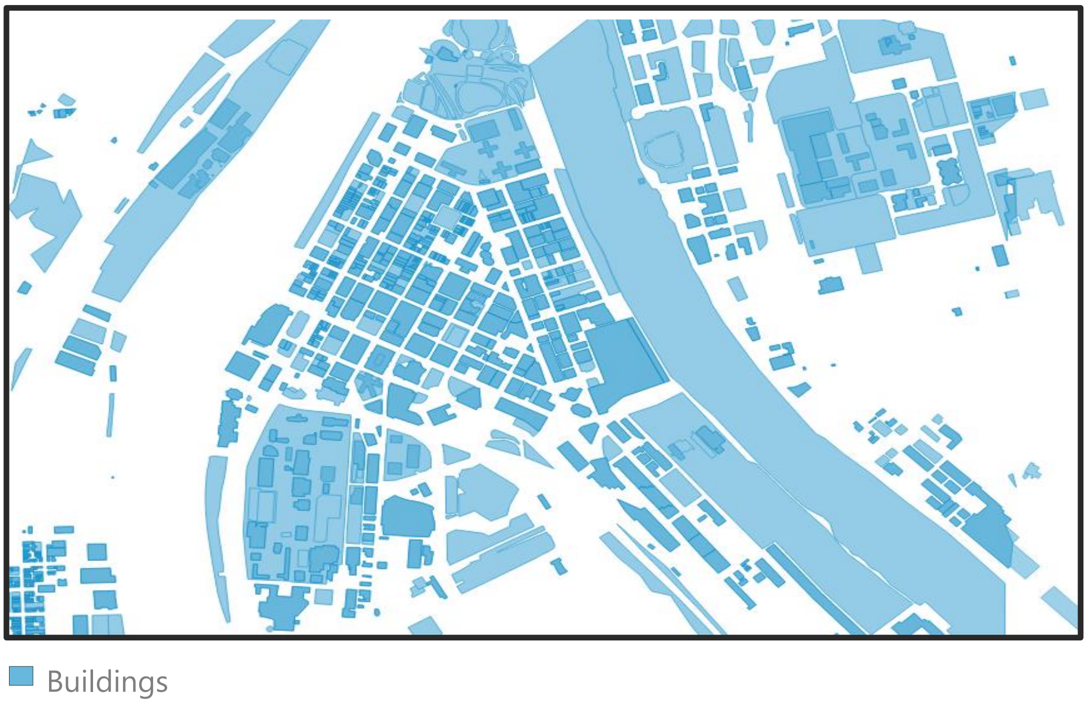


Figure 31: An example of our Spatialite Database that contains the geographic entities of Pittsburgh.

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 called the families, and interviewed 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.

3.3.1.1. Test scenario

We compared our solution to CB-SMoT algorithm [89] described in Chapter 2. CB-SMoT is widely used to extract visited places from locations' trajectories. We have implemented this algorithm using Weka-STPM platform [91], 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 [133] database and the output was the inferred semantic trajectories (a following of visited places retrieved using Weka-STPM).

Our solution was implemented on a Huawei P7 android phone where we have deployed our application that contains the online recognition algorithm, family coordination dataset and a SQLite database [134] 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 test 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 don't have access to the battery's life data in the family coordination dataset. The ability to save the phone's battery and its impact on the accuracy will be presented in the second part.

3.3.1.2. Results Analysis

Results presented in Table 5 represent the comparison of our approach with three versions of CB-SMoT algorithm, by varying its $MinTime$ parameter each time, it was set to 60s in version 1, to 180s in version 2 and to 500s in version 3.

We have tested 10525 activity 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, 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 mentioned in Equations (21) and (22) respectively.

$$Accuracy = \frac{Correct}{number\ of\ tested\ activities} \tag{21}$$

$$Error = \frac{missed + false}{number\ of\ tested\ activities} \tag{22}$$

Table 5: Comparison of our approach to CB-SMoT algorithm

	Our approach	**CB-SMoT V1 $MinTime$=90s**	**CB-SMoT V2 $MinTime$=180s**	**CB-SMoT V2 $MinTime$=500s**
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%

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 CB-SMoT was around 50%. These results are explained in the following points:

Our algorithm succeeds to recognize more activities because it supports the recognition of activities with movements contrariwise of CB-SMoT that recognizes only stop and move activities (see Figure 32).

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 5. And in both cases, the error will be increased since it is calculated in function of the number of false and missed activities.

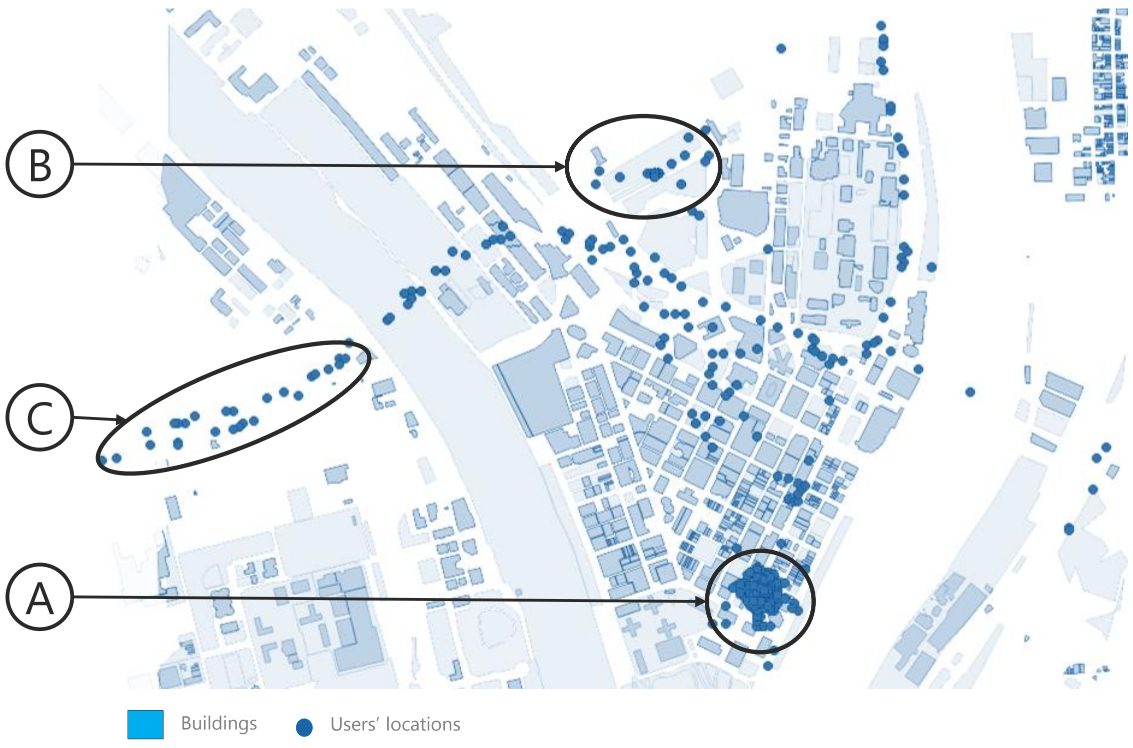


Figure 32: An example of one user's GPS locations projected on the geographic entities layer of Pittsburgh; (A): stationary activity, (B): activity with movement, (C) moves.

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

3.3.2. LIFEMAP DATASET

3.3.2.1. 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.

3.3.2.2. Test scenario

In this step we will compare our approach to the LifeMap application used to recognize users' mobility. The project can be found in [132], the LifeMap dataset in [135] 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 from 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. 33).

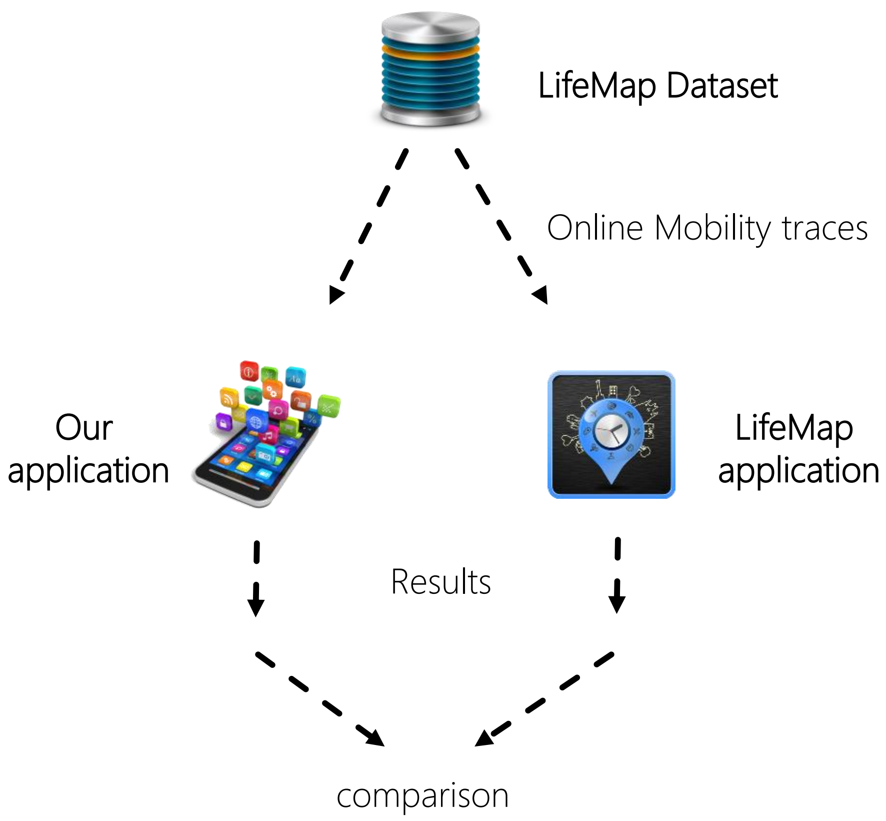


Figure. 33. Test scenario to compare our solution with LifeMap application using LifeMap Dataset.

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 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 in function 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 from the GPS satellites (in the case of LifeMap) than when it retrieves it from a database (in 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 let 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 geo-database [134] where we stored the background geographic data needed in our spatial recognition (see Figure. 34).

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. In order 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).

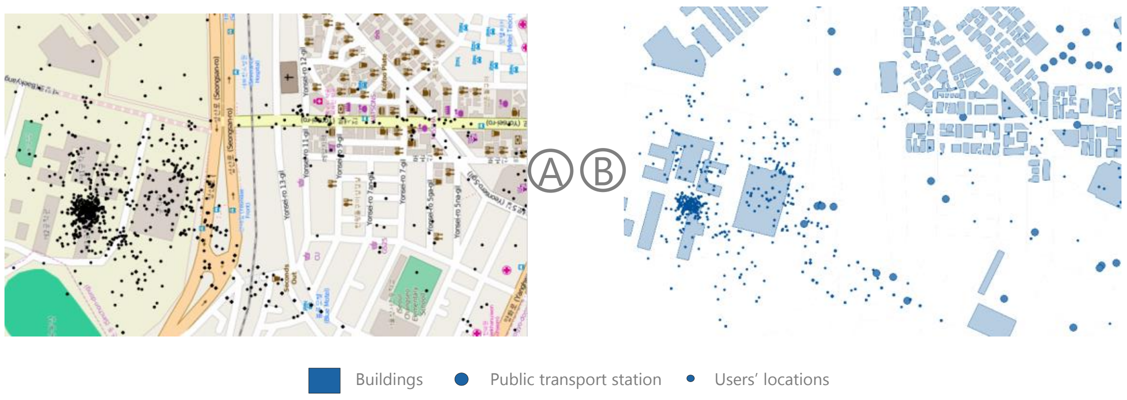


Figure. 34. An example of one user's trajectory projected on: (A) OpenStreetMap raster layer and (B) Vector layer retrieved from our geo-database.

Our application was deployed on the Samsung Galaxy S smartphone, one of the smartphones' model 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.

3.3.2.3. Test results

The total number of recorded hours of battery status in LifeMap dataset is 48900 hours. In order to experiment 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 compared hours is 1632 hours. Due to insufficient space, we present in Figure 35 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 start to save battery life.

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 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 Figure 35. 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).

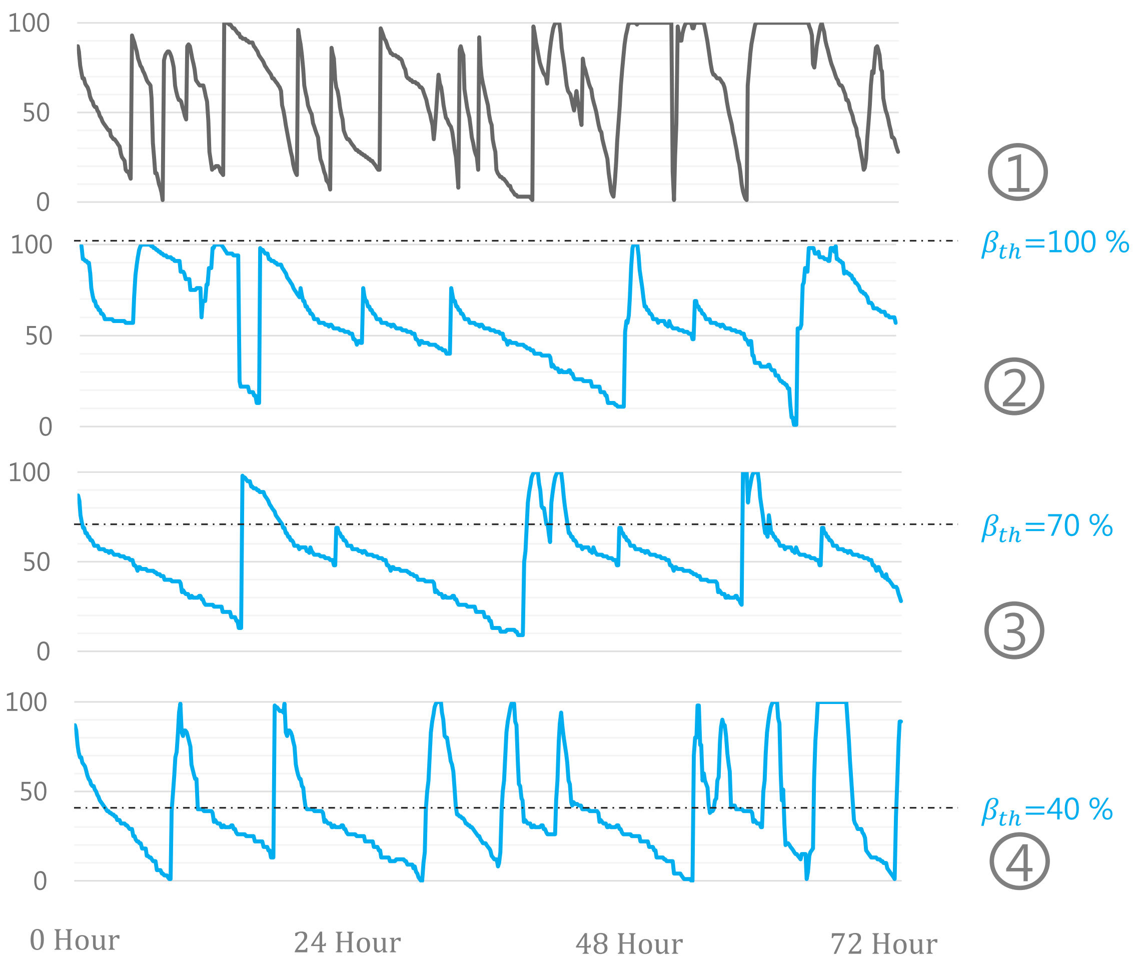


Figure 35. Results comparison between LifeMap and our solution for 72 hours of activity recognition. (1) LifeMap, (2) our solution using β_{th} = 100 %, (3) our solution using β_{th} = 70%, (4) our solution using β_{th} = 40 %.

Thus, β_{th} has an impact on the resource consumption, but what about the accuracy? Is it affected by the variation 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.

Table 6: The impact of varying β_{th} on the accuracy

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

The authors of LifeMap reported in [136] 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 6). This is justified by the fact that the LifeMap's technique for detecting important places is based on the time spent by the user at 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, more β_{th} is set to high value more accuracy drops. This is justified by the fact that when β_{th} 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 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 36 where the accuracy and an estimation of battery life are presented in function of β_{th} . We notice that the higher value of β_{th} is, more we save resources and lose precision.

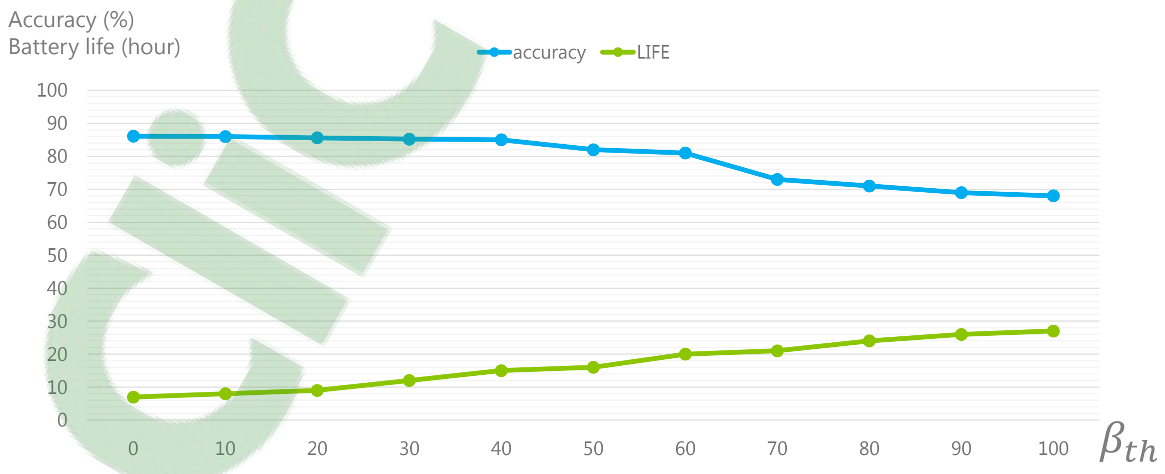


Figure 36: The impact of varying β_{th} on the accuracy.

In order to find an optimal solution that lets saving both battery life and accuracy, we have to resolve the optimization problem that maximizes the battery life function L

(β_{th}) and the accuracy function A (β_{th}) at the same time. This is a multi-objective optimization problem where we optimize two different functions. One of the existing solutions is the weighted sum method [137]. The solution is based on selecting a scalar weight s_i and maximizing the following composite objective function:

$$U = s_1 * \mathrm{L}\left(\beta_{th}\right) + s_2 * \mathrm{A}\left(\beta_{th}\right) \tag{23}$$

In our case the importance of the accuracy and the battery life are 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 Figure 37). Thus, if we need to maximize the accuracy and the battery life at the same time, β_{th} should be set around 60%.

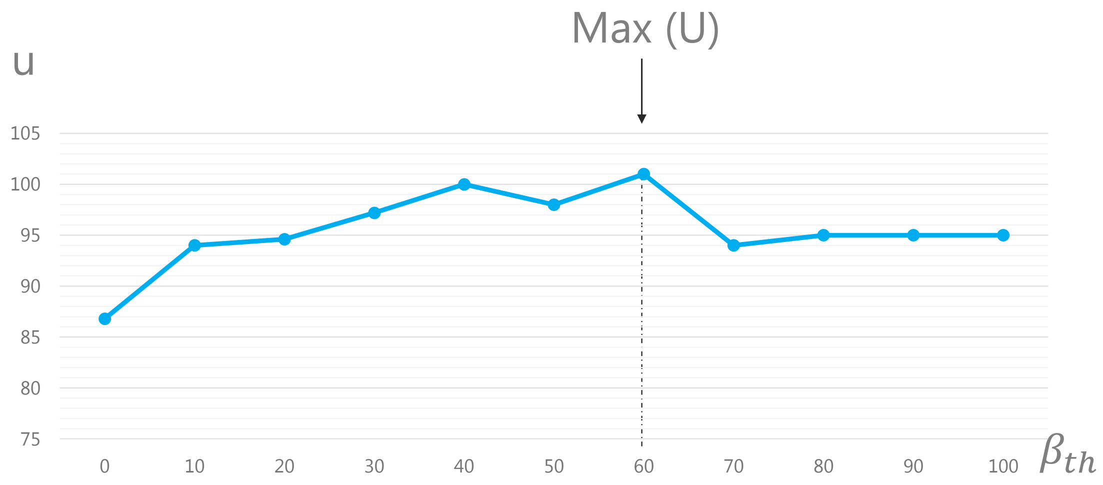


Figure 37: Optimization of β_{th} to maximize the accuracy and the battery life.

The average precision of our approach when varying β_{th} from 0% to 100% is 79% (see Figure 36). For a system that proposes an online activity recognition system while keeping a long battery lifetime, 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 small time outdoor, we set β_{th} small value in order to promote the accuracy and vice versa.

3.4. CHAPTER CONCLUSION

In this chapter we proposed a new battery-aware technique for extracting semantically and incrementally important geographical locations from users' movements. We associate the places visited by individuals during their movements to meaningful human activities using an algorithm that clusters incrementally users' movements 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 to 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 in function of the remaining battery level and the users' behaviors.

Our approach has been experimented using two real case-studies 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, 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 from an individual to another one based on several factors such as age, health, and sex. The implementation of such pattern profiling models will improve the accuracy of our activity recognition process.

The third enhancement is about proposing new ways of spatial analysis for future cities. As seen in the spatial recognition step (3.2.2), storing the geographic database on mobile phones may generate problems linked to the big storage space that may be required to keep such big database on mobile phones. As such, we propose in the next chapter a novel approach to answer this problem. We propose a new spatial exploration system that offers geographic information without using the Internet and without using any predefined geographic database.

CHAPTER 4

NOMABLUE: SPATIAL EXPLORATION IN SMART CITIES

4.1. CHAPTER INTRODUCTION

Smart Cities are employing information and communication technologies in the quest for sustainable economic development and the fostering of new forms of collective life. They facilitate connections between citizens and organizations that are of paramount important for their long-term sustainability. As cities become more complex and their communities more dispersed, questions such as 'where can I find ...' are increasingly pertinent. In this chapter, we introduce NomaBlue, a new vision of spatial recognition in smart cities. The proposed system is based on an intelligent nomadic data collection and users collaboration using smart Bluetooth technology. We demonstrate using evaluations that our approach is capable of proposing an efficient spatial recognition service while supporting a range of users' constraints. Our system doesn't require an access to the Internet, it can operate in any indoor/outdoor area, it doesn't require pre-defined geographic databases, and it uses a new concept of nomadic data collection and sharing to speed-up the circulation of information in smart cities.

The concept of Smart City (SC) as a means to enhance the quality of citizens' lives has been gaining increasing importance in the agendas of research communities. There is no unique definition for a smart city. The interpretations and definitions used by different interest groups, stakeholders and regions vary substantially. However, there is wide agreement about the fact that SCs are characterized by a pervasive use of information and communication technologies [138], which, in various urban domains, help cities make better use of their resources, and assure future viability and prosperity in metropolitan areas.

One of the evolving domains of Smart Cities is smart buildings in which researchers are adopting sustainable building technologies to create smart living and working environments. These entities will keep a range of continues calculations to propose efficient services to citizens, e.g. Aspects related to the quality of life in a residential building such as comfort, lighting, and Heating [139]. Furthermore, these smart buildings will provide a real-time extern information to the city and its citizens, information that will facilitate the urban development and the citizens' quality of life. Consequently, people will no longer visit an ordinary place, but a smart place of interest SPOI. It is a smart urban geo-referenced object where a person may carry out a specific activity. These SPOIs provide enhanced information both unobtrusively and in real time. For instance, SPOI will provide continuously a set of static metadata such as the nature and the type of the building, number of floors, opening hours, …etc. and dynamic metadata such as temperature, humidity, the number of people inside the building, discounts of the day if the building is a shop, menu of the day if the building is a restaurant, …etc.

In SCs, the update of this metadata needs to be instantaneous in order to provide an efficient information about the users' context. This characteristic will increase the use of such service to become a primary input for a new class of mobile services such as smart traffic monitoring, social networking, marketing and cognitive assistance.

Though, existing geographic data providers [63],[123], [139] encounter difficulties to meet the problem of updating information in smart cities, information that can change every hour. In fact, existing geo-exploration techniques are based on pre-defined spatial databases that provide a static information about the concerned geographic entity such as the geographic shape, amenity, entrance…etc. However, this information is insufficient when used in smart cities because of the inability to provide richer and updated information. The main reason is that the majority of spatial data used around the world suffers few months' delay [49]. For instance, let us take the case of a recommendation application that suggests a set of shops located in the surroundings of users, and imagine that a new shop has just opened in the neighborhood. As the existing spatial analysis techniques refer to the pre-defined spatial databases for the recommendation process, this new shop is unrecognizable by these services until the next database update (which can take several months). Moreover, every shop may have a specific daily information that it would like to share, e.g., new products, discounts, new opening hours …etc. Handling these daily updates may become unfeasible using the existing spatial analysis techniques due to the time and the cost needed for such daily data collection and management.

In order to address such weaknesses, we propose a new vision of spatial recognition that offers to the citizens of SCs, an instantiate spatial service without any use of pre-defined spatial databases. Basing on a nomadic data collection and sharing, our work will offer an updated information about the Smart Cities' geographic entities. In this chapter, we introduce NomaBlue, a smart Bluetooth-based and battery-friendly offline service that operates on users' phones without any additional connections requirement.

Citizens in smart cities play an important role in the evolution patterns of SCs as they represent one of the mobile agents of the city. They move in the various corners of the city taking their mobile devices with them everywhere. On the other side, 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 discreetly and incessantly. As such, we will combine the sensing power of mobile phones and smart buildings in order to provide an efficient smart spatial recognition service.

The main idea of our system is based on the observation of a real-life situation. When a person is searching for some place in the city, he would probably start asking people around him if they know where to find this place, he would continue asking the people that he meets until finding his way. NomaBlue reproduces the same process but more efficiently; when two users meet each other in the street, they share a quantity of data instantly and unobtrusively using low energy Bluetooth signals, without any physical contact or even knowing what the other user is searching for.

In order to well introduce our approach, we will introduce next some basic knowledge on geographic data sources as well as social interactions and Bluetooth systems in the cities.

4.1.1. GEOGRAPHIC DATA SOURCES

Geospatial analysis in urban areas has been well-explored in the last decades [49], [130][130]. As such, Geographic Information System GIS has taken off in a big way. Moving from the realms of academic research, the technology was first harnessed to the needs of large information-hungry organizations such as local authorities, environmental agencies, emergency services and utilities providers. More recently, GIS has leapfrogged onto the back of advances in desktop and mobile computing to find applications in every conceivable area of business activity [65], [125].

Location-aware mobile applications are fed from geographic warehouses in two ways: online and offline modes [49]. Online services require the Internet to communicate with the centralized geographic servers, this is insured using cartographic Web services such as Web mapping, feature and processing services [130]. The geographic data can be produced manually by the application developers or by referring to one of the geo-data provider companies such as Google [140], ESRI [125] and OpenStreetMap (OSM) [64].

Let’s take the example of the Waze application [141], it is a community-based traffic and navigation app that shares real-time traffic and road information, the aim is to save everyone time and gas money on their daily commute. Every Waze user that went to share or get traffic information needs to be connected to the Internet because they use Web Feature Services [49] to store and defuse the users’ notifications. Our work is similar to Waze in some points since both works are community-based systems. Moreover, similarly to our work, Waze try to fill the gap between users and geographic data providers by offering information that does not exist in the providers databases, e.g., nearby police activity, accidents, traffic cameras, potholes,…etc. However, in our case, users don’t need to be connected to the Internet and don’t need to harvest and share any information manually since in NomaBlue, the data collection and sharing processes are made without any user intervention.

The second technique to provide geo-data is the offline mode that is designed to avoid using the Internet to get data. The main idea is to store the geographic database of the city in the user’s mobile phone. In fact, many mobile GIS solutions such as [142] offer an offline version to avoid the constraints linked to the use of networks. For instance, Esri company, one of the world’s biggest GIS companies, have provided a whole runtime [126] to support offline mapping. It includes map viewing, interaction, editing and routing while fully disconnected from wireless. However, this technique requires the storage of the whole heavy geographic database in mobile phones, those devices characterized by a limited storage and calculation capacities. Moreover, the user has to download a different database each time he moves to a new city, else, the system is unusable. Unlike these systems, our approach is designed to operate from the first instants without using any predefined geographic database. As the data collection is based on a collaborative data sharing, the system can operate in any area that contains a minimum of collaborators from whom the user will get city information.

4.1.2. Social Interactions in the Cities

In addition to user mobility, another defining characteristic of mobile systems is user social interaction. A variety of new applications focus on facilitating social activities in pervasive systems. For example, new Internet dating services allow clients to use their cell phones' Bluetooth radios to detect when they are in the proximity of a person that matches their interests [143]. Other companies are offering file-sharing software for mobile phones that allows users to share ring-tones, music, games, photos, and video [144]. The digital encounter concept used in these apps is well explained in [145] where authors explore the types of encounters that technology enables. They also investigate the relation between consciousness of communication and intention of interaction in a city context by comparing two prototypes that generate different types of digital encounters.

Authors in [146] have gone further in exploiting social interactions in mobile systems where they investigate how mobile systems could exploit people's social interactions to improve these systems' performance and query hit rate. They study three diverse mobile systems: DTN routing protocols, firewalls preventing a worm infection, and a mobile P2P file-sharing system to confirm that mobile systems can benefit substantially from exploiting social information.

