UNIVERSITÀ DEGLI STUDI DI MILANO
FACOLTÀ DI SCIENZE E TECNOLOGIE

CORSO DI LAUREA MAGISTRALE IN
INFORMATICA

HYBRID SEMANTIC AND STATISTICAL INCREMENTAL ACTIVITY RECOGNITION

Relatore:	prof.	Claudio Bettini
Correlatore:	Dott.	Gabriele Civitarese

Tesi di laurea di:
Riccardo Presotto
Matr. Nr. 901311

Anno accademico 2017/2018

Contents

Introduction **4**

1 State of art in activity recognition using mobile devices **7**
- 1.1 Activities . . . 8
- 1.2 Data sources . . . 8
- 1.3 Context information modelling . . . 12
- 1.4 Activity recognition techniques . . . 15
  - 1.4.1 Statistical learning models . . . 16
  - 1.4.2 Context-based activity recognition . . . 18
- 1.5 Previous works of EWLab and goal of this thesis . . . 19

2 System architecture **20**
- 2.1 Architecture modules . . . 20
  - 2.1.1 Sensor and context data . . . 22
  - 2.1.2 Incremental activity recognition module . . . 23
  - 2.1.3 Semantic refinement module . . . 24
  - 2.1.4 Prediction confidence evaluation module . . . 25

3 Technical solution **26**
- 3.1 Incremental activity recognition . . . 27
  - 3.1.1 Data cleaning . . . 28
  - 3.1.2 Smartphone orientation adjustment . . . 29
  - 3.1.3 Data aggregation and segmentation . . . 30
  - 3.1.4 Features extraction and standardisation . . . 30
  - 3.1.5 Activity model bootstrap . . . 34
  - 3.1.6 Classification . . . 34
- 3.2 Semantic refinement . . . 35
  - 3.2.1 Activities translation into ontological facts . . . 35

3.2.2 Context reasoning and refinement 37
3.3 Prediction confidence evaluation 39
3.3.1 Semi-supervised model update 39
3.3.2 Incremental data balancing 40

4 Experimental evaluation 42
4.1 Setup for data collection and the dataset 43
4.2 Metrics 45
4.3 Results 46

5 Discussion and future works 59

Introduction

In the last decade, the development of electronic and computer systems together with the phenomenon of miniaturisation, allowed the spreading of mobile devices with unprecedented characteristics. The computational power, the small size and the every low price of these technologies favoured the increasing use of smartphones, smartwatches and fitness trackers so much to consider them an integral part of everyday lives.
These statements are confirmed by the latest research published by Cisco in June 2016 which point out how the mobile devices connected to the Internet are constantly growing. In fact, it is estimated that in 2020, 71% of the overall IP traffic will be generated by the “other devices ”(all devices other than personal computers), with a clear prevalence of smartphones which will be responsible for 30% of the overall traffic. Instead, wearable devices dedicated to health and well-being (including smartwatches and fitness trackers), it is expected that they will increase from 144 to 729 million of active units.

These mobile devices often integrate different types of sensors (i.e. accelerometer, magnetometer, light sensor, GPS, etc.), which carried by a user allow to provide precise and constant information about his movements, his state of health and the context that surrounds him/her. The extracted information, can be used for heterogeneous purpose: from pure entertainment to monitoring of anomalous behaviour up to the development of systems to guarantee safety and protection for users. In general, the main purpose of these technologies is to develop systems that improve and make easier the user life.
Over the past few years, were been created many intelligent applications that continuously monitor our daily activities to provide context-aware services [27]. In this context is placed the recognition of human activities. With this term, we intend a process that relying on the available data it is capable to infer the current user activity. The majority of activity recogni-

tion algorithms in the literature rely on supervised machine learning to infer the most likely performed activities by analysing inertial sensors data [25]. One of the major drawbacks of those solutions is the cost of collecting the amount of labeled data required to reach high recognition rate. Moreover, standard classifiers are trained once with available data, and the recognition model cannot evolve over time. To overcome this issue, semi-supervised and incremental approaches for activity recognition have been proposed [1]. Those methods only require a small amount of training data to initialise the recognition model, while techniques like co-learning, self-learning or active learning are used to assign labels to unlabelled sensor data [23, 2, 32, 35]. While the majority of semi-supervised methods showed to be effective on classifying few physical activities (e.g., walking, running, biking, etc.), their effectiveness on more complex and context-dependent activities is still unclear. Moreover, discriminating those activities which have similar motion patterns is still problematic. For instance, activities like *walking* and *taking the stairs*, or *standing* and *taking the elevator* are easily confused between them by purely statistical methods based on inertial sensors. Considering the context which surrounds the user could be valuable information to mitigate these issues [31, 33]. Indeed, a rich description of the user's context (e.g., semantic location, weather, traffic condition, speed, etc.) has the potential to enable the recognition of a wide set of activities which are a) highly dependent to the current context and b) difficult to recognise only considering inertial sensors. However, semi-supervised approaches rely on a small set of labeled data, while activities can be performed in a large number of possible context conditions. For this reason, directly using context-data as additional features may not be as effective as expected. On the other hand, common-sense knowledge about activities and contexts can be used to improve a statistical classification purely based on inertial sensor data [33].

In this thesis, we propose a real-time activity recognition method which combines semi-supervised learning and semantic context-aware reasoning. An incremental classifier is in charge of analysing inertial sensors data to provide probability distributions over the possible activities. A knowledge-based reasoning engine is then used to exclude from the statistical predictions those activities which are highly unlikely considering context-data. The system's output is the most likely activity from the resulting context refinement. Context-refined predictions are then used to update the incremental classifier. When the system is confident about the refined prediction, it is provided as a new labelled sample to the incremental classifier. On the other hand, when the system is not confident we ask the ground truth to the user.

In order to evaluate our method, we used a large dataset acquired in previous works of EW Lab that contains both inertial sensor data and rich context information. Results on this dataset show that context-refinement is effective in improving the recognition rate and, at the same time, allow to trigger a significantly lower number of queries.

My personal contributions to this work are the following:

- I have worked on the pre-processing, segmentation and features extraction of inertial sensors data provided by the existing EW Lab's dataset.
- I have developed a semi-supervised statistical activity recognition system capable to evolve its learning model thanks both self-learning and active-learning.
- I have integrated into our system the SMOTE data-balancing technique.
- I have integrated a symbolic reasoning module relies on an Ontology in a statistical semi-supervised activity recognition model, with the aim of excluding the not context consistent activities from the statistical classifier probability vector output.
- I have performed an extensive evaluation of our approach showing the positive impact of context-data to improve semi-supervised activity recognition.

Chapters of this thesis are organised as follows: Chapter 1 presents the state of the art of the research area related to the recognition of human activities then, it illustrates the most exploited sensors and the different techniques used in this field. Chapter 2.1.2 introduces the general architecture of the solution proposed for our hybrid semi-supervised and context-aware activity recognition system. In Chapter 3, we present the implementation of the proposed architecture by explaining in detail the data processing, the statistical model classification, the semantic-refining and the model updating process. Finally, in Chapter 4, we illustrate the experimental evaluation of the proposed solution, the comparison between different ways of exploiting context data and the tuning of our parameters and thresholds. In the end, in Chapter 5, we summarise the results obtained and discuss the open problems and the possible research directions to improve our system.

Chapter 1

State of art in activity recognition using mobile devices

In the context-aware services gained particular importance the human activities recognition services.
With human activity recognition, we mean a process that, starting with observations about the user and the environment that surrounds him, infers the carried out activity in a specific time interval. The attractiveness of this system is due to the possibility to become a starting point to future applications able to adapt dynamically their functionalities to the user behaviour.
The current chapter shows the state of the art in the human activities recognition scope. In Section 1.1 we discuss about the different types of recognisable activities by an activity recognition system and the different kind of technologies used to capture their discriminating information.
In Section 1.2 are then introduced the different types of sensors that allow collecting data exploited in the activity recognition task.
The chapters continue with a summary description of the concept of the *Ontology* where we explain why and how it is important for our purposes.
Finally, we present in Section 1.4 the most common techniques used in the activity recognition process discussing their strengths and weaknesses.

1.1 Activities

In context-aware applications, the term *activity* is used to denote a vast set of actions that one or more people can carry out.
In literature they are grouped into two subsets: *low-level* activities and *high-level* activities. The former identifies fisical activities (i.e., walking, running, sitting etc.) which are characterised by specific physical movements. The second indicate complex actions which imply the interactions with other objects (i.e., prepare a meal, driving a car, washing dishes etc.). In both cases, the general goal of activity recognition systems is to detect the current user activity using available data.
In literature, there exist two main approaches to collect user activity information: using *wearable Inertial sensors* or by *environmental sensors*. Wearable inertial sensors allow detecting simple motion pattern produced by a person's movement. For example, an accelerometer placed on the user's leg can measure different accelerations of his leg while he is running or walking. Differently, Environmental sensors are dislocated on specific objects in the environment and collect information about how the user interacts with them. Using wearable sensors, in many cases, it is possible to detect only *low-level* activities, while environmental sensors allow recognising also more complex activities like the *high-level* ones.
Furthermore, there exists also hybrid systems that exploit both approaches to increase the system reliability and extend the set of recognisable activities.

1.2 Data sources

The new generation of mobile devices (like smartphone and smartwatch) integrate many sensors that can be exploited to get useful data to characterise user's movements and get environmental information. Moreover being practically always connected to the internet, they can rely on web services to obtain additional data about high-level information (i.e weather condition, traffic etc..). We divided the available data sources into three categories relying on their characteristics:

- **Integrated sensor** they are built in sensors that provide both information on the user's movements (i.e. gyroscope magnetometer accelerometer) and on the environment that surrounds him/her (i.e. light sensor, microphone etc.)

– **Accelerometer**: it allows detecting acceleration intensity in a three-dimensional space. The accelerations measurements are expressed relatively of x, y and z-axes. This sensor works thanks to the inertia principle: a mass subjected to an acceleration moves from the rest position proportionally to the detected acceleration. Generaly an accelerometer consists of two components, one fixed and one movable. The movable component is compressed and expanded proportionally to the amount of external stress received. These movements are captured by the fixed part which comes in contact to the movable one. The intensity and the direction of these variations characterise the acceleration measurements.

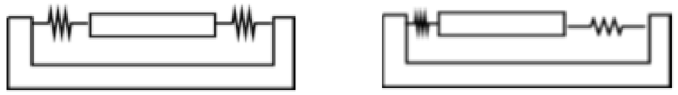

(a) state of rest (b) state of stress

Figure 1.1: Accelerometer in action.

– **Magnetometer:** it is able to measure the magnetic fields intensity variation. There are two types of magnetometers: scalars and vectorial. The former measures the magnetic field module. The vectorial type sensors measure the magnetic field component along a particular spatial axis. In general, in mobile devices, there are three vectorial magnetometers which allow detecting magnetic variations along the three x, y and z-axes. The magnetometer can be used in the same way as a compass, as its measurements allow to get the device orientation.

– **Gyroscope:** it estimates the device angular velocity on one or more axes. More precisely, through the gyroscope it is possible to detect the movements of the device in terms of roll, pitch, and yaw respectively along the x, y and z-axes. For this reason, the gyroscope is also used for advanced motion sensing applications such as simulations in video gaming.

– **Light sensor:** this sensor allows measuring the intensity of electromagnetic radiation that reaches the sensor. In mobile devices field, the light sensor is usually placed on smartphone and smartwatch' displays, as it is used to estimate the amount of ambient

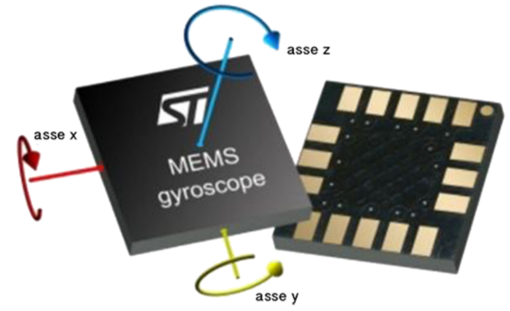


Figure 1.2: A gyroscope integrated sensor.

light that reaches the display and consequently adjust the display brightness. In our application, light intensity data are considered context information. Indeed, the amount of ambient light could be a piece of characterising information between different activities. For example, *taking stairs* is an activity that is usually carried out indoors where there is low light, while *cycling* it is performed outdoors preferably on sunny days.

  – **Microphone:** it is a sensor capable of recording the environmental sound/noise. This sensor is installed into common smartphones (and some smartwatches). More precisely, we use microphone to measure the environmental noise. In an activity recognition system, the microphone's data could indicate, for example, if the user is in a crowded street (characterised by a high level of noise) or in a quieter space such as a hotel room.

- **Location:** most of the actual mobile devices are able to calculate their position independently. The widespread techniques for this purpose refer to the GPS and GSM standards, but there are also some devices that can exploit additional data (such as Wi-Fi signals) to obtain positioning information.

  – **GPS (Global Position System) Receiver**: GPS system is a network of about 30 satellites that orbiting the Earth at an altitude of $20,000\ km$. Each satellite continuously transmits a radio signal that shows the satellite position associated with the transmitting timestamp. A GPS receiver continuously reads the available satellites signals and use them to compute its position:

it calculates the distance to each satellite from which it has received a signal, and then perform triangulation on the computed distances to obtain its position in therm of polar coordinates. In our case, the device position is used to characterise the context in which the device is located, exploiting the Reverse Geo-Coding process that allows transforming geographic coordinates into semantic places. Furthermore, position information is useful for determining the device speed. In the activity recognition field, speed is of primary importance to discriminate between activities that have similar motion pattern. For example, the activities *sitting on a bus* and *sitting on a train* are characterised by distinct speeds.

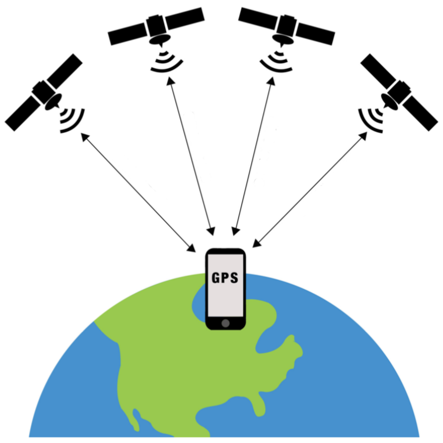


Figure 1.3: example of GPS system.

- **Wi-Fi connection**: it can indicate (in reference to the distance to the Wi-F1 emitter) if a device is inside or outside a building. Also, since the position of the Wi-fi emitters is known, it is possible to approximate the device location basing on the emitter position to which it is connected.
- **GSM network connection**: in the absence of GPS data a GSM connection can be used to approximate the device position using techniques similar to those used for GPS: here the position is estimated performing the triangulation on the transmitting antennas' distances to which the device is connected.

- **web services:** mobile devices connected to the Internet can make requests to services able to provide rich context information.These services generally offer Web APIs that take input data from the user's device and compute a series of information relied on the input received. The first step is extracting useful information from the device itself. Then a specific query is created and sent to the web service. Finally, it processes the query and returns the response to the device. As an example, information commonly used to build queries is the location of the device. Starting from this, the requests can be made to the web service in order to obtain the *semantic place*, the *weather condition* or the *traffic conditions*.

1.3 Context information modelling

In the last two decade, there was a growing body of research on the use of context-awareness for investigates approaches to modelling context information used by context-aware applications. In fact, a good context information modelling formalism reduces the complexity of context-aware applications and improves their maintainability and evolvability.
The goal of this Section is to give a summary view of the state-of-the-art in context modelling, explaining more in detail the ontological based techniques. The prominent approaches to context modelling are the following:

Object-role based

Object-role based are originated from attempts to create sufficiently formal models of context to support query processing and reasoning, as well as to provide modelling constructs suitable for use in software engineering tasks such as analysis and design. Early context modelling approaches, such as attribute-value pairs [24], could not satisfy these requirements, particularly as the types of context information used by applications grew more sophisticated.
To overcome these shortcomings was developed a Context Modelling Language (CLM) [19, 17, 18] that have its early roots in database. In particular it is based on Object-Role Modelling (ORM) [16] which was developed for conceptual modelling of database. CML provides a graphical notation designed to support the software engineer in analysing and formally specifying

the context requirements of a context-aware application. The formality of ORM and the CML extensions makes it possible to support a straightforward mapping from a CML-based context model to a runtime context management system that can be populated with context facts and queried by context-aware applications.

Spatial models

Spatial models are based on the concept of space. In fact, it is an important context information in many context-aware applications. In the most frequently used context definition by Dey et al. [10], space can be seen as a central aspect of context entities: *"An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves"*.
Most spatial context models organise their context information by physical location. This could be the location of the real world entities which is described in the context information (e.g., the boundaries of a room), the location of the sensor that measures the context information, or, for non-physical context information, an associated location as metaphor (e.g., Stick-E-Notes or virtual information towers). This location information is either pre-defined (if the entities are static), or it is obtained by positioning systems which track mobile objects and report their position to a location management system. Basically, two kinds of coordinate systems are supported by positioning systems: *Geometric coordinates* and *Symbolic coordinates.* The former represents points or areas in a metric space, such as coordinates of GPS (latitude, longitude, and elevation above sea level) and using geometric functions such as the Euclidian distance allows distance calculation and nearest neighbour queries. The second are represented by an identifier, such as a room number or the ID of a cell or access point in wireless telephone or local area networks. Here there is no spatial relation offered by symbolic coordinates. In order to allow spatial reasoning about inclusion and distances explicit information about the spatial relations between pairs of symbolic coordinates has to be provided.

Ontology-based models of context information

Context, as intended in this thesis, can be considered as a specific kind of knowledge. The trade-off between expressiveness and complexity of reasoning has driven most of the research in symbolic knowledge representation in the last two decades, and description logics [4] have emerged among other logic-based formalisms, mostly because they provide complete reasoning supported by optimised automatic tools. Since ontologies are essentially descriptions of concepts and their relationships, the subset of the OWL language (Web Ontology Language) admitting automatic reasoning is indeed a description logic. Ontology-based models of context information exploit the representation and reasoning power of these logics for multiple purposes: (a) the expressiveness of the language is used to describe complex context data that cannot be represented, for example, by simple languages like CC/PP; (b) by providing a formal semantics to context data, it becomes possible to share and/or integrate context among different sources; (c) the available reasoning tools can be used both to check for consistency of the set of relationships describing a context scenario, and, more importantly, to recognise that a particular set of instances of basic context data and their relationships actually reveals the presence of a more abstract context characterisation (e.g., the user's activity can be automatically recognised). In the following, we illustrate the two fundamental concepts on which the Ontology is based:

- **Context information representation**: the formalism of choice in ontology-based models of context information is typically OWL-DL [22] or some of its variations. By means of OWL-DL it is possible to model a particular domain by defining *classes*, *individuals*, *characteristics of individuals* (datatype properties), and *relations between individuals* (object properties). Complex descriptions of classes and properties can be built by composing elementary descriptions through specific operators provided by the language. For instance, given two atomic classes *Person* and *Female*, the class *Male* can be defined as:

  $$Male \equiv Person \sqcap \neg Female$$

  More complex definitions can be obtained by using operators such as property restrictions that can force all/some values of a certain property to belong to a given class, or can force a property to have at least k values. Hence, complex context data, intended as those data that can be inferred by means of reasoning tasks on the basis of raw data directly acquired from sensors, and other complex context data, can be represented by structured OWL-DL expressions. These data typically include information regarding the sociocultural environment of users, complex user preferences regarding the adaptation of services, and activities. For example, Definition 1.1 is used to describe *BusinessMeeting* as including any activity performed in a conference room within a company building, and having at least two actors, each of which is an employee:

$$BusinessMeeting \subseteq Activity \sqcap \geq 2 hasActor \sqcap \forall\ hasActor.Empoyee \sqcap \exists\ hasLocation.(ConfRoom \sqcap CompanyBuilding) \quad (1.1)$$

- **Support for reasoning:** a further benefit of ontologies with respect to simpler representation formalisms consists in the support of reasoning tasks. Indeed, on the basis of the asserted knowledge it is possible to (i) automatically derive new knowledge about the current context, and (ii) detect possible inconsistencies in the context information. With respect to (i), ontological reasoning can be executed for inferring new context information based on the defined *classes* and *properties*, and on the individual objects retrieved from sensors and other context sources. For instance, it is possible to derive the set of individual objects that are related to a given one by a particular property (e.g., the set of activities taking place in a specific location), or to calculate the most specific class an individual object belongs to (e.g., the fact that the activity performed by a given employee is a business meeting). With respect to (ii), we point out that *consistency checking* is crucial in the definition of an ontology, as well as in its population by new instances. Hence, automatic consistency checking can be performed to capture possible inconsistencies in the definition of the classes and properties of the ontology (e.g., a class being a subclass of two disjoint classes), or in its population (e.g., a person being in different rooms at the same time).

1.4 Activity recognition techniques

The recognition of physical activities using commonly available mobile devices (e.g., smartphones and smartwatches) is a widely explored research area [27]. Most of the method in literature create the learning model relying only on inertial sensor data using different statistical approaches. Some examples are described in Section 1.4.1.

The main limitation of the most common statistical methods based on internal sensors data, it is that they allow recognising with acceptable confidence a restricted number of physical activities. For this reason, some works have been proposed to integrate the context data in the classification process to improve the performances of statistical approaches based on inertial sensors data. We discuss these methods in Section 1.4.2.

1.4.1 Statistical learning models

In literature, are defined various approaches to the physical activities recognition problem. The two most common approaches are: *supervised* and *unsupervised* learning technique.
Both techniques involves two distinct phases:

- **Training**: in this phase a substantial number of data is provided as input to a recognition algorithm with the aim of creating the *learning model*.
- **Recognition phase**: the second step allows exploiting the *learning model* created during the previous phase. Starting from this, a specific recognition algorithm, or a classifier, produces a prediction for each given input data.

The main difference between *supervised* and *unsupervised* approaches consists of the type of data provided as input in the training phase.

- **Supervised learning**: the majority of approaches in the literature to infer activities from inertial sensors data rely on *supervised methods* [25, 15, 36, 6, 8]. In this approach, in the training step, the learning model is created exploiting a wide set of *labelled* data. Differently, the recognition phase uses *unlabelled* data. Here, the aim of the classifier is to analyse a not annotated data stream and query the learning model to make a prediction on the performed activity. While these methods allow reaching high recognition rates, the acquisition of a wide labelled dataset of activities is costly and often unfeasible. The annotation task is generally performed manually and therefore it is an operation that takes a long time to be finalised. For this reason, supervised classifiers are trained once with the available data. The learning model cannot evolve over time unless it is rebuilt by the start.

- **Unsupervised learning**: in order to reducing the effort required to label data in supervised techniques, few works proposed *unsupervised* learning techniques [26, 39, 28]. In fact, unsupervised learning exploits *unlabelled* data for the training phase. More precisely, during this step, the algorithm attempts to construct the learning model without knowing the activities associated with the input data received. Here, the activity patterns are hence found from unlabelled measurements with data mining techniques.
  Then, in the recognition phase, in the same way as supervised models, the learning model is exploited to perform a prediction on the unlabelled data supplied as input to the classifier. Their strength is to make it possible to extend the model on the fly, without the need for any external intervention. However, the performances of unsupervised classifiers are generally lower than the supervised ones.

In order to combine the strengths of both *supervised* and *unsupervised* approaches, *semi-supervised* learning methods for activity recognition have been proposed [1, 35].

Semi-supervised learning

In these methods, the training phase is no longer performed only at the beginning, but also throughout the recognition phase. In fact, Semi-supervised learning exploits small labeled training set to initialise the *learning model*, which is continuously enhanced using unlabelled data. In the recognition phase, most confident predictions are generally used to update the learning model. In the literature, the semi-supervised approaches which have been mainly considered for activity recognition are *Self-learning*, *Co-learning*, and *Active learning*.
Self-learning methods exploit the starting small training set to classify unlabelled data [32]. The most confident predictions are hence used to update the classifier.
Co-learning involves multiple classifiers trained on different data perspectives. Those classifiers collaboratively improve their models exploiting their prediction's confidences [29, 13].
The strength of these two methods is to be able to auto-update the learning model, without any user involvement. In some cases, it can also become its

weakness. In fact, the predictions made by the classifier are not always correct and the learning model could be updated with incorrectly labelled data that lead to a decreasing the reliability of the classifier.
Differently from self-learning and co-learning, *Active learning* requires explicit feedback from the users in order to obtain labels for the most informative data (i.e., data where the classifier is uncertain about the performed activity) [20, 2, 23]. Here, when the classifier is not quite sure of a prediction, it asks a question to the user about the activity label, avoiding any risk of incorrect updating of the learning model. The side effect of this method is that the user must be involved in the learning process and that he/she might not be willing to supply the requested labels often. In general, we can say that active learning proved to be particularly effective for semi-supervised activity recognition. However, for the sake of acceptability, the number of triggered queries should be low.

1.4.2 Context-based activity recognition

Statistical prediction techniques based on inertial sensor data are generally affected by errors due to the statistical component. In the literature there are several works that try to overcome the limits of statistical techniques, trying to improve the accuracy and efficiency of the activity recognition systems. Some proposals focus on improving the data pre-processing phase with the purpose to eliminate the influence of physical phenomena (such as gravity) from sensor data [3].
Other solutions, instead of working on getting better specific phases of statistical prediction process, attempt to improve the efficiency of the prediction system including useful information to characterise the *context* in which the user is immersed.
Context-dependent techniques combine statistical classifiers with symbolic reasoning methods to construct a recognition model that allows to overcome the limits of statistical model and improve the system efficiency. The principal symbolic technique used in this field is the *Ontology*.
In the following, we present an example of a system that exploits context-based techniques to improve statistical prediction. COSAR (Combined Ontological / Statistical Activity Recognition) [33] is a mobile system developed for the recognition of human activities, which uses the context refinement on the output prediction produced by the statistical recognition classifier. More precisely, COSAR proposes a technique that can be used for the activity

recognition task both from the analysis of inertial sensors data and environmental sensors information. The inertial sensors data collected, are used by the statistical reasoning module to make a multinomial prediction on the user activity. Then, context information is exploited by the ontological reasoning module that refines the multinomial prediction, determining which activity is most likely to be performed by the user.

1.5 Previous works of EWLab and goal of this thesis

In previous works of EWlab was created architecture capable to collect data about the physical movements made by a user to carry out an activity and the context information in which it takes place. More precisely different users, during the performances of various activities, were been equipped with a smartphone and a smartwatch that, thanks to specific applications developed to this task, have collected the data stream from their inertial sensors(i.e. accelerometer, magnetometer, gyroscope etc.) and a rich context information (i.e. GPS position, altitude, weather forecast, road traffic etc.). Collected measurements were therefore organised and stored in a specific dataset.
In another previous work, was developed an ontology by using the activities context data collected, with the aim of integrating into a future work a context-reasoner system capable to improve the performances of an activity-recognition classifiers. In this thesis, we have developed an incremental semi-supervised activity recognition classifier in charge of analysing inertial sensors data from smartphone and smartwatch to provide probability distributions over the possible user's activities. Then, we have adapted the already created ontology to our work, exploiting it to build a context-refinement module capable to exclude the not context-consistent activities. Finally, the refined prediction is used to continuously update our semi-supervised classifier through a combination of self-learning and active-learning.

Chapter 2

System architecture

In this chapter we introduce the architecture of our semi-supervised and context-aware activity recognition system. This architecture allows producing an activity statistical prediction on a stream of inertial sensor data provided from smartphones and smartwatches. Then it exploits an Ontology to applies knowledge-based reasoning in order to exclude from the statistical prediction those activities which are not context-consistent. The proposed system is also capable to continuously update itself using both *active-learning* and *self-learning* technique: a specific module evaluates the prediction confidence and decides whether using directly the predicted label to update the model (self-learning) or ask it to the user (active-learning).