4.1.3. Bluetooth Systems

The constrained physical and link layer technologies in smart cities are generally characterized by low energy consumption and relatively low transfer rates. One of the most prominent solutions in this category is the Bluetooth Low Energy (BLE) or also called Bluetooth 4.0 or Smart Bluetooth [147]. BLE advertising beacons are particularly attractive because of the promise of long battery lives of many years, and so low maintenance requirements. Moreover, the low price of Bluetooth beacons (around 20 $) represents an attractive solution for transmitting SPOI metadata [148]. Beacons transmit a low-power signal that can be picked up by nearby Bluetooth-enabled mobile devices, including smartphones. They broadcast short-range signals that can be detected by apps on mobile devices in close proximity to a beacon (20-200m)[148].

Moreover, the future Bluetooth 5.0 is supposed to quadruple range, double the speed and increase data broadcasting capacity by 800% [149]. Extending range will deliver robust, reliable Internet of Things (IoT) connections that make full-home and building and outdoor use cases a reality. Higher speeds will send data faster and optimize responsiveness. Increasing broadcast capacity will propel the next generation of “connectionless” services like beacons and location-relevant information and navigation. These Bluetooth advancements open up more possibilities to build accessible and interoperable Bluetooth apps.

Bluetooth has a very important role to play in the development of smart cities. Bluetooth is an emerging platform for future telecommunications and thus has a lot to offer. Since the birth of BLE technology in June 2010, we have witnessed valuable works that use this low-energy technology in several areas [150]. In 2014, Regent Street (RS) in London is set to become the first shopping street in Europe to pioneer a mobile phone app which delivers personalized content to shoppers during their visit [151]. Every store along the mile-long RS has installed BLE beacons to broadcast offers to shoppers as they walk past the shop front. The mobile app available to download from the App Store and Google Play, uses Beacon devices which are placed in stores to push personalized messages and offers to consumers' smartphones whilst they stroll down the street and shop. Moreover, users have the ability to pay their bills transmitted automatically by beacons to their phones at the end of their shopping.

In fact, RS application is a good example that supports the usability and feasibility of our approach. RS application can be seen as a use case of NomaBlue, since the spatial exploration is made using BLE devices. However, the RS app wait until shoppers walk past by the beacon to alerts their phones about the shops information. Our work is different from this, if NomaBlue is deployed in RS, the users will be notified about the shops information without necessarily passing by them. The concept of collaborative data exchange that we propose transfers permanently the set of the daily offers and upcoming events from one user to another along the RS.

Furthermore, a British department store group called House of Fraser [152] announced in August 2015 the introduction of beacon-equipped mannequins in its Aberdeen store to provide customers with a more engaging retail experience. When a customer with an enabled smartphone app is within 50 meters of the mannequin, the beacon sends a signal providing them with useful information: details about the clothes and accessories the mannequin is wearing, the price, where the items can be found within the store and links to purchase the items directly from the retailer’s website.

Samsung’s digital discovery tool [153] is another example of BLE usability in SCs. Samsung, Sydney Opera House’s principal partner, created the discovery program to enhance junior tours of the opera house for years 3 and 4 students. BlueCats company [154] provided offline beacon caching to enable the devices to range beacons without Internet connectivity. The ‘Quest to Stop the Mischief-making-Opera Ghost’ app, existing only on the tablets designated for the project, enhances student engagement with the Opera House including questions and information relating to the Australian national curriculum such as the creative arts, drama and indigenous history. The BlueCats BLE beacons used in this project enabled a more immersive, and engaging experience with the tour and the rich history of the Opera House.

All of these projects are city probes: a technological perturbation of the city experience made with the aim of understanding more about how to design user experiences in the city. As seen above, significant efforts have been undertaken for an efficient usage of BLE devices in future cities. In this context, we propose a new spatial recognition system that offers an offline use of geographic data using BLE devices. Unlike the existing solution, our approach is disconnected from the Internet, it doesn't require pre-defined geographic databases and use a new concept of nomadic data collection and sharing to speed-up the circulating information in SCs.

The following sections detail our contribution: Section 2 presents our approach; Section 3 illustrates some usage examples of NomaBlue; Section 4 describes the experimentations. Finally, conclusion and future focus, are summarized in Section 5.

4.2. Nomadic Data Collection For Spatial Recognition

We adopt that in SCs, every building is a kind of alive entity that can interact with its environment. These entities have the intelligence of capturing some information and transmitting others dynamically in function of the situation context. In our work we will focus on the transmission capability of smart buildings. Every SPOI contains one or more Bluetooth transmitters that represent its identity. As said earlier, the constrained physical and link layer technologies in smart cities are generally characterized by low energy consumption. Thus, we adopt that the Bluetooth devices used in the Smart buildings are Bluetooth 4.0 which is an upgrade of Bluetooth 3.0 that includes a power-saving feature called "low-energy technology". It also relies on high-speed data transfers introduced in

Bluetooth 3.0. A BLE device can operate for several years without changing its battery [155].

Thus, every transmitter propagates continuously a low energy Bluetooth signal that contains all kinds of SPOI metadata, static or dynamic, calculated automatically (e.g. number of people inside the SPOI) or put manually by a human (e.g. a slogan of a restaurant), diffused all-time or at specific moments. Actually, the metadata can contain all types of transmittable human information about the SPOI.

For instance, let us take the example shown in Figure 38, imagine that the blue rectangles represent shops located in an urban area and the small dark points represent Bluetooth transmitters located inside every shop. These BLE devices will transmit a constant signal containing information that the shop owner would share with the community. The system accepts a BLE transmitter administrator for each building (in our example, he can be the shop owner) that has the ability to update information contained inside the transmitter using a mobile administrator application, see Figure 38 where the building administrator updates information of the BLE transmitter.

The metadata can contain for example the opening hours, the type of business, the daily or weekly discounts and the congestion inside the shop calculated by counting sensors that reflects the number of visitors. Users moving in the surroundings of this area are continuously informed about the metadata described earlier, consequently, if a user is searching for a specific product, he will get immediately all the opened shops having discounts concerning this product. Moreover, a user can choose between the shops in function of the congestion inside the shop or the waiting time in the checkout line. We will see next, how NomaBlue keeps users aware of the information updated in the neighborhoods.

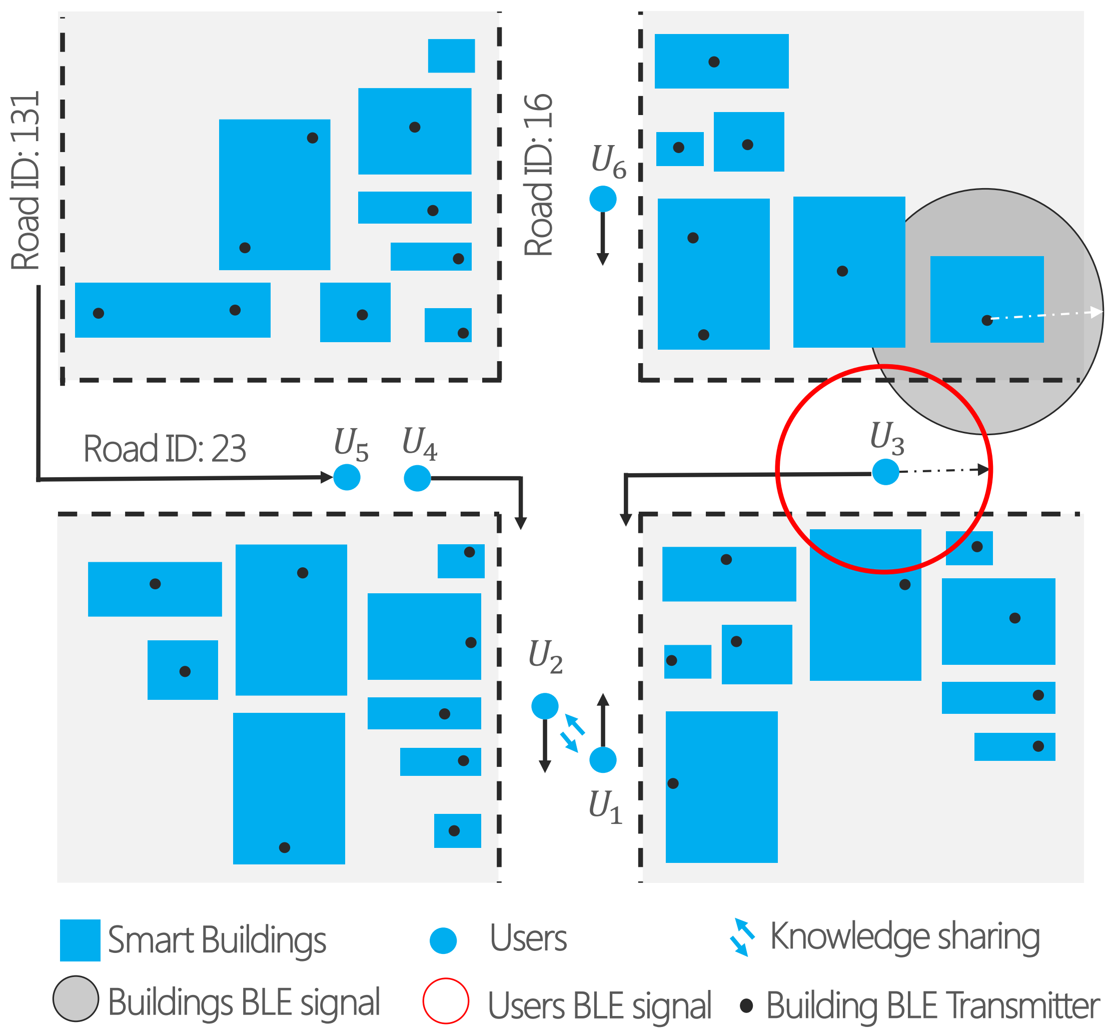


Figure 38: An example of a smart city disposition

We assume that every user has a Bluetooth 4.0 equipped mobile phone. When a person moves in the city, our system tracks continuously the Bluetooth signals found in the neighborhood in order to explore the nearby SPOI. Thus, each time that the application (deployed on the user phone) detects a new BLE signal, it connects to it and extracts the concerned metadata. This metadata is processed and stored in the user's phone (See U_3 in Figure 38). Note that in real life, a geographic area may contain a multitude of BLE signals, not necessarily only those of SPOI. Identifying and securing SPOIs' BLE signals are necessary steps, but as our contribution is principally focused on the nomadic data sharing, we will not go deep in the description of this step. Generally, BLE device companies such as [154] provide their own embedded identification and security protocols. Moreover, they provide a cloud platform to store and protect every beacon data, and offer mobile tools to manage the fleet of beacons, edit its properties and view collected insights.

Therefore, every user U_i collects a quantity of data Q_i that represents the set of already explored SPOIs. In order to propose a better spatial exploration, we introduce a new sharing concept. On every two user encounter in the street, they share a sub-quantity of data instantly and unobtrusively. The shared data will let the user being aware of an unexplored region yet, this characteristic will create a dynamic that will speed-up the flow of the circulating information, and will keep it updated and available for everyone. For instance, suppose that U_1 receives a part of U_2's knowledge (see Figure 38), U_1 will accordingly have information about all the buildings that are situated on the other side of the street without any effort.

However, the number of persons present in the street may become higher in big cities, doing bilateral sharing for every two nearby users may become very resource-hungry for mobile phones. As such, we present in the following sections how we proceed to explore the maximum city space in the least time while minimizing the data sharing.

4.2.1. NOMADIC SPATIAL EXPLORATION

Whenever the user moves, the NomaBlue mobile app in his device connects to every new building BLE signal to explore the SPOI data. The exploration is made under a frequency rate that is linked to the user speed. In order to simplify our approach, we will adopt that the Bluetooth sampling rate T_s for building exploration is 30 sec.

The set of collected data is stored incrementally in the mobile phone. As said previously, the buildings can transmit any communicable human information, so, it was crucial to introduce an ontology part in our system that structures the set of information that the buildings want to send.

Our work in this context is inspired by the OSM ontology [65]. Each building entity 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.

Tags offer a meaningful human semantic information about the SPOI. Tags are written as key=value. The key describes a broad class of features, for example names, types, height…etc. The value details the specific feature that was generally classified by the key. For example, buildings contain the tag building that can take one of the following values:

$$Buildings\ =\ \{apartments, hotel, house, commercial, cathedral, kiosk\ldots etc.\}$$

After that, if the building is commercial we can add the tag shop with one of the following values:

$$Shop\ =\ \{alcohol, convenience, mall, supermarket\ldots etc.\}.$$

Furthermore, we can add other relevant information such as:

$$\{floor =*, wheelchair =*, Internet\ access =*, opening\ hours =*,$$

$$payment\ methods =*, phone =*, website =*\ \ldots etc.\}.$$

Consequently, the data stored in users' phones will look like the example presented in Table 7. At every new BLE signal detected, our system read the Bluetooth data and save a new record in the SPOIs table.

Table 7: A simplified example of the SPOIs table

ID	Longitude	Latitude	Tags
1	-71.0572413	48.4210453	Building= hotel, Name= the hotel, Stars=4, Rooms=25, Beds=45, Breakfast=yes, Internet_access=wlan …
2	-71.0572532	48.4210250	Building=stadium, Sport=soccer, Capacity=70K, Owner=The city…

For information about the technical way of reading Bluetooth data, please refer to the Bluetooth Generic Attribute Profile (GATT) [155] that defines the way that two Bluetooth Low Energy devices transfer data back and forth using concepts called Services and Characteristics.

So far, we have seen that every user in the city harvests data from the buildings BLE transmitters. But as said earlier, if we aim for a more efficient spatial exploration system, we have to introduce a resource-sharing process that let exploring city areas rapidly and efficiently, without necessarily going there. We are going to see in the following parts, under what rules our sharing process is designed.

4.2.2. KNOWLEDGE SHARING

In the parallel of exploring the buildings BLE signals in the SC, the users have the ability to send and to receive information unconsciously and promptly. This sharing process can be seen as a collaborative social mechanism to keep the circulating information updated and widely shared. Therefore, we have implemented some sharing rules in order to maximize the user explored city space while minimizing the exploration time.

Sharing knowledge at every two users meeting is a power-hungry technique. Consequently, in order to propose a phone battery-friendly solution, we adopt that the knowledge sharing is not processed at every moment but at a specific moment called T_e.

In fact, the idea behind T_e is quite similar to the human heart rate. The heart rate can vary according to the body's physical needs, including the need to absorb oxygen and excrete carbon dioxide. When the need to these parameters increases, the heart rate increases too, and vice versa. In a similar way, the time exchange frequency T_e is calculated in function of the necessity of obtaining new knowledge (metadata of new SPOIs). When the user needs to further explore his surroundings the T_e frequency decrease (i.e. the exploration rate increase) and vice versa.

Suppose that the user knowledge linked to the explored SPOIs is represented by a circle that contains all the explored buildings such represented in Figure 39. As such we introduce the knowledge circle *KC*, as an area which its center is the center of gravity of the explored entities.

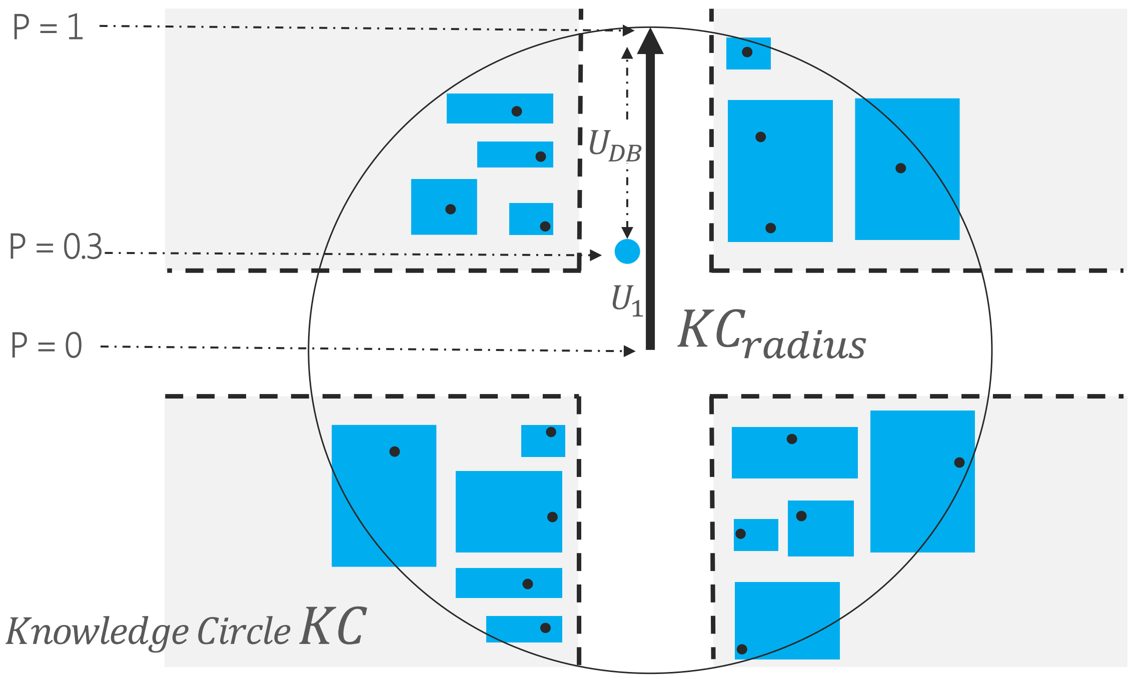


Figure 39: An example of one user's Knowledge circle KC. Blue squares are explored entities. Outside the circle, the area is unknown.

Users' knowledge can be divided into several areas depending on the user patterns. For instance, a user's knowledge can be divided between two distant cities because he has the habit of spending time in both of them. As such KC has to behave accordingly, because if we form one circle for both cities, the KC will certainly contain unexplored areas (compare (A) and (B) in Figure 40).

Moreover, users' knowledge may contain noise, for instance, the geographic coordinate of a BLE transmitter may be introduced wrongly (the buildings will be considered very distant from the user). Another example of the noise generation is when a user receives a small quantity of data from a user that came from a distant provenance. As seen in Figure 40, including noise in KC fake the user's knowledge representation.

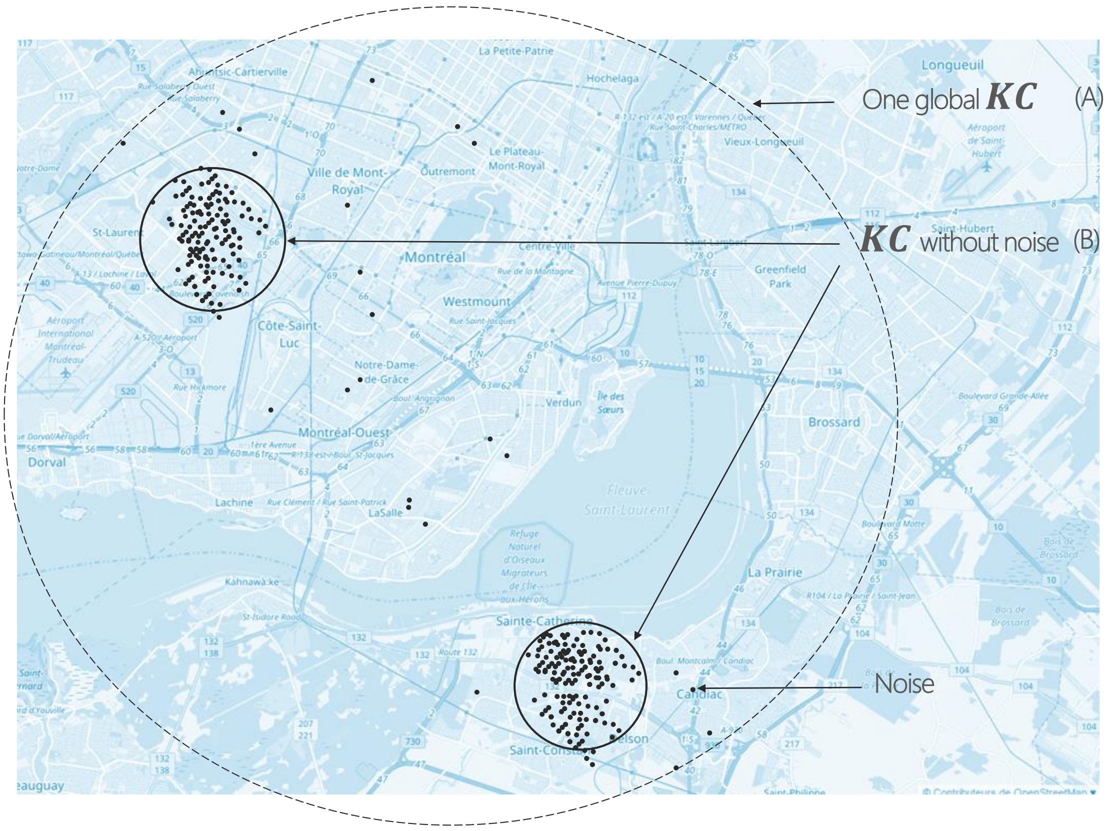


Figure 40: Example of a user's knowledge presentation in Montreal City using one global KC (A) and multiple KCs without noise (B). The user's knowledge is scattered between two regions forming two clusters and some noise between them.

As such, the calculation of KC must be handled carefully to avoid including unexplored areas inside the circle. A user may have several KCs depending on his pattern (ex. (B) in Figure 40). Basing on these facts, we calculate KC by using an incremental version of DBSCAN algorithm [90]. Given a set of points in some space, DBSCAN groups together points that are closely packed together, i.e., points with many nearby neighbors, marking as outliers points that lie alone in low-density regions, i.e., whose nearest neighbors are too far away.

This algorithm can be tuned with two parameters: ε which determines how far to search for points near a given point, and minPts, which determines how many points should be present in the neighborhood of a given point in order to keep expanding a given cluster. In NomaBlue, we use $\varepsilon = 100m$ and $minPts = 5$, which means that the minimum number of entities to form a *KC* is 5 entities in a diameter of 100 m. These parameters are learned from the simulated dataset presented in the experimentation section.

By definition, a user moves inside one of his *KCs*. Our work is to make sure that the user will never be out of his *KC* to avoid going to an unexplored area. Thus, we introduce the position indicator *P* that is calculated as follows:

$$P = U_{DB} / KC_{radius} \tag{24}$$

Where U_{DB} is the distance from the user to the *KC* border and KC_{radius} is the radius of *KC*. If the user has multiple *KCs*, *P* is calculated using the *KC* in which the user is located.

P can be seen as a knowledge security indicator. When the user is in the center of his knowledge $P \rightarrow 0$, which means that the area around him is well explored. However, $P \rightarrow 1$ when a user approaches the border of the explored area, this means that the user will soon be in a situation of knowledge deficit.

T_e is linked to P. When $P \to 0$, the user is near the center of his knowledge. Consequently, there is no need to explore further the neighborhood, so, the frequency time of knowledge sharing T_e will be increased. On the other hand, when $P \to 1$ because the user is near the KC border, we need to enrich the user knowledge quickly by providing new SPOIs, so, T_e will take small frequencies. Based on these conclusions, we calculate T_e as mentioned in the Equation (25) where D_e is the duration of the exchange.

$$T_e = e^{2(1-p)}\, D_e \tag{25}$$

Note that we set D_e as a user-defined threshold in order to simplify our proposal. It can be calculated dynamically by adjusting it in function of the user situation, i.e., speed and number of neighbors.

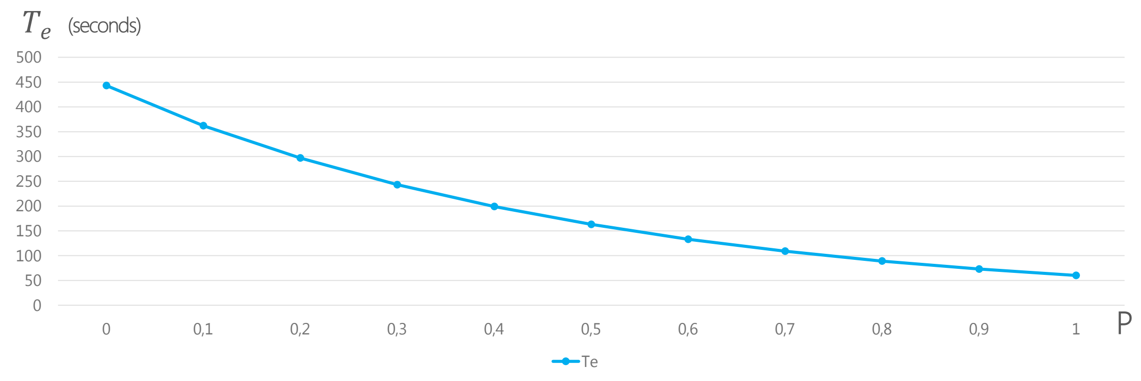


Figure 41: Relation between T_e and P. Using $D_e = 60\ sec$

The Figure 41 illustrates an example of T_e variation in function of P. For instance, when the user position at the moment T is $P = 0.3$, given Equation (25), the next moment for sharing knowledge will be: $T + 250\ sec$ (see Figure 41). So the algorithm will wait 250s to receive knowledge from other users during $D_e = 60\ sec$. After that, the T_e will be updated in function of the new P and so on.

As seen so far, we have discussed how we choose the appropriate time to share knowledge in order to reduce the power consumption of our approach, this time is calculated in function of the users' necessity to explore the city. The next step will continue in this battery saving perspective where we will discuss the protocols of the sharing process.

4.2.3. PAIRING SELECTION

On every T_e, the user will unobtrusively make a data exchange with the nearby other users during a duration D_e. As the time disposed for this step may be short, and in order to minimize the data exchange to preserve phone resources, the sharing process must undergo a pairing selection. The pairing is a mechanism in which two devices establish a relationship by creating a shared secret key known as a link key. If both devices store the same link key, they are said to be paired or bonded, and so, they can communicate mutually.

During the sharing duration D_e, the user may have several other users with whom he can share data. However, we will select during this process only few users basing on two rules; the selected user has to be free (not already paired with another user), and the second rule is that there is no need to receive the same data from several users. This technique aims to discard the useless calculations that drain the phone resources. Consequently, we select each time, a free user with a different profile to be sure that he brings new knowledge. The user profile is deduced from his geographic provenance, or in other terms, we will choose each time a new user that came from different provenance.

For instance, let us take the example presented in Figure 38, there are six users U_i in this city area. For the user U_1, suppose that at the moment T_e, he starts receiving data from other users, the first user that he meets is U_2, so he keeps receiving data from him during a pairing duration D_p (will be introduced later). After that U_1 will choose a new user from whom receiving data, he has the choice between U_3, U_4, U_5, U_6, the algorithm will notice that all the users came from different provenances except U_4 and U_5, consequently, U_1 will receive data from U_3 , U_6 and only the first come between U_4 and U_5.

To differentiate between the user provenances, we introduce the concept of user provenances signature US_i. It is a ID combination of the last three roads that the user has passed through. For example, in Figure 38, when U_4 is near U_1 , he will have the following signature: $US_4 = (16,23,131)$. Thus, at each meeting of a new user, the algorithm checks for his US_i, if the signature is already known, the algorithm will pass to another user, otherwise, it requests a pairing between the two users.

4.2.4. SENDING DATA

We are aware that sending files in urban areas might suffer from some difficulties linked to the interruptions that may break the sending process. Moreover, as the Bluetooth 4.0 is not designed for sending heavy data, we will process by sending a multitude of small compressed files, this allows to insure that a minimum of data is sent if any interruption happens. The theoretical data transfer speed of BLE is up to one Mbit/s, but this rate concerns the physical Layer transfer speed and doesn't account for a protocol overhead. The achievable data rate is a device dependent value between 10 kB/s and 70 kB/s [155].

As such, we send small files that contain 10 SPOIs metadata, the size of each file when compressed is around 1000 b, i.e. less than 1 Kb, this number is obtained from the average size of 1000 compressed files, each file contains 10 different real OSM POIs with different numbers of tags (see Figure 42). Thus, we can send theoretically between at least 5 and 10 files (50-100 SPOIs data) per second, and up to 700 files (7000 SPOIs data).

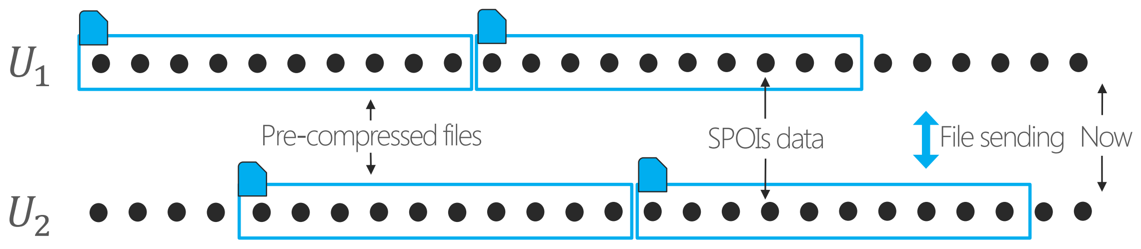


Figure 42: An example of the sending data process

The compression process is a background service that operates incrementally. On every new 10 POIs, the algorithm compress them into a separated file using zlib compression library [156]. Consequently, when the algorithm needs to send packets to another user, it will find these files ready to be sent one by one (see Algorithm 6, lines [1,6]) .

Sharing data between two users accept two scenarios. The first one consists of sharing in the two directions, i.e. every user send and receive data. This scenario happens when both the two users are on their sharing time intervals [T_e, T_e+D_e] (i.e. both users need to receive data).

The second scenario is when only one user is on his sharing time interval (i.e. only one user needs to receive data). For the bidirectional sharing, the algorithm in each device sends and receives the small already compressed files alternately, while in the unidirectional sharing, the sending user will keep sending the compressed files continually (see Algorithm 6, lines [7,10]).