2.1 Architecture modules

Our hybrid semi-supervised and context-aware activity recognition algorithm relies on the architecture shown in Figure 2.1.
The input data are provided by a smartphone and a smartwatch that, thanks to specific mobile application, are able to collect inertial sensors data and a rich context information while a user carry out an activity (see Section 2.1.1). These data are hence processed by our algorithm that involves three steps. At first, the stream of raw inertial sensors measurements is processed by the Incremental Activity Recognition module explained in Section 2.1.2. This module first applies pre-processing methods like raw data cleaning, alignment, segmentation and feature extraction. Then, a semi-supervised classifier associates to each feature vector a probability distribution over the

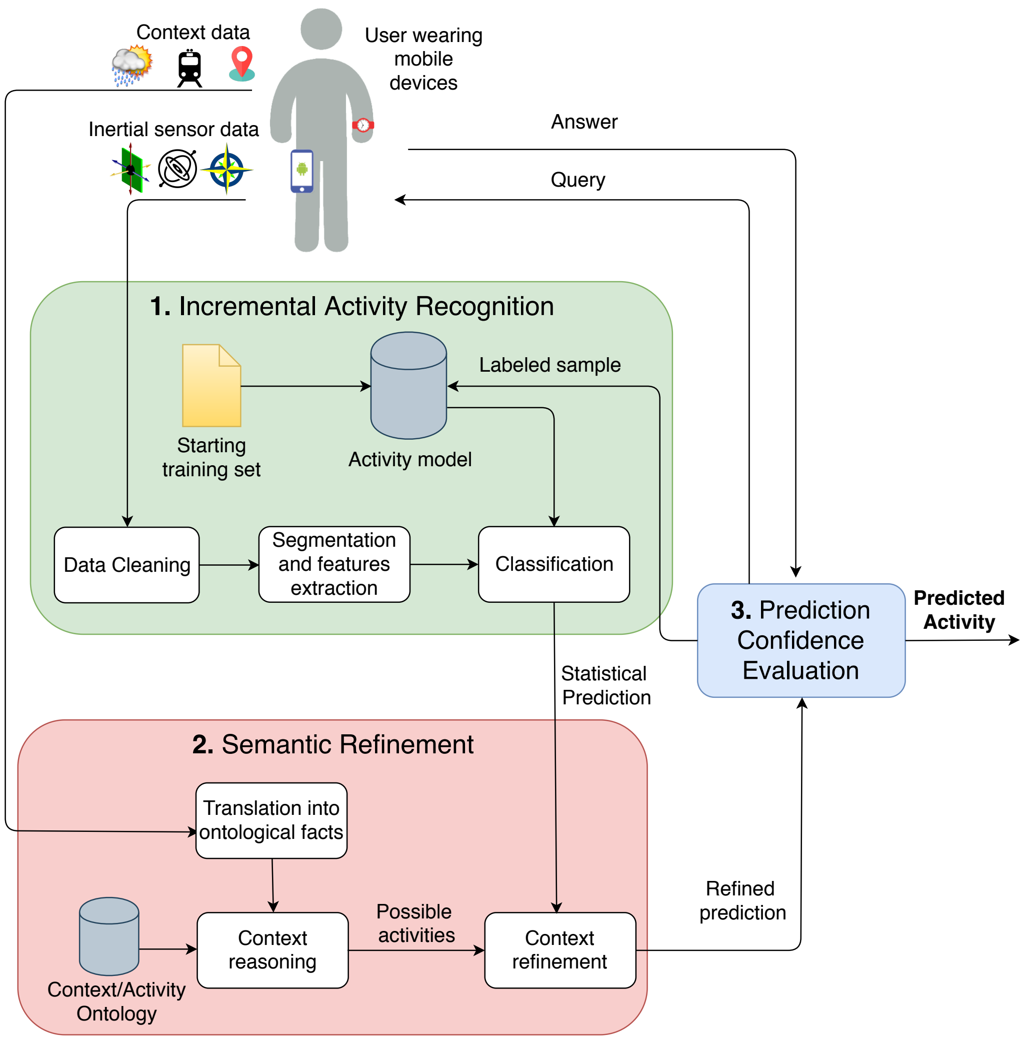


Figure 2.1: Overall architecture of our system.

possible activities.
The second step of our algorithm consists of the SEMANTIC REFINEMENT phase. It is introduced in Section 2.1.3 and it translates context data provided by mobile devices into ontological facts. Then, it applies knowledge-based reasoning in order to exclude from the semi-supervised prediction those activities which are not context-consistent.
The third and the last step of our method is explained in Section 2.1.4 and

it is called PREDICTION CONFIDENCE EVALUATION. It allows to evaluate the refined prediction confidence and decide whether to updates the semi-supervised activity model with the predicted activity label or trigger a question to the user.

2.1.1 Sensor and context data

In the following, we focus on how we have acquired the input data of our architecture and explain their characteristics.
Our system exploits both inertial sensors data and context information provided from mobile devices like smartphones and smartwatches. The inertial sensors' data are used to make a statistical prediction over the possible user's performed activity. The context information allows refining the statistical prediction excluding the not context-consistent activities.
In our setup, each user has been equipped with a smartphone and a smartwatch in order to collect data required for the activity recognition task. More precisely, the users carry a smartphone in the pants' front pocket and a smartwatch on the dominant hand's wrist.
The user's mobile devices acquire data from different sources. On one hand, inertial sensors (e.g., accelerometer, magnetometer and gyroscope) are in charge of continuously stream data about physical movements of the user. On the other hand, data from other built-in sensors and devices (e.g., GPS) in combination with publicity available web services (e.g., weather service) are used to obtain data about user's context. Finally, the data collected take two different paths: the stream of inertial sensors data are sent to the Incremental activity recognition module, while the context information is provided to the semantic refinement module.

Example 2.1.1 *Consider a user with a smartphone placed in pants' front pocket and a smartwatch on his hand's wrist which is riding a bike on a sunny Sunday morning at the park. Inertial sensors of both considered devices collect the user's movements due to ride a bike: the sensors of the smartphone measure linear acceleration, angular velocity, and orientation changes of the user's leg; while the smartwatch's ones get the same type of measurements but about the user's arms when he/she moves the handlebars. At the same time a lot of contextual data are collected and processed: from the GPS position and reverse geocoding's services is understood that the user is outdoors in a park, from weather forecast web API which is a sunny day, from the mobile* *devices' clock which is morning, from the sequence of the GPS positions that the user is moving with a certain speed, etcetera.*

2.1.2 Incremental activity recognition module

This module is in charge of analysing and manipulate only the inertial sensor data provided by the mobile devices with the aim to extract their features and supply them as input to a semi-supervised classifier. The INCREMENTAL ACTIVITY RECOGNITION module does not take into account context data because semi-supervised methods rely on a rather small set of labelled data, while activities can be performed in a wide variety of different contexts. While it is feasible for a classifier to discriminate different motion patterns even with few labelled samples, learning their correlations with all the possible context conditions may be problematic. This is especially true when considering a wide set of activities.

Generally, the inertial sensors stream provided by the inertial sensors is usually affected by noise. Hence, before to apply them any manipulation, they need to be cleaned to reduce noise and preserve only the informative part. Since to collect data we used different devices (smartphone and smartwatch), measurements providing by the different sources are aggregated in a unique data stream by using their timestamp to temporally align them. Then, the aligned data stream is divided into time windows of n seconds each.
Ended the segmentation process, for each segment we computed its statistical features with the aim of extracting the useful inertial signals information that characterises each activity. Features segments are hence provided as input to the classification module. This module, relying on semi-supervised classifier, produces for each features vector a probability distribution over the possible activities. It is important to point out that the activity model, before being performed any prediction, needs to be initialised with a small labelled training set in an offline phase.
The activity model is also trained run-time using a mixed approach between active-learning and self-learning: each system prediction is evaluated by the Prediction confidence module (Section 2.1.4) that decide whether to provide to the classifier the segment label directly using the predicted one or asking it to the user. In our architecture, the semi-supervised activity model is stored on a server and shared between all the participating users, which can collaboratively update it using our context-driven semi-supervised framework.

2.1.3 Semantic refinement module

Before to start to explain this module, it is important to note that "*context*" is a very broad term which in the literature is used to model users situations at several levels of abstractions [7]. For instance, even the performed activity can be considered as high-level contextual information.
In this thesis, with context data we mainly indicate the information about the environment which surrounds the user. Examples of such context data are user's current semantic location, his/her proximity to transportation routes, the current weather, the time of the day, the day of the week, etcetera.

The SEMANTIC REFINEMENT MODULE applies knowledge-based reasoning to context data in order to exclude from the statistical semi-supervised prediction those activities which are not consistent according to the current context. In particular, context information collected from mobile devices, before being used for the context refinement, are pre-processed and translated into high-level facts. Those facts are then provided to an ontology which models activities, contexts and the relationship that exists between them. Knowledge-based reasoning is then applied to evaluate which activities are *context-consistent*.
Finally, the activity probability distribution provided by the INCREMENTAL ACTIVITY RECOGNITION MODULE is refined excluding the not context-consistent activities. Since some activities may have been deleted from the probability vector, it is re-normalised before being sent to the PREDICTION CONFIDENCE EVALUATION module.

Example 2.1.2 *Suppose a user who is using our system is driving a car. Inertial sensor data from smartwatch and smartphone are sent to the* INCREMENTAL ACTIVITY RECOGNITION *module that produces a statistical prediction over the possible activities. In this case we suppose that the classifier created the following probability distribution: sitting: 50%, moving by car: 45%, cycling: 5%. The* SEMANTIC REFINEMENT *module received as context information that the user is moving with a certain speed. This causes the exclusion from the prediction list of the sitting activity that is defined in the ontology as static ones. The refined prediction list is hence normalised updating the prediction confidences of each activity, obtaining the following probability vector: moving by car: 90%, cycling: 10%.*

2.1.4 Prediction confidence evaluation module

The third and the last step of our method consists of using the refined prediction to update the semi-supervised activity model.
In particular, the Prediction Confidence Evaluation module evaluates the system confidence on the refined prediction deciding if exploit a self-learning or an active-learning approach to update the recognition model. Whether the predicted activity label confidence is upper that specific threshold ρ, it is provided as a new labelled example to the incremental classifier (self-learning). Otherwise, whether the activity label confidence is lower than another threshold π, it is triggered a query to the user in order to obtain the ground truth about the current activity label and hence update the learning model accordingly (active-learning).

Example 2.1.3 *In the Example 2.1.2 we obtained this refined and normalised probability vector: moving by car: 90%, cycling: 10%. Moreover we suppose to set the threshold value* $\rho = 80\%$.
The Prediction confidence evaluation *extracts the most confident prediction (moving by car) and then evaluate it. Here, the predicted label's confidence is upper than the* ρ *threshold. The* Prediction confidence evaluation *module hence use the predicted label to update the activity model using the self-learning approach.*

Chapter 3

Technical solution

In this chapter, we present the technical solutions used to realise each module of the architecture illustrated before. At first, in Section 3.1, we describe how raw data captured by the inertial sensors are pre-processed by the INCREMENTAL ACTIVITY RECOGNITION module to clean up them to noise.
After that, the cleaned data are temporally aligned and aggregated in segments of n seconds. Then, from each inertial sensors data segments, are computed their characteristic features vector. Finally, an online incremental classifier produces a prediction for each features vector over the possible activities.
At the same time, as described in Section 3.2, context data are processed by the SEMANTIC REFINEMENT MODULE that incorporates an Ontology able to models the relationships between context and activities. Then, the context-reasoning basing on the ontology excludes activities which are unlikely considering the current context.
The refined prediction is then processed by the PREDICTION CONFIDENCE EVALUATION module (Section 3.3) that decides whether to update the learning model with the predicted label or ask it to the user. In particular, if the prediction confidence is below a certain threshold trigger a query to the user. Otherwise, if the prediction confidence is upper to another threshold the predicted label is considered correct.

3.1 Incremental activity recognition

In the following, we describe the inertial sensors data flow of the activity recognition process. As shown in Figure 3.1, raw sensor data, before being used by the classifier, are been pre-processed.
More precisely the pre-processing phase consists of the following steps:

- *data cleaning*: extracting from the sensor's data the informative part excluding the noise (Section 3.1.1).

- *smartphone orientation adjustment*: finding the device orientation and adjust it in case of unaligned measurements with the device coordinate system (Section 3.1.2).

- *data aggregation and Segmentation*: temporally align sensors data streams and divide them into n seconds segments (Section 3.1.3).

- extracting specific features from each n seconds segment (Section: 3.1.4)

Once the pre-processing phase has been completed, before proceeding with the classification, the activity model needs to be trained with x examples for each activity included in the dataset. This process is called *bootstrap* and it is described in Section 3.1.5.
Now, features segments are finally ready to be classified. In Section 3.1.6we explain how our incremental classifier produces an activity probability distribution for each features segment.

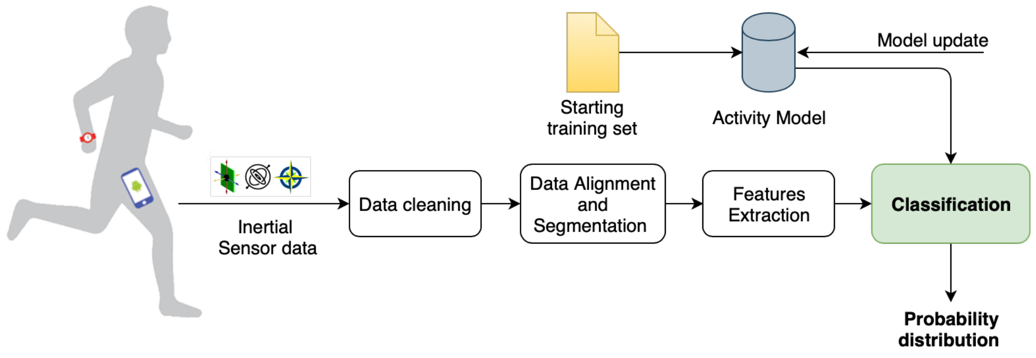


Figure 3.1: Scheme of the Incremental Activity Recognition phase.

3.1.1 Data cleaning

Inertial sensors data may be affected by noise. It is hence necessary to extract the informative component from the signal in order to create a more accurate recognition model.
In particular, we apply a median filter on each axis of each inertial sensor. For each value v_i in the signal, we consider its w neighbours:

$$n(v_i, w) = [v_{i-\frac{w-1}{2}}, \ldots, v_i, \ldots, v_{i+\frac{w-1}{2}}]$$

For each v_i in the signal, we replace its original value with the median value computed on $n(v_i, w)$. An example of the median filter applied to the signal on the x-axis of an accelerometer is shown in Figure 3.2.

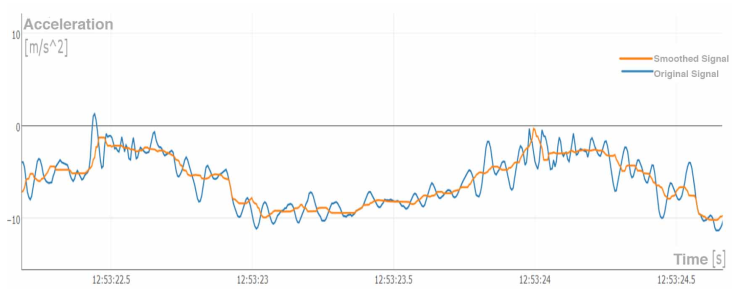


Figure 3.2: Example of median filter application on an accelerometer signal. $w = 9$.

The blue line represents the raw data signal, while the orange one is the result of applying the median filter.
As it easy to see, the general trend of the signal is maintained, while the number of peaks (which likely reflects the signal noise) is considerably reduced.
It is important to note that the w parameter highly impacts the smoothing of the original signal and hence the recognition rate.
A high value of w would excessively smooth the signal, hence removing important signal characteristics is useful to discriminate activities. On the other hand, a low value of w would maintain the intrinsic noise of sensor data. It is hence important to select a value of w which maximises the classifier's accuracy.

3.1.2 Smartphone orientation adjustment

For the sake of this thesis, we assume that the user carries the smartphone in a pocket. As Figure 3.3 shows, the smartphone in the pocket has two main degrees of freedom: the top may be upward (U) or downward (D), while the screen may be outward (O) or inward (I).

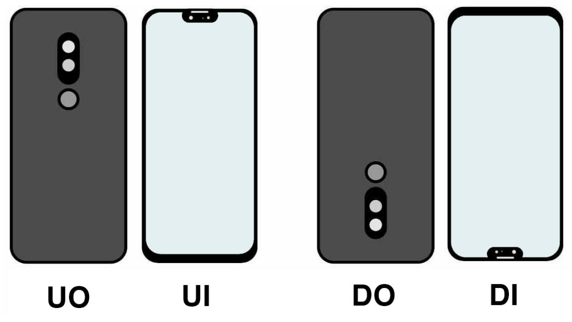


Figure 3.3: The smartphone's orientations considered in this work. UO = Top upward and screen outward, UI = Top upward and screen inward, DO = Top downward and screen outward, DI = Top downward and screen inward.

Figure 3.4 shows the coordinate system (i.e., x, y and z axis) with respect to the "*standard*" smartphone's orientation (i.e., top upward and screen outward).

We performed some preliminary experiments to evaluate the impact of the

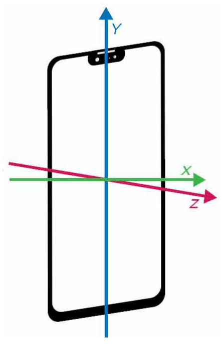


Figure 3.4: Smart-phone's coordinate system.

smartphone's orientations on the recognition rate. We determined that the variations with respect to the y-axis (i.e., whether the top is upward or downward) have a high impact on the classifier's accuracy. On the other hand, the orientation with respect to the x and z axes (i.e., whether the top is outward or inward) does not have a significant impact on activity recognition. Hence, in order to build a robust recognition model, we make uniform the y-axes of each inertial sensors. In particular, when the values on the y-axis of the magnetometer (which reveals the smartphone orientation) are negative, we invert the sign of the y-axis of every considered inertial sensor. The only device affected by this problem is the smartphone because the smartwatch is worn on the wrist in a static position.

3.1.3 Data aggregation and segmentation

Since a user may carry multiple mobile devices (e.g., a smartphone and a smartwatch), it is first necessary to temporally align their raw inertial sensors' data streams. The raw sensors' data consist of measurement for each axis (x, y, z) and a timestamp expressed in milliseconds. This timestamp allows to uniquely place the data in time and therefore it is used to temporally align measurements providing from different device sensors. Once the alignment phase is completed, the segmentation is performed. Each segment is defined as the set of inertial sensor data acquired during a specific time window of n seconds. Each of them starts the next second with respect to the end of the previous segment, hence segments are contiguous and non-overlapping. The length n is the same for all segments, and it must be chosen carefully according to the complexity of the considered activities [5]. In fact, small window size will not contain enough data to characterise the activity, on the other hand, with too large windows, you may include data in the window of multiple activities, creating confusion in the classification. An example of the data aggregation and segmentation process is illustrated in Figure 3.5.

3.1.4 Features extraction and standardisation

For each segment, we extract a wide set of statistical features which are well-known in the activity recognition literature [27]. In particular, for each axis of each inertial sensor, we extract:

- **Average**: it is considered an estimator of the gravity component and makes it possible to distinguish easily when an activity is static or not.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

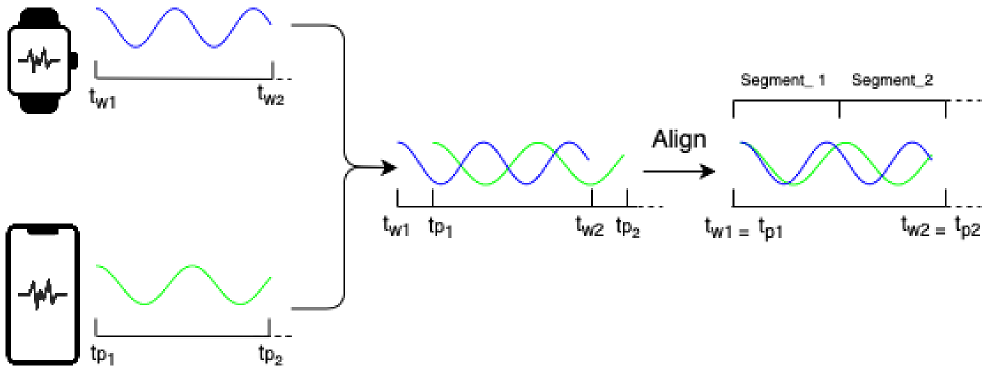


Figure 3.5: Data aggregation and alignment.

- **Difference between max and min**: it is computed as the difference between the maximum and minimum value in a n seconds window. It allows detecting the "step consistency" that is useful to discriminate between activities like running or walking.

- **Variance**: (σ^2): it provides a measurement of the signal variability, and enhance how the signal value departs from its average.

$$\sigma^2 = \sum_{i=1}^{n} \frac{(x_i - \mu)^2}{n}$$

- **Standard deviation**(σ) it is a static dispersion index of the signal values. It can give an indication of the stability of the signal.

$$\sigma = \sqrt{\sum_{i=1}^{n} \frac{(x_i - \mu)^2}{n}}$$

- **Median**: This feature is not often used to discriminate activity but is useful to replace possible signal missing values.

$$< x >= \sqrt{\sum_{i=1}^{n} \frac{(x_i - \mu)^2}{n}}$$

- **RMS (Mean Squared Error)**: it is widely used features in activity recognition tasks [27], and it is computed with the following formula:

$$RMS = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i)^2}$$

- **Symmetry**: it is a distribution index that allows assigning a symmetry weight to each value in the n seconds segment in reference to the axes (x, y, x). A zero value indicates symmetry, others value define how the signal departs from the symmetry.

- **Kurtosis (γ^2)**: This value is calculated using the Pearson index $\beta_2 = \frac{m_4}{m_2^2}$, that is the ratio between the centred moment of order 4 and the square of the centred moment of order 2.

$$\gamma^2 = \beta_2 - 3$$

- **Zero-crossing rate**: it is the frequency with which the signal cross zero along specific axes. This feature is used to recognise activities such as walking, running or going upstairs/downstairs etc. , that, in order to be carried out, need an oscillatory and repetitive movement of the legs.

- **Number of peaks**: it counts the number of Maximum and minimum signal pitches. It is useful to discriminate between dynamical and static activities(i.e., the movements of the limbs needed to run produces more peaks than sitting).

- **Energy**: this features is calculated applying Function 3.1.4 for each measurement:

$$E = \sum_{n=-\infty}^{\infty} \mid x(n) \mid^2$$

Then, for each inertial sensor we compute the *Pearson correlation* for each combination of its axes.

- **Pearson correlation**:It is used to measure the linear correlation between two signals and indicates the tendency of a signal to change in relation to the changes of a second one. This feature can be calculated in different ways, but the most popular method is the*Pearson index* ρ. It is computed as the ratio between co-variance of the signal along x and y axis and the product of their standard deviations.

$$\rho_(x, y) = \frac{cov(x, y)}{\sigma_x \sigma_y}$$

Finally, for each inertial sensor we compute the *magnitude* on all of its axes.

- **Magnitude**: this feature allows detecting the strong changes in the performance of an activity. A significant example could detect a sudden interruption during a run due to a user falling.

$$magnitude = \frac{1}{n} \sum_{i=1}^{n} \sqrt{x^2 + y^2 + z^2}$$

Hence, given k 3-axis inertial sensors equipped in the user's mobile devices, we compute $k \times 37$ features.

Example 3.1.1 *Consider a user which carries a smartphone and a smartwatch, both equipped with 3-axis accelerometer, gyroscope and magnetometer. Hence, the overall number of inertial sensors is* 6. *In this scenario, our feature extraction mechanism would compute, for each segment,* $6 \times 37 = 222$ *features.*

In the end, we also applied **standardisation** to each feature in order to further improve the recognition rate [14].
More precisely, the aim of standardization process is to transform data to make them as similar as possible to a Gaussian distribution with zero average and unit variance.
This operation allows to avoid that some features, characterised by a high variance, dominates the classifier learning function compromising the other features contribution.
For each feature X we computed standardization as follow:

$$\bar{X} = \frac{X - \mu}{\sigma}$$

Where μ indicate the average value of the feature X, σ is the feature variance and $\bar{X}$ is the result of the standardisation process applied on X.

3.1.5 Activity model bootstrap

A crucial aspect of our semi-supervised framework is the activity model initialisation. Indeed, without a proper bootstrap mechanism, the semi-supervised model would have to discover each activity "on-the-fly", with a negative impact on the recognition rate. Hence, we initialise the semi-supervised model acquiring t seconds of labeled data for each activity in order to obtain a balanced labeled dataset. Clearly, the parameter t has a high impact both on the recognition accuracy and on the number of queries triggered to the users. However, as we motivated in the introduction, it is unfeasible to obtain a wide labeled dataset (i.e., choosing a high value of t). The choice of t mainly depends on the number of considered activities and their complexity. In order to reduce the effort of acquiring and annotating data, it is hence important to use a small value of t which allows achieving a reasonable recognition accuracy. We adopt an empirical approach to determine the value of t.

3.1.6 Classification

In our solution, we decided to use a machine learning algorithm to perform the activities classification process. More precisely, we exploit an *online incremental classifier* to make predictions over the possible activities. This type of classifier allows to build and refine the model as new data arrive at different points in time. Our updating model implementation is explained in Section 3.3.

For each feature vector fv, computed from a segment s, the incremental classifier h outputs a probability distribution over the set of considered activities $\mathbf{A} = \{A_1, A_2, \ldots, A_m\}$:

$$h(fv) = \langle p_1, p_2, \ldots, p_m \rangle$$

where $0 \leq p_i \leq 1$ is the probability $P(A_i|s)$ that s was generated by the activity A_i, $\sum_i^m p_i = 1$, and $m = |\mathbf{A}|$.

The probability distribution $h(fv)$ is hence the list of activity recognised by the classifier, each one associated with specific confidence. This result is not directly used as our system output but forwarded to our Semantic Refinement module which will refine it using context information.

3.2 Semantic refinement

The Semantic Refinement module is in charge of analysing the context which surrounds the user to refine the prediction $h(fv)$ obtained by the Incremental Activity Recognition module. In order to achieve this task, this module relies on an ontology which models the relationships between contexts and activities. In particular, ontological reasoning is applied to exclude from the statistical prediction those activities which are unlikely considering the current context.
In the following, we describe in details our context-aware semantic reasoning mechanism (see Figure 3.6).

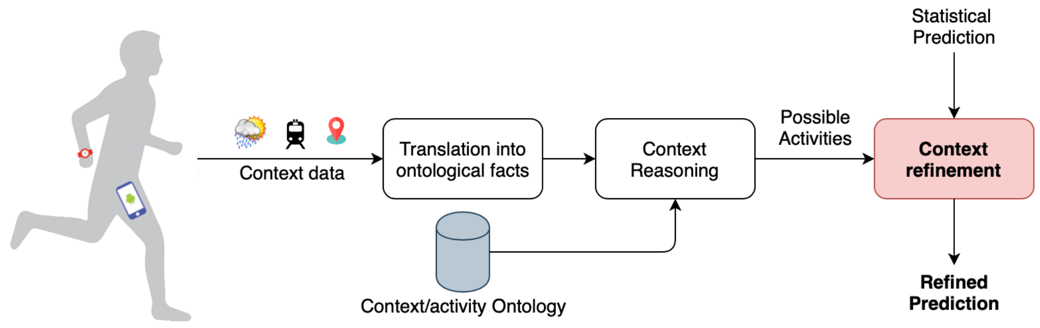


Figure 3.6: Scheme of the semantic refinement module.

3.2.1 Activities translation into ontological facts

Before introducing this section, it is important to point out that the purpose of this thesis in not been to model a new ontology but it was to embed an existing one in a semi-supervised activity recognition system.
In order to enable semantic reasoning we used an extended version of the *ActivO* ontology [33] developed in previous works of EW Lab.
This ontology defines a wide set of activities, semantic locations, artifacts (e.g., used by the user or part of the semantic locations), user's postures, time granularities (e.g., the day of the week, time of the day) and environmental information (e.g., temperature and light conditions).
Details about the original *ActiveO*'s implementation can be found in [33]. The extended version has taken advantage of Protégé [1] to expand *ActivO*

[1]`https://protege.stanford.edu/`

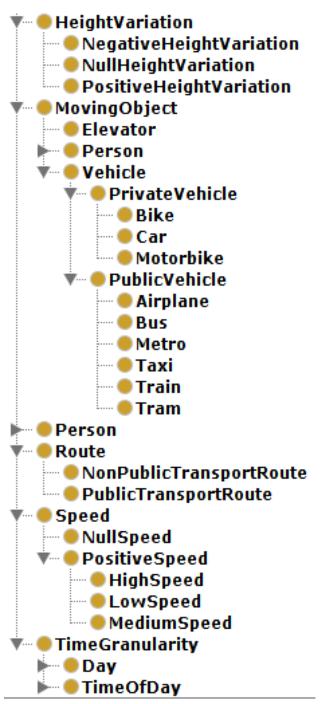
with several new activities, contextual data and their relationships. An example of those entities are shown in Figure 3.7. Examples of such entities are:

- `HeightVariation`: it models height variations of the user. The subclasses are `PositiveHeightVariation`, `NegativeHeightVariation` and `NullHeightVariation`.
- `Route`: it models the current route which the user is traveling (if any). A subclass of this entity is `PublicTransportationRoute`, which defines that the user is traveling along the route of a public transport.
- `Speed`: it models the current speed of the user. The subclasses are `PositiveSpeed`, `NegativeSpeed` and `NullSpeed`.