The pairing duration between two users D_p is calculated as illustrated in Equation (26), Nbn is the number of nearby not paired users.

$$D_p = D_e/Nbn \tag{26}$$

The aim of such technique is to harvest the maximum data from different provenance, consequently, a bigger KC. For instance, suppose that $D_e = 60sec$, if the algorithms detects only one nearby user, the whole duration will be dedicated to exchanging with that use. However, if the algorithm detects three potential users, the duration D_e will be divided into three intervals, i.e. 20 sec for each user.

This pairing duration D_p is recalculated after every pairing step, because the number of nearby users may change meanwhile.

Algorithm 6: The overall algorithm of NomaBlue

```
Input: A user position;
Output: A set of SPOIs data;
1:  On every $T_s$
2:      Read and store Beacons data;
3:      If ( number of non-compressed POIs >= 10 )
4:          Compress data;
5:      End
6:  End
7:  On every $T_e$
8:      Repeat until $T \geq T_e + D_e$
9:      On every $D_p$
10:             Choose one  free user with different signature $US_i$;
11:         If ( selected user is on his exchange time )
12:             Bilateral exchange (compressed files);
13:         Else
14:             Unilateral exchange (compressed files);
15:         End
16:         Recalculate $D_p$
17:     End
18:     End repeat
19:     Calculate $KC$, $P$ and new $T_e$ ;
20: End
```

As described in Algorithm 6, NomaBlue users achieve two parallel processes, the first process (see lines [1,6]) is the SPOIs exploration, where every user scan the nearby BLE of SPOIs, on a sampling frequency rate called T_s.

Simultaneously, the second process (see rows [7,20]) is achieved on every time exchange rate T_e. At this moment, the users exchange information basing on their knowledge acquisition necessity, they share their knowledge about the explored SPOIs during an exchange duration D_e.

After describing the overall process of NomaBlue System, we will see in the following section our vision for NomaBlue usability in the real world.

4.3. EXAMPLES OF NOMABLUE USAGE

In this part we will present few examples of use cases where NomaBlue can be used to enhance the existing spatial exploration systems.

4.3.1. GEOSPATIAL DATA COLLECTION

NomaBlue is not a rival of the existing geospatial data provider companies such Google maps [140] and OSM [64], on the contrary, it can represent a support service for these existing technologies. Updating the geospatial data is a complicated and expensive task, and the most important, as seen in the related works, it is a task that take a long time. NomaBlue can represent a nomadic data update system that helps the geographic servers to stay updated and informed by any change in the cities' entities. Thanks to the versioning data stored in each SPOI, it is possible to retrace any update in the SPOIs data. NomaBlue can be programed to transfer periodically the users' data into a central server that treats the various data provided by each user, and use it to update the geospatial databases.

4.3.2. MARKETING

Companies spend valuable efforts in marketing to let the user informed by any information that the company would like to share to improve its sales. By using technology to help salespeople provide a more personalized shopping experience, brands can retain the high level of customer service that people expect, while also offering new services that mimic what shoppers previously have found only online. NomaBlue can be used by a recommendation system that proposes to a user a set of pertinent shop in his neighborhood basing on his personal preferences, wherever he is indoor or outdoor. Moreover, the users will receive alerts directly to their phones providing information on new products, upcoming events and exclusive offers available only to those shopping in the stores that day, while being disconnected from the Internet.

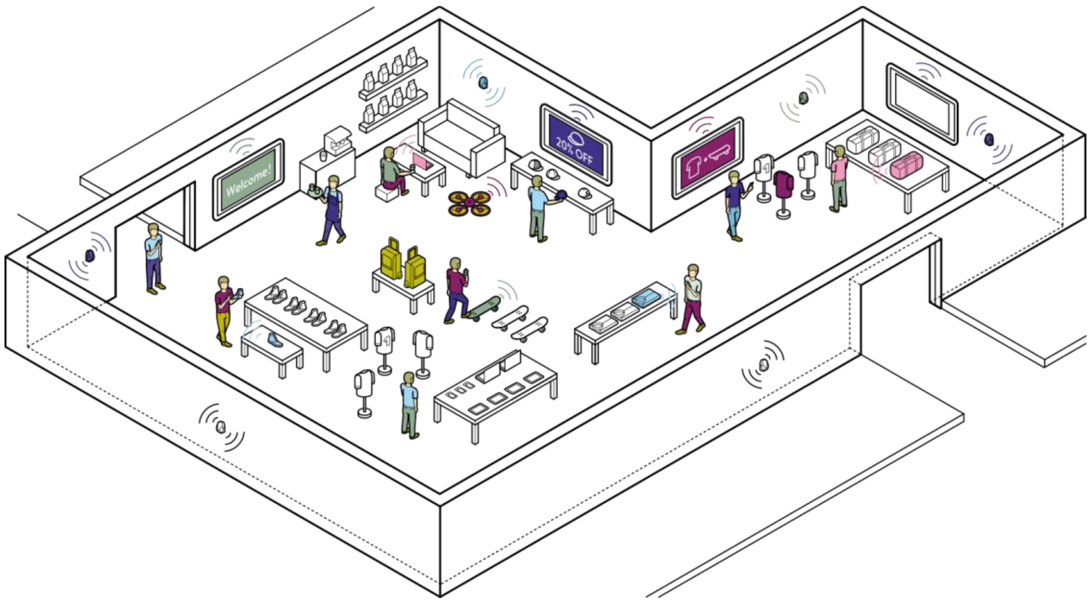


Figure 43: Example of SPOI containing more than one BLE transmitter, this scenario show how the beacons are used to interact with users inside the shop. Source:[157]

Suppose that a mall adheres to such a solution, NomaBlue can offer to one user that has never visited this mall before, a set of relevant information about the pertinent shops and products inside the mall while just entering the mall (see Figure 43). We believe that the first interactions with other users who leave the mall and who potentially explored the shops in the mall will bring enough user knowledge to guarantee a good shopping experience.

4.3.3. NAVIGATION

NomaBlue can be used to enhance the navigation systems. It can operate without using the Internet or any prior geographic database. Suppose that a user has moved to a new city. While just walking in public areas, the user will acquire enough knowledge from other users to navigate easily in the city without having any pre-defined database. The sharing capability can be seen as if the user ask permanently the other nearby users where he can find a pharmacy, a cinema, a restaurant…etc., but with a discreet and fast way. Thus, NomaBlue can support the existing navigation platforms when for any reason, the background geographic data is not available (the Internet absence, Web server failure ...etc.).

4.4. EXPERIMENTAL EVALUATION

We will test our approach using real and simulated datasets to validate the feasibility and the performance of our system. Firstly, we are going to present the experimentations conducted inside the campus of the university of Quebec in Chicoutimi and secondly we are going to expand the tests by simulating the movement and the interaction of up to 100 000 users in the whole city of Montreal (4900 km^2).

4.4.1. Real Dataset

We have deployed 10 BlueCats beacons [154] on one of the university floors, and we asked 10 students to move freely on the same university floor holding their android phones that contain the NomaBlue android app. The sampling frequency T_s was set to 30 sec and the exchange duration D_e was set to 60 sec.

The experiment lasted 60 minutes while changing the beacon locations every 10 minutes. The beacons were located either on the corridors, in offices or in classrooms (see Figure 44). Each beacon transmits the following data: longitude and latitude, the corridor ID in which the beacon is located, versioning information and additional information such as the type of the entity (class, office or laboratory), the capacity of the class, the name of the office occupant, the name of the laboratory...etc.

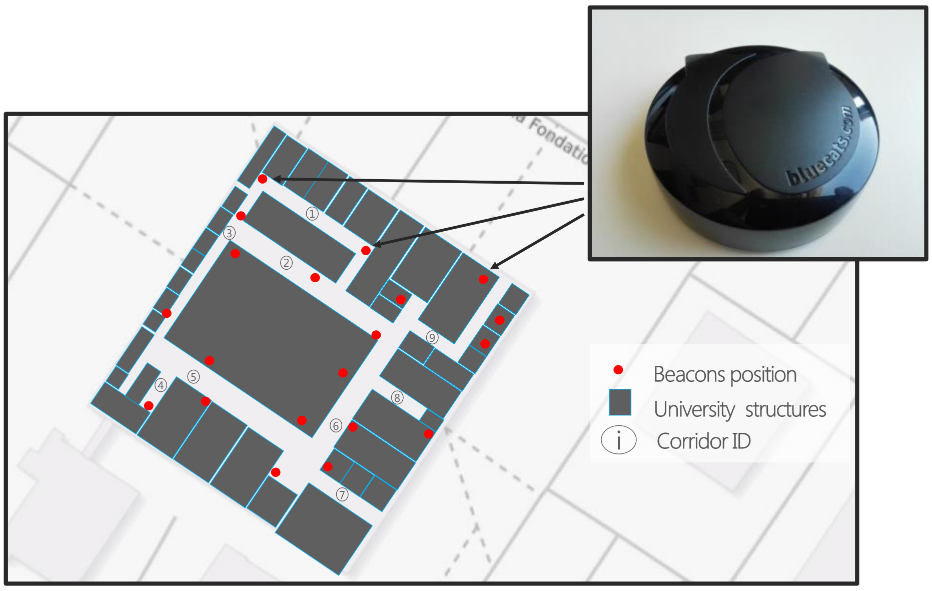


Figure 44: Experimentation process on the university campus

The results showed that the average time exchange frequency T_e during the whole experiment was around 249 sec. In Figure 45 we present the average T_e and P evolution during the whole testing process. We noticed that in most of time, the average position indicator P is between 0 and 0.4, which globally implicated a T_e value between 200 and 440. This means that the users were in most of the time, in the center of their *KCs*.

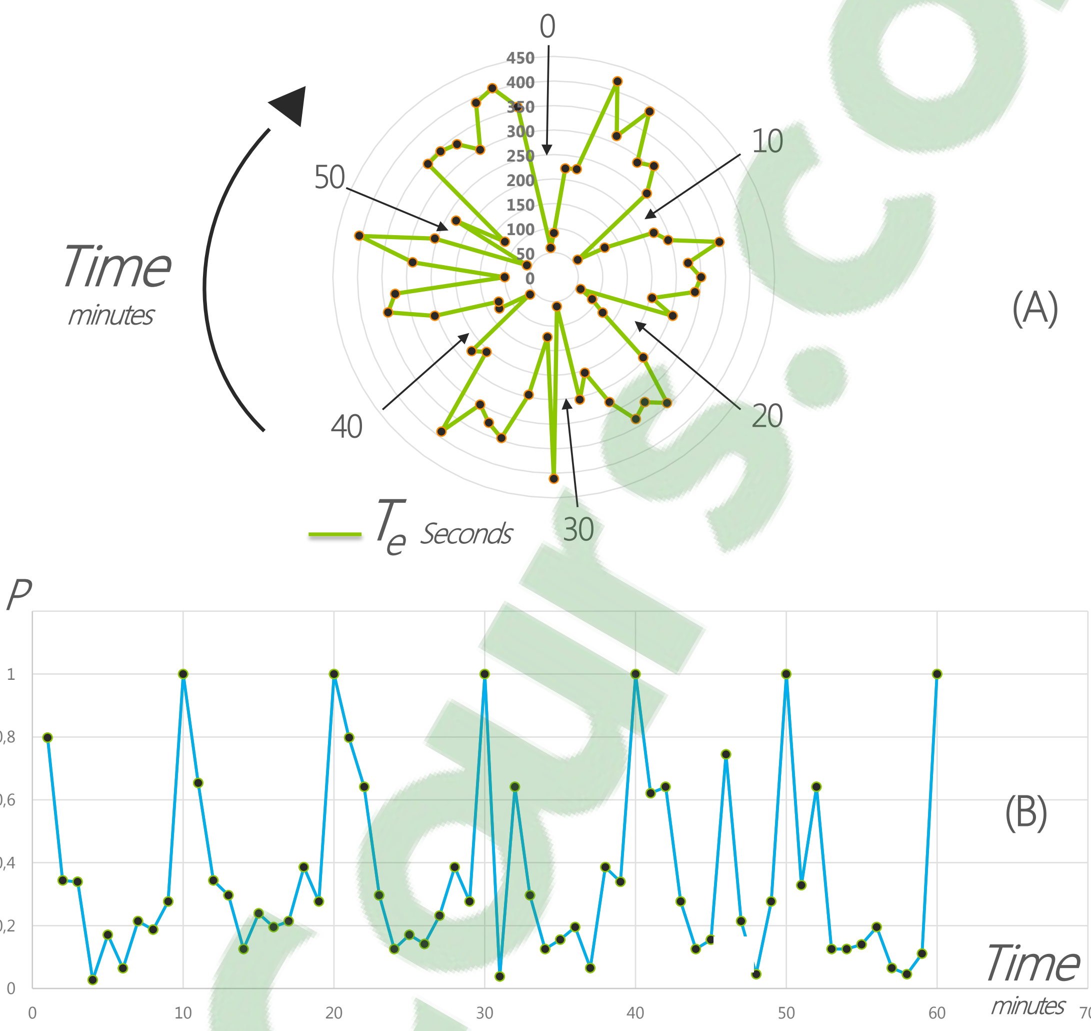


Figure 45: The evolution of T_e in (A) and P in (B) during 60 minutes

We also noticed that every 10 min, $P = 1$ and $Te = 60$ sec. This is justified by the fact that these moments are the moments when we changed the beacon positions (10,20,30,40,50 min), when the beacon positions change, the knowledge circle *KC* automatically shrinks because the users ignore the new beacon distributions, the algorithm interprets the situation as a knowledge deficit because the distance between the user and the border of his explored area is null (*P*=1). Consequently, T_e is set to its smallest value $Te = De = 60\ sec$ to obtain rapidly information from the other users.

In order to test if the fact of introducing T_e has an impact on preserving the battery life, we have tracked the phone's memory usage of our method, a second version of our method (V2) where we discarded the concept T_e (anytime knowledge transfer) and Waze app described in the related works. Results presented in Table 8 represent the mean RAM usage of each application for 60 minutes.

Table 8: RAM usage experiment

	Our approach	**Our approach V2**	**Waze**
RAM usage	26 Mo	48 Mo	67 Mo

Our approach shows a promising RAM usage rate, better than the other solutions. Using a time exchange frequency T_e in our approach has a big impact on preserving the mobile resources, the memory saving in this case was around 45% compared to the V2 of our approach where we discarded the concept of T_e, and we transferred knowledge all the time.

We are aware that this first part of the experimentation does not reflect the real world situation, since the number of users and implemented beacons are small. Consequently, we are going to go deeper in the second test using a simulated dataset that contains many more users and beacons.

Before going to the second test, we have gathered an important parameter that we need to use in the simulated experiment. We perceived during the first test that the average range of the Bluetooth signal transmitted by the beacons was around 48 m depending on the position of the beacons and the number of obstacles. Moreover, we have tested the range of three beacons in outdoor environment for 30 minutes, the average range was around 70 m. Thus, in the simulation process, we are going to simulate a Bluetooth signal that vary between 40m to 70m.

4.4.2. SIMULATED DATASET

We have simulated the movement of up to 100 000 citizens moving in the entire city of Montreal (70Km x 70Km), see (2) in Figure 46. The simulation model was developed using Netlogo platform [158], an agent-based programming language and integrated modeling environment that enables exploration of emergent phenomena.

The developed model called CityAgents can be found in [159]. The main idea of our simulation is to reproduce the movement of citizens while remaining as realistic as possible. Every user moves freely in the Montreal City by following the path of the city roads, meanwhile, he has the possibility of spatial exploring and sharing data as described earlier. However, we suppose that the pairing process between users may fail, this probability is calculated as follows: $NUN * PF$, where NUN is the number of users in the neighborhoods and PF is a random failure probability ratio. We calculate this probability in function of the number of nearby users because it is known in the literature that the number of Bluetooth signals may create a noise that interferes the pairing process [155].

The model uses two background spatial layers provided by OSM: the road layer that defines how the users move in the city, and the buildings layer that contains more than 84200 entities that are used to feed the BLE beacons in terms of information (see Figure 46). In fact, we adopt that every building contains a BLE transmitter and that the information transmitted is the data contained in the OSM buildings layer.

Our simulation model contains two types of agents, firstly, the citizens are agents that move freely in the city, every citizen has a random destination, where when reached, the user stay a random duration ranging from 0 to 60 minutes, after that, a new random destination is assigned to the agent. The citizens move with a speed ranging from 2 km/h to 7 km/h.

The second type of agents, called in NetLogo environment: breed, is the buildings. We design the buildings as living entities that transmit information using its BLE beacons. Beacons are located inside every building and transmit a set of information in a range varying between 40m to 70m;

When users are paired, we use a file transfer speed ranging from 5 to 10 kb/s (this speed represent the minimum transfer speed of BLE technology as described earlier), with a transfer failure probability same as PF.

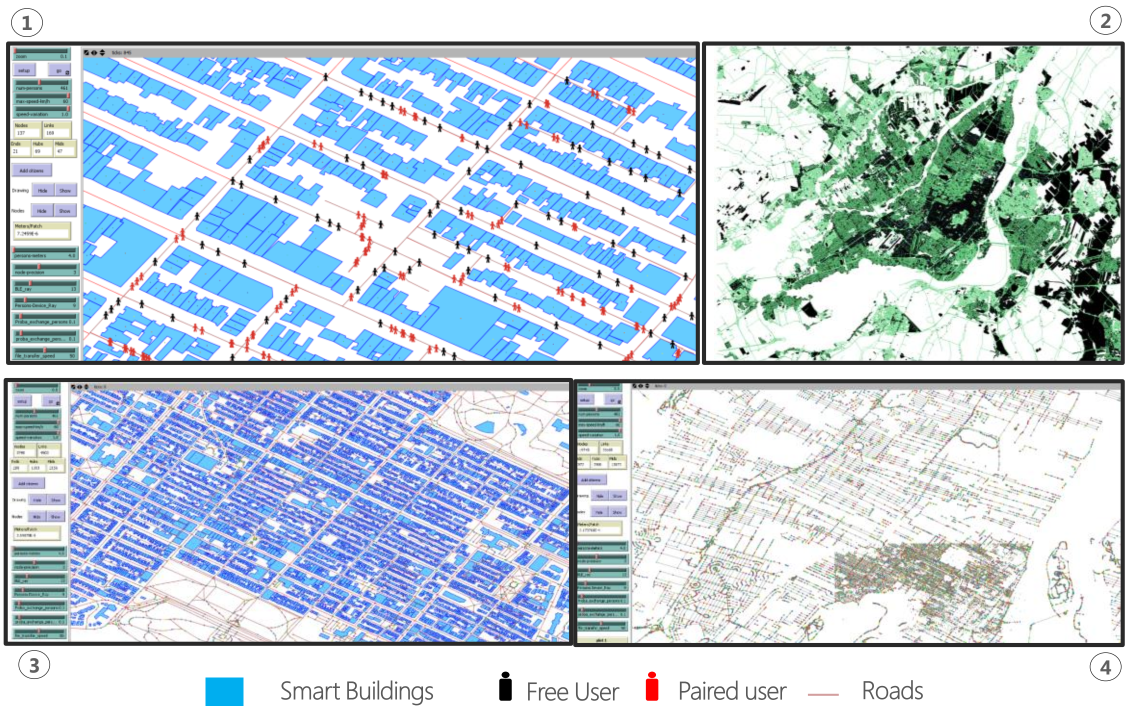


Figure 46: CityAgents: our simulation platform with different zoom levels. (1)(3)(4) The simulation UI. (2) The whole Montreal buildings and roads layers projected from our database.

Before presenting the results, we would like to affirm that the aim of the simulation setting is to approach as much as possible the real world situation, although that these parameters are too hard to calculate in the real world. For this reason, the simulation was not launched once, but several times, in order to ensure that the results are not sensible to one specific parameter value. Therefore, we have developed a java application that takes the NomaBlue simulation as a core, and launch it 1000 times. Every simulation lasted 30 days, which generated a total simulation time around 83 years. Thanks to the time acceleration feature of Netlogo, the real time needed for the whole experimentation was around 50 hours of continuous calculation. Besides, The Java application was designed to make the complex tasks that NetLogo cannot achieve easily such as user's knowledge storage and the mathematical calculations.

In order to measure the effectiveness of our approach, we introduce an exploration satisfaction indicator ESI. The aim of this indicator is to calculate the efficiency of the spatial exploration process in a radius rad. We adopt that every user, during the simulation, is searching for a random geographic entity located in the city.

The ESI_{rad} is the time needed to acquire knowledge about this unknown entity located in the radius rad. When done a new unknown entity is assigned to this user and so on. We repeat this process until all the entities in the radius rad are explored. The user's ESI_{rad} is the average time of all search processes and the global users ESI indicator is the average of all users' ESI_{rad} indicators.

In Table 9 we present the global ESI_{rad} results in radius 100m, 500m, 3km, 15km and 35km using a duration exchange $D_e = 90\ sec$ while varying the probability of failure PF of the pairing and sharing processes.

Table 9: The variation of ESI in function of the exploration radius.

	Radius	**100m**	**500m**	**3km**	**15km**	**35km**
	NB users	10	500	1000	10k	100k
ESI (rounded seconds)	$\text{PF1} = 1/rand(100)$	9	132	311	1398	3471
	$\text{PF2} = 1/rand(\ 1k)$	8	127	305	1382	3134
	$\text{PF3} = 1/rand(10k)$	8	112	288	1340	3160
	$\text{PF4} = 1/rand(100k)$	8	112	287	1303	3091
	Average ESI	**8**	**112**	**298**	**1356**	**3214**

We notice from the average ESI in Table 9 that our approach showed a promising efficiency rate in small areas (when the researched entity is near to the user), the user has a high probability to find the location of the unknown building in the first searching moments. For instance the average search time ESI is around 8 sec when the search area is 100m. However, when we spread the searching radius, the system takes several minutes up to 50 minutes when the searching area is the whole Montreal City. This is justified by the fact that we supposed in the first of the experiment that the users are pedestrians, so their speed ranges between 2 and 7 Km/h. However in real life, the ESI indicator will be much smaller because the users' speed will be higher since they move by using transportation. This high speed will bring the knowledge of distant places rapidly.

In order to confirm this idea, we simulated the users' movements for 10 days by including a ¼ of the population that takes transportation when moving in the city (car, bus..., etc.). The speed of this population was set to a random value ranging from 20 to 50 Km/h, results are presented in Table 10.

Table 10. The variation of ESI when the user's speed is increased.

Radius	**100m**	**500m**	**3Km**	**15km**	**35Km**
NB users	10	500	1000	10k	100k
ESI (rounded seconds)	7	103	288	723	915

We noticed that in this case, the ESI is much smaller than when all the population are pedestrians. For instance, the ESI indicator for the whole city of Montreal dropped from 3214 sec (53 min) to 915 sec (15 min), see the last row and column of Table 9 and Table 10.

We varied the probability of failure PF in order to test the sensitivity of our system to the user device failures. From Table 9, we noticed that when NomaBlue is applied to small places with a small number of users (up to 15km and 10k users) the system is not highly affected by the failure probability, even when it is set to a high value (PF $1 = 1/random(\ 100)$). However when we increase the number of users, the sensitivity to the failure increases too. For instance, in Table 9, when the number of users is set to 100k and the searching radius to 35km, the ESI value for PF 1 is much higher than that for PF 4.

After analyzing our system sensitivity to external parameters such as the failure probability, we will analyze next, the sensitivity to the internal parameters, D_e and T_e. Remember that T_e is an exchange frequency calculated dynamically in function of the users' knowledge acquisition necessity, and D_e is a user fixed threshold that represents the exchange duration. In Figure 47, we present the variation of ESI_{3Km} in function of these two parameters.

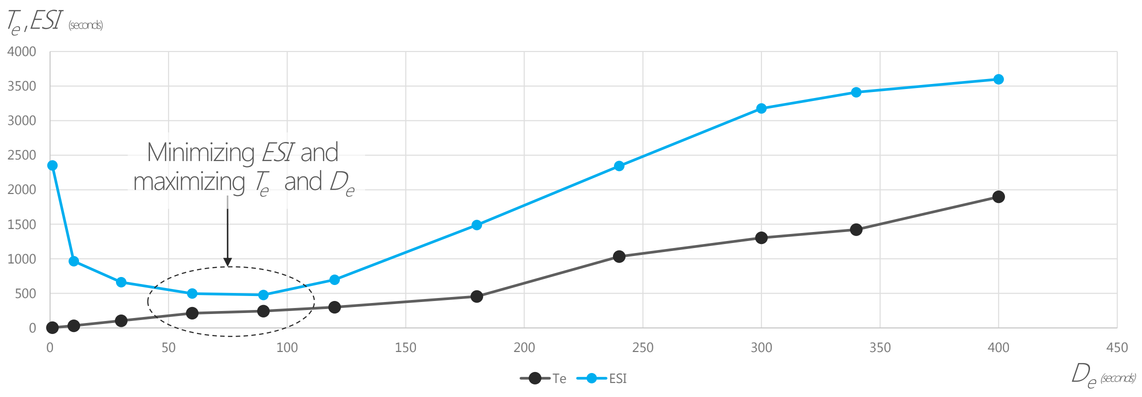


Figure 47: *ESI* evolution in function of D_e and T_e

Varying D_e has an impact on ESI, contrary to what one may think, when the D_e increases, ESI increases too. This is justified by the fact that when the exchange time is high, the exchange frequency T_e is so high that it reduces the efficiency of the spatial exploration (e.g. D_e = 400sec $\rightarrow T_e$ = 33min). On the other side, when D_e is too small, the efficiency is reduced too, because the time allocated for the knowledge exchange between users does not allow to gather enough information, moreover, it increases the number of transfer interruptions.

Consequently, the main idea is to maximize D_e and T_e (in order to preserve the phone resources) and to minimize ESI at the same time. In Figure 47, we have noticed that the best value of D_e is when ESI is around 500 sec. Thus, D_e parameter should be set between 50 sec and 100 sec.

Discussion

We have presented above the exploration satisfaction indicator ESI, we have seen that this indicator varied in function of the studied area size. However, this indicator does not reflect the time needed to a user to obtain the information. This indicator was introduced to test the effectiveness and the robustness of our system. For instance, we reported a value of ESI_{500m}= 120sec, this value is calculated in an environment where we suppose that each second, every city structure is providing a new information. In real life, the frequency of updating information is much higher depending on the application requirement, e.g. every day if the information is daily discounts. Furthermore, NomaBlue is designed as a background service that operates continuously without waiting for a user request, which means that the service will keep harvesting the city information in order to provide an instantiate information. After that, when the user demands a specific information, the system will provide it directly (for the example cited earlier, without waiting 120 sec) because it has been already collected.

4.5. CHAPTER CONCLUSION

In this chapter, we introduced NomaBlue, a new spatial exploration technique in SCs based on a nomadic data collection and sharing. Our system senses the nearby BLE beacons to retrieve information about the city buildings, and shares this information with the other NomaBlue users without using Internet connection. Our proposal has been tested using both real and simulated data to discuss the feasibility of our system. The experimentation has shown that this system is capable of creating an efficient dynamic flow of information in the SCs that improves the user's urban explorations.

We believe that NomaBlue is a promising project that can be used in various fields in spite of not reaching its maturity yet. Further experimentations in real cities have to be done to adjust and improve this work. Future experimentation can be done in Montreal City using LIMVI laboratory [160], a mobile laboratory on smart cities and mobile computing. It includes a highly equipped and adapted recreational vehicle to conduct studies and validations directly in urban environments using several wireless sensors for urban deployment.

In the context of this thesis, NomaBlue is used to replace the local OSM database used in the spatial recognition step (3.2.2) in Chapter 3 when the approach is used in SCs. As such, this thesis proposes two ways to search for the user surrounded geographic entities. The first one by storing the OSM database on the mobile phone, this solution is suitable to avoid Internet dependency in standard cities. The second way is to use NomaBlue as a background service in SCs. NomaBlue offers a spatial database for users without any Internet connection or any predefined database.

After having presented our battery-aware activity recognition and the novel spatial exploration system in Chapter 3 and Chapter 4 respectively, we present next a new activity prediction approach that aims to predict incrementally users next visited paces.

CHAPTER 5

ACTIVITY PREDICTION

5.1. CHAPTER INTRODUCTION

Few decades ago, understanding human behaviors was considered as a mystery where predicting peoples' future was impossible. Many changes have been noticed since this era. Thanks to current advances in location-tracking technologies and data mining techniques, predicting users' behaviors has become possible. Our precedent work presented in Chapter 3 propose an online activity recognition system that offers the possibility to understand what people are doing at a specific moment by inferring incrementally people's interesting places from raw position data. In this chapter, we are proposing an evolution of our precedent system that estimates the users' actions in the future by predicting their next visited point of interest.

We are addressing the issue of predicting the next location of an individual based on the observations of his mobility habits. One of the major problems met when trying to incrementally learn users' routine is the concept drift [161]–[163]. The concept drift means that the statistical properties of the target variable (in our case, the users' habits), which the model is trying to predict, change over time in unexpected ways [87]. For instance, assuming that we learn a user's habits using a traditional algorithm, a user lived in "address1" for one year, after that, he moved to "address 2". This shifting will lead to not only shift the address but probably the habits too. Existing algorithms will take few months to detect that the user is making new habits. For instance, if we take the support of 60 %

using Apriori algorithm [164], it will take 7 months to detect the new habits, in the meantime, all the next locations proposed by these algorithms are probably false since the predictions are based on the old routines and not the new ones.