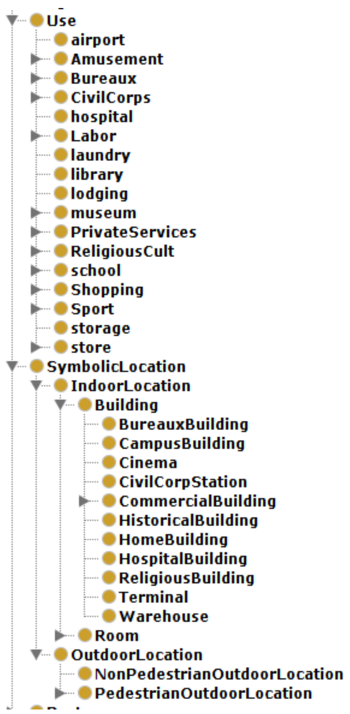
The Ontology considers several sources of context data: *user's semantic place*, *user's recent route*, *weather conditions*, *proximity to public transportation stops and routes*, *surrounding traffic condition*, *user's height variations*, *user's speed*, *surrounding light*, *environment's noise level* and *temporal context (e.g., time of the day, day of the week, month, ...)*.
Figure 3.7a shows a portion of those context data modelled in our ontology, while Figure 3.7b focuses on the set of considered semantic locations, including the ones classified by Google Places API [2]. It is important to note that are be distinguished symbolic locations/buildings (and their characteristics) from their use. This allowed to better model activities in terms of symbolic locations.

Due to the intrinsic open-world assumption of ontological reasoning, the necessary conditions have been explicitly stated in the used Ontology to make the activities possible or impossible.
As we will explain later, such constraints are necessary to enable our context-aware refinement which relies on *consistency* reasoning. For instance, the activity `GoingStairs` (Figure 3.8a) should take place at a location which may have stairs and the person should have a non-negative height variation. Another example is the activity `MovingByCar` (Figure 3.8b): The used ontology enforces that it should take place in an outdoor location which includes a road or a street and that the car's speed should be positive.

[2]`https://developers.google.com/places/supported_types`



(a) An excerpt of context hierarchy



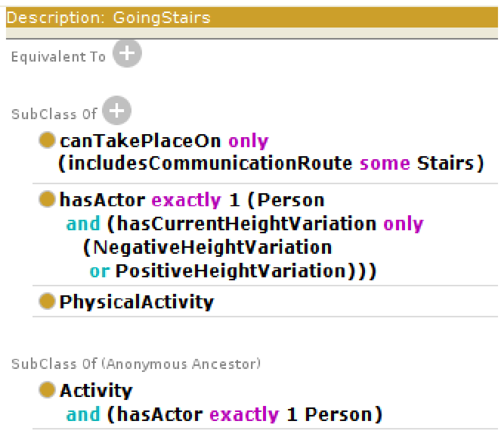
(b) An excerpt of symbolic locations hierarchy

Figure 3.7: Excerpts of our ontology.

3.2.2 Context reasoning and refinement

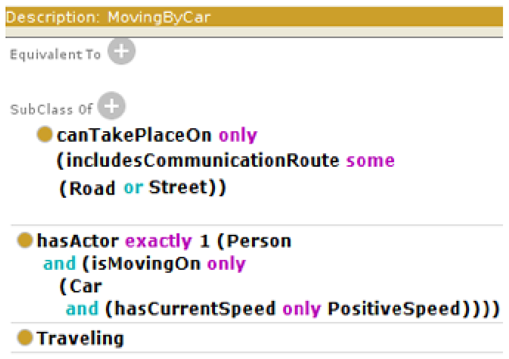
The Semantic Refinement module is in charge of analyzing the context which surrounds the user to refine the prediction $h(fv)$ obtained by the Incremental Activity Recognition module.
In order to achieve this task, this module relies on the ontology introduced in Section 3.2.1 that models the relationships between contexts and activities. In particular, ontological reasoning is applied to exclude from the statistical prediction those activities that are unlikely considering the current context. In the following, we describe in details our context-aware semantic reasoning mechanism.

Example 3.2.1 *Suppose that Bob is using our system. When the context-aware reasoning task is triggered, an individual* `Person(Bob)` *is added to the ABox. Then, the context data gathered by the mobile devices is analysed*



```
Description: GoingStairs
Equivalent To
SubClass Of
  canTakePlaceOn only
   (includesCommunicationRoute some Stairs)
  hasActor exactly 1 (Person
    and (hasCurrentHeightVariation only
      (NegativeHeightVariation
       or PositiveHeightVariation)))
  PhysicalActivity
SubClass Of (Anonymous Ancestor)
  Activity
    and (hasActor exactly 1 Person)
```

(a) Definition of the activity "going stairs"



```
Description: MovingByCar
Equivalent To
SubClass Of
  canTakePlaceOn only
   (includesCommunicationRoute some
    (Road or Street))
  hasActor exactly 1 (Person
    and (isMovingOn only
     (Car
      and (hasCurrentSpeed only PositiveSpeed))))
  Traveling
```

(b) Definition of the activity "moving by car"

Figure 3.8: Examples of activity definitions in our ontology.

in order to expand the ABox. Suppose that some web services provided the information that Bob is in a park and that the speed value obtained by the GPS sensor is 11 *km/h. First, we have to instantiate the individuals for the context data:* `Park(place)`, `MediumSpeed(speed)`. *Note that the raw speed value obtained by the GPS has been discretized in order to be mapped to an ontological concept. Then, the relationships between Bob and context data are added to the ABox:* `hasCurrentSymbolicLocation(Bob, place)`, `hasCurrentSpeed(Bob, speed)`. *Finally, in order to test whether the activity* `Running` *is context-consistent, we simply add to the ABox the axioms* `Running(currentActivity)` *and* `isPerforming(Bob, currentActivity)`. *The consistency of the ABox with respect to the TBox will determine if the running activity is consistent according to the current Bob's context.*

Hence, given the current context C and the marginal probabilities obtained by the semi-supervised classifier $h(fv) = \langle p_1, p_2, \ldots, p_m \rangle$, the goal of context refinement is to exclude those activities that are not *context-consistent* according to C.
For each activity class ac_i such that $p_i > 0$, we compute its consistency according to context C as explained above. Each activity which is not *context-consistent* is removed from the probability vector. The refined vector is finally normalized in order to preserve the properties of a probability distribution.

The output is a new refined probability vector $\langle r_1, r_2, \dots, r_c \rangle$ such that each A_i is a *context-consistent* activity according to C, $0 \leq r_i \leq 1$ and $\sum_i^m r_i = 1$. Note that an activity is usually not *context-consistent* when ontology's necessary constraints are violated.

Example 3.2.2 *Suppose that a user is going up with the elevator. According to the* Incremental Activity Recognition *classifier, the current probability distribution is* 45% *standing,* 40% *elevator up,* 10% *walking and* 5% *going upstairs. Thanks to the barometer sensor, it is possible to deduce that the person currently had a positive height variation. According to the ontology, standing is not context-consistent since it should not involve height variations. Hence, the resulting context-refined probability distribution is* 73% *elevator up,* 18% *walking and* 9% *going upstairs.*

3.3 Prediction confidence evaluation

As we introduce in Section 3.3.1, our system relies on a hybrid active-learning/self-learning activity model update system. More precisely, the refined prediction given by the semantic refinement module is analysed by the *Prediction confidence evaluation* module that is in charge to determine how to update the model. This decision process implementation relies on two thresholds that define when the prediction confidence value is sufficient (or not) to consider the prediction reliable.
Since in everyday life we usually perform some activities more often than others, we have also implemented an incremental data balancing method. It is described in Section 3.3.2 and allows to decrease the unbalancing on the number of activity examples given to train in run-rime the semi-supervised model.

3.3.1 Semi-supervised model update

In order to update the activity model, we apply an uncertainty sampling strategy to identify in real-time the confidence level of each refined prediction [30].
Given a context-refined prediction $\langle r_1, r_2, \dots, r_c \rangle$, we denote $r^\star = \max_i r_i$ as the probability value of the most likely activity.
If $r^\star$ exceeds a threshold π, we consider the system very confident on the

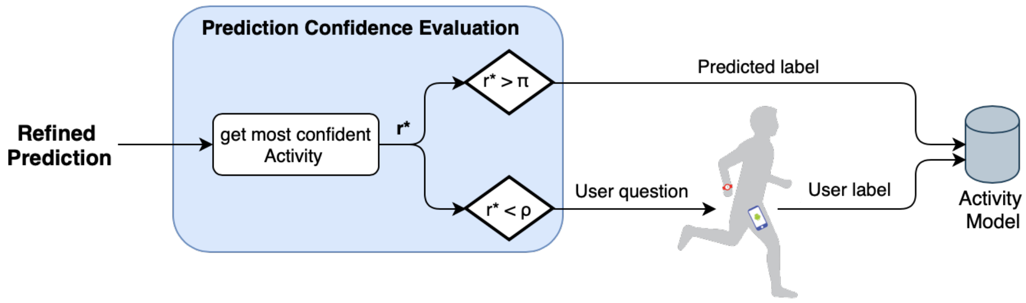


Figure 3.9: Scheme of the prediction confidence evaluation.

current classification and we update the semi-supervised activity model with a new labeled example (self-learning). Otherwise, if $r^\star$ is below a threshold ρ, where $\rho < \pi$, we consider the system uncertain about the current activity being performed by the user. In this case, an active learning process is started by asking the user to provide the ground truth about the current activity (active learning), in order to update the model accordingly. When $\rho < r^\star < \pi$ we do not consider the current prediction to update the semi-supervised activity model. How we selected these threshold values is described in Section 4.3. It is also important to remember that the user's answer does not change the classifier prediction, but it is only used to label the features segment and hence update run-time the learning model. The entire decisions process of the semi-supervised model update, it is illustrated in Figure 3.9.

3.3.2 Incremental data balancing

During everyday life, some activities are performed on average with a lower frequency than others (e.g., the amount of time that a subject spends on an elevator is usually less than the time he/she spends walking). Hence, adding new labeled samples to the incremental classifier without taking in account this aspect may lead to an unbalanced classification model and subsequently to a poor recognition rate on the "minority" activity classes.
For this reason, our ontology also describes (using OWL2 properties) which activity classes are known to be "minority" according to common-sense knowledge. We define M as the set of "minority" activity classes according to the ontology. We adopt the well-known SMOTE technique [9] in real-time to balance the activity model by generating synthetic samples of "minority" activity classes.

In particular, when a segment S labeled as $ac \in M$ is provided as a new labeled example to update the model, we create q additional synthetic labeled samples using SMOTE to further improve the classifier. To perform SMOTE oversampling on segment S labeled as $ac \in M$, we start considering its k nearest neighbors. Then, we take the features segment S_n between one of those k neighbor and S. We hence multiply S_n by a random number x which lies between 0 and 1. Finally, we add the resulting features vector to the current segment S obtaining the new synthetic segment.

Chapter 4

Experimental evaluation

In order to evaluate our system, we exploit an existing dataset collected in previous work in EW lab that contains both labeled inertial sensors data and context data of various activities. Details of how these data were collected and how is structured the resulting dataset are introduced in Section 4.1. Then the chapter continues with Section 4.2 that consist of an overview of the different metrics used to evaluate our method.
Finally, in Section 4.3, we propose the empirical results obtained by the proposed semi-supervised and context-aware activity recognition framework. Furthermore, we also implemented two additional methods which do not rely on the SEMANTIC REFINEMENT module. The former is called *No context*, since it considers only inertial sensor data to recognise activities. The second is called *Context as features* because it incorporates context data directly in the feature vectors generated from the inertial sensors data, without using any semantic Refinement technique. To evaluate and compare these methods in terms of recognition rate and number of questions, we used a *leave-one-subject-out* cross-validation approach. The results showed that context data (whether it is used as features or not) significantly impacts the overall recognition rate. It is also clear that our method, thanks to the Semantic Refinement module, allows triggering most frequently our self-learning technique to update the activity model asking much fewer questions than the other two methods.

4.1 Setup for data collection and the dataset

In the following, we describe the experimental configuration used in previous works of the EW lab that allowed to obtain the activity data exploited as input to our system.
In this setup, the users carried a smartphone in the pants' front pocket and a smartwatch on the dominant hand's wrist.
More specifically, they used a Nexus 5x as smartphone and an LG G Watch R as smartwatch. Dedicated applications have run on the devices to continuously collect sensor measurements from their accelerometers, gyroscopes and magnetometers. Context data were acquired by the smartphone application considering built-in sensors as well as publicly available web services.

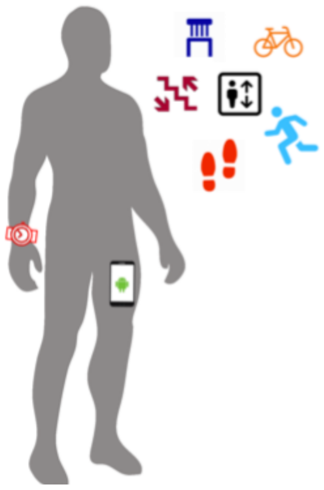


Figure 4.1: Device placement on user body: smartphone is placed in the front pants pocket, Smartphone on dominant hand wrist.

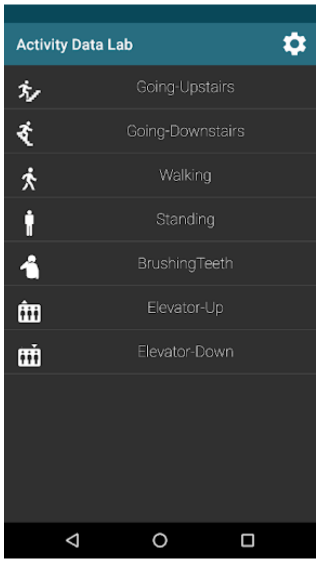
The considered built-in sensors were: the barometer to get insights about height variations, the luminosity sensor, the microphone to obtain the environment's noise level and the GPS to obtain the user's location and speed.
The considered web services were the following:

- **Google's Places API** [1]: to obtain the most likely semantic places where the user is performing the current activity.

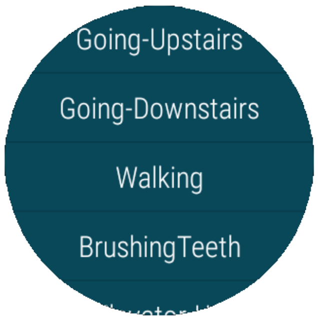
[1]`https://developers.google.com/places/web-service/intro`

- **OpenWeatherMap** [2]: to obtain the climatic conditions (e.g., sunny, cloudy, rainy), temperature, wind speed, etcetera.
- **Bing's Traffic API** [3]: to obtain nearby traffic situation like road conditions, presence of road works, presence of car accidents, etcetera.
- **Transitland**[4]: to obtain information about transportation routes and stops close to the user.

The application has also collected temporal context like: the moment of the day (e.g., morning, afternoon, evening), the day of the week, the season, etcetera. Then, every 5 seconds, context was transmitted to the server.



(a) Smartphone interface



(b) Smartwatch interface

Figure 4.2: Annotation interfaces.

Besides data acquisition, mobile applications allowed the users to annotate data in real-time. Examples of such interfaces are shown in Figure 4.2. The monitored activities were the following: *walking*, *running*, *standing*, *lying*, *sitting*, *stairs up*, *stairs down*, *elevator up*, *elevator down*, *cycling*, *moving*

[2]`https://openweathermap.org/`
[3]`https://docs.microsoft.com/en-us/bingmaps/rest-services/traffic/`
[4]`https://transit.land/`

by car, *sitting transport*, *standing transport* and *brushing teeth*. Overall were recorded almost 9 hours of labeled sensor data ($\sim$ 350 activity instances) from 26 volunteers aged between 20 and 28.
Table 4.1 summarises how many minutes of data were acquired for each activity.

Activity	**Minutes**
Standing	52
Sitting	96
Walking	96
Running	24
Cycling	24
Brushing teeth	16
Stairs up	16
Stairs down	16
Elevator up	8
Elevator down	16
Sitting transport	60
Standing transport	60
Moving by car	40
Overall	524

Table 4.1: Number of minutes acquired for each activity.

As it easy to see, the dataset is unbalanced. Indeed, activities like *taking the elevator* or *brushing teeth* have been executed for a significantly shorter time than others like walking. The annotated sensors data have been acquired in different contexts, which include being at the office, going around in the city (Milan), driving, using public transportations, cycling and being at home.

4.2 Metrics

Despite our system is semi-supervised, in order to measure its effectiveness we exploited a labelled dataset. Indeed, for each prediction of our classifier, we have used his ground truth to evaluate it.
For each activity we calculate the number of *true positive* (TP), *false negative* (FN) and *true negative* (TP). TP counts the number of times a predicted activity is equal to the ground truth. FP indicates the number of cases when

an activity is predicted without being performed (i.e. a FP for the activity A_1 occurs when the classifier predicts the activity A_1 but the ground truth is A_2).
Finally, FN counts cases in which the effective activity was miss-predicted (i.e. a FN for the activity A_1 occurs when the classifier predicts the activity A_2 but the ground truth is A_1).
Our classifier's performances on each different type of activity are hence computed using the following metrics:

- Precision: measures how precise/accurate is the model in respect to the positive prediction, providing a percentage about how many of them are actually positive

$$Precision = \frac{VP}{TP + FP}$$

- Recall: calculates how many of the actual positives our model captures through labelling them as positive (True Positive)

$$Recall = \frac{VP}{TP + FN}$$

- F1 score: provides summary information on the quality of the learning model, combining the two previous metrics.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Figure 4.3 shows a graphical representation of *Precision* and *Recall*.

4.3 Results

In the following, we show the results of our method considering the dataset described above in Section 4.1. As incremental classifier, we use Online Random Forest [34] taking advantage of the Java implementation proposed in [38]. The motivation is that Online Random Forest is the incremental version of the well-known classifier Random Forest, which proved to be one of the most effective classifiers for activity recognition [37]. As OWL2 reasoner we used HermiT [12] in combination with the Java OWL APIs [21].

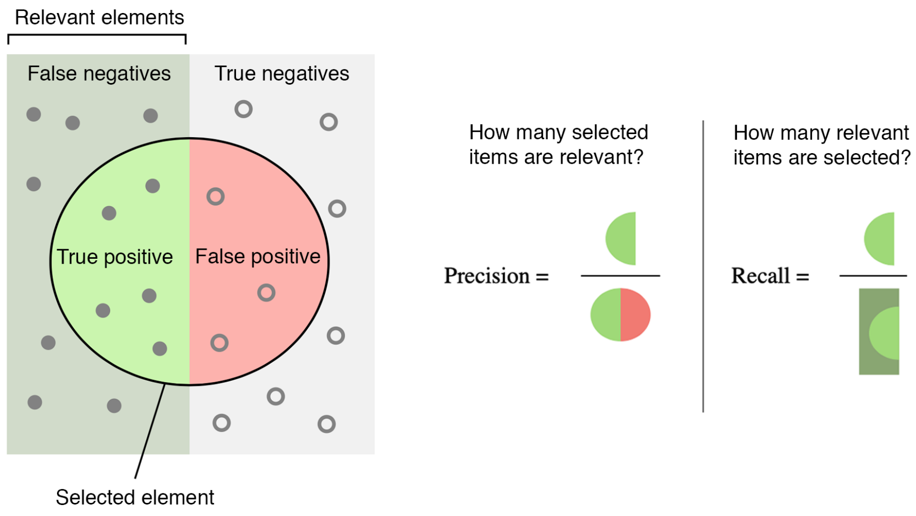


Figure 4.3: Precision and Recall.

In order to evaluate the effectiveness of our technique, we also implemented two additional methods that do not rely on the SEMANTIC REFINEMENT module. The former is called *No context*, since it considers only inertial sensor data to recognise activities.
In particular, it combines the INCREMENTAL ACTIVITY RECOGNITION module (see Section 3.1) and the PREDICTION CONFIDENCE EVALUATION module (see Section 3.3) without applying our context-refinement method.
The latter is called *Context as features*. Similarly to *No context*, this method does not rely on the SEMANTIC REFINEMENT module to refine activity predictions. However, this method incorporates context data directly in the feature vectors generated by the feature extraction mechanism presented in Section 3.1.4. In particular, this methods extracts a) statistical features (average, variance, difference between max and min) from *numeric* context data like speed or height variations and b) binary features for *symbolic* context data (i.e., semantic place, weather condition, proximity to transportation routes, etc.).
We used a *leave-one-subject-out* cross-validation approach to evaluate and compare these methods in terms of recognition rate and number of questions. At each fold, we apply our context-aware semi-supervised method to 25 users to collaboratively update the activity model, which is initialised considering 1 minute of samples for each activity. The remaining user is finally used

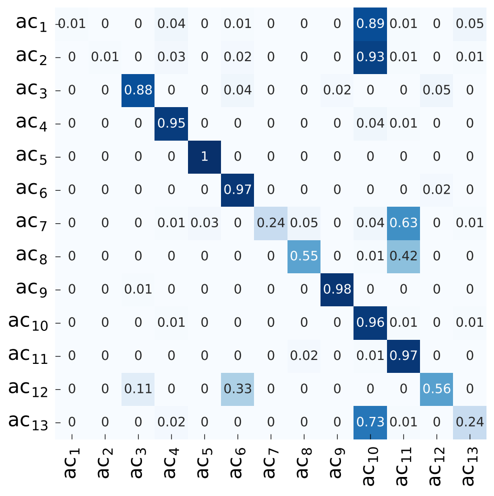
to evaluate the recognition rate and the number of questions of our method considering the resulting model. We have then empirically determined that the optimal window size is $w = 4$, while the thresholds for semi-supervised updates are $\pi = 0.7$ and $\rho = 0.45$ respectively.
The reasons for how we have empirically determined these specific threshold values are explained below in Subsection 4.3.
Table 4.2 shows the results (in terms of overall F1 score) of the considered methods. The results clearly show that context data (whether it is used as

Activity	**Without Context**	**Context as features**	**Our method**
Elevator up	0.0	0.04	**0.70**
Elevator down	0.02	0.71	**0.83**
Moving by car	**0.85**	**0.85**	0.74
Brushing teeth	0.87	0.91	**0.93**
Running	0.97	0.97	**0.99**
Sitting	0.96	**0.97**	**0.97**
Going upstairs	0.38	0.69	**0.76**
Going downstairs	0.65	0.87	**0.92**
Cycling	**0.97**	0.90	0.89
Standing	0.86	0.93	**0.95**
Walking	0.89	0.94	**0.95**
Sitting transport	0.60	0.78	**0.88**
Standing transport	0.36	**0.95**	**0.95**
Avg F1	0.64	0.81	**0.88**

Table 4.2: Recognition rate (F-1 score) of our method compared with alternative approaches.

features or not) significantly impacts the overall recognition rate.
Indeed, it is evident that activities like going upstairs/downstairs and sitting/standing transport (which are more difficult to recognise only considering motion patterns) highly benefits from context data. The positive impact of context in reducing confusion between activities is also depicted in Figure 4.4.

In general, our method outperforms the *Context as features* approach. Moreover, our approach is crucial to make the classifier able to recognise *elevator up* activity. This is due to the fact that statistical methods confuse this activity with *standing*, since they have very similar motion patterns.


(a) Without context

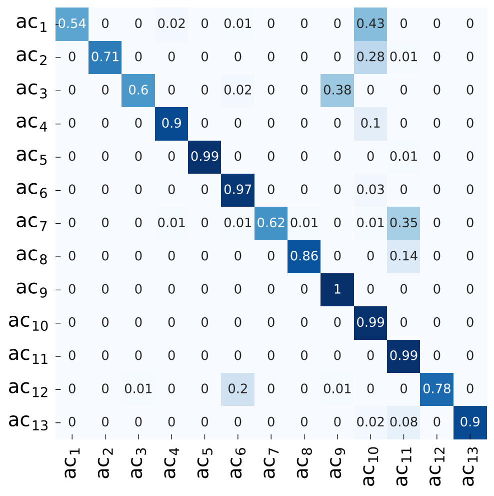

(b) Our method

Figure 4.4: Comparison of confusion matrices. ac_1 = Elevator Up, ac_2 = Elevator Down, ac_3 = Moving by Car, ac_4 = Brushing Teeth, ac_5 = Running, ac_6 = Sitting, ac_7 = Going upstairs, ac_8 = Going Downstairs, ac_9 = Cycling, ac_{10} = Standing, ac_{11} = Walking, ac_{12} = Sitting Transport, ac_{13} = Standing Transport.

However, we point out that the recognition rate of our method on the *moving by car* activity is lower than the ones obtained by the other approaches. Indeed, how Figure 4.4 shows, this activity is often confused by our method with *cycling*.
This is due to the fact that context data which characterises those activities are similar (e.g., they are both performed outdoor, with a variable speed, etc.). Hence, the semi-supervised model updates may propagate mis-predictions when the classifier has few examples of those activities.
Besides the recognition rate, the most important result is about the number of questions triggered by our method. In fact, as Figure 4.5 shows, our method generates a significantly lower number of questions (6%) with respect to *No context* (40%) and *Context as features* (35%).

Indeed, our semantic refinement technique exploits the ontology to remove from the prediction unlikely activities. This increases the confidence of the remaining activities and, at the same time, triggers our semi-supervised technique to update the activity model without bothering the user. Results

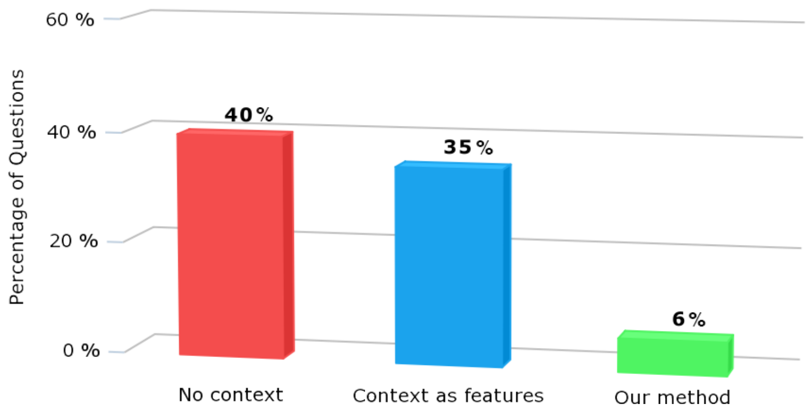
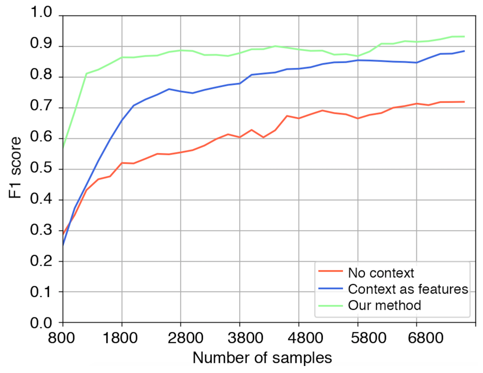


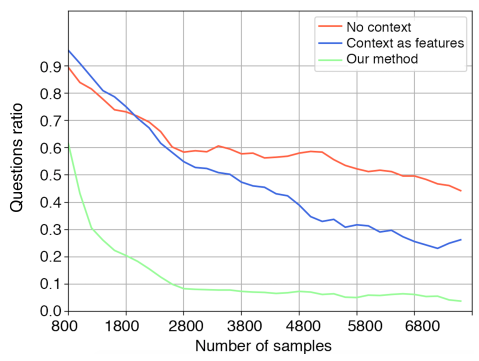
Figure 4.5: Percentage of triggered queries of our method compared with alternative approaches.

indicate that the resulting system should provide a much better user experience by limiting the number of times a user is interrupted with a question. Hence, our system is way more acceptable for a real-life deployment.
In order to evaluate how the recognition rate and the number of triggered questions evolve over time, we used the method proposed in [11]. First, we initialise the model as described above, considering 1 minute of samples for each activity. Then, for each sample of the dataset (considering all 26 subjects), we use the current recognition model to classify it and, depending on the prediction's confidence, to update the model. The classification's output (i.e., the resulting most likely activity) and the corresponding ground truth are stored to evaluate the recognition rate. In particular, we use a sliding window of 800 samples with an overlap of 75% to periodically compute the overall F-1 score and the percentage of questions triggered to the users. Figure 4.6 shows the evolution of the F-1 score and the number of questions of the three considered methods. It emerges that, with respect to *No context* and *Context as features* approaches, our solution quickly reaches high recognition rates and a significantly lower number of questions.