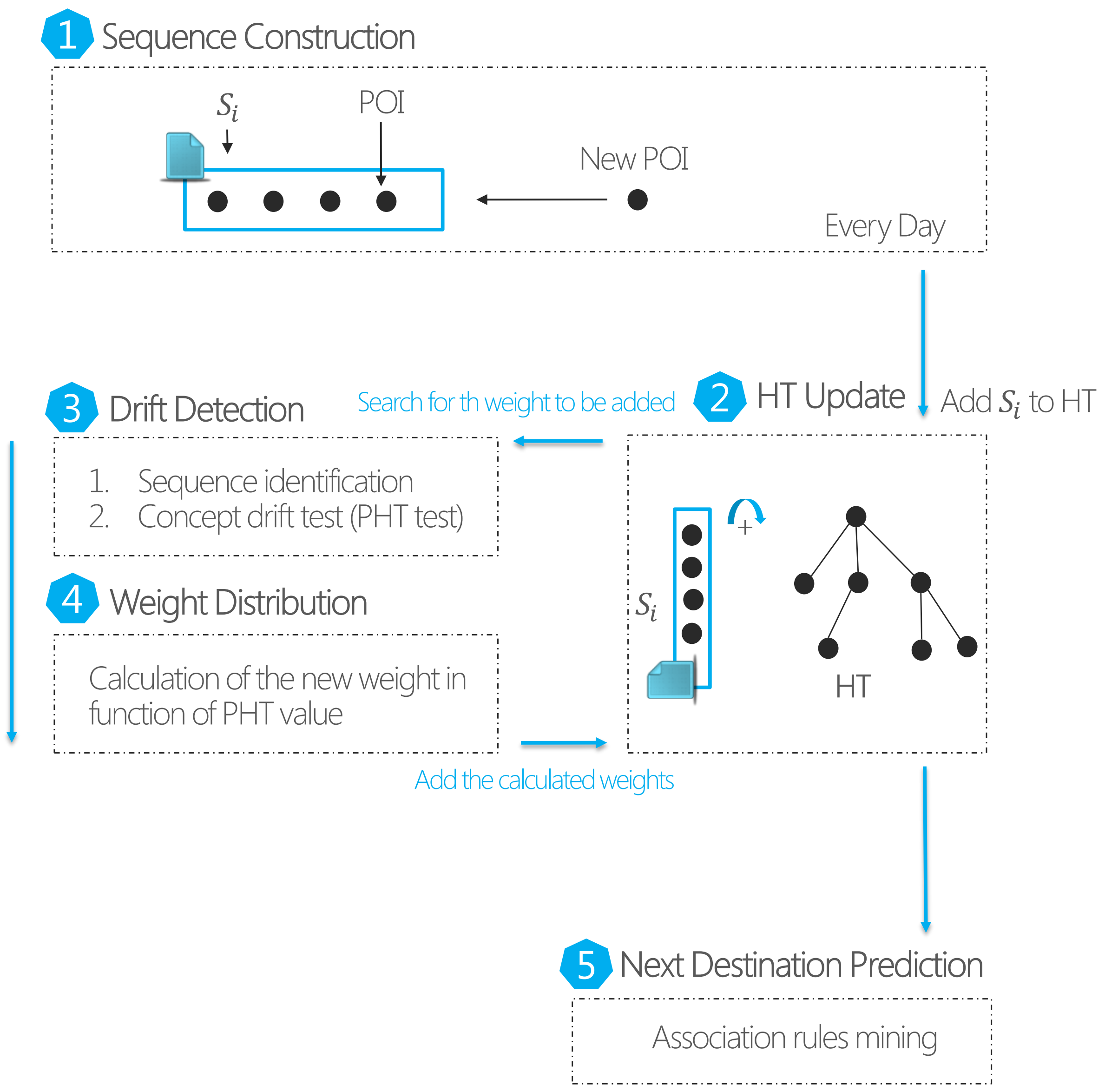
Current works [78], [106], [108], [136], [165] that try to learn users' routines and predict their future routines fail in the ability to deal with the changes in users' behaviors. Moreover, there are only few works that attempt to incrementally predict the users' next location.

We bring a novelty via a novel online algorithm that extracts association rules carrying data drift during the learning process. Hence, the main idea is to help the new habits to become quickly frequent. We introduce a new criterion of support calculation based on a weight distribution instead of the classic number of occurrences.

The following sections detail our contribution: Section 2 presents our approach; Section 3 describes the experimentation. Finally, conclusion and future works are summarized in Section 4.

5.2. PREDICTION OF NEXT DESTINATIONS FROM IRREGULAR PATTERNS

Our solution learns users' habits by analyzing their visited places POIs. Our approach begins by constructing a sequence of POI_i that represents the tracking of users' daily habits. Every sequence is stored incrementally in a tree structure called Habits' Tree 'HT'. On every sequence arrival, our algorithm checks for a drift in the distribution of sequences and allocates a new weight to the sequence concerned. When achieved, the new sequence is added to the user's HT and finally, the algorithm predicts the next POI using the association rules drawn from HT (see Figure 48).



1 Sequence Construction
S_i
POI
New POI
Every Day
Add S_i to HT
2 HT Update
Search for th weight to be added
3 Drift Detection
1. Sequence identification
2. Concept drift test (PHT test)
4 Weight Distribution
Calculation of the new weight in function of PHT value
Add the calculated weights
S_i
HT
5 Next Destination Prediction
Association rules mining

Figure 48: The overall approach of our prediction model

In the following sections, we are going to present every part of our model starting by the sequence construction step.

5.2.1. SEQUENCE CONSTRUCTION

Users' habits are composed of daily routines that define the pattern of users' movements in the city. This step aims to represent users' habits via a series of routines called sequence S_i. Every sequence contains a set of disjoint singletons POI_i (i.e. we can't find the same POI many times in the same sequence) and terminates at the end of the day (daily habits). For example, assuming that the user achieved the following activities during a day: Home, Work, Restaurant, Work, Gym, Home; the algorithm will construct incrementally two sequences from these habits:

S_1 : {Home, Work, Restaurant}.

S_2: {Restaurant, Work, Gym, Home}.

In fact, when

Algorithm 7 detects a new POI that already exists in the sequence, like “work” in the example above, it stops constructing S_1 and creates a new sequence S_2. The reason of our proceeding is to optimize the storage of the sequences (this point will be discussed further).

The sequences S_i are stored in a Habits’ tree HT. It is a new data structure that we proposed and that takes the form of a special tree. In HT, every node represents a POI and is characterized by a weight w_i that represents the weight of POI_i’s in HT. Moreover, every node has an identifier ID_i that aims to identify the sequences by an integer (see Sequence Identification in 5.2.3).

To give a brief idea of the way the data are structured in HT, we illustrated in Figure 49 how our algorithm stored the two sequences S_1 and S_2 from the example above in a form of a series of nodes characterized by <name, w, ID >.

Algorithm 7 is executed on every POI_i arrival, when S_i is finally constructed we move to the next step: the habits’ tree update.

Note that an improvement can be made at this step to respect the definition of streaming learning, i.e. every new POI is treated instantly without waiting for the construction of S_i.

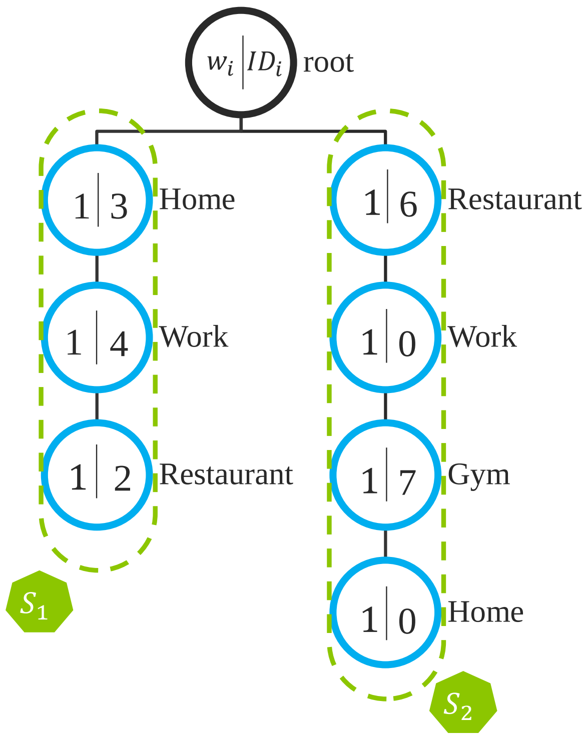


Figure 49: HT Structure construction

Algorithm 7: Sequence construction

```
Input: A $POI_j$;
Output: Sequence $S_i$;
1:  $S_i$ = null ;
2:  **For each** new $POI_i$
3:      **If** (! $S_i$.contains($POI_i$) and  StillTheSameDay) then
4:      $S_i$ = $S_i$ + $POI_i$
5:      **Else** //new sequence
6:      **Return** $S_i$;
7:      **If** (StillTheSameDay) $S_i$ = Last $POI_i$
8:      **Else** $S_i$ = null ;
9:  **End for each**
```

5.2.2. HABITS’ TREE UPDATE

The work of [111] inspired us to plan this part. Authors in that work proposed a new algorithm for mining incrementally association rules called DB-Tree. DB-Tree is a generalized form of FP-Tree (FP-Growth [166]) which stores in descending order of support all items in the database and counts all items in all transactions in the database in its branches. The DB-Tree is constructed in the same way as done in FP-Tree except that it includes all the items instead of only the frequent 1-items.

In our habits’ tree, tree data structure wasn’t chosen arbitrarily. Indeed, using this structure and storing all the sequences (frequents and not frequents) eliminate the need of rescanning the entire database to update the structure. Algorithms such as Apriori and FP-tree rescan the entire database when previously not frequent POI_j become frequent in the new update. As such, our algorithm scans only the branches concerned by the new sequence, which optimizes the computational complexity. Additionally, storage optimizations are achieved using this structure, sharing paths between items in tree structure leads to much smaller size than that in a traditional database (see Figure 49, Figure 50, Figure 51).

Let us take the following example: we take the HT presented in Figure 49 and we add two new sequences:

S_3 = {Home, Work, Gym, Cinema}.

S_4 = {Home, Gym}.

From the updated HT presented in Figure 50, we can observe how the notion of sharing paths in the nodes Home and Work leads to a compression of the database dimension (this characteristic will be discussed deeper in the experimental section). Additionally, we can notice that the algorithm added these sequences with different weight than the previous. The main reason is that it detects new behaviors, more details will be provided in the next sections.

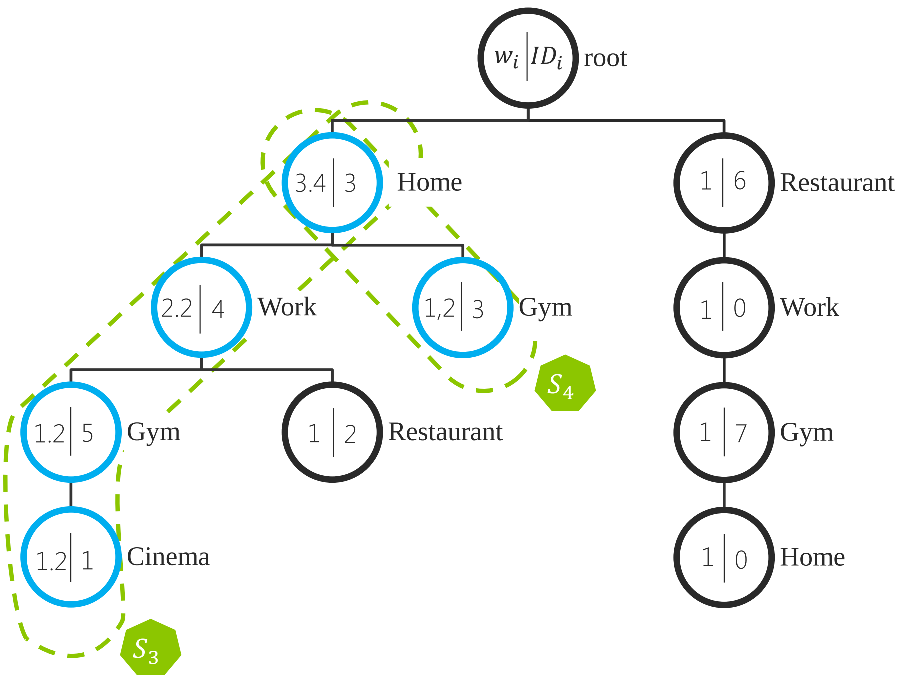


Figure 50: Sharing paths in HT structure

On the arrival of a new sequence S_i , the Algorithm 8 recursively processes each POI_i in S_i . If the POI_i exists in HT, the concerned node's weight is updated, otherwise, the algorithm adds a new node with a new random ID_i, and a new weight w_i where the details of calculation is given in the next section.

For example, supposing that after a certain time of learning, a user's HT is structured like in Figure 50, the next day, the user did the following sequence:

S_5 = {Home, Work, Gym, Friend's Home}.

Figure 51 shows how our algorithm updates the nodes: Home, Work, Gym; and adds a new POI: Friend's Home. Note that Friend's Home was added to HT with an unknown weight because it will be calculated in the next section (5.2.4).

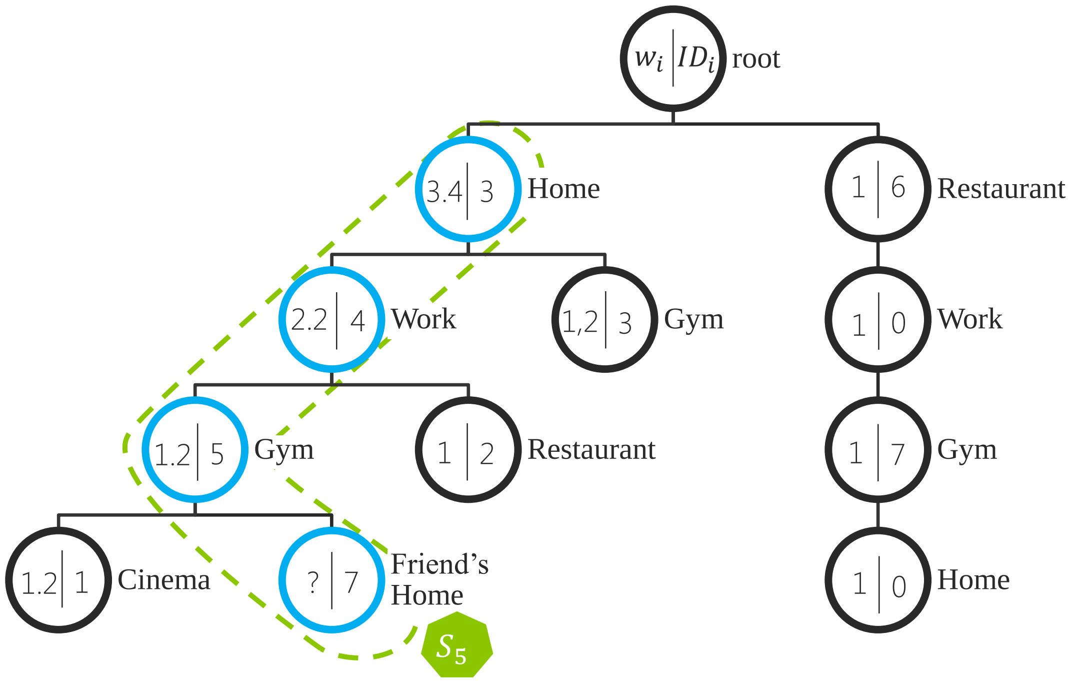


Figure 51: An example of HT updated

So after, seeing how we proceed to structure HT on every S_i arrival, it is time to present how we calculate the weight that will be added in every node. Actually, the weight is calculated in two ways depending on if the new S_i is frequent or not, the formula is given as follows:

$$\begin{cases} w_i = 1 + wd_i \text{ , if } S_i \text{ is not frequent} \\ w_i = 1 \qquad\qquad \text{ , if } S_i \text{ is frequent} \end{cases} \tag{27}$$

The drift's weight wd_i is an extra weight that will be added if the S_i is considered as a not frequent habit. As we see, our distribution behaves in two ways: a traditional way when the sequence S_i is frequent (adding only 1 in every node), and a special way when S_i is not frequent (adding $1 + wd_i$), see lines [4,9] in Algorithm 8. The reason why we proceed this way is that we are trying to help only the new habits that aren't frequents to become frequents, once arrived, we stop our help not to promote a sequence (habit) relative to another.

Algorithm 8: Habits' tree update

```
Input: A sequence $S_i$;
Output: HT;
1: For each (POI in $S_i$ )
2:     HT.add POI
3: End for each
4: If ($S_i$ is frequent )
5:     $w_i$ = 1;
6: Else
7:     $w_i$ = 1+ Extra weight distribution ($S_i$);
8: End
9: HT.add Weights ($w_i$);
```

The next section presents how we detect the drift in the user habits in order to calculate this extra weight wd_i. Our contribution aims to track the drift in users' habits, and to distribute the weights basing on these behavior changes**.**

5.2.3. DRIFT DETECTION

This part aims to track changes in users' habits, or in other terms, it aims to check if the new sequence is an old or a new routine. Our technique is divided into two steps: firstly we formulate mathematically each new sequence and secondly we use this number to test if this sequence is new or not in order to figure out if it is a concept drift or not.

5.2.3.1. Sequence identification

In order to be able to use a concept drift test, it is crucial to parse the information contained in S_i into a quantifiable entity. In other terms, instead of comparing a sequence of strings, we are going to parse every sequence into a number to facilitate the comparison. In Fact, the introduction of ID_i in each node was in this perspective. Every new sequence will be represented by a variable called xi where xi is obtained by the concatenation of each ID_i present in S_i.

For instance, in the example cited in Figure 49, if we want to calculate the mathematical representation of $S_2 = \{\text{Restaurant}, \text{Work}, \text{Gym}, \text{Home}\}$. The variable $x2$ will take a value 6070 (i.e. ID $= 6$ for Restaurant, ID $= 0$ for Work, ID $= 7$ for Gym, ID $= 0$ for Home). Similarly, the identifier of S_1 is $x1$= 342 (i.e. ID $= 3$ for Home, ID $=$ 4 for Work, ID $= 2$ for Restaurant).

Once the sequence is identified by an integer, we use this number in the next step, the concept drift test.

5.2.3.2. Concept Drift Test

We use the Page Hinkley Test (PHT) [167] to detect changes in users' habits, PHT is a sequential analysis technique typically used for monitoring change detection. It allows efficient detection of changes in the normal behavior of a process which is established by a model. The PHT was designed to detect a change in the average of a Gaussian signal [161]. This test considers a cumulative variable U_T defined as the cumulated difference between the observed values (in our case the sequences' identifier xi) and their mean till the current moment.

The procedure consists of carrying out two tests in parallel. The first makes it possible to detect an increase in the average. We calculate then:

$$\begin{cases} U_t = \sum_{d=1}^{t} (x_d - \overline{x_d} - \delta), U_0 = 0 \\ m_t = \min(U_t), t \geq 1 \\ PHT = U_t - m_t \end{cases} \tag{28}$$

The second allows detecting a decrease in the average as follows:

$$\begin{cases} U_t = \sum_{d=1}^{t} (x_d - \overline{x_d} + \delta), U_0 = 0 \\ M_t = \max(U_t), t \geq 1 \\ PHT = M_t - U_t \end{cases} \tag{29}$$

Where $\overline{x_d} = (\sum_{d=1}^{t} x_d)/t$ and δ corresponds to the magnitude of changes that are allowed. When the difference PHT is greater than a given threshold (λ) a change in the distribution is assigned. The threshold λ depends on the admissible false alarm rate. Increasing λ will entail fewer false alarms, but might miss or delay some changes. Controlling this detection threshold parameter makes it possible to establish a trade-off between the false alarms and the miss detections.

In order to avoid issues linked to the parameterization of λ, we were inspired by the work of [168] where authors propose a self-adaptive method of change detection by proving that λ_t can be self-adapted using this equation: $\lambda_t = f * \overline{x_d}$ where f is a constant called the λ factor, which is the number of required witnesses seeing the changes.

In our case, habits' drift that we are tracking are brutal, in the sense that the user doesn't usually change his habits gradually, so, we don't need a large number of witnesses to detect the changes, that's why we put $f = 2$ in case of increasing average and $f = 1/2$ in case of decreasing average.

For instance, assuming that the user has been doing the sequence S_5 =Home, Work, Gym, Friend's Home for 10 days, after that he changes his habit from S_5 to S_4 = Home, Gym (see Figure 50 and Figure 51).

We are going to illustrate the detailed calculation of PHT after the change of habit (the 11th day). First, we need to represent the two sequences mathematically, as the concatenation process seen before, every sequence will be presented by xi_d where i is the number of sequence S_i and d is the number of the day. As such, $x5_{10} = 3457$ and $x4_{11} =$

33 are the representation of S_5 in the 10th day and S_4 in the 11th day respectively (see Figure 50 and Figure 51). The new habit number $x4$ is less than $x5$, so we are tracking a decrease in the average, consequently we use the equations (29) as follows :

$$\begin{cases} U_{11} = \sum_{d=1}^{11} (x_d - \overline{x_d} + \delta), U_0 = 0 \\ M_{11} = \max(U_d), t \geq 1 \\ PHT = M_{11} - U_{11} \end{cases}$$

$$\begin{cases} U_{11} = U_{10} + (x4_{11} - \overline{x4_{11}} + \delta) \\ M_{11} = \max(U_d), t \geq 1 \\ PHT = M_{11} - U_{11} \end{cases}$$

As during the 10 days before the change of habit, the user was doing the same habits:

$$U_{10} = \sum_{j=1}^{10} \left(x5_j - \overline{x5_j} + \delta\right) = \sum_{j=1}^{10} (3457 - 3457 + \delta) = 10\,\delta$$

As said before δ represents the minimum of allowed changes. For us $\delta = 1$, so $U_{10} = 10$.

$$\begin{cases} U_{11} = 10 + (33 - 3457 + 1) \\ M_{11} = \max(U_d), t \geq 1 \\ PHT = M_{11} - U_{11} \end{cases}$$

$$\begin{cases} U_{11} = -3413 \\ M_{11} = 10 \\ \text{PHT} = 3423 \end{cases}$$

After calculating the threshold $\lambda_{11} = 1/2 * \overline{x5_{10}}$ = 1728.5, we notice that we are in the presence of a change of habits because PHT $> \lambda_{11}$.

After having a value (PHT) that represents the stability of our users' habits, we are going to use this variable to distribute the weight in every new POI's node in HT.

5.2.4. Extra weight distribution

As said earlier, our approach behaves in two ways (see Algorithm 8): a traditional way when the sequence S_i is frequent (adding only 1 in every node), and a special way when S_i is not frequent (adding $1 + wd_i$), the extra weight wd_i is calculated using an exponential function like follows:

$$wd_i = 1 - e^{-\frac{1}{2}p_t} \tag{30}$$

$p_t = \frac{PHT}{\lambda_t}$ represents an indicator of the user's state. The greater is PHT than λ_t more we are sure that the user is doing something new, and vice versa. For example, when there are no new habits in the user behaviors $PHT = 0$, so, $p_t = wd_i = 0$, see Figure 52. By against, each time p_t approaches the value 1, or exceeds it, we conclude that the user has a drift in his habits.

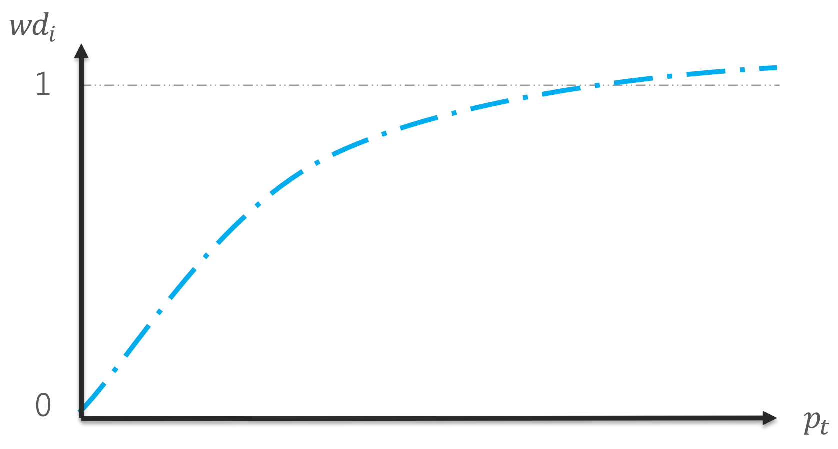


Figure 52: Mathematical representation of wd_i

For instance, in the past section we calculated the PHT = 3423 value on the 11th day and we found that it exceeds the threshold λ_{11}=1728.5. In this case the weight added will be calculated as follows:

$$w_{11} = 1 + wd_{11}$$

$$wd_{11} = 1 - e^{-\frac{1}{2}\frac{3423}{1728.5}} = 0.62$$

So the weight that will be added to HT will be w_{11}= 1.62.

Algorithm 9: Extra weight distribution

```
Input: sequence $S_i$;
Output: Weight $w_i$ ;
    1:  $xi$ = HT.GetSequenceId($S_i$);
    2:  Calculate $\lambda_t, U_T, m_t, M_t$;
    3:  Calculate $PHT$;
    4:  Calculate $wd_i$;
    5:  **Return** $wd_i$
```

After calculating the weight that will be added on S_i arrival by using Algorithm 9, the next step is to get the association rules from HT to predict the next activity.

5.2.5. NEXT DESTINATION PREDICTION

5.2.5.1. Association Rules Mining

Our technique is inspired by FP-growth algorithm and one of its incremental versions called DB-Tree [111]. The main difference between our work and theirs is that we introduced a weight distribution function that tracks the habits drift. Secondly, our structure doesn't order the tree's items in descending order of support like done in DB-tree. In fact, DB-tree and FP-growth don't make a difference between {Home, Work, Restaurant} and {Restaurant, Work, Home}. Obviously, we can't use such technique when analyzing users' habits, otherwise, it will lead to gross errors.

Suppose we have a database with a set of items like illustrated in Figure 50, I={Home, Gym, Work, Restaurant, Cinema} and $MinSupport = 60\%$ of database transactions. To compute the frequent POI_i after constructing the habit tree HT, the algorithm mines the frequent POI_i that satisfy the minimum support represented by the $MinSupport$ percentage of the maximum item's weight in HT. From Figure 50, we have the weight of every item like follows (note that for every item, we add up the corresponding weights contained in the entire tree, for example Home's weight = 3.4 + 1.0 because it appears twice in HT):

I = {(Home: 4.4), (gym: 3.4), (work: 3.2), (restaurant: 2), (Cinema: 1.2)}

The minimum support will be: $MinSupport = \frac{60}{100}(4.4) = 2.64$. Thus, the frequent POI_i are all items greater or equal to 2.64 as {(Home: 4.4), (gym: 3.4), (work: 3.2)}.

The next step is mining the frequent patterns from HT and association rules that are similar to those in FP-growth [111] that was already described in the related works.

5.2.5.2. *Next Activity*

After mining the association rules from HT, we get from the same example in Figure 50 these rules:

$Work, Gym \rightarrow Home$

$Work \rightarrow Gym$

$Home \rightarrow Gym$

$Home \rightarrow Work$

$Home, Work \rightarrow Gym$

Predicting the next activity relies on choosing the most appropriate association rules with the highest weight that represent the user situation, and using the resulting clause as predicted next activity. For example, if we know that the user has gone from home to work, using the last association rules: $Home, Work \rightarrow Gym$, we can predict that he will go next to the gym.

Finally, we present in Algorithm 10 the whole process that predicts the next activity from users' current location and some of their past locations if exist. First we construct a sequence of POI from the current location, then we calculate the weight that will be added to HT and we update the tree using this weight. We search for the association rules and we predict the next location based on the user's past activities (if exist).

Algorithm 10: Final algorithm for activity prediction

```
$S_i$ = Sequence construction ($POI_j$);
Output: Next activity;
    1:  $S_i$ = Sequence construction ($POI_j$);
    2:  $w_i$ = Extra weight distribution ($S_i$) ;
    3:  HT = Habits 'tree update($S_i$);
    4:  **Return** Next location = Activity prediction (HT,UPA);
    5:  $S_i$ = Sequence construction ($POI_j$);
```

Note that the sequence of these steps is not essentially like mentioned in Algorithm 10. We present in this algorithm the whole process to explain how to start from a simple localization to predict the next user's activity. In real life, these processes can be used differently, for example, there is no need to search for the association rules on every sequence arrival. The most correct way is to update HT on every S_i (because the update does need a whole scan of the tree, so it's not expensive in terms of calculation), and to search for the association rules in an appropriate time depending on the application requirements.

For example, supposing that we try to assist a patient of Alzheimer's disease, the user tends to forget his next activities. The appropriate time that we are talking about is when the proposed system detects an anomaly in the user's behaviors because the user makes mistakes as he doesn't know where to go next. At this moment the system searches for the association rules in order predict the next probable activity.

5.3. Experimental evaluation

In the experimentations, we address the following questions: (i) how does our algorithm compare with other states of the art, (ii) how does the disparity of habits affect the algorithm results and (iii) how does our algorithm behave in a mobile environment.

5.3.1. Datasets

5.3.1.1. Synthetic Data

We asked three users with different profiles to note their daily habits for three months. The choice of users wasn't arbitrary, we chose them with different habits disparity level: (i) user 1 with very recurrent habits, (ii) user 2 with moderately recurrent habits and (iii) user 3 with very low recurrence level. The dataset contains 107 different activities (POIs).

5.3.1.2. Real Data

To push even further the level of our experiment, we used a renowned dataset from the Microsoft research project GeoLife [77]. The GPS trajectory dataset was collected in (Microsoft Research Asia) Geolife project by 182 users in a period of over three years (from April 2007 to August 2012). A GPS trajectory of this dataset is represented by a sequence of time-stamped points, each of which contains the information of latitude, longitude and altitude. This dataset contains 17,621 trajectories with a total distance of about 1.2 million kilometers and a total duration of 48,000+ hours (see Figure 53).