In order to further show the impact of context reasoning on activity recognition, we have also evaluated our system considering different sets of activities.
Figure 4.7 shows how our system performs on a restricted set of simple physical activities which usually are considered in the majority of existing works. Those activities are poorly characterised by the context which surrounds the



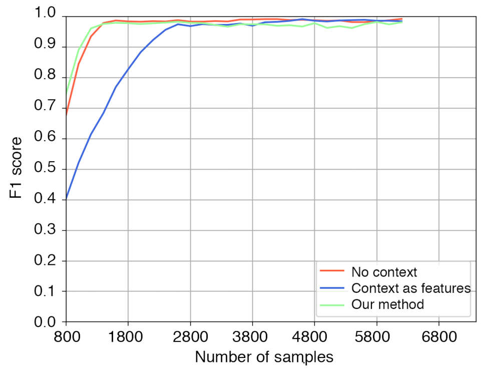
(a) F1 score



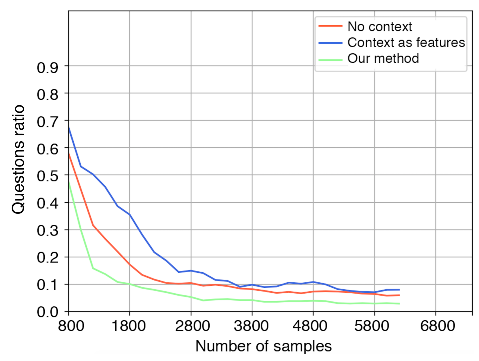
(b) Questions ratio

Figure 4.6: Evolution of the recognition model over time. Considered activities: Running, Sitting, Cycling, Standing, Walking, Elevator up, Elevator down, Going Upstairs, Going Downstairs, Brushing Teeth, Moving by car, Sitting transport, Standing transport.

user, while their motion patterns can be easily discriminated by purely statistical models. Indeed, the results show that the recognition rate and the



(a) F1 score

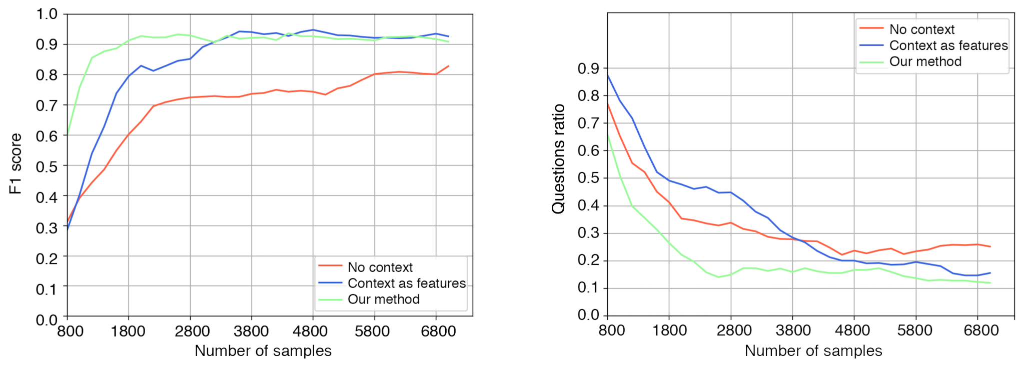


(b) Questions ratio

Figure 4.7: Evolution of the recognition model over time. Considered activities: Running, Sitting, Cycling, Standing, Walking

number of questions reached by our method are similar to the ones obtained by the *No Context* approach. It also strikes out that *Context as features* method has a slower learning improvement, due to the fact that additional context features add complexity to the semi-supervised activity model.

Finally, we also considered an “intermediate” set of activities. That set includes more context-dependent activities, like *moving by car*, *brushing teeth*, *elevator* and *stairs*. The results are shown in Figure 4.8.



(a) F1 score (b) Questions ratio

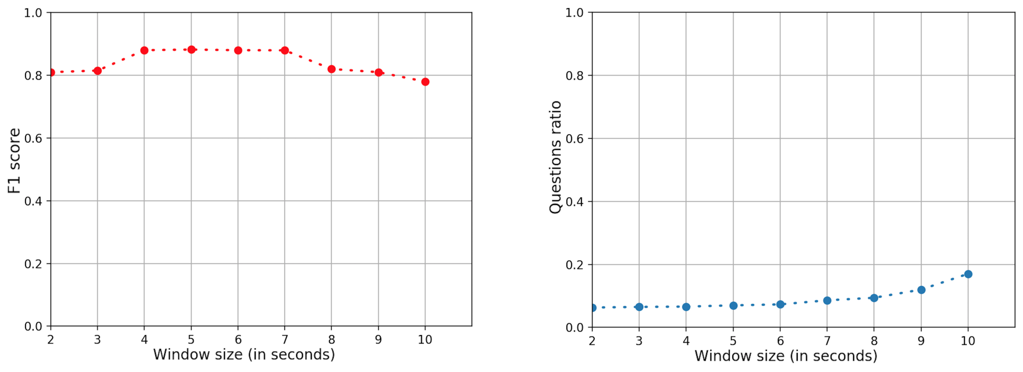
Figure 4.8: Evolution of the recognition model over time. Considered activities: Running, Sitting, Cycling, Standing, Walking, Elevator, Stairs, Brushing Teeth, Moving by car.

From the plots, it emerges that context-data has a high impact both on the recognition rate and on the number of questions. Indeed, the *No context* method struggles to reach the performances of context-based solutions. Moreover, our method reaches high recognition rates and a low number of questions very quickly with respect to the *context as features* approach. Hence, it emerges that considering context-data allows to significantly expand the set of considered activities.
Combining these results with the ones shown in Figure 4.6, it emerges that our method maintains the same trend independently from the considered set of activities. On the other hand, *Context as Features* method’s performance degrade increasing the complexity of the considered activities.
These results confirm that our method is scalable with respect to the set of considered activities and that it is effective in significantly reducing the number of triggered questions.

Thresholds

As introduced above, we have empirically determined that the optimal window size is $w = 4$, while the thresholds for semi-supervised updates are

$\pi = 0.7$ and $\rho = 0.45$. More precisely, Figure 4.9, shows how different w values impact on the overall classification result in term of F1 score and Questions ratio.



(a) F1 score (b) Questions ratio

Figure 4.9: Comparison between *F1 score* and *Questions ratio* by using different window sizes. In graphs (a) and (b), each point represents the overall result obtained using a *leave-one-subject-out* cross-validation approach, respectively in terms of *F1 score* and *Questions ratio*.

It is clear that small windows sizes (2 or 3 seconds) allow triggering to the user a few numbers of questions, but the F1 score value remains low. Increasing the w value from 4 to 7 seconds we assist to an improvement in the F1 score rate, while the number of user questions remains almost unchanged (or it grows a little). Using time windows of over 7 seconds causes deterioration of performance both on the F1 score and on the number of triggered questions. From this analysis emerges that $w = 4, 5$ or 6 allows obtaining the best trade-off between the number of user's questions and F1 score. Here It is important to point out that whether we use a time window of n seconds, the classifier works with a n seconds delay due to the time it takes to create each segment. Hence, as the length of the time windows affects the speed of the real-time classification, we chose $w = 4$.

Others important parameters in our model are ρ and π thresholds. The ρ threshold is used by the PREDICTION CONFIDENCE EVALUATION module (explained in Section 2.1.4), to determine when to ask a question to the user relying on the prediction confidence value. More precisely, it indicates the minimum value of the predicted label confidence accepted to consider the prediction reliable enough to do not trigger a question to the user. Differently, the π threshold is the characterising parameter of the *self-learning* approach. The PREDICTION CONFIDENCE EVALUATION MODULE exploit this value to determinate if update directly the *learning model* with the predicted label. In particular, π represents the minimum confidence value that a prediction must have to use its label to update the learning model. In fact, despite the CONTEXT-REFINEMENT MODULE allows to eliminate many of the statistical model mistakes, not all predictions are correct. To decrease the risk of updating the model with an incorrectly predicted label, its confidence is first evaluated and compared with π. Analyzing the figure 4.10 it is clear that a

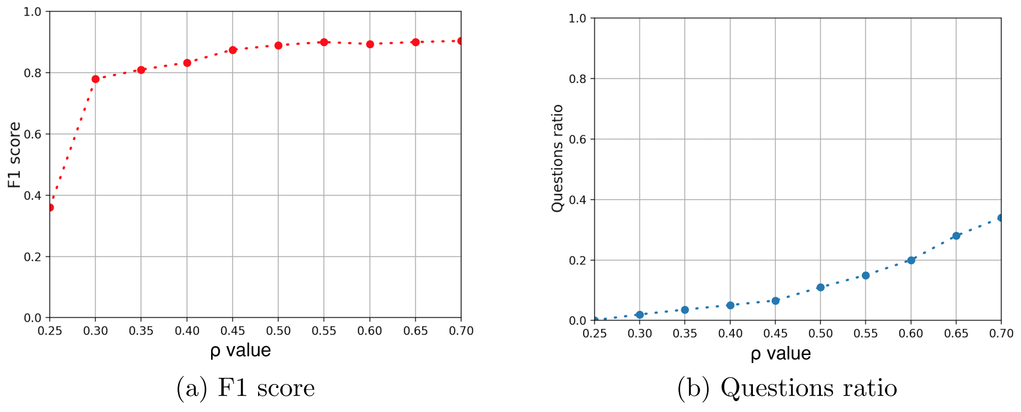
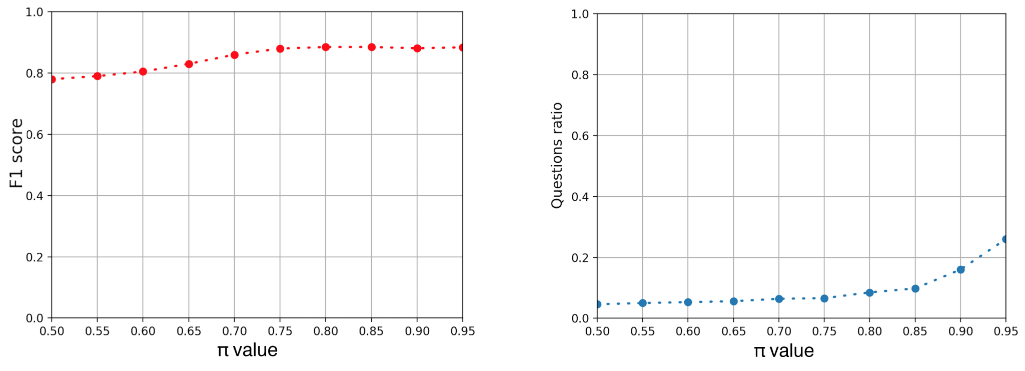


Figure 4.10: Comparison between *F1 score* and *Questions ratio* by using different ρ thresholds. In graphs (a) and (b), each point represents the overall result obtained using a *leave-one-subject-out* cross-validation approach, respectively in terms of *F1 score* and *Questions ratio*.

too low ρ value (i.e., 0.25) leads the PREDICTION CONFIDENCE EVALUATION module to assume that almost every prediction is correct (even if they are not) and hence do not trigger any question to the user. Increasing slightly the ρ value (from 0.30 to 0.45) it is evident how the F1 score increases significantly, despite a small rise in the number of user questions.

Proceeding to increment the ρ value, the F1 score remains almost stable, but the number of questions keep increasing inevitably. This is due to that choosing a high ρ threshold, leads the system to consider the predictions often uncertainty and hence triggers more questions to the user than necessary. In fact, it is clear that the F1 score does not grow accordingly with the rise of

the number of user's questions. This means that, increasing the ρ value and hence triggering more and more questions, does not help to further improve the learning model.
For these reasons, we have selected for our system a ρ threshold of 0.45 that it is the value that allows obtaining a good F1 score without bothering too much the user. Figure 4.11 shows how with a low π value (from 0.50 to 0.70)



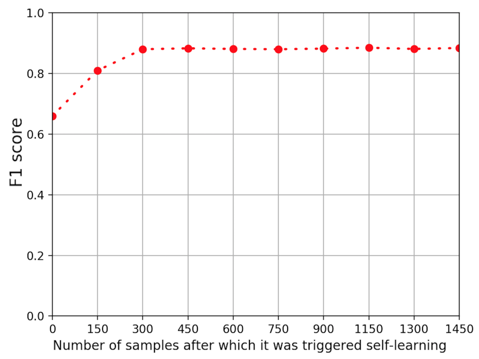
(a) F1 score (b) Questions ratio

Figure 4.11: Comparison between *F1 score* and *Questions ratio* by using different π thresholds. In graphs (a) and (b), each point represents the overall result obtained using a *leave-one-subject-out* cross-validation approach, respectively in terms of *F1 score* and *Questions ratio*.

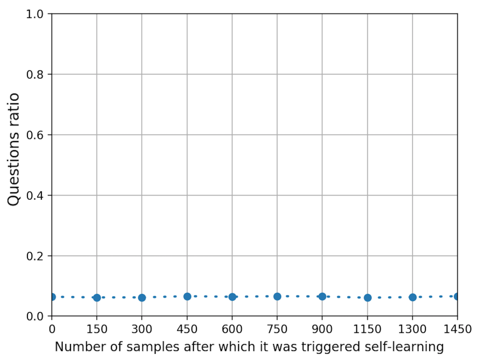
the F1 score struggles to improve even if the percentage of user's questions remains very low. Increasing the π value from 0.75 to 0.95 it is evident how the F1 score is stabilised on its maximum value while the number of question rise accordingly to the π increases. The continuous rise in the number of user questions is due to that high values of π the PREDICTION CONFIDENCE EVALUATION module tends to consider as unacceptable many predictions and consequently struggles to self-train itself. This leads to the conclusion that to reach the best performance in terms of F1 score, it is better to update the model with only predictions whose confidence is at least above 70%. It is also clear that a too high π value leads to only increase the number of questions without any benefit to the F1 score. We have hence chosen $\pi = 0.75$ because it allows the best trade-off between the number of user questions and the overall F1 score.