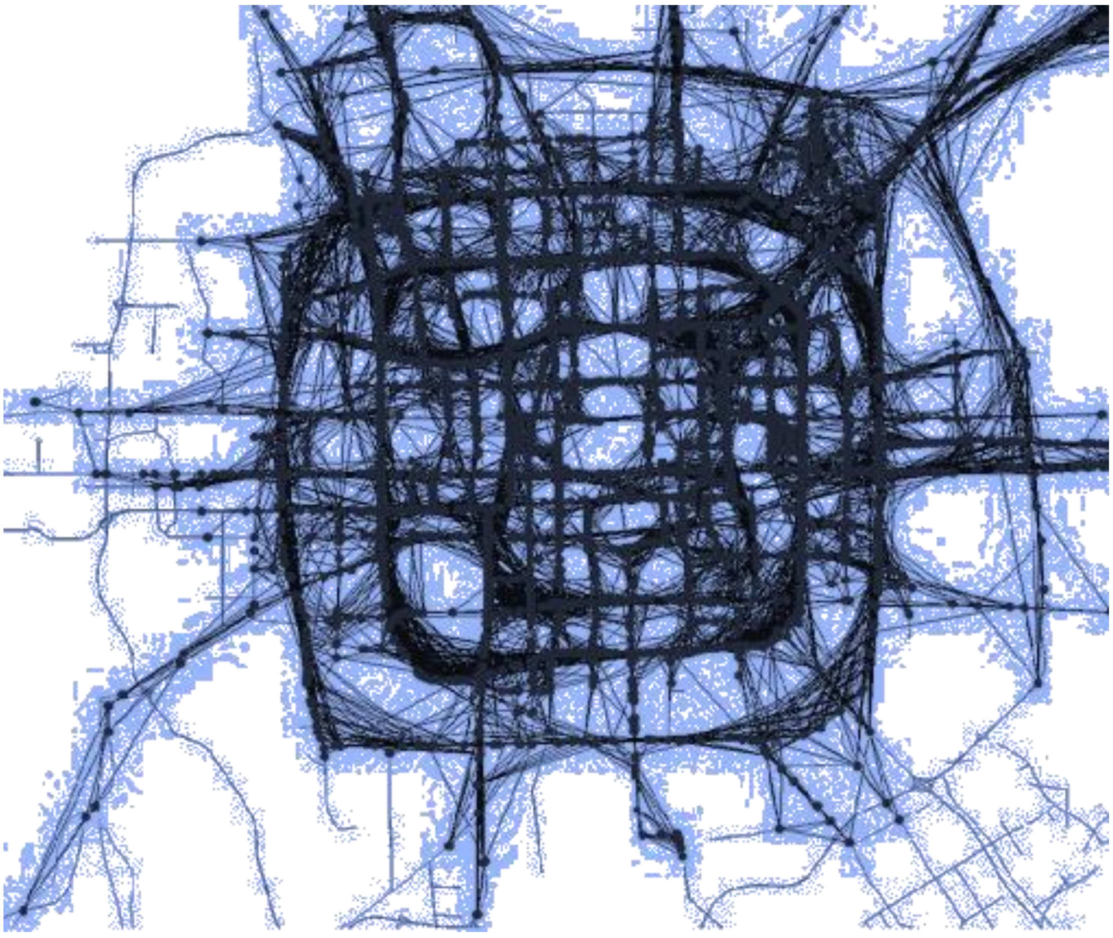


Figure 53: A Global view of Geolife trajectories

These trajectories were recorded by different GPS loggers and GPS-phones, and have a variety of sampling rates. 91 percent of the trajectories are logged in a dense representation, e.g. every 1~5 seconds or every 5~10 meters per point. This dataset recorded a broad range of users' outdoor movements, including not only life routines like go home and go to work but also some entertainment and sports activities, such as shopping, sightseeing, dining, hiking, and cycling. The final dataset includes 2831 different visited places (POIs).

5.3.2. TESTING PROCESS

Test process in association rules represent a delicate step, since how to test is related to the field of study. We created a specific testing process to represent as much as possible the activity prediction situation. The concept is based on the introduction of a second virtual user that will follow the real users' movements all the time. However, during every sequence of POI, the virtual user will have a memory lapse (he will forget his next destination), in this case our algorithm will predict a next localization that will be compared to the real next activity (see Figure 54).

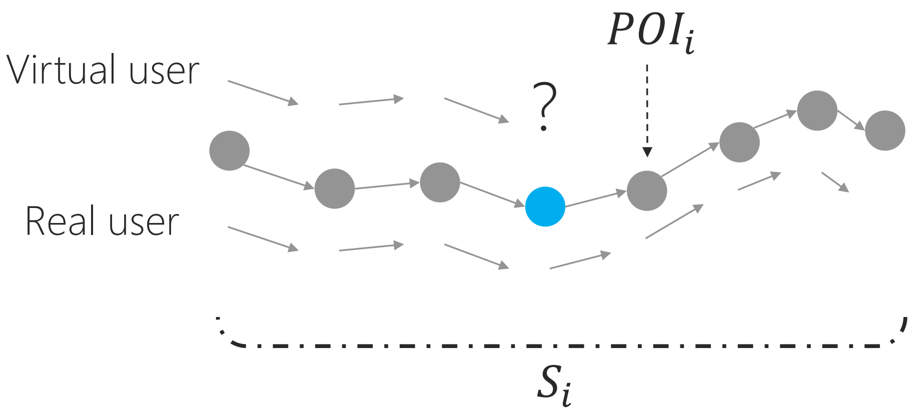


Figure 54: Testing process with real and virtual users

We adopt that the memory lapse happens randomly. In fact, on every S_i arrival, our algorithm generates a random position between 1 and S_i's length, this position represents the POI's position where the virtual user will forget his next destination.

The precision represents the number of sequences where activities were well predicted on the total number of sequences, by against, the global error GE, which is calculated using the number of sequences where the activities' predictions were mistaken on the total number of sequences.

The global error contains two types of error: learning error LE and habit error HE. LE represents an error in the prediction of the next activity when referring to the past activities. For example, the user has the habit of going sometimes from home to work and other times to drive his child to school. If we do a test starting from the POI "Home", our algorithm will predict for example work as next destination because it's the most recurrent activity after home. All the time when the user will go driving his child to school and when we predict work as next activity, the algorithm will record a LE.

HE represents the disability to predict next activities because the user's precedent POIs are not frequent. This error can be seen as a similarity index, the greater is HE, the more data is scattered.

We evaluated our approach by highlighting three dimensions: first we tested our algorithm with a standard dataset, secondly with a dataset that contained a concept drift, and finally we tested the performance of our work on the mobile environment.

5.3.3. STANDARD INCREMENTAL ACTIVITY PREDICTION EXPERIMENT

In this step we used four users' data: the three from the synthetic dataset and one user from Geolife dataset. Results presented in Table 11 represent the precision, GE, LE and HE of our algorithm on every user data.

Table 11: Standard incremental activity prediction results

	Simulated data (107 POIs)			Geolife
	User 1	User 2	User 3	User 4
Precision	83%	71%	61%	68 %
GE	17%	29%	39%	32 %
LE	11%	14%	13%	9 %
HE	6 %	15%	26%	23 %

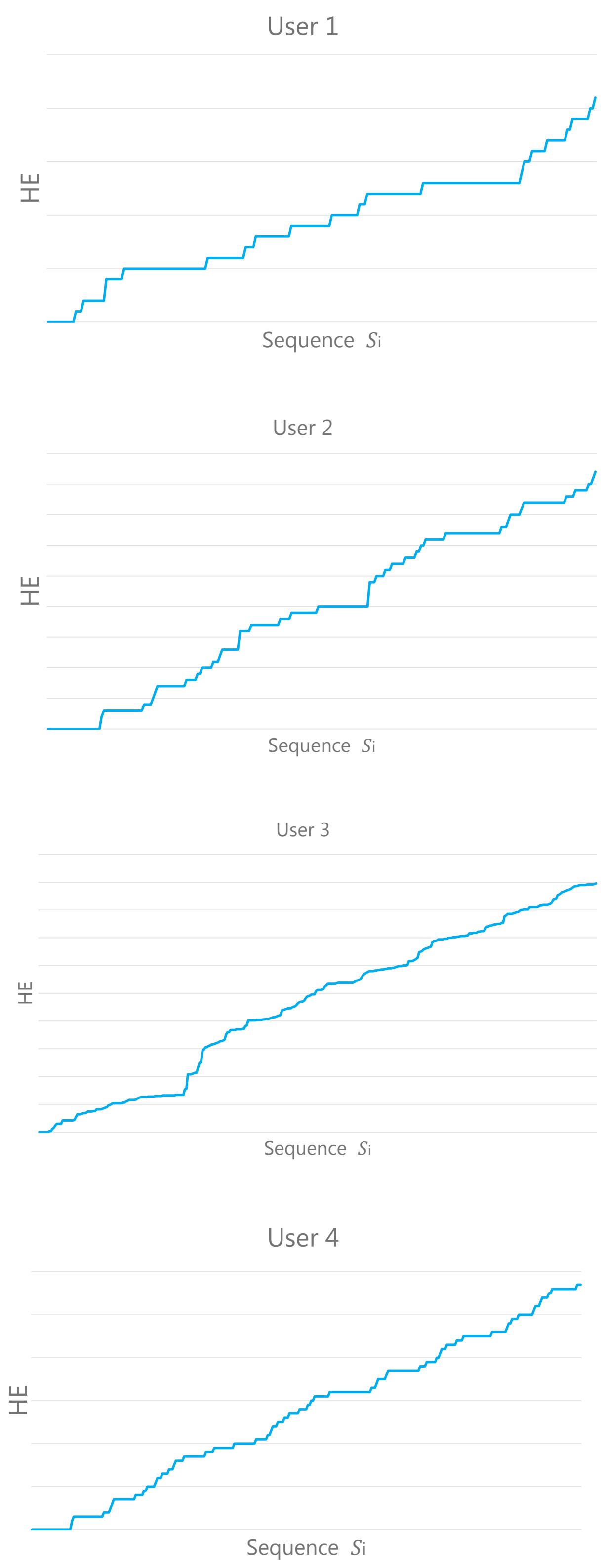


Figure 55: Habit error HE evolution in the four users' data

From Table 11, our algorithm predicts the next activities of user 1 (with very recurrent habits) with a precision of 83 % and a global error GE of 17% divided into 11% of learning error LE and 6% of habit error. User 2 and user 3 show less precision rate with respectively 71% and 61% of precision. User 4 data that contains 726 sequences and a total of 2831 POIs (classes) shows a precision of 68% and a global error of 32%.

Discussion

First, it is clear that our approach shows an interesting result with an average precision of 70.75%. Moreover, by analyzing the distribution of errors in every user's data, we notice a correlation between the global error GE and the habit error HE. Indeed, the variation of learning error LE is so small that we preclude the possibility of linking between GE and LE.

We conclude that the error in our approach is sensitive to the users' habits similarity, which is somewhat logical because the definition word "Habit" is a routine of behavior that is repeated regularly, so, regularity in users' movements is important to succeed in predicting his next location.

In Figure 55, we track the evolution of HE in every user's data. Results show globally two kinds of graphics: stepped graphic concerning user 1 and user 2 owning to the fact that those two users have some new behaviors, when arrived, the algorithm makes a mistake in that moment because the new sequence is unknown, but it catches up quickly in the next moments to recognize the sequence as frequent and predict the right activity. The second type of graphic concerning user 3 and user 4 is a moderately smoothed graphic which approaches a straight function, the user 3 and user 4 have dispersed habits which explain the continuous increase of HE over time.

5.3.4. Incremental activity prediction with concept drift experiment

In this section, we compare our approach with an incremental version of FP-growth called DB-Tree [3] and CVFDT (Concept-Adapting Very Fast Decision Tree) [23], an extension of VFDT algorithm that handles the concept drift.

The ideal dataset to experiment the three algorithms is a dataset where the user has made a relocation (change of address and probably of habits) using a dataset that contains a concept drift. For that, we paired two users' dataset from Geolife users' datasets into one dataset to say that the first user changes his address and his habits to the second user's address and habits, the new dataset contained 389 POIs.

Table 12: Comparison between our algorithm, DB-tree and CVFDT

	Our algorithm	DB-tree	CVFDT
Precision before the relocation	72 %	63 %	60%
Precision after the relocation	69 %	51 %	60.5 %
Global precision	70.5 %	57 %	60.25 %

Table 12 presents the results of our experiment; we divided it into three indicators, precision before and after the relocation, and the global precision.

Discussion

Globally, our algorithm shows an interesting precision result with 70.5 % of precision contrary to DB-tree and CVFDT which show a low rate with respectively 57% and 61.5% of precision.

To analyze the result presented in Table 12, we tracked the error evolution of the three algorithms (see Figure 56, Figure 57, Figure 58).

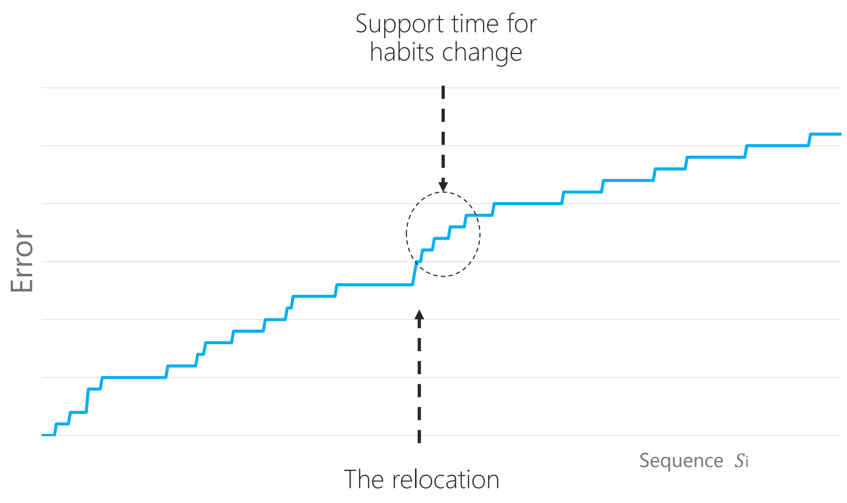


Figure 56: Error evolution in our algorithm

After the relocation, our algorithm shows strength to these shifts (precision decreases only from 72% to 69%) and support time for habits change is small because the algorithm detects a change in the user's habits and starts to add a supplement weight (drift's weight wd_j) until that the new sequences become frequents.

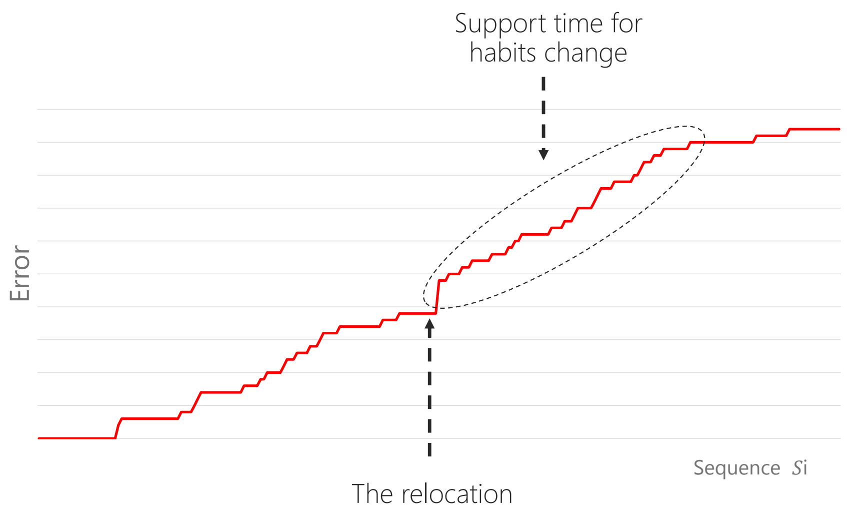


Figure 57: Error evolution in DB-tree algorithm

Contrariwise, in Figure 57, DB-tree encounters difficulties to revive its model after the relocation to detect the new behaviors (see the drop of the precision from 63% to 51% in Table 12). Indeed, as DB-tree traits all the sequences with the same manner (adds 1 to the concerned nods in the tree), it will take much time to the new habits to become frequents, which explains the important support time for habits change in Figure 57.

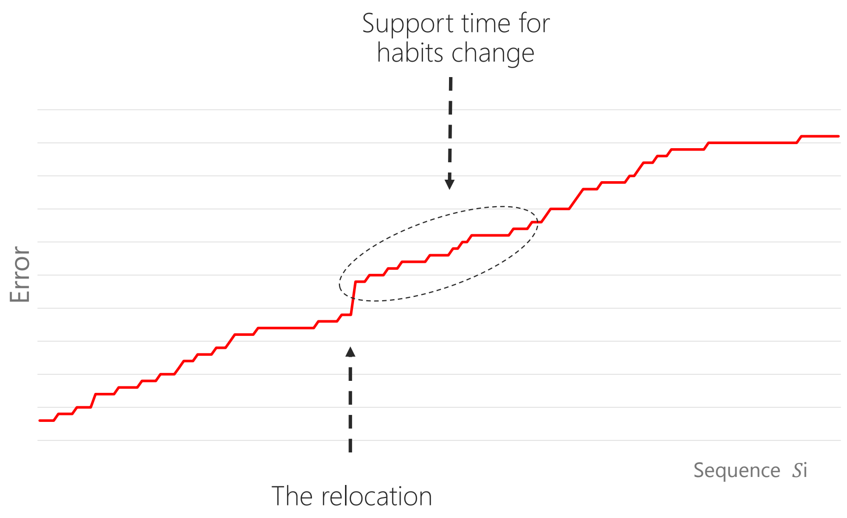


Figure 58: Error evolution in CVFDT algorithm

In Table 12 CVFDT behaves better than DB-tree, but its global accuracy still considered as low (60.25 %), back to the fact that theoretically CVFDT needs a massive set of examples to start improving its accuracy. The literature has mentioned a threshold of 100k examples [169].

Despite the overall low accuracy, CVFDT seems unaffected by the change of habits. The recorded support time for habits change is smaller than DB-tree's one (Figure 58). This is justified by the fact that when CFDT detects a concept drift, it starts to build an alternate sub-trees using the new habits. These alternate sub-trees will replace the original ones when the error in the new sub-tree is less than the original error. The time needed to do this substitution is represented by our variable called support time for habits change.

After the experimentation of our approach in terms of precision and support of the concept drift, we are going to test in the next section the computational impact of the algorithm on the mobile resources.

5.3.5. EXPERIMENTATION OF MOBILE RESOURCES USAGE

This work can be used in any environment (mobile, desktop or Web applications) and using any architecture (local or distributed design). In spite of that, we are going to test our solution in a mobile environment, principally for these reasons: (i) users movements are usually collected incrementally using a mobile device, so, it is more consistent to continue predicting incrementally the users' movements on the same device. (ii) Mobile environment requires careful handling of the reduced storage and computing capacities. If we prove that our solution is optimal for the mobile environment, it is clear that it will be useful for the other environments that have fewer requirements.

We tested our algorithm using an Android smartphone from Sony (Sony Xperia S) with 1 GB of Ram and 1.5 GHz dual-core processor.

The first test concerns the RAM usage, we added our solution to our precedent work presented in Chapter 3 where we recognized incrementally users' activities. Then we compared the set with a well-known GIS solutions "Waze Social GPS Maps & Traffic", one of the best free navigation applications that won the best overall mobile app award at the 2013 Mobile World Congress. The reason for such selection is that Waze has a lot in common with our approach. In fact it gathers complementary map data and traffic information from its users like police traps (can be seen as a POI in our case), and learns from users' driving times to provide routing and real-time traffic updates.

Results in Table 13 that represent the average consumption of mobile's memory of every application for 12 hours show that our solution is not greedy regarding memory usage with 40 Mo of RAM usage compared to Waze with 67 Mo.

Table 13: Comparing our solution to Waze application in terms of memory usage

	Our solution	Waze
Memory usage	40 Mo	67 Mo

The second test concerns the storage capacity usage. We had to compare our solution with two algorithms: (i) DB-tree that uses the same tree structure as us, but that stores in descending order of support all items in the database; (ii) Apriori algorithm [105] that uses a traditional database structure to observe the impact of using a tree structure. Apriori is a widely used algorithm for association rules learning that uses a standard database design. We tracked the variation of the database size in every solution in function of the number of sequences arrived (from 1 to 1 million sequences). In order to get such an important number of sequences, we created an algorithm that generated random sequences containing between 2 and 20 POIs using 500 different POIs. Results exposed in Figure 59 show how much the use of tree structure is benefic to the size of the database, thanks to sharing paths between items in the tree structure, our database had much smaller size (367 Mo) than in a traditional database (1200 Mo).

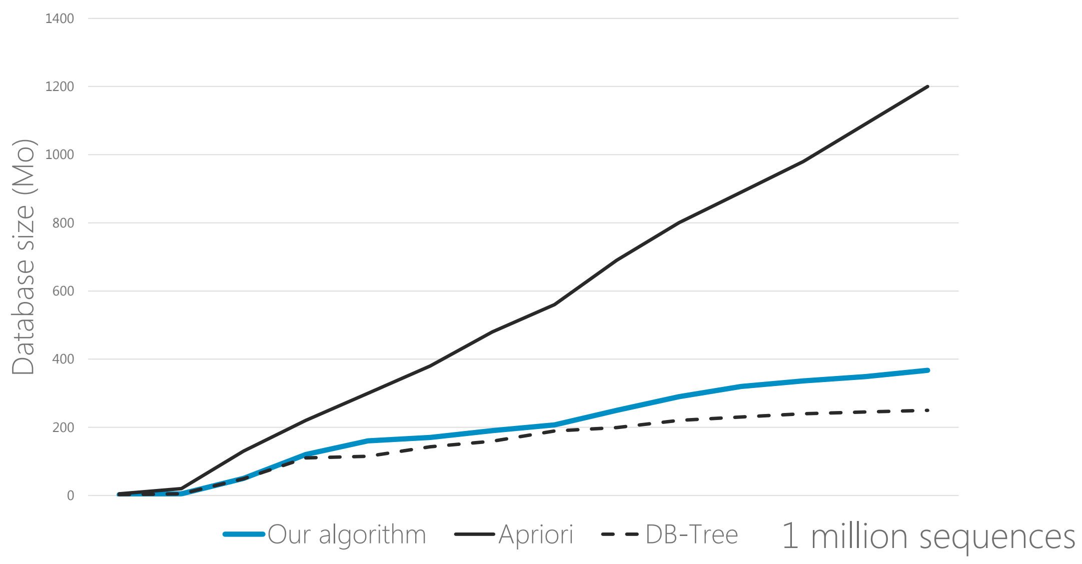


Figure 59: Database size comparison between our solution and Apriori algorithm

In the other hand, the maximum size of our tree (367 Mo) that is reached using one million sequences represents a size widely acceptable by the requirements of mobile environment storage. Note that if we take an average of 2 sequences per day, 1 million sequences represents more than 1388 years, for one user, it is clear that this size in unreachable.

DB-tree had a slightly smaller database (250 Mo for one million sequences) than our algorithm, this is justified by the fact that DB-tree does not take into consideration the order of items, which means that “home, work, restaurant” and “restaurant, work, home” will be stored in the same branches. However, this technique can’t be used when analyzing users’ habits, because the order of habits is a very important parameter. Otherwise, it will lead to gross errors.

Though, when comparing our algorithm to DB-tree in the real world, the difference will be neglected since the number of sequences will be much lower, for instance, from the experiment presented in Figure 59, the average database size for our algorithm will be 260 Ko/year, for DB-Tree it will be 180 Ko/year. If the two algorithms will continue running for 10 years, the databases size will be 2.6 Mo and 1.8 Mo respectively, the difference is too small to be considered.

5.4. CHAPTER CONCLUSION

In this chapter, we proposed a new algorithm based on the online learning of users’ habits to predict users’ next locations taking into account the changes that can occur in their routines. Our original contribution includes a new algorithm of online mining association rules that support the concept drift.

Our approach has been experimented in real case studies using both synthetic and real data retrieved from Geolife project to test the accuracy of our predicting technique. We compared our solution to a set of algorithms like Apriori, CVFDT and FP-growth algorithms via several indicators like supporting users’ habit changes and mobile resource usage. Results show that our proposal is well positioned compared to its similar, and represents an interesting solution to predict users' next activities without depleting the resources of users' mobile devices.

Several promising directions for future works exist. First, if this work is used in a big data context, some efforts shall be done to optimize the construction and the research process in the tree structure to minimize the response time of our algorithm. Secondly, the clustering of users' profiles represents an interesting research field. In that direction, the habits' tree represents a good structure that summarizes users' routines, clustering users' profiles basing on their habits will be reduced to the comparison of two trees (habits' trees). Thirdly, this work has to be improved by introducing a temporal dimension to the habits' tree in order to improve our algorithm precision, for example, routines made at weekends are different of those made in working days.

CHAPTER 6

HYBRID ACTIVITY RECOGNITION

6.1. CHAPTER INTRODUCTION

Activity recognition applications that extract users visited places are supposed to operate 24 hours a day, 7 days a week. Searching for users visited places at every moment like done in the majority of related works leads to excessive power consumption that drains mobile batteries rapidly. We propose a novel approach that minimizes power consumption by reducing the activity calculation. The proposed algorithm learns users' habits and chooses an appropriate time to search for their performed activities. For instance, suppose that the user has the habit of going from home to work every morning. Theoretically, there is no need to process the user movements every time he goes from home to work since it represents useless calculations.

Moreover, nearly all outdoor activity recognition approaches use a fixed activity's minimum duration threshold that represents the minimum time that the user has to spend in the POI (place of interest) to be declared as a visited place. This threshold prevents false activity detection like 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. On the other hand, setting it to a high value will miss detect some activities like buying cigarettes at the convenience store.

Moreover, human behaviors differ according to a lot of factors like age, sex, health status, etc. Consequently, generalizing these solutions to cover a multitude of user profiles may become hopeless when fixing these thresholds. Consequently, we are, to the best of our knowledge, the first to propose not only a dynamic approach to learn the activity's minimum duration threshold automatically, but to propose a specific threshold for each POI too. Our approach will assign to each POI a minimum duration threshold to be able to detect both the short and the long activities.

In this chapter, we will present an innovative hybrid battery-friendly method that aggregates the activity recognition model presented in the Chapter 3 and the prediction model presented in the Chapter 5 in order to recognize users' location activities accurately without draining the battery of their phones. The proposed method succeed in detecting incrementally users' visited places without any previously fixed threshold. We will also prove that our proposal reduces outstandingly the battery consumption while keeping the same high accuracy rate.

The following sections detail our contribution: Section 2 presents our approach in terms of three major components: activity recognition, prediction and verification; Section 3 describes the experimentations by highlighting two dimensions: accuracy and power saving. Finally, the conclusion is summarized in Section 4.

6.2. LEARNING HUMAN HABITS USING MOBILE PHONES

We will analyze incrementally users' mobility to extract their performed activities. The main idea of our approach is to minimize the calculation during the analyze process by using a mixture of activity recognition and prediction algorithms.

Our system is divided into three parts; the first part aims to recognize users' activities when they visit places for the first time; the second part is activity prediction where we predict the next activity to avoid processing already recognized activities; and finally, activity verification which is a post-processing step that aims to verify if the predicted activity is the right one. Suppose that the user has gone from home to work (see Figure 60). For the first time when the user visits these locations, we will recognize the two POIs linked to the activities staying at home and working in the office using our activity recognition model presented later.

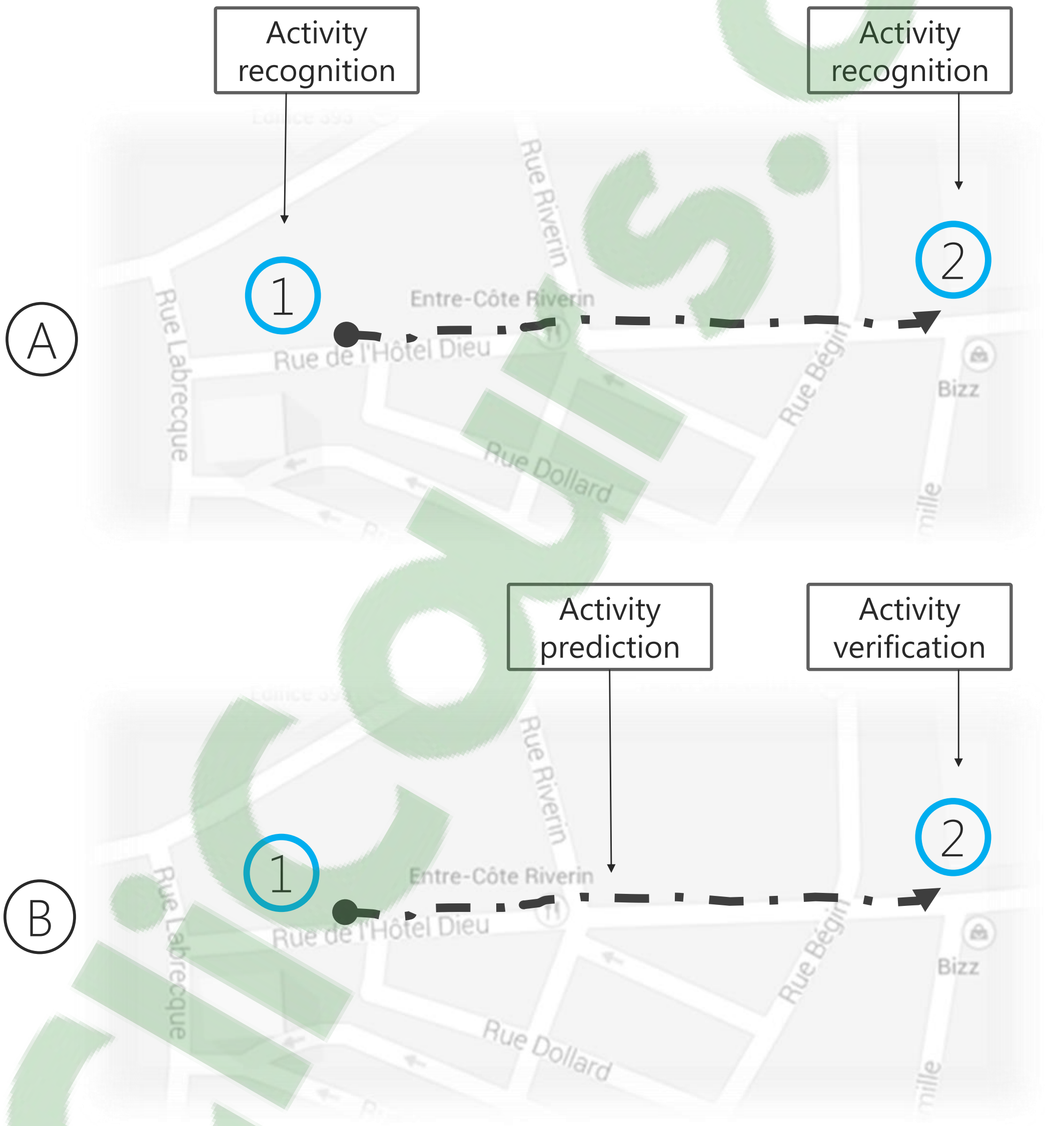


Figure 60: The three parts of our hybrid approach. (A) Home (B) Work.

The next time that the user will go from Home to Work, our approach will not use the activity recognition model since it represents a significant source of power consumption. Nevertheless, it will use only the association rules driven from the prediction model to estimate the next destination. Meanwhile, all the position points between Home and Work will be stored without any processing until we confirm that the predicted activity is that one performed by the user. This verification is achieved by using the verification model (see (B) in Figure 60).

If we confirm that the predicted activity is that one performed by the user we delete the recorded position points between Home and Work because the prediction was made successfully. Otherwise, we reapply the activity recognition model for the whole recorded points to figure out where has the user gone from Home.

After presenting the general idea of our approach, we are going to detail in the following, the three parts of our system.

6.2.1. ACTIVITY RECOGNITION

When a new POI is detected, we execute the activity recognition model presented in Chapter 3. After that at the end of the day, the daily users visited places are passed to the activity prediction model in order to learn the user habit.

6.2.2. ACTIVITY PREDICTION

Remember that the prediction model presented in Chapter 5 starts by constructing a sequence of POIs that represents the daily users visited places. Each POI in this sequence is obtained using the activity recognition model presented in Chapter 3.

6.2.3. ACTIVITY VERIFICATION

After predicting the next activity, we need to confirm that the predicted activity is that one performed by the user. For this purpose, we introduce a new structure of POIs. A

POI is no longer considered as a static entity where the user carry out an activity, but a geographic entity that is characterized by a minimum duration d_{min} that represents the minimum duration of an activity, and a distance r that represents a ray where the activity can be performed. Unlike the related works, we will learn these parameters by adjusting them incrementally and dynamically according to a user's behaviors (see Figure 61).

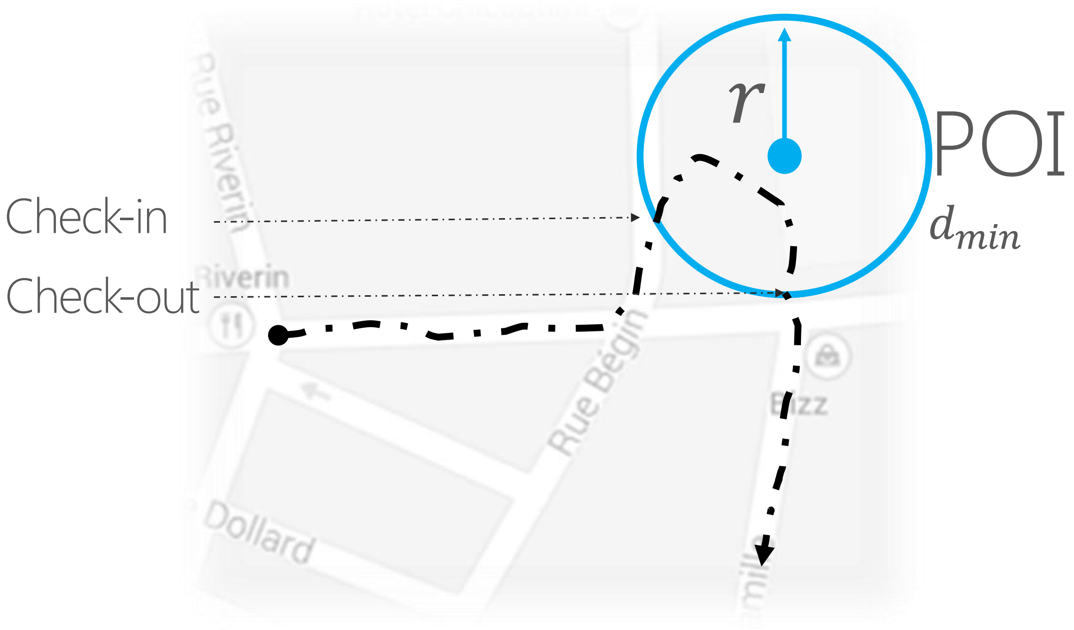


Figure 61: The d_{min} and *r* characteristics of a POI

Despite that in the real world a POI can contain several activities, related works have linked a POI to only one activity, like assigning a mall to shopping, even if it can contain a multitude of activities like going to a restaurant and cinema. Thus, our approach handles the POIs as geographic areas that may contain several activities. Each activity is characterized by temporal edges learned from the user behaviors.

6.2.3.1. Calculating d_{min}

Each duration represents a time spent by the user inside or at the surroundings of a given POI. It is calculated using the time of check-in and check-out (see Figure 61).

In order to calculate the minimum duration threshold d_{min}, we need to understand how the user behaves inside this POI. As said previously, a user may perform more than one activity at the same place, the d_{min} calculation starts by regrouping the set of duration using Fuzzy C-Means (FCM) [170]. The reason is that this algorithm allows a time duration to belong to more than one cluster which solves the problem of values on the borderline (see Figure 62).

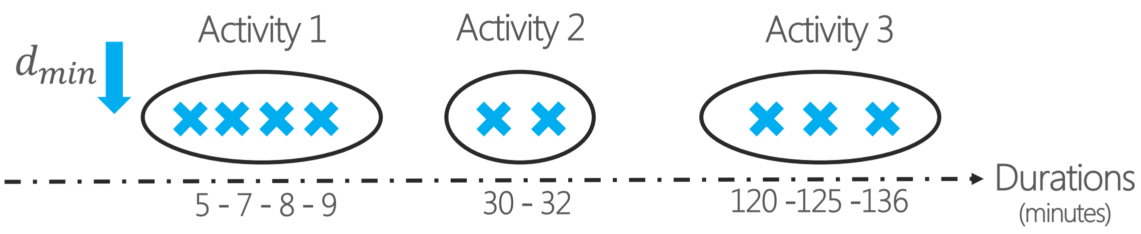


Figure 62: Activities' duration clustering inside a POI

However, FCM requires a fixed number of clusters, in our case we don't have a prior information about the number of activities inside this POI. So, in Equation (31) we propose a criterion to calculate incrementally the optimal number of clusters C_N; the average deviation of each value from the median M of its most probable cluster. When more than one cluster is analyzed, the criterion value is the sum of each cluster average.

$$C_N = \sum_{i=1}^{N} \frac{\sum_{j=1}^{n_i} |x_{ij} - M_i|}{n_i} \tag{31}$$

N represents the number of clusters, x_{ij} is the duration of the activity j in the cluster i, n_i is the number of activities in the cluster i and M_i is the median of the cluster i.

The algorithm considers that the optimal clusters number is N if the N+1 clusters' criterion value doesn't improve significantly the one with N clusters. We judge that a gain in C is significant or no if it exceeds a gain threshold Cth_{N+1}, see Equation (32). This threshold represents the quotient of the last gain on the number of clusters, see Equation (33).

$$C_N - C_{N+1} > Cth_{N+1} \tag{32}$$

$$Cth_{N+1} = \frac{|C_N - C_{N-1}|}{N+1} \tag{33}$$

Let the durations in Figure 62 be an example to illustrate this clustering step. Suppose that we initially have one cluster $\{5,7,8,9,30,32,120,125,136\}$. Its median will be 30. Given Equation (31), the criterion will be: $C_1 = 42{,}6$.

For two clusters, FCM will construct two clusters {5,7,8,9,30,32} and {120,125,136}, thus, $C_2 = 8{,}5 + 5{,}3 = 13.8$, we note that $C_2 < C_1$. In order to know if this gain is significant or not, we calculate Cth_2 by using Equation (33) as follows:

$$Cth_2 = \frac{|C_1 - C_0|}{2} = \frac{|0 - 42{,}6|}{2} = 21.3$$

Using the condition in equation (32), we note that $C_1 - C_2 > Cth_2$, so, the new criterion C_2 brings a significant gain. Consequently, we increase the number of clusters to two.

For three clusters, FCM will construct the following clusters: {5,7,8,9} {30,32} and {120,125,136}, thus, $C_3 = 1{,}25 + 1 + 5{,}3 = 7.55$, we note that choosing three clusters improved the criterion C as $C_3 < C_2$ and $C_2 - C_3 > Cth_3$.

In order to know if we stop at three clusters, we have to test the criterion of four clusters. Thus, for N=4, FCM will construct {5} {7,8,9} {30,32} and {120,125,136}. Consequently, $C_4 = 0 + 0.7 + 0.7 + 5{,}3 = 6.7$. We note that $C_4 < C_3$ but this gain is not significant as the condition in equation (32) is not verified: $Cth_4 = \frac{|13{,}8 - 7{,}55|}{4} = 1.56$ and $C_3 - C_4 \ngtr Cth_4$. The gain in the criterion C is too small for N=4. So, there is no need to add another cluster, at this moment, the algorithm stops the clustering and declares that the number of clusters is $N = 3$ (see Figure 62).

After clustering the durations inside the POI, it's time to calculate the value of d_{min}. Remember that our solution is online, which means that d_{min} can change at every new visit to this POI. So, initially d_{min} takes the value of the smallest value of the first cluster, for instance, d_{min} in Figure 62 is 5 minutes. But at the arrival of a new duration, we compare it to d_{min}, if it is higher than d_{min} we declare that the user has visited this POI and we recalculate the criterion C to figure out if we have to add a new cluster or not.

However, if the duration is less than d_{min}, the new duration can be a new d_{min} or an error (the user just passed by the POI without performing an activity there). Accordingly, we calculate the criterion C for the new clusters including the new duration. If C_{new} is significantly higher than C_{old} we keep d_{min} and we conclude that the new duration is an error, otherwise, d_{min} takes the value of the new duration.

For instance, let us take the previous example presented in Figure 62, if the new duration is 1 minute, the new C_3 will be 8.5. We note that new C_3 is higher than the old $C_3 = 7.55$. Consequently, we assume that the user has not visited the POI. However, if the new duration is 4.5 min, the new C_3 will be 7.8 which is not significantly higher than the old $C_3 = 7.55$. Thus, we admit that the user has visited the POI, we accept this duration as a borderline duration and we put $d_{min} = 4.5$.

6.2.3.2. *Calculating r*

The value *r* represents a ray where the user has to spend a minimum duration d_{min} to declare that the user has visited the POI. Contrary to the similar works that use a fixed ray to determine the nature of the user performed activity, our work stands-out by proposing a dynamic approach to calculate *r*. We calculate a specific *r* for each POI basing on the popularity of this POI, which means that when the POI is popular in the user routines, the radius will be stretched until reaching a max value called r_{max} (see (A) in Figure 63). Basing on these rules, the ray r is calculated as follows:

$$r = supp(POI).r_{max} \tag{34}$$

The value $supp(POI)$ is the support of the concerned POI in the user Habit's tree HT. It is an indication of how frequently the item-set appears in the user habits (see Figure 51 in 5.2.2 Habits' tree update), it is calculated as follows:

$$supp(POI) = numbre\ of\ occurence\ of\ POI\ in\ HT/Total\ number\ of\ sequences \tag{35}$$

For instance, suppose that the result of Equation (35) is $supp(POI) = 0.85$. Given Equation (34), the radius r will be 85% of r_{max}.

As said previously, we calculate a specific r for each POI basing on two parameters: the popularity of this POI calculated by the $supp(POI)$ value and the r_{max} that represents the maximum area that this POI can reach. By definition, the area dedicated to a POI cannot intersect another POI's area. As such, r_{max} will be stretched as long as there is no other POIs in the neighborhoods. In other terms, the maximum ray of our POI r_{max} is calculated in function of the surroundings of the POI. If there is no other POI in the neighborhood, the r_{max} will be stretched to cover a wider area, and vice versa.

Basing on these conclusions, we calculate r_{max} by using Voronoi diagram [171]. It is a diagram that aims to partition a plane with n points (in our case POIs) into convex polygons such that each polygon contains exactly one generating point (POI) and every point in a given polygon is closer to its generating point (POI) than to any other. Thus, r_{max} represents the distance between the POI and his closer Voronoi border (see Voronoi border in Figure 63 B).

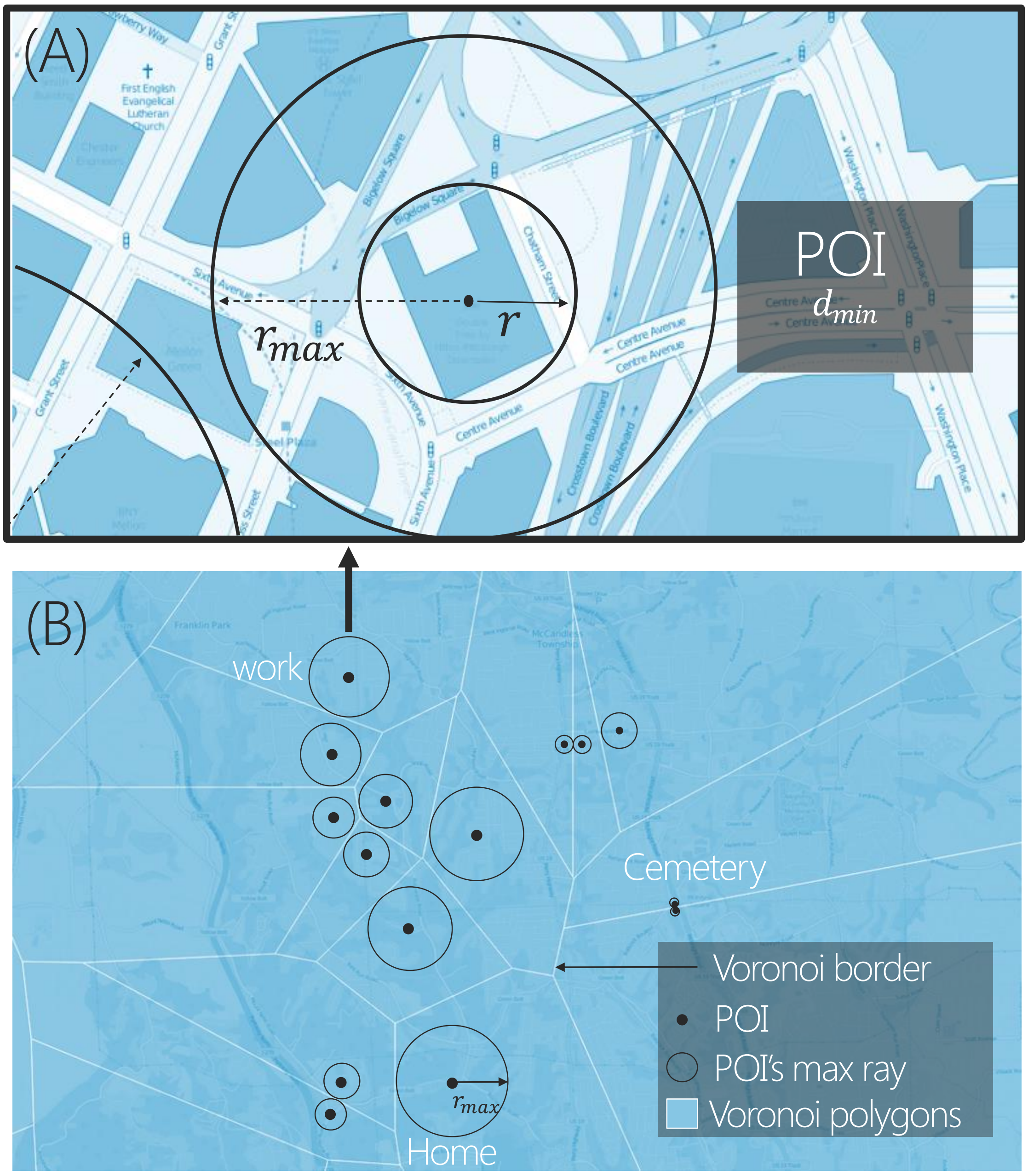


Figure 63: An example of POIs' rays calculation. (A) relation between r and r_{max} (B) calculating r_{max} by using Voronoi diagram.

Unlike the other solutions [89], [95], [96], our approach calculates on every new POI arrival a new r according to the POI's popularity in the user's habits. As such, the most visited places will have wider rays than the others.

The radius is used in the testing step to test if the prediction was right or not. A wider radius let detecting the performed activity rapidly since as soon as the user enters the radius of the POI, we start calculating his dwell time inside this area, when it exceeds d_{min} we declare that the user has truly visited this place (this will be discussed deeper in the next section). By enlarging r, we aim to speed up the testing process in order to decrease the power consumption.

Let us take the example presented in Figure 63 B, the user’s popular places such as Home and Work have wider radius than the less visited places like Cemeteries. For Home and Work, the algorithm will take a small time to declare that the user has visited these places. This is justified by the fact that as soon as the user approaches these POIs (entering the large POI areas), there is a high probability that he will visit them. Hence, we anticipate that the user will achieve an activity in these places despite the fact that he has not reached them yet. On the opposite, the algorithm will wait until the user reaches physically the unpopular POIs such as the Cemetery to declare that he visits them. The main reason is that as these POIs are unpopular, the probability that the user will visit them is small.

6.2.3.3. *Testing Process*

Remember that the verification process is designed to figure out if our prediction was right and correct it when needed by re-running the activity recognition model. The prediction error can fall under two cases: the user didn’t go to the predicted place and the user has gone to the predicted place but he performed other activities meanwhile. To detect these errors we introduce two types of tests: trajectory duration test and activity duration test.

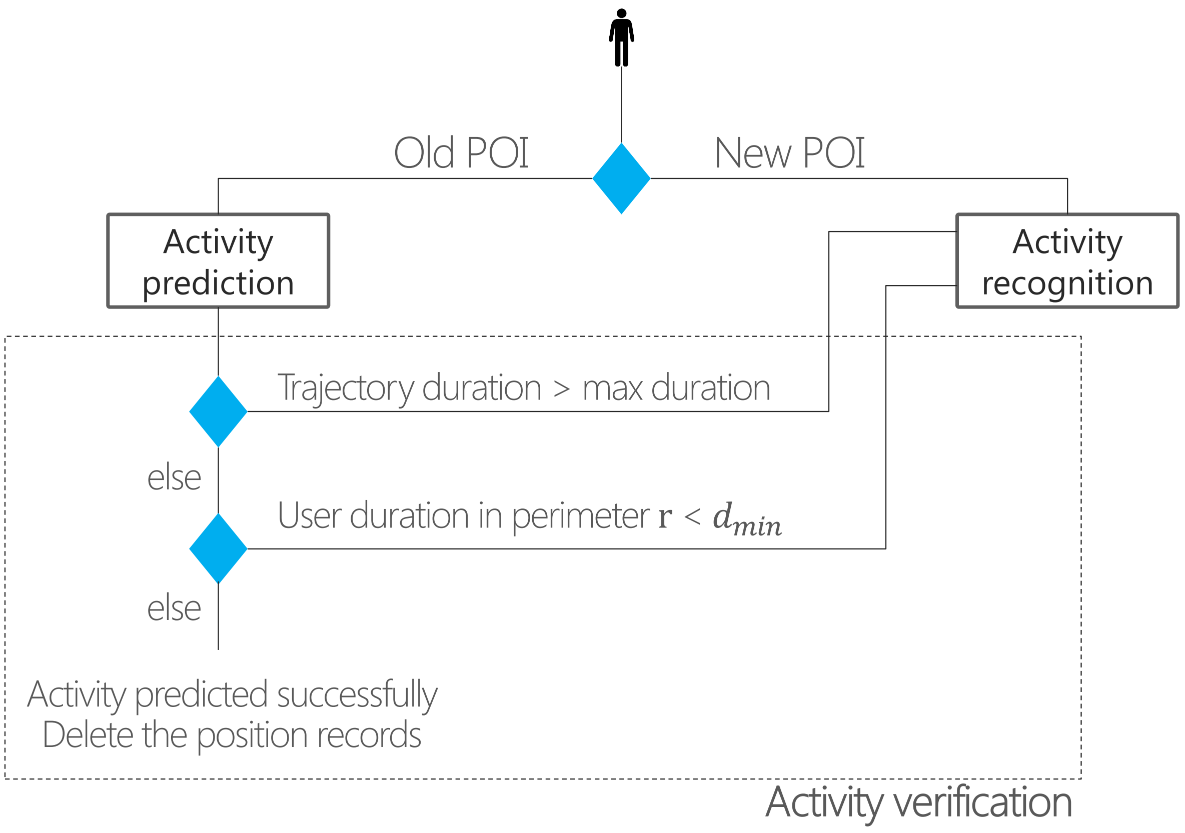

Old POI
New POI
Activity prediction
Activity recognition
Trajectory duration > max duration
else
User duration in perimeter r < d_{min}
else
Activity predicted successfully
Delete the position records
Activity verification

Figure 64: Testing process in the activity verification step

In Figure 64, we present our testing process for the activity verification step. The first test is to compare the duration of the user trajectory and the trajectory's max duration between the two POIs (the source POI and the predicted POI) to test if the user has gone to the estimated location. If the user's trajectory duration exceeds the max durations we can say that the user has probably gone somewhere else, because he spent more time than usual to reach the predicted POI. Consequently, we reapply the activity recognition model because our prediction was probably wrong.

The second test compares the duration of the activity. If the duration of the user's staying in the POI's perimeter, defined by r, is less than d_{min} we can conclude that the user just passed by the area.

Algorithm 11: The overall algorithm of our hybrid approach

```
Input:  user positions;
Output: the activity of the person;
1:  If (HT contains current POI)
2:      Next POI = Prediction POI
3:      Wait until position inside POI
4:      Activity verification (next POI )
5:  Else
6:      Activity recognition until next POI
7:  End
8:      Update HT
9:      Cluster durations inside POI
10:     Calculate $d_{min}$
11:     Delete position records
```

Algorithm 12: Activity verification (POI)

```
Input: a POI;
1:  If ( trajectory duration < Max trajectory durations)
2:      If dwell time in $r$ $< d_{min}$
3:      Calculate the new criterion $C$
4:          If (new $C \gg$ old $C$ )
5:          Activity recognition from the previous POI
6:          Else
7:          $d_{min}$ = duration in $r$
8:          End
9:      Else
10:         //Predicted activity is true
11:     End
12: Else
13:     Activity recognition from the previous POI
14: End
```

6.2.3.4. *Examples of Activity Verification*

To illustrate how our approach operates, we will give the following example of one user's mobility during four days. Suppose that user's habits include going to four destinations like presented in Figure 65 and his daily routines as described in Table 14.

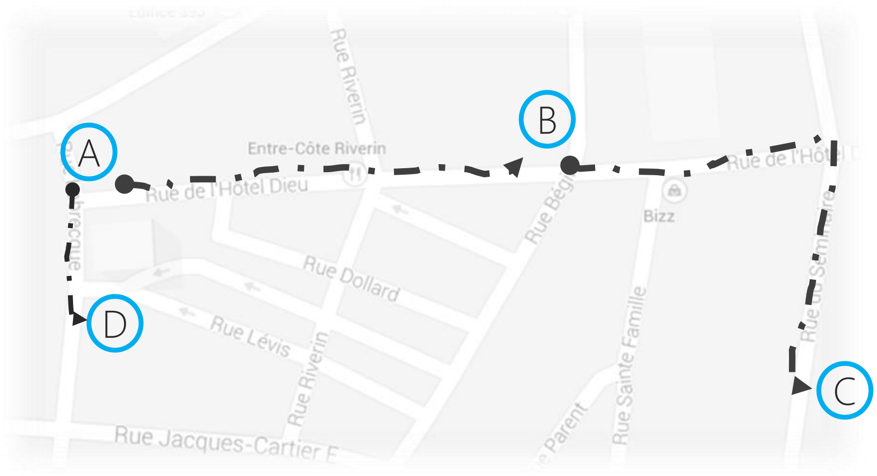


Figure 65: An example of a user mobility

Table 14: A user mobility description for four days.

Day	Source	Destination	Trip duration (minutes)	Stay Duration (minutes)
Day 1	A	B	21	20
Day 2	A	B	23	30
	B	C	12	10
Day 3	A	D	11	4
Day 4	A	C	41	23

In day one, the user has gone from A to B, at this moment we don't have any prior information about the user's habits, so, our algorithm starts by applying the activity recognition model for the whole position records between A and B. When done, the algorithm updates the user's habit tree HT to add the sequence (A-B) and records the duration in the activity verification model in order to calculate the d_{min}.

In day two, the user has gone from A to B and from B to C. So, when he goes out from A, the algorithm checks in the prediction model if there are any predictions starting from A, the model finds B as next destination. Consequently, the algorithm records the position points starting from A until figuring out if the user has really gone to B. After 23 minutes, the algorithm detects that the user has entered in the perimeter r of the POI B, so, it starts calculating the user's duration of stay in B. The verification model compares the duration of stay = 30 min with d_{min} = 20 min. Since the new duration > d_{min}, the algorithm declares that the user has really visited the POI B, so, it deletes the position records between A and B and updates the habit's tree HT. For the segment B to C, the algorithm achieves the same process as day one, it puts $d_{min} = 10$ for the POI C and updates HT, HT will contain A-B-C with the number of occurrences 2-2-1 respectively.

In day three, the user goes from A to D. When he goes out from A, the algorithm checks HT and predicts B as next destination, after that it uses the prediction model to check for the veracity of this prediction. At this step, even if the algorithm does not use any recognition model, but it continues to record the position points. After 23 minutes of recording, which represents the maximum trip duration between A and B, the algorithm notes that the user has not yet entered the B area. Consequently, it reapply the activity recognition model for the whole position records starting from A, and figures out that the user has gone to the place D. Next, the algorithm deletes the position records, sets d_{min} of D to 4 minutes and updates HT to contain the sequences A-B-C and A-D with the number of occurrences 3-2-1 and 3-1 respectively.

In day four, the user has gone from A directly to C. Using the prediction model, the algorithm will predict B as next destination. During the trip, the verification model detects that the user's duration of stay in B is less than d_{min}. Consequently, it understands that the user has just passed the area of B and did not achieve any activity there. So, the algorithm reapply the activity recognition model to find where the user is going. The model finds that the user has gone to C, so it sets d_{min} of C = 41 min, it deletes the position records between A and C and updates HT that will contain the sequences: A-B-C, A-D and A-C with the number of occurrences: 4-2-1, 4-1 and 4-1 respectively.

6.3. EXPERIMENTAL EVALUATION

We validate our proposal by comparing the four following solutions:

- Our hybrid solution
- Our old solution: using only the activity recognition model without any prediction.
- CB-SMoT approach
- LifeMap application

We will re-conduct the same experimentation done in Chapter 3 to test is our hybrid approach improves the battery consumption. As such, we will test the accuracy of our approach using the Family Coordination dataset [131] by comparing it with two solutions. A first solution where we apply only our activity recognition model for every POI. The second solution is CB-SMoT method described in [111]. Then, we will test our approach's ability to save battery life by comparing our solution to LifeMap application described in [135].

6.3.1. FAMILY COORDINATION TEST

We used the family coordination dataset [131] that was already introduced in section 3.3.1. We compared our solution to our old solution where we apply only our activity recognition model during all the experimentation process without using any prediction.

The two versions of our approach were deployed using Huawei P7 android phone. We have installed both solutions, the family coordination dataset and a SpatiaLite database. This geographic database 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 test our approach in terms of accuracy. Consequently we suppose that the phone's battery is full charged during all the processes principally because we don't have access to the battery's life data in the family coordination dataset. The ability to save the phone's battery will be presented in the second part.

Results presented in Table 15 represent the comparison of our approach to our old solution, we have tested 10525 activity 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, 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 using the Equation (36) and Equation (37) respectively.

$$Accuracy = \frac{Correct}{number\ of\ tested\ activities} \tag{36}$$

$$Error = \frac{missed + false}{number\ of\ tested\ activities} \tag{37}$$

Table 15: Comparison of our hybrid approach to our old Activity recognition model

	Our hybrid solution	Activity recognition model
Tested activities	10525	10525
Correct	8052	8313
Missed	1830	1812
False	644	578
Accuracy	76,5%	78%
Error	23,5%	22%

Our hybrid method has shown similar results to our old approach (using the activity recognition model all the time) in terms of accuracy and error rates, 76.5%, 78% and 23.5%, 22% respectively. This is justified by the fact that our hybrid approach has the ability to recognize the errors generated by the predictions, and to correct them by reapplying the activity recognition model. Consequently, the hybrid approach acts like if we have applied the activity recognition model all the time, but with much less power consumption. Indeed, in order to confirm this point, we have tracked the phone's memory usage for each method for 12 hours, and we have added the results of Waze [141] and CB-SMoT [111] presented earlier in section 3.3.1. Results presented in Table 16 represent the average RAM usage of each application.

Table 16: Comparing our hybrid solution to our old solution, CB-SMoT and Waze application in terms of memory usage

	Our Hybrid solution	Our old solution	CB-SMoT	Waze
RAM usage (Mega Octets)	19	35	38	67

Our hybrid solution shows a promising RAM usage rate that is better than the other solutions. It is clear that the hybrid method has proven itself in terms of saving the RAM usage, but, is it the same case for the battery's life? We are going to answer this question by going further in the experimentation of our battery-saving technique in the following test.

6.3.2. LIFE MAP TEST

A this step, we used the LifeMap dataset [135] that we have already introduced in section 3.3.2. We used the LifeMap dataset to test our battery-friendly approach. To do so, 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 that of LifeMap application. The test scenario is the same as described in section 3.3.2. Actually, we added our hybrid approach to the experiment presented in section 3.3.2.2 and we compared our hybrid solution to LifeMap and our old solution.

The total number of recorded hours of battery status in LifeMap dataset is 48900 hours noted from the mobility of 68 persons. However, some of these traces don't reflect a user's mobility in the real world, since some users in LifeMap experiment did not have a repetitive behavior during the experiment (several visits of the same POI). Consequently, we have chosen 5 users that had the most regular mobility to reflect fairly the real world situation.

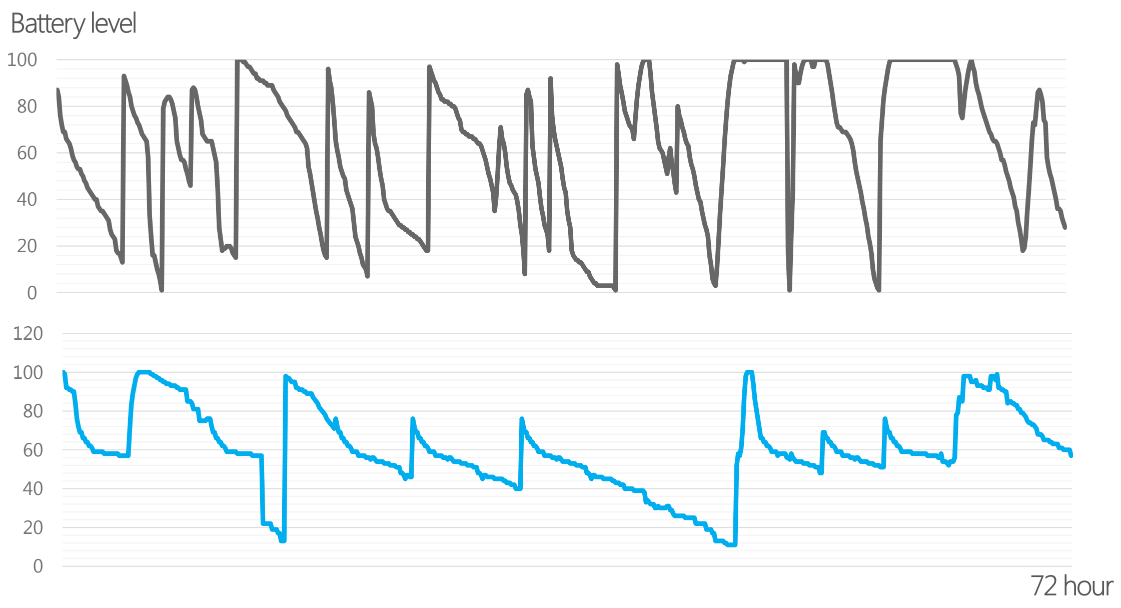


Figure 66: Results comparison between LifeMap (A) and our hybrid solution (B) for 72 hours of activity recognition.

The total number of hours experimented from these 5 users is near 2900 hour. We have used five smartphones to record the power consumption of our hybrid approach for each user and compare it to LifeMap results. Due to insufficient space, we present in Figure 66 the tracking of one user's battery life for 72 hours using LifeMap and our hybrid solution. However, the results derived from the 2900 total hours will be presented in Table 17 and Figure 67.

Our approach shows an interesting battery saving capacity. We notice from Figure 66 that our approach needed only 3 to 4 battery recharges contrariwise LifeMap that needed more than 10 recharges for 72 hours.

However, the number of recharges is not an efficient indicator that quantifies the power consumption, since, like noticed in Figure 66, users tend to recharge partially their phones. Therefore, we have introduced a new indicator to quantify the power consumption called PC. We put $PC = T_r/T_d$, where T_r is the global battery recharging time and T_d is the global battery discharging time. Note that we exclude the time when the battery was full but still under recharge because it can falsify the results presented in Table 17.

Table 17: *PC* and accuracy (calculated using equation (25)) comparisons between our hybrid solution, our activity recognition model and LifeMap application

	Our hybrid solution	Our activity recognition model	LifeMap
PC	9.4%	15.9 %	16.7 %
Accuracy	76.5%	77 %	73%

The PC comparison between our hybrid approach and LifeMap presented in Table 17 confirmed that our solution saves battery life by about 45% while keeping a better accuracy than LifeMap's one (73%) and the same accuracy as if we applied the activity recognition model continuously (77%). This is justified by the fact that our hybrid approach has the ability to recognize the errors generated by the predictions, and to correct them by reapplying the activity recognition model. Consequently, our hybrid approach acts like if we have applied the activity recognition model all the time, but with much less power consumption. To go deeper in the analysis of the PC indicator, we tracked in Figure 67, the average daily PC value of the 5 users for 23 days.

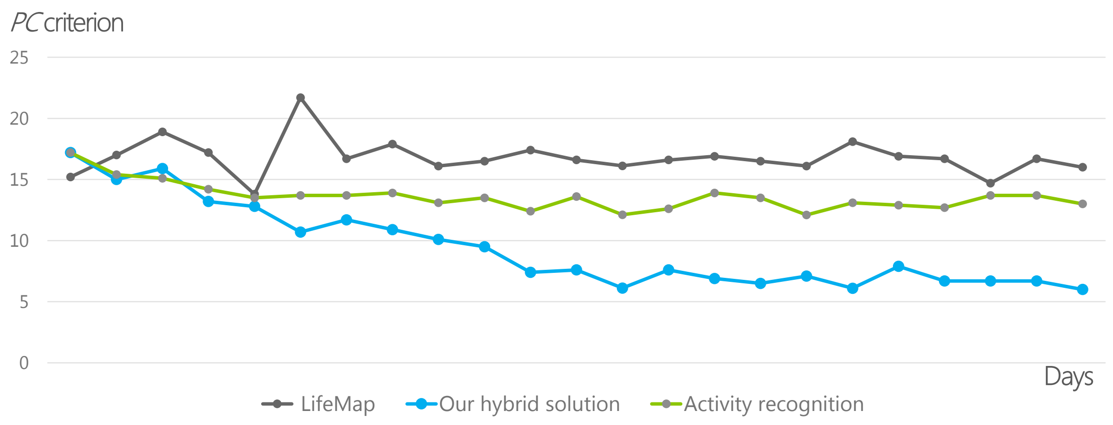


Figure 67: The average daily PC value of the 5 users, comparison between our hybrid solution, our activity recognition model and LifeMap for 23 days.

We notice that the PC value of LifeMap and our old solution are stable contrariwise our hybrid solution that starts from a value of 17% to fill under the 7%. This is justified by the fact that our solution consumes more power in the first times of the experimentation because it applies the activity recognition model frequently in a perspective of learning the user's habits. When done, the approach will consume less power because it will refer each time to the prediction model. We believe that, in the long term, when a user's habits are well learned, the PC value of our approach will be stabilized under 7% , which will lead to about 60 % of power saving.

6.4. CHAPTER CONCLUSION

In this chapter, we proposed an improvement of the model presented in Chapter 3. We presented a new battery-saving technique for extracting incrementally important geographical locations from users' movements. We learned users' habits to reduce the computational complexity of our approach, the proposed approach is divided into three parts; the first part aims to recognize users' activities when they visit places for the first time, the second part is activity prediction where we predict the next activity to avoid processing already recognized activities, and finally, activity verification which is a post-processing step that aims to verify if the predicted activity is the right one.

Our approach has been deeply experimented using two real datasets to test the accuracy and the ability to save batteries of our approach. These tests demonstrate that our proposal is capable of reducing the battery consumption up to 60% while maintaining the same accuracy as the similar solutions. This solution constitutes a promising technique capable of online recognizing a person’s activities without depleting the phone resources. We believe that this approach will represent a strong support for several mobile location-aware applications, in several fields, ranging from traffic management to advertisement and social studies.

PART IV

CONCLUSION AND APPENDIX

CHAPTER 7

GENERAL CONCLUSION

This thesis research project presented in the six previous chapters has proposed original solutions to the challenges arising from using smartphones in the field of outdoor activity recognition. We have seen how ubiquitous and powerful smartphones have become in the last decades. We have seen that using them to recognize users outdoor activities allowed to take a step forward for the activity recognition field, principally, because smartphones are everywhere while being able to see, hear and sense their environment. However, the limited battery life of mobile phones represents a major obstacle for context detection, because the embedded sensors in mobile phones represent a significant source 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. Unfortunately, the limited battery capacity of mobile phones has not had the full intention of the research community. The majority of related works, as seen in Chapter 2, are basing their efficiency factors on the accuracy of their models while neglecting the model impact on the phone resources. In order to report a method as most suitable for activity recognition on mobile phones, it is important to consider the trade-off between accuracy and resource consumption. As such, we focused this thesis on filling the gap between the accurate activity recognition service and the wise use of the mobile resources.

To address these problems and the limits of related works, we have proposed, through this thesis an online battery-aware activity recognition system that let extracting semantically and incrementally important geographical locations from users' moves and switch them to meaningful human activities. Our approach operates entirely on mobile phones without using the Internet or predefined geographic databases.

Our approach has been deeply experimented using a range of real datasets to test the accuracy and the ability to save batteries of approach. These tests demonstrate that our proposal is capable of reducing the battery consumption up to 60% while maintaining the same accuracy as the similar solutions. This solution constitutes a promising technique capable of online recognizing a person's activities without depleting the phone resources. We believe that this approach will represent a strong support for several mobile location-aware applications, in several fields, ranging from traffic management to social studies and healthcare. Applications that operates 24 hours a day, 7 days a week. Applications in which the battery consumption matters.

7.1. REALIZATION OF THE OBJECTIVES

The first phase aimed to gain knowledge of the targeted area of research by conducting a review of the literature on the problem of activity recognition in general. The first part has allowed having an overview of the field of online activity recognition, particularly in an applicative context of mobile environment. In Chapter 2, we reviewed the most important approaches regarding human outdoor activity recognition. We have discussed the advantages and disadvantages of each model in the optic of discovering what would be needed for this thesis project. We brought some common understandings concerning the principal notions of outdoor activity recognition like trajectories, semantic enrichment of trajectories, sensing techniques, inference techniques...etc. We have divided the related works into two types; location systems that determine the user's activity in terms of location and motional systems that recognize activities in terms of motion state. As our work fits more in the category of location systems, we have developed this section into two types; desktop systems that require the use of computers for the heaviest tasks like the classification, and mobile systems that try to achieve all the recognition processes in the mobile phone. We have also explored the existing works that predict users' next destinations from their historic patterns, and we have highlighted some important battery-aware works since we aim proposing battery-friendly solutions throughout all the models that we will develop either for the recognition or the predictive systems.

The second phase consisted of elaborating a complete solution of activity recognition based on smartphones. This part was elaborated in the form of four layers as described above. The first layer included an online activity recognition model that recognizes users visited places incrementally and without using the Internet, a model that is aware of the user behavior and the limited phone resources. This layer was presented in Chapter 3. The second layer was to adapt the first layer to smart cities by adding a supplementary spatial exploration model that is designed to operate in future cities, this layer is presented in Chapter 4. In the third layer presented in Chapter 5, we propose an online algorithm that learns users' habits and predict their next activities carrying the change that can occur in their habits. The last layer presented in Chapter 6 consolidates all the previous layers to propose a hybrid system, a system that switches between Activity recognition and prediction in order to reduce the phone battery consumption efficiently.

The third and fourth phases are dedicated to the implementation and the validation of our approach. Both of them are divided on each of the four previous chapters. The validation of the four layers was presented in sections 3.3, 4.4, 0 and 6.3 respectively. These tests demonstrate that our proposal is capable of inferring users' outdoor activities while modulating the utilization of their mobile resources. Furthermore, our approach reduces the battery consumption up to 60% while maintaining the same high accuracy. Among other things, this final phase of the research allowed us to identify the strengths and weaknesses of each part and of the overall online solution. In addition, we were able to identify interesting improvement for the future that we will discuss later in this chapter.

7.2. REVIEW OF THE DEVELOPED MODEL

The model of activity recognition described in this thesis proposes several interesting innovations in relation to the scientific literature. First, the majority of existing solutions are based on supervised learning where they require a training dataset to previously train their model. However, as users' profiles may differ in several ways, the training dataset may become unrepresentative, thus, not effective in the real world. Moreover, supervised algorithms are facing a major problem linked to the new situations that don’t exist in the training dataset. Our approach it is completely unsupervised, that is to say that we don’t need any centralized training data source. Our approach is ready to use without any background knowledge of the human behavior, it can operate on the first moment of deployment on the user phone.

The majority of related works are based on the classification of historical records of people’s trajectories using non-incremental data mining algorithms. These methods fail in their ability to instantly detect the user performed activities. As such, they cannot be used to fields such as assistance in which we need a real-time access to people’s state. Research on offline activity recognition has been reviewed in several earlier studies in detail. However, work done on online activity recognition is still in its infancy and is yet to be reviewed. Our work is completely online that let detecting incrementally users’ activities without the massive use of historical records. This point let preserving the phone resources.

The third contribution continued in the direction of online learning where we proposed a new version of online K-means that is designed for streaming services on mobile phones (presented in section 3.2.1). We proposed a novel self-adaptive clustering approach that adjusts the computational complexity of the algorithm according to 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 proposes a temporal data window characterized by a variable size in function of a person's travel behavior and his phones' remaining resources.

Our fourth contribution concerns the type of the recognized activities. As described in Chapter 2, a big part of related works has focused recognizing users' outdoor activities on detecting stops in their trajectories. Unfortunately, these works fail to detect activities that needs a movement to be executed such as shopping and running in a park. We demonstrate that the novel approach that we propose, succeeds to recognize both stationary and activities with movements.

As presented in Chapter 4, the outdoor activity recognition systems need a spatial analysis of background geographic data in other to obtain a semantic information about the performed activity. However, in future cities (smart cities), the geographic data can change frequently. As such, we proposed in the same chapter a new spatial exploration technique in smart cities based on a distributed users' collaboration. This technique aims to discard using the predefined geographic databases that can be very expensive to get and very difficult to update in smart cities, future cities where information can change every hour and every day. The proposed service called NomaBlue is disconnected from the Internet, it can operate in any indoor/outdoor area and it doesn't require any pre-defined geographic databases. This system is used for this thesis to feed our activity recognition model in terms of cities geographic data. However, it can be used in several other fields such as: geospatial data collection, marketing and navigation.

After recognizing users' activities in smart cities, we proposed an incremental approach to predict their next activities. Our sixth contribution includes a new algorithm for online prediction of users' next visited locations that not only learns incrementally the users' habits, but also detects and supports the drifts in their patterns. We proposed a new algorithm of online association rules mining that supports the concept drift.

Our last contribution includes a novel hybrid approach that combines activity recognition and prediction algorithms in order to online recognize users' outdoor activities without draining the mobile resources. Our approach minimizes activity computations by wisely reducing the search frequency of activities.

7.3. KNOWN LIMITATIONS OF THE PROPOSED MODEL

Despite the success of the model proposed in this thesis, we believe that it has some limitations. First, we recognize the user activity depending on his neighborhood, i.e., when we detect that he is probably doing some activity, we check for his surrounded geographic entities to figure out what is he doing. For instance, if we find that the user is near a mall, we use the taxonomy presented in section 3.2.3 to figure out that the user is doing shopping. However, in the real world, this process is very complicated because the user may perform several activities inside these POIs, for instance, a mall can contain cinemas, restaurants and a set of shops. We tried in section 6.2.3.1 to answer this problem by clustering the user dwell times inside the POI, which means that we succeed to recognize that the user is doing different activity types inside the mall basing on his dwell time there (we detect a multitude of dwell time clusters). However, we don't have any semantic information about the detailed activity, i.e. whether he is in the cinema, the restaurant or doing shopping.

The second limitation is that the association of a geographic entity with an activity is made on distances, which means that when we detect that the user is doing an activity, we choose the nearest geographic entity to the user and we adopt that it is the visited place. However, in big cities where the urban architecture is very dense, this method may fail because there are a lot of probable geographic entities that can be the visited place. In this, case, relying solely on the distances is probably not effective.

7.4. Prospects and Future Works

Although the online activity model that we developed possesses its downsides, we are very optimistic about its future. This thesis project has laid a foundation on an emerging field of research that should provide a lot of challenges to the community for many years to come. In this section, we discuss the future work on online recognition of users' activities by using smartphones.

First, 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 movement's pattern of individuals varies between young and old, healthy and sick, male and female. The implementation of such rules will improve the accuracy of our activity recognition process.

As presented earlier, our system encounters difficulties to obtain information about the exact activity performed inside the POI. This issue can be answered by proposing another layer that is specialized in detecting fine grained indoor movements, this can be used by Wi-Fi triangulation [172], PDR technique [173] or even by using Bluetooth beacons [174] as described in Chapter 4. For instance, when we detect that the user is inside a mall, we can check for the nearby Bluetooth signals to calculate his exact location inside the POI, after that, if we detect that he is near the cinema for instance, we can assume that he is watching a movie.

In the prediction part of our approach some efforts shall be done to optimize the construction and the research process in the tree structure to minimize the response time of our algorithm. Secondly, the clustering of users' profiles represents an interesting research field. In that direction, the habits' tree represents a good structure that summarizes users' routines. Clustering users' profiles basing on their habits will be reduced to the comparison of two trees (habits' trees). Which means that it will be possible to regroup users inside a community basing on their mobility patterns. The solution will be based on comparing their habits tree that we proposed in section 3.2. This profile clustering is useful for several fields such as smart traffic monitoring and collaborative transportation. Thirdly, prediction part of our system has to be improved by introducing a temporal dimension to the habits' tree in order to improve our algorithm precision, for example, routines made at weekends are different of those made in working days.

7.5. Personal Assessment on This Research

In conclusion, I would like to make a brief personal assessment of my initiation to the world of research. The journey made throughout this project was quite a hard and continuous work. However, it was very rewarding, worthy of all these short nights for which I traded hours of sleep for acquisition new precious knowledge in the targeted areas of this thesis. This experience allowed me to develop important new skills such as a rigorous research methodology and communication skills. This rewarding experience also allowed me to make few contributions to the scientific community in my field of research that I presented at the occasion of notorious international conferences [50]–[52], [55], [57] and journals [53], [54], [56], [58]. After such a positive introduction to research, I only look toward beginning a career as a researcher and pushing the limit of science in new territories. My last words go to all the persons that supported me, one way or another, intentionally or not, in my quest to obtain an expertise, new skills set and priceless knowledge.

APPENDIX

The world is in the midst of a unique and irreversible process of demographic transition that will result in older populations everywhere. Developed nations are simply not working quickly enough to cope with a population graying faster than ever before. By the year 2050, for the first time in history, seniors older than 60 will outstrip children younger than 15 [175]. This demographic transition will unfortunately increase chronic diseases afflicting the elderly, new challenges linked to supporting them are already appearing in the horizons. Alzheimer's disease is currently ranked as the sixth leading cause of death for older people in the United States, [176]. In Canada, an estimated 564,000 people were living with dementia in 2016 [177]. Alzheimer is the most common form of dementia caused by an irreversible, progressive brain disorder that slowly destroys memory and thinking skills, and eventually the ability to carry out the simplest tasks, causing the loss of their autonomy and hence, their ability to take care of themselves. Therefore, at a certain stage in the evolution of the disease, these people must be assisted continuously for the rest of their lives. Health authorities have deployed some mechanisms to support these persons, they are attended at all times by a health professional or a caregiver. As a result, these persons are often placed in a specialized center thus reducing their quality of life and generating significant costs to the healthcare system. With the increasing numbers of elderly people, these mechanisms certainly need to be substituted by smarter and cheaper solutions that keep elderly autonomy and relieve the healthcare system.

Recent advances in information technology and geo-tracking techniques represent promising opportunities that can address the problems discussed above. Assistive technologies promote greater independence by enabling users to perform tasks that they were formerly unable to accomplish by proposing smart systems that replace as much as possible health professionals and caregivers [178]. To do so, these systems require continuous information on the individual's condition obtained by enhancing their environment using a variety of sensors (e.g. global positioning system, electromagnetic contacts, motion detectors, touch pad, radio-frequency identification tags, etc.) while ensuring to limit their intrusion in order to help the patients to perform their routines without invading their privacy. The LIARA laboratory (Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activités) where this work has been conducted is specialized in human cognitive assistance. As such this work will be used in this direction.

Human living is based on a set of social interactions that require that the individual goes outside, so it is crucial to adapt these techniques to open urban environments, taking into account the different geographic constraints that may challenge the monitoring of users' outdoor routines (shopping, going to restaurants, jogging…etc.).

Assistance process is seen as a succession of three important phases. First, it is necessary to develop an efficient system, capable of inferring exactly the nature of the outdoor activity performed by the patient using user-friendly solutions. This step is followed by an interrogation step where the system wonders if the user is doing something wrong, and finally, supporting users when anomalies are detected. For instance, assuming that we have some beforehand knowledge about the daily destinations of Alzheimer's patients. The mobile-based activity recognition system that we proposed is used to detect every anomaly in their behavior by comparing the planned and real performed activities or by detecting stress situations, if an error is detected, the assistance processes will be launched, like reminders, suggesting a new destination or back home roads, see Figure 68. The activity prediction that we proposed is used to predict the user next activity when an error is detected, for instance because the user has forgotten his next destination. The message type used when aiding users must be chosen carefully in order to stimulate the brain reactivity of the individual so that he corrects himself. When continuous support is provided to an Alzheimer patient, the disease degeneration is decelerated and the patient can remain independent longer [179].

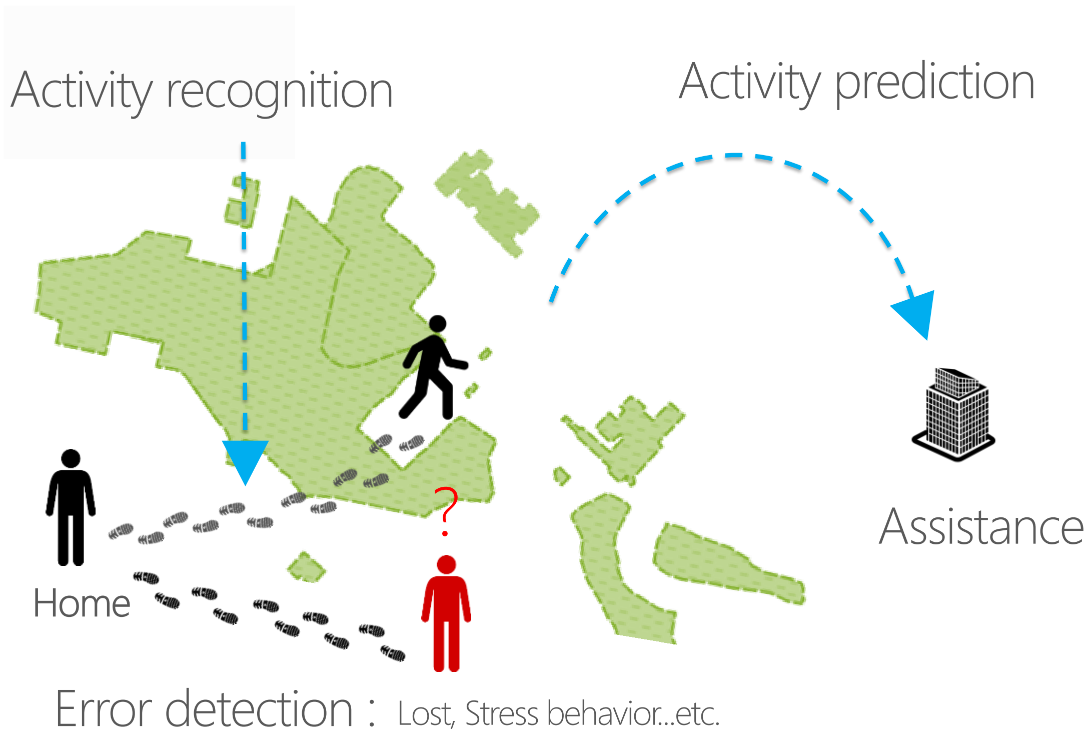


Figure 68: Using our approach to assist people surfing from Alzheimer's disease

BIBLIOGRAPHY

[1] D. Ferguson, “Mobile Phones,” in *Mobile .NET*, Berkeley, CA: Apress, 2002, pp. 71–97.

[2] *Smart Phone and Next Generation Mobile Computing*. Elsevier, 2006.

[3] G. Kearsley, “Bell laboratories,” 1981.

[4] “LAN/LAN and LAN/WAN internetworking launch,” *Comput. Commun.*, vol. 13, no. 6, p. 377, Jul. 1990.

[5] J. L. Burbank, J. Andrusenko, J. S. Everett, and W. T. M. Kasch, Eds., “Fourth-Generation (4G) Cellular Communications,” in *Wireless Networking*, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2013, pp. 469–558.

[6] E. miluzzo, “Smartphone Sensing,” Dartmouth College Hanover, New Hampshire., 2011.

[7] K. Curran, Ed., *Ubiquitous Developments in Ambient Computing and Intelligence: Human-Centered Applications*. IGI Global, 2011.

[8] M. Rehman, C. Liew, T. Wah, J. Shuja, and B. Daghighi, “Mining Personal Data Using Smartphones and Wearable Devices: A Survey,” *Sensors*, vol. 15, no. 2, pp. 4430–4469, Feb. 2015.

[9] Xing Su, Hanghang Tong, and Ping Ji, “Activity recognition with smartphone sensors,” *Tsinghua Sci. Technol.*, vol. 19, no. 3, pp. 235–249, Jun. 2014.

[10] H.-A. Jacobsen and V. Muthusamy, “Green Middleware,” in *Green IT: Technologies and Applications*, J. H. Kim and M. J. Lee, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 341–360.

[11] K. Ebi, “National and Global Monitoring and Surveillance Systems for the Health Risks of Global Change,” in *Global Environmental Change*, B. Freedman, Ed. Dordrecht: Springer Netherlands, 2014, pp. 649–655.

[12] A. Salibian and T. Scholz, “Smartphones in Surgery,” *J. Healthc. Eng.*, vol. 2, no. 4, pp. 473–486, Dec. 2011.

[13] A. Henrysson and M. Ollila, “Augmented reality on smartphones,” 2003, pp. 27–28.

[14] M. D. Braga, G. L. A. Mota, and R. M. E. M. Da Costa, “Technologies Integration of Immersive Virtual Reality on Smartphones with Real-Time Motion Capture,” 2016, pp. 127–134.

[15] O. Czogalla and S. Naumann, “Pedestrian Guidance for Public Transport Users in Indoor Stations Using Smartphones,” 2015, pp. 2539–2544.

[16] M. Sarwar, “Impact of Smartphones on Society,” 2013.

[17] S. Poslad, *Ubiquitous Computing*. Chichester, UK: John Wiley & Sons, Ltd, 2009.

[18] G. Banavar and A. Bernstein, “Software infrastructure and design challenges for ubiquitous computing applications,” *Commun. ACM*, vol. 45, no. 12, Dec. 2002.

[19] H. Ghasemzadeh, R. Fallahzadeh, and R. Jafari, “A Hardware-Assisted Energy-Efficient Processing Model for Activity Recognition Using Wearables,” *ACM Trans. Des. Autom. Electron. Syst.*, vol. 21, no. 4, pp. 1–27, Jun. 2016.

[20] J. J. Ockerman, “Understanding Mobile Human-Computer Interaction,” *Int. J. Hum.-Comput. Interact.*, vol. 22, no. 3, pp. 289–290, May 2007.

[21] F. Li, Y. Bao, D. Wang, W. Wang, and L. Niu, “Smartphones for sensing,” *Sci. Bull.*, vol. 61, no. 3, pp. 190–201, Feb. 2016.
[22] “Artifical Inteligence,” *NIPPON SHOKUHIN KOGYO GAKKAISHI*, vol. 40, no. 3, pp. 224–224, 1993.
[23] O. Löfgren, “Modes and Moods of Mobility: Tourists and Commuters,” *Cult. Unbound J. Curr. Cult. Res.*, vol. 7, no. 2, pp. 175–195, Jun. 2015.
[24] L. Hong, “Smartphone sensing and inference of human behavior and context,” *Dartm. Coll. Libr. Press*.
[25] L. Liao, “Location-based activity recognition,” University of Washington, 2006.
[26] E. Kim, S. Helal, and D. Cook, “Human Activity Recognition and Pattern Discovery,” *IEEE Pervasive Comput.*, vol. 9, no. 1, pp. 48–53, Jan. 2010.
[27] L. Piyathilaka and S. Kodagoda, “Human Activity Recognition for Domestic Robots,” in *Field and Service Robotics*, vol. 105, L. Mejias, P. Corke, and J. Roberts, Eds. Cham: Springer International Publishing, 2015, pp. 395–408.
[28] J. Bacon, “Toward pervasive computing,” *IEEE Pervasive Comput.*, vol. 1, no. 2, p. 84, Apr. 2002.
[29] A. McCullough, P. James, and S. Barr, “A Service Oriented Geoprocessing System for Real-Time Road Traffic Monitoring: Real-Time Road Traffic Monitoring,” *Trans. GIS*, vol. 15, no. 5, pp. 651–665, Oct. 2011.
[30] J. T. Foley, M. W. Beets, and B. J. Cardinal, “Monitoring Children’s Physical Activity With Pedometers: Reactivity Revisited,” *J. Exerc. Sci. Fit.*, vol. 9, no. 2, pp. 82–86, Dec. 2011.
[31] Y. Zhao and H. Jiang, “A Hybrid Biometric Personal Identification Method Based on Chinese Signature,” in *Advances in Computer Science, Intelligent System and Environment*, vol. 104, D. Jin and S. Lin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 309–314.
[32] F. Paetzold and U. Franke, “Road recognition in urban environment,” *Image Vis. Comput.*, vol. 18, no. 5, pp. 377–387, Apr. 2000.
[33] A. Tellez, “Social Networking for Promoting Physical Activity: (521582014-123),” 2009.
[34] O. D. Lara and M. A. Labrador, “A Survey on Human Activity Recognition using Wearable Sensors,” *IEEE Commun. Surv. Tutor.*, vol. 15, no. 3, pp. 1192–1209, 2013.
[35] W. Song, J. W. Lee, B. S. Lee, and H. Schulzrinne, “Finding 9-1-1 callers in tall buildings,” 2014, pp. 1–9.
[36] Q. Li, J. A. Stankovic, M. A. Hanson, A. T. Barth, J. Lach, and G. Zhou, “Accurate, Fast Fall Detection Using Gyroscopes and Accelerometer-Derived Posture Information,” 2009, pp. 138–143.
[37] D. j. Patterson, “Intelligent ubiquitous computing to support Alzheimer’s patients: Enabling the cognitively disabled,” in *Adjunct*, 2002.
[38] S. Kozina, H. Gjoreski, M. Gams, and M. Luštrek, “Efficient Activity Recognition and Fall Detection Using Accelerometers,” in *Evaluating AAL Systems Through Competitive Benchmarking*, vol. 386, J. A. Botía, J. A. Álvarez-García, K. Fujinami, P. Barsocchi, and T. Riedel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 13–23.
[39] Z. S. de Urturi, A. M. Zorrilla, and B. G. Zapirain, “Serious Game based on first aid education for individuals with Autism Spectrum Disorder (ASD) using android mobile devices,” 2011, pp. 223–227.
[40] A. J. Siegel, “Suicide Prevention by Smartphone,” *Am. J. Med.*, vol. 129, no. 8, p. e145, Aug. 2016.

[41] D. J. Cook and N. C. Krishnan, *Activity Learning, Discovering, Recognizing, and Predicting Human Behavior from Sensor Data*. Hoboken, NJ, USA: John Wiley & Sons, Inc, 2015.
[42] D. J. Cook, N. C. Krishnan, and P. Rashidi, “Activity Discovery and Activity Recognition: A New Partnership,” *IEEE Trans. Cybern.*, vol. 43, no. 3, pp. 820–828, Jun. 2013.
[43] *Data Mining: Practical Machine Learning Tools and Techniques*. Elsevier, 2011.
[44] K. Bouchard, “Unsupervised Spatial Data Mining For Human Activity Recognition Based On Objects Movement And Emergent Behaviors,” University of Quebce At Chicoutimi, 2014.
[45] D. Choujaa and D. Narnker, “Activity Recognition from Mobile Phone Data: State of the Art, Prospects and Open Problems,” Imperial College London, 2009.
[46] M. Shoaib, S. Bosch, O. Incel, H. Scholten, and P. Havinga, “A Survey of Online Activity Recognition Using Mobile Phones,” *Sensors*, vol. 15, no. 1, pp. 2059–2085, Jan. 2015.
[47] C. Seeger, A. Buchmann, and K. Van Laerhoven, “myHealthAssistant: A Phone-based Body Sensor Network that Captures the Wearer’s Exercises throughout the Day,” 2011.
[48] N. Ravi, *Activity recognition from accelerometer data*, AAAI., vol. 5. 2005.
[49] D. Holland, “Online Gis and Spatial Metadata,” *Photogramm. Rec.*, vol. 18, no. 101, pp. 80–81, Mar. 2003.
[50] M. Boukhechba, A. bouzouane, B. Bouchard, G.-V. Charles, and G. Sylvain, “Online recognition of people’s activities from raw GPS data: Semantic Trajectory Data Analysis,” presented at the The 8th ACM International Conference on Pervasive Technologies Related to Assistive Environments, 2015.
[51] M. Boukhechba, A. Bouzouane, B. Bouchard, C. Gouin-Vallerand, and S. Giroux, “Battery-aware mobile solution for activity recognition from GPS data,” presented at the IEEE International Conference on Mobile Services, San Francisco, 2016, pp. 1–8.
[52] M. Boukhechba, A. Bouzouane, S. Gaboury, C. Gouin-Vallerand, S. Giroux, and B. Bouchard, “Hybrid Battery-friendly Mobile Solution for Extracting Users’ Visited Places,” *Procedia Comput. Sci.*, vol. 94, pp. 25–32, 2016.
[53] M. Boukhechba, A. Bouzouane, B. Bouchard, C. Gouin-Vallerand, and S. Giroux, “Energy Optimization for Outdoor Activity Recognition,” *J. Sens.*, vol. 2016, pp. 1–15, 2016.
[54] M. Boukhechba, A. Bouzouane, S. Gaboury, C. Gouin-Vallerand, S. Giroux, and B. Bouchard, “Learning Human Habits Using Mobile Phones,” pp. 1–16, 2016.
[55] M. Boukhechba, A. Bouzouane, B. Bouchard, C. Gouin-Vallerand, and S. Giroux, “Online Prediction of People’s Next Point-of-Interest: Concept Drift Support,” in *Human Behavior Understanding*, vol. 9277, A. A. Salah, B. J. A. Kröse, and D. J. Cook, Eds. Cham: Springer International Publishing, 2015, pp. 97–116.
[56] M. Boukhechba, A. Bouzouane, S. Gaboury, C. Gouin-Vallerand, S. Giroux, and B. Bouchard, “Prediction of next destinations from irregular Patterns.,” pp. 1–25, 2016.
[57] M. Boukhechba, A. Bouzouane, S. Gaboury, C. Gouin-Vallerand, S. Giroux, and B. Bouchard, “NomaBlue : Nomadic Data Collection For Spatial Recognition In Smart Cities,” presented at the IEEE International Conference on Pervasive Computing and Communication (PerCom)., 2017, pp. 1–10.

[58] M. Boukhechba, A. Bouzouane, S. Gaboury, C. Gouin-Vallerand, S. Giroux, and B. Bouchard, “A novel Bluetooth Low Energy Based System For Spatial Exploration In Smart Cities,” pp. 1–12, 2016.
[59] S. von Watzdorf and F. Michahelles, “Accuracy of positioning data on smartphones,” 2010, pp. 1–4.
[60] P. A. Zandbergen and S. J. Barbeau, “Positional Accuracy of Assisted GPS Data from High-Sensitivity GPS-enabled Mobile Phones,” *J. Navig.*, vol. 64, no. 3, pp. 381–399, Jul. 2011.
[61] A. H. Sayed, A. Tarighat, and N. Khajehnouri, “Network-based wireless location: challenges faced in developing techniques for accurate wireless location information,” *IEEE Signal Process. Mag.*, vol. 22, no. 4, pp. 24–40, Jul. 2005.
[62] Youngjune Gwon, R. Jain, and T. Kawahara, “Robust indoor location estimation of stationary and mobile users,” 2004, vol. 2, pp. 1032–1043.
[63] S. Spaccapietra, C. Parent, M. L. Damiani, J. A. de Macedo, F. Porto, and C. Vangenot, “A conceptual view on trajectories,” *Data Knowl. Eng.*, vol. 65, no. 1, pp. 126–146, Apr. 2008.
[64] T. Brinkhoff, “Open Street Map As Data Source For Built-Up And Urban Areas On Global Scale,” *ISPRS - Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. XLI-B4, pp. 557–564, Jun. 2016.
[65] M. Ruta, F. Scioscia, S. Ieva, G. Loseto, and E. Di Sciascio, “Semantic Annotation of OpenStreetMap Points of Interest for Mobile Discovery and Navigation,” 2012, pp. 33–39.
[66] N. Eagle and A. (Sandy) Pentland, “Reality mining: sensing complex social systems,” *Pers. Ubiquitous Comput.*, vol. 10, no. 4, pp. 255–268, May 2006.
[67] N. Eagle and A. S. Pentland, “Eigenbehaviors: identifying structure in routine,” *Behav. Ecol. Sociobiol.*, vol. 63, no. 7, pp. 1057–1066, May 2009.
[68] K. Farrahi and D. Gatica-Perez, “Daily Routine Classification from Mobile Phone Data,” in *Machine Learning for Multimodal Interaction*, vol. 5237, A. Popescu-Belis and R. Stiefelhagen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 173–184.
[69] K. Farrahi and D. Gatica-Perez, “What did you do today?: discovering daily routines from large-scale mobile data,” 2008, p. 849.
[70] P. Bezousek and V. Schejbal, “Coherent multilateration systems,” 2008, pp. 60–65.
[71] “Differential GPS and Assisted GPS,” in *Introduction to Wireless Localization*, Singapore: John Wiley & Sons, Ltd, 2012, pp. 157–184.
[72] A. Quigley and D. West, “Proximation: Location-Awareness Though Sensed Proximity and GSM Estimation,” in *Location- and Context-Awareness*, vol. 3479, T. Strang and C. Linnhoff-Popien, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 363–376.
[73] V. Stewart, S. Ferguson, J.-X. Peng, and K. Rafferty, “Practical automated activity recognition using standard smartphones,” 2012, pp. 229–234.
[74] E. Miluzzo, N. D. Lane, S. B. Eisenman, and A. T. Campbell, “CenceMe – Injecting Sensing Presence into Social Networking Applications,” in *Smart Sensing and Context*, vol. 4793, G. Kortuem, J. Finney, R. Lea, and V. Sundramoorthy, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 1–28.
[75] E. Miluzzo *et al.*, “Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application,” 2008, p. 337.

[76] P. Formento, R. Acevedo, S. Ghoussayni, and D. Ewins, “Gait Event Detection during Stair Walking Using a Rate Gyroscope,” *Sensors*, vol. 14, no. 3, pp. 5470–5485, Mar. 2014.
[77] Y. Zheng, L. Wang, R. Zhang, X. Xie, and W.-Y. Ma, “GeoLife: Managing and Understanding Your Past Life over Maps,” 2008, pp. 211–212.
[78] M. Morzy, “Prediction of Moving Object Location Based on Frequent Trajectories,” in *Computer and Information Sciences – ISCIS 2006*, vol. 4263, A. Levi, E. Savaş, H. Yenigün, S. Balcısoy, and Y. Saygın, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 583–592.
[79] A. Ng, “Generative learning algorithms,” 2008.
[80] S.-B. Cho, K.-J. Kim, K. S. Hwang, and I.-J. Song, “AniDiary: Daily Cartoon-Style Diary Exploits Bayesian Networks,” *IEEE Pervasive Comput.*, vol. 6, no. 3, pp. 66–75, Jul. 2007.
[81] A. Lan and H. Muller, *Context awareness via gsm signal strength fluctuation*. 2016.
[82] M. Kose, O. Durmaz Incel, and C. Ersoy, “Performance evaluation of classification methods for online activity recognition on smart phones,” 2012, pp. 1–4.
[83] A. LaMarca *et al.*, “Place Lab: Device Positioning Using Radio Beacons in the Wild,” in *Pervasive Computing*, vol. 3468, H.-W. Gellersen, R. Want, and A. Schmidt, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 116–133.
[84] S. Lloyd, “Least squares quantization in PCM,” *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982.
[85] K. Zhang, H. Li, K. Torkkola, and M. Gardner, “Adaptive Learning of Semantic Locations and Routes,” in *Location- and Context-Awareness*, vol. 4718, J. Hightower, B. Schiele, and T. Strang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 193–210.
[86] A. Ram, S. Jalal, A. S. Jalal, and M. Kumar, “A Density Based Algorithm for Discovering Density Varied Clusters in Large Spatial Databases,” *Int. J. Comput. Appl.*, vol. 3, no. 6, pp. 1–4, Jun. 2010.
[87] Z. Indré, “Learning under concept drift: an overview,” Vilnius University, 2010.
[88] F. d’Alché-Buc, “Incremental Learning Algorithms for Classification and Regression: local strategies,” 2002, vol. 627, pp. 320–329.
[89] A. T. Palma, V. Bogorny, B. Kuijpers, and L. O. Alvares, “A clustering-based approach for discovering interesting places in trajectories,” 2008, p. 863.
[90] S. Chakraborty, N. K. Nagwani, and L. Dey, “Performance Comparison of Incremental Kmeans and Incremental DBSCAN Algorithms,” *Int. J. Comput. Appl.*, vol. 27, no. 11, pp. 14–18, Aug. 2011.
[91] V. Bogorny, H. Avancini, B. C. de Paula, C. R. Kuplich, and L. O. Alvares, “Weka-STPM: a Software Architecture and Prototype for Semantic Trajectory Data Mining and Visualization: Weka-STPM: a Software Architecture and Prototype,” *Trans. GIS*, vol. 15, no. 2, pp. 227–248, Apr. 2011.
[92] S. Václav, “WEKA and event related potentials,” *Front. Neuroinformatics*, vol. 2, 2008.
[93] “CLARANS (Clustering Large Applications Based Upon Randomized Search),” in *SpringerReference*, Berlin/Heidelberg: Springer-Verlag, 2011.
[94] L. Huang, Q. Li, and Y. Yue, “Activity identification from GPS trajectories using spatial temporal POIs’ attractiveness,” 2010, p. 27.
[95] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello, “Extracting places from traces of locations,” 2004, p. 110.

[96] L. Spinsanti, C. Fabrizio, and R. Chiara, “Where you stop is who you are: understanding people’s activities by places visited.,” Lausanne, Switzerland b KDDLab, ISTI, CNR, Pisa, Italy., 2010.
[97] Y. Takeuchi and M. Sugimoto, “CityVoyager: An Outdoor Recommendation System Based on User Location History,” in *Ubiquitous Intelligence and Computing*, vol. 4159, J. Ma, H. Jin, L. T. Yang, and J. J.-P. Tsai, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 625–636.
[98] J. Lester, T. Choudhury, and G. Borriello, “A Practical Approach to Recognizing Physical Activities,” in *Pervasive Computing*, vol. 3968, K. P. Fishkin, B. Schiele, P. Nixon, and A. Quigley, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1–16.
[99] S. Sharma, J. Agrawal, and S. Sharma, “Classification Through Machine Learning Technique: C4. 5 Algorithm based on Various Entropies,” *Int. J. Comput. Appl.*, vol. 82, no. 16, pp. 28–32, Nov. 2013.
[100] J. Cheng, U. M. Fayyad, K. B. Irani, and Z. Qian, “Improved Decision Trees: A Generalized Version of ID3,” in *Machine Learning Proceedings 1988*, Elsevier, 1988, pp. 100–106.
[101] M. Kose, I. Ozlem, and E. Cem, “Online human activity recognition on smart phones,” presented at the Workshop on Mobile Sensing: From Smartphones and Wearables to Big Data, 2012.
[102] Y. Zheng, Y. Chen, Q. Li, X. Xie, and W.-Y. Ma, “Understanding transportation modes based on GPS data for web applications,” *ACM Trans. Web*, vol. 4, no. 1, pp. 1–36, Jan. 2010.
[103] S.-K. Chan and W. Lam, “Pseudo Conditional Random Fields: Joint Training Approach to Segmenting and Labeling Sequence Data,” 2010, pp. 767–772.
[104] G. M. Sangeetha and Prashanth, “Training the SVM to Larger Dataset Applications using the SVM Sampling Technique,” *Indian J. Sci. Technol.*, vol. 8, no. 15, Jul. 2015.
[105] Y. L. Gai, “Research on Data Mining and Apriori Algorithm,” *Adv. Mater. Res.*, vol. 546–547, pp. 497–502, Jul. 2012.
[106] S. Gambs, M.-O. Killijian, and M. N. del Prado Cortez, “Next place prediction using mobility Markov chains,” 2012, pp. 1–6.
[107] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, and L. Terveen, “Discovering personal gazetteers: an interactive clustering approach,” 2004, p. 266.
[108] A. Asahara, K. Maruyama, A. Sato, and K. Seto, “Pedestrian-movement prediction based on mixed Markov-chain model,” 2011, p. 25.
[109] T. Meiser and B. Ohrt, “Modeling Structure and Chance in Transitions: Mixed Latent Partial Markov-Chain Models,” *J. Educ. Behav. Stat.*, vol. 21, no. 2, pp. 91–109, Jan. 1996.
[110] S. Scellato, M. Musolesi, C. Mascolo, V. Latora, and A. T. Campbell, “NextPlace: A Spatio-temporal Prediction Framework for Pervasive Systems,” in *Pervasive Computing*, vol. 6696, K. Lyons, J. Hightower, and E. M. Huang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 152–169.
[111] C. I. Ezeife and Y. Su, “Mining Incremental Association Rules with Generalized FP-Tree,” in *Advances in Artificial Intelligence*, vol. 2338, R. Cohen and B. Spencer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 147–160.
[112] C. Borgelt, “An implementation of the FP-growth algorithm,” 2005, pp. 1–5.
[113] H. Galeana-Zapién, C. Torres-Huitzil, and J. Rubio-Loyola, “Mobile Phone Middleware Architecture for Energy and Context Awareness in Location-Based Services,” *Sensors*, vol. 14, no. 12, pp. 23673–23696, Dec. 2014.

[114] F. Ben Abdesslem, A. Phillips, and T. Henderson, “Less is more: energy-efficient mobile sensing with senseless,” 2009, p. 61.
[115] M. B. Kjærgaard, J. Langdal, T. Godsk, and T. Toftkjær, “EnTracked: energy-efficient robust position tracking for mobile devices,” 2009, p. 221.
[116] Y. Wang *et al.*, “A framework of energy efficient mobile sensing for automatic user state recognition,” 2009, p. 179.
[117] M. A. Viredaz, L. S. Brakmo, and W. R. Hamburgen, “Energy Management on Handheld Devices,” *Queue*, vol. 1, no. 7, p. 44, Oct. 2003.
[118] P. Pillai and K. G. Shin, “Real-time dynamic voltage scaling for low-power embedded operating systems,” 2001, p. 89.
[119] B. Zhao, H. Aydin, and D. Zhu, “Reliability-aware Dynamic Voltage Scaling for energy-constrained real-time embedded systems,” 2008, pp. 633–639.
[120] E. Shih, P. Bahl, and M. J. Sinclair, “Wake on wireless:: an event driven energy saving strategy for battery operated devices,” 2002, p. 160.
[121] T. D. Schneider, “Shannon Entropy (Shannon Uncertainty),” in *Dictionary of Bioinformatics and Computational Biology*, Chichester, UK: John Wiley & Sons, Ltd, 2004.
[122] *Fuzzy Logic*. Elsevier, 1994.
[123] J. van Heijenoort and H. Reichenbach, “Elements of Symbolic Logic.,” *J. Symb. Log.*, vol. 31, no. 4, p. 675, Dec. 1966.
[124] “GIS Cloud,” *Offline Maps are out! | GIS Cloud*, 2016. .
[125] S. Shekhar and H. Xiong, “ESRI,” in *Encyclopedia of GIS*, S. Shekhar and H. Xiong, Eds. Boston, MA: Springer US, 2008, pp. 290–290.
[126] “ArcGIS Runtime SDKs,” *ArcGIS for Developers*, 2016. [Online]. Available: https://developers.arcgis.com//arcgis-runtime/. [Accessed: 01-Oct-2016].
[127] “Geofabrik north America-NewYork.” [Online]. Available: http://download.geofabrik.de/north-america/us/new-york.html. [Accessed: 12-Oct-2016].
[128] “GEOFABRIK // Home.” [Online]. Available: http://www.geofabrik.de/. [Accessed: 12-Oct-2016].
[129] “OpenStreetMap,” *OpenStreetMap*. [Online]. Available: https://www.openstreetmap.org/. [Accessed: 12-Oct-2016].
[130] “Geographic Information Analysis and Spatial Data,” in *Geographic Information Analysis*, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2010, pp. 1–32.
[131] S. Davidoff, J. Zimmerman, and A. K. Dey, “How routine learners can support family coordination,” 2010, p. 2461.
[132] J. Chon and Hojung Cha, “LifeMap: A Smartphone-Based Context Provider for Location-Based Services,” *IEEE Pervasive Comput.*, vol. 10, no. 2, pp. 58–67, Apr. 2011.
[133] “Introducing PostgreSQL,” in *Beginning PHP and PostgreSQL 8*, Apress, 2006, pp. 573–577.
[134] “Introducing SQLite,” in *The Definitive Guide to SQLite*, Apress, 2006, pp. 1–16.
[135] “LifeMap Dataset,” *Crawdad*. [Online]. Available: http://crawdad.org/yonsei/lifemap/20120103/.
[136] Y. Chon, E. Talipov, H. Shin, and H. Cha, “Mobility prediction-based smartphone energy optimization for everyday location monitoring,” 2011, p. 82.

[137] J. Koski, “Multicriteria Truss Optimization,” in *Multicriteria Optimization in Engineering and in the Sciences*, W. Stadler, Ed. Boston, MA: Springer US, 1988, pp. 263–307.
[138] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, “Internet of Things for Smart Cities,” *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, Feb. 2014.
[139] A. V. Medina, I. M. Gómez, M. Romera, J. A. Gómez, and E. Dorrozoro, “Indoor Position System Based on BitCloud Stack for Ambient Living and Smart Buildings,” in *IT Revolutions*, vol. 82, M. Liñán Reyes, J. M. Flores Arias, J. J. González de la Rosa, J. Langer, F. J. Bellido Outeiriño, and A. Moreno-Munñoz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 127–136.
[140] T. Ballard, “Google Maps and Google Earth,” in *Google This!*, Elsevier, 2012, pp. 113–124.
[141] “Waze,” 2016. [Online]. Available: https://www.waze.com/en/. [Accessed: 01-Oct-2016].
[142] “Gis2go,” *Mobile GIS App for ArcGIS - gis2go.com* |, 2016. [Online]. Available: http://www.gis2go.com/home-en.html. [Accessed: 01-Oct-2016].
[143] “BlueDating,” 2016. [Online]. Available: http://www.csg.ethz.ch/research/projects/Blue_star. [Accessed: 01-Oct-2016].
[144] “Leawo iTransfer,” *Leawo iTransfer for Mac*, 2016. [Online]. Available: http://www.leawo.org/itransfer-mac/. [Accessed: 01-Oct-2016].
[145] A. F. gen. Schieck, V. Kostakos, and A. Penn, “Exploring Digital Encounters in the Public Arena,” in *Shared Encounters*, K. S. Willis, G. Roussos, K. Chorianopoulos, and M. Struppek, Eds. London: Springer London, 2009, pp. 179–195.
[146] A. G. Miklas, K. K. Gollu, K. K. W. Chan, S. Saroiu, K. P. Gummadi, and E. de Lara, “Exploiting Social Interactions in Mobile Systems,” in *UbiComp 2007: Ubiquitous Computing*, vol. 4717, J. Krumm, G. D. Abowd, A. Seneviratne, and T. Strang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 409–428.
[147] J. Decuir, “Introducing Bluetooth Smart: Part II: Applications and updates.,” *IEEE Consum. Electron. Mag.*, vol. 3, no. 2, pp. 25–29, Apr. 2014.
[148] S. Statler, *Beacon Technologies*. Berkeley, CA: Apress, 2016.
[149] “Bluetooth 5.0,” *Bluetooth 5 Coming Soon | Bluetooth Technology Website*, 2016. [Online]. Available: https://www.bluetooth.com/news/pressreleases/2016/06/16/-bluetooth5-quadruples-rangedoubles-speedincreases-data-broadcasting-capacity-by-800. [Accessed: 01-Oct-2016].
[150] C.-M. Chang, S.-C. Li, and Y. Huang, “Crowdsensing route reconstruction using portable bluetooth beacon-based two-way network,” 2016, pp. 265–268.
[151] “Discover your Regent Street,” 2016. [Online]. Available: http://www.regentstreetonline.com/zeitgeist/regent-street-app. [Accessed: 01-Oct-2016].
[152] “House of Fraser,” *House of Fraser :: Gifts, Fashion, Beauty, Home & Garden*, 2016. [Online]. Available: http://www.houseoffraser.co.uk/. [Accessed: 01-Oct-2016].
[153] “Samsung Digital Discovery Tour,” *Sydney Opera House*, 2016. [Online]. Available: http://www.sydneyoperahouse.com/visit/education/samsung_digital_discovery_tour.aspx. [Accessed: 01-Oct-2016].
[154] “BlueCats Beacons,” *BlueCats Beacons*, 2016. .

[155] D. Giovanelli, B. Milosevic, and E. Farella, “Bluetooth Low Energy for data streaming: Application-level analysis and recommendation,” 2015, pp. 216–221.
[156] “zlib,” 2016. [Online]. Available: http://zlib.net/. [Accessed: 01-Oct-2016].
[157] “Estimote,” 2016. [Online]. Available: http://estimote.com/. [Accessed: 03-Oct-2016].
[158] A. Banos, C. Lang, and N. Marilleau, *Agent-based spatial simulation with NetLogo. Volume 1, Volume 1,*. London: Elsevier, 2015.
[159] “CityAgents,” *GitHub*, 2016. [Online]. Available: https://github.com/Boukhechba/CityAgents. [Accessed: 01-Oct-2016].
[160] “LIMVI Laboratory,” *A new laboratory on Smart Cities and Mobile Computing is born! | Charles Gouin-Vallerand’s Blog & Website*, 2016. .
[161] J. Gama, R. Sebastião, and P. P. Rodrigues, “Issues in evaluation of stream learning algorithms,” 2009, p. 329.
[162] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, “Learning with Drift Detection,” in *Advances in Artificial Intelligence – SBIA 2004*, vol. 3171, A. L. C. Bazzan and S. Labidi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 286–295.
[163] Heng Wang and Z. Abraham, “Concept drift detection for streaming data,” 2015, pp. 1–9.
[164] L. Vu and G. Alaghband, “Efficient Algorithms for Mining Frequent Patterns from Sparse and Dense Databases,” *J. Intell. Syst.*, vol. 24, no. 2, Jan. 2015.
[165] A. Kapterev, “Where to Go Next,” in *Presentation Secrets*, Hoboken, NJ, USA: John Wiley & Sons, Inc., 2014, pp. 265–278.
[166] J. Hipp, U. Güntzer, and G. Nakhaeizadeh, “Algorithms for association rule mining --- a general survey and comparison,” *ACM SIGKDD Explor. Newsl.*, vol. 2, no. 1, pp. 58–64, Jun. 2000.
[167] E. S. Page, “CONTINUOUS INSPECTION SCHEMES,” *Biometrika*, vol. 41, no. 1–2, pp. 100–115, 1954.
[168] X. Zhang, C. Germain, and M. Sebag, “Adaptively detecting changes in Autonomic Grid Computing,” 2010, pp. 387–392.
[169] . S. R., “EFFICIENT CLASSIFICATION OF BIG DATA USING VFDT (VERY FAST DECISION TREE),” *Int. J. Res. Eng. Technol.*, vol. 5, no. 1, pp. 102–107, Jan. 2016.
[170] N. B. Karayiannis, “Generalized fuzzy c-means algorithms,” 1996, vol. 2, pp. 1036–1042.
[171] D. Watson, “Spatial tessellations: concepts and applications of voronoi diagrams,” *Comput. Geosci.*, vol. 19, no. 8, pp. 1209–1210, Sep. 1993.
[172] B. Kim, W. Bong, and Y. C. Kim, “Indoor localization for Wi-Fi devices by cross-monitoring AP and weighted triangulation,” 2011, pp. 933–936.
[173] J. Meyers, “Dead Reckoning,” *Sewanee Rev.*, vol. 123, no. 4, pp. lxvii–lxix, 2015.
[174] C.-H. Yun and J. So, “A Bluetooth Beacon-Based Indoor Localization and Navigation System,” *Adv. Sci. Lett.*, vol. 21, no. 3, pp. 372–375, Mar. 2015.
[175] A. Scuteri and P. M. Nilsson, “Aging Population,” in *Early Vascular Aging (EVA)*, Elsevier, 2015, pp. 17–20.
[176] M. J. Prince, “Prevention report from Alzheimer’s disease international (ADI),” *Alzheimers Dement.*, vol. 11, no. 7, p. P210, Jul. 2015.
[177] “Alzheimer Society Research Program | Alzheimer Society of Canada.” [Online]. Available: http://www.alzheimer.ca/en/Research/Alzheimer-Society-Research-Program. [Accessed: 13-Oct-2016].

[178] C. Ramos, J. C. Augusto, and D. Shapiro, “Ambient Intelligence&#x02014;the Next Step for Artificial Intelligence,” *IEEE Intell. Syst.*, vol. 23, no. 2, pp. 15–18, Mar. 2008.

[179] S. Horton-Deutsch, P. Twigg, and R. Evans, “Health care decision-making of persons with dementia,” *Dementia,* vol. 6, no. 1, pp. 105–120, Feb. 2007.