Self-learning triggering

Another important aspect to discuss is when it is better to trigger the *self-learning* approach to update the model. Indeed, our system before starting to make any predictions is subjected to a short bootstrap phase, in which is given one minute of labelled data to prime the learning model.
Just completed the bootstrap phase, the learning model is still unstable, and therefore firsts classifier predictions are not very reliable. Figure 4.12 shows that triggering the self-learning in this starting situation leads to propagating the classification error and the F1 score remains low.
Postponing the activation of self-learning after the classifier has processed 150 samples, it is clear the increase of F1 score, which reaches its maximum level after 350 analysed samples. Continuing to increase the number of samples provided to our system, the value of the F1 score remains stable. It is also



(a) F1 score



(b) Questions ratio

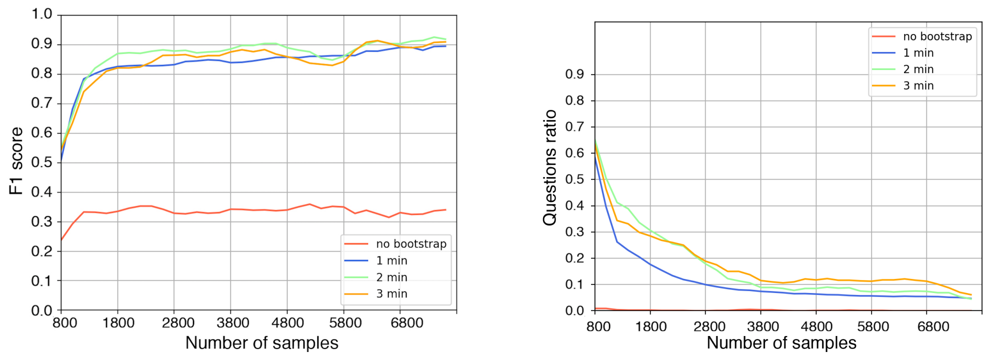
Figure 4.12: Comparison between *F1 score* and *Questions ratio* triggering the self-learning approach after a different number of analysed samples. In graphs (a) and (b), each point represents the overall result obtained using a *leave-one-subject-out* cross-validation approach, respectively in terms of *F1 score* and *Questions ratio*.

evident that regardless of when self-learning is activated the number of user questions is not affected.
This trend is due to that whether the confidence of the most probable activity exceeds the π threshold the PREDICTION CONFIDENCE EVALUATION updates the learning model without trigger a user question (independently of whether the prediction is correct or not).

For these reasons, in our implementation are necessary at least 350 samples to exploit the active-learning method on its best.

Model bootstrap

The last fundamental issue to discuss is how many labelled example to use in the bootstrap phase. Analyzing Figure 4.12 (a) it is evident how, without bootstrap, F1 score struggles to improve. In the same way (as is shown in Figure 4.12 (b)) the number of questions always remains low. This is due to



(a) F1 score (b) Number of user questions

Figure 4.13: Evolution of the F1 score and the percentage of user questions by giving different minutes of labelled data to pre-train the learning model.

that the not bootstrapped classifier fails to realize the presence of all the activities to be recognised, and assumes that some similar activities (i.e. going upstairs and going downstairs) are the same activity. So it does not trigger any questions to the user but at the same time, the F1 score does not increase.

It is also clear that by initialising the classifier with only 1 minute of samples for each activity, the trend changes considerably. Indeed, using this setup, as soon as the semi-supervised classification starts, a lot of questions are triggered to the user which strength the model awareness and allows growing the F1 score. Then, with the increase in the number of examples given to the classifier, the number of users' questions is rapidly reduced, reaching about 6%, while the F1 score stabilises at its maximum value (88%). By further increasing the number of pre-training samples for each activity, the improve-

ment is evident only in the initial phase of the classification, especially in the number of questions asked to the user (Figure 4.12 (b)). The more data is processed, the more the difference becomes irrelevant.
This analysis makes clear that the pre-training phase is fundamental to initialise the classifier to notice the presence of different activities, which otherwise, have to be recognised on the fly. Since one of the main advantages of using semi-supervised methods is to make the data labelling phase less burdensome, we decided to use only one minute of pre-training examples of each activity.

Chapter 5

Discussion and future works

In this thesis, we proposed a novel real-time activity recognition method that combines semi-supervised learning and semantic context-aware reasoning. To validate our system, we used a large dataset acquired in previous works of EW Lab that contains both inertial sensors data and rich context information. We developed a statistical incremental classifier able to process these data and provide a probability distributions over the possible activities. Then, this probability vectors were used as input to the semantic refinement module, which exploiting a knowledge-based reasoning engine, has excluded from the statistical predictions the highly unlikely activities by considering context-data. The system's output was hence been the most likely activity from the resulting context-refinement. Finally, the refined prediction was evaluated by a specific module that, rely on its confidence, decided if update the model using a self-learning approach, or ask the ground truth to the user.

In order to evaluate the effectiveness of our technique, we also implemented two additional methods that do not rely on any ontology-based reasoning. The former is called *No context*, since it considers only inertial sensors data to recognise activities. The latter is called *Context as features*. This method incorporates context data directly in the feature vectors generated by inertial sensors data. We then used a *leave-one-subject-out* cross-validation approach to evaluate and compare these methods in terms of recognition rate and number of questions.
From this comparison emerged that ontological reasoning on context-data is effective in improving the recognition rate of a semi-supervised classifier. However, the performances in terms of F1 score are not too distant from the ones obtained using context-data directly as features. More precisely, with

our method, we obtained an overall F1 score of 88%. By using *Context as features* approach the F1 score was 81% and by exploiting the *No context* method it was 64%.
The main advantage of our symbolic method is the drastically reduced number of user' queries. In fact, it was reduced from 40% - 35% (using respectively *No context* and *Context as features*) to only 6%. This is due to that our semantic refinement technique exploits the ontology to remove from the prediction unlikely activities. This increases the confidence of the remaining activities and, at the same time, triggers our semi-supervised technique to update the activity model without bothering the user. Results indicate that the resulting system should provide a much better user experience by limiting the number of times a user is interrupted with a question.

Strong points

The drastically reduced number of user queries allows our method to be more acceptable in a realistic scenario.
We also believe that our method is significantly more scalable. One aspect is that not every context source may be continuously available at the same time. Using context as features imply possible missing values in the feature vectors that can potentially be used to update the classifier, which in turn may negatively impact the recognition rate. Another aspect is whether it becomes necessary to include additional context data while the system is running. In the case of using context as features, there is a need for re-training the model from scratch with new labelled data. In our method, considering new context data simply means extending the ontology. Moreover, the ontology can also be periodically revised and improved.
About Model initialisation and personalization, our semi-supervised classifier respect to a supervised classifier requires a small set of labelled data acquired from a few users to be initialised.
However, in the first phase of the system's deployment, there is a need for a set of volunteers in charge of using the system to collaboratively update the initial model. Indeed, initially the classification is not robust and it generally triggers a high number of queries which is not acceptable for final users. When enough training data is collected by the volunteers, we consider the shared model as stable.
For those users, context-aware refinement and model updates are still part of the activity recognition pipeline, in order to personalise the model on each

subject. Hence, the final users download the stable model on their mobile devices to perform activity recognition locally and as it is easy to see from the results, the number of queries required by our system drops down quickly.

Weaknesses and future works

A major limitation of our approach is the rigid formalism for semantic reasoning, which does not take into account the intrinsic uncertainty of knowledge and sensor-based systems. Hence, in future works, we will evaluate alternative probabilistic tools to model the context. We also want to consider temporal sequences of activities and contexts in our semantic refinement method. Hence, we will investigate how to include temporal reasoning in our knowledge-based model. Finally, we will investigate how our system can be used to provide a personalised activity recognition model for each user. Different subjects may have different physical characteristics and habits. Hence, there is a high variance of activity execution moralities and contexts. Personalising the recognition model using our system and, at the same time, learning personalised contexts may further enhance the recognition rate. We will evaluate the feasibility of running our method locally on the subjects' mobile devices, evaluating the performances of this solution.

Bibliography

[1] Zahraa S Abdallah, Mohamed Medhat Gaber, Bala Srinivasan, and Shonali Krishnaswamy. Activity recognition with evolving data streams: A review. *ACM Computing Surveys (CSUR)*, 51(4):71, 2018.

[2] Zahraa Said Abdallah, Mohamed Medhat Gaber, Bala Srinivasan, and Shonali Krishnaswamy. Adaptive mobile activity recognition system with evolving data streams. *Neurocomputing*, 150:304–317, 2015.

[3] A. Avci, S. Bosch, M. Marin-Perianu, R. Marin-Perianu, and P. Havinga. Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In *23th International Conference on Architecture of Computing Systems 2010*, pages 1–10, Feb 2010.

[4] Franz Baader, Diego Calvanese, Deborah McGuinness, Peter Patel-Schneider, and Daniele Nardi. *The description logic handbook: Theory, implementation and applications.* Cambridge university press, 2003.

[5] Oresti Banos, Juan-Manuel Galvez, Miguel Damas, Hector Pomares, and Ignacio Rojas. Window size impact in human activity recognition. *Sensors*, 14(4):6474–6499, 2014.

[6] Ling Bao and Stephen S. Intille. Activity recognition from user-annotated acceleration data. In *Pervasive Computing: Second International Conference, PERVASIVE 2004, Linz/Vienna, Austria, April 21-23, 2004. Proceedings*, pages 1–17, Berlin, Heidelberg, 2004. Springer.

[7] Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, and Daniele Riboni. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2):161–180, 2010.

[8] Andreas Bulling, Ulf Blanke, and Bernt Schiele. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Computing Surveys*, 46(3):33:1–33:33, 2014.

[9] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[10] Anind K Dey. Understanding and using context. *Personal and ubiquitous computing*, 5(1):4–7, 2001.

[11] João Gama, Raquel Sebastião, and Pedro Pereira Rodrigues. On evaluating stream learning algorithms. *Machine learning*, 90(3):317–346, 2013.

[12] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. Hermit: an owl 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[13] Donghai Guan, Weiwei Yuan, Young-Koo Lee, Andrey Gavrilov, and Sungyoung Lee. Activity recognition based on semi-supervised learning. In *Embedded and Real-Time Computing Systems and Applications, 2007. RTCSA 2007. 13th IEEE International Conference on*, pages 469–475. IEEE, 2007.

[14] Isabelle Guyon and André Elisseeff. An introduction to feature extraction. In *Feature extraction*, pages 1–25. Springer, 2006.

[15] Norbert Györbíró, Ákos Fábián, and Gergely Hományi. An activity recognition system for mobile phones. *Mobile Networks and Applications*, 14(1):82–91, 2009.

[16] Terry Halpin. Object-role modeling (orm/niam). In *Handbook on architectures of information systems*, pages 81–103. Springer, 1998.

[17] Karen Henricksen and Jadwiga Indulska. Modelling and using imperfect context information. In *PerCom Workshops*, pages 33–37. Citeseer, 2004.

[18] Karen Henricksen and Jadwiga Indulska. Developing context-aware pervasive computing applications: Models and approach. *Pervasive and mobile computing*, 2(1):37–64, 2006.

[19] Karen Henricksen, Jadwiga Indulska, and Andry Rakotonirainy. Modeling context information in pervasive computing systems. In Friedemann Mattern and Mahmoud Naghshineh, editors, *Pervasive Computing*, pages 167–180, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[20] Enamul Hoque and John Stankovic. Aalo: Activity recognition in smart homes using active learning in the presence of overlapped activities. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2012 6th International Conference on*, pages 139–146. IEEE, 2012.

[21] Matthew Horridge and Sean Bechhofer. The owl api: A java api for owl ontologies. *Semantic Web*, 2(1):11–21, 2011.

[22] Ian Horrocks, Peter F Patel-Schneider, and Frank Van Harmelen. From shiq and rdf to owl: The making of a web ontology language. *Web semantics: science, services and agents on the World Wide Web*, 1(1):7–26, 2003.

[23] HM Sajjad Hossain, Md Abdullah Al Hafiz Khan, and Nirmalya Roy. Active learning enabled activity recognition. *Pervasive and Mobile Computing*, 38:312–330, 2017.

[24] G. KLYNE. Composite capability/preference profiles (cc/pp) : Structure and vocabularies. *W3C Working Draft*, 2001.

[25] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

[26] Yongjin Kwon, Kyuchang Kang, and Changseok Bae. Unsupervised learning for human activity recognition using smartphone sensors. *Expert Systems with Applications*, 41(14):6067–6074, 2014.

[27] Oscar D Lara, Miguel A Labrador, et al. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys and Tutorials*, 15(3):1192–1209, 2013.

[28] Min-Seok Lee, Jong-Gwan Lim, Ki-Ru Park, and Dong-Soo Kwon. Unsupervised clustering for abnormality detection based on the tri-axial accelerometer. *ICCAS-SICE*, 2009:134–137, 2009.

[29] Young-Seol Lee and Sung-Bae Cho. Activity recognition with android phone using mixture-of-experts co-trained with labeled and unlabeled data. *Neurocomputing*, 126:106–115, 2014.

[30] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine Learning Proceedings 1994*, pages 148–156. Elsevier, 1994.

[31] Lin Liao, Dieter Fox, and Henry Kautz. Location-based activity recognition. In *Advances in Neural Information Processing Systems*, pages 787–794, 2006.

[32] Brent Longstaff, Sasank Reddy, and Deborah Estrin. Improving activity classification for health applications on mobile devices using active and semi-supervised learning. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2010 4th International Conference on Pervasive Computing Technologies for Healthcare*, pages 1–7. IEEE, 2010.

[33] Daniele Riboni and Claudio Bettini. COSAR: Hybrid reasoning for context-aware activity recognition. *Personal and Ubiquitous Computing*, 15(3):271–289, 2011.

[34] Amir Saffari, Christian Leistner, Jakob Santner, Martin Godec, and Horst Bischof. On-line random forests. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1393–1400. IEEE, 2009.

[35] Maja Stikic, Kristof Van Laerhoven, and Bernt Schiele. Exploring semi-supervised and active learning for activity recognition. 2008.

[36] Lin Sun, Daqing Zhang, Bin Li, Bin Guo, and Shijian Li. Activity recognition on an accelerometer embedded mobile phone with varying positions and orientations. In *International conference on ubiquitous intelligence and computing*, pages 548–562. Springer, 2010.

[37] Timo Sztyler and Heiner Stuckenschmidt. On-body localization of wearable devices: An investigation of position-aware activity recognition. In *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–9, Washington, D.C., 2016. IEEE Computer Society.

[38] Timo Sztyler and Heiner Stuckenschmidt. Online personalization of cross-subjects based activity recognition models on wearable devices. In *Pervasive Computing and Communications (PerCom), 2017 IEEE International Conference on*, pages 180–189. IEEE, 2017.

[39] Dorra Trabelsi, Samer Mohammed, Faicel Chamroukhi, Latifa Oukhellou, and Yacine Amirat. An unsupervised approach for automatic activity recognition based on hidden markov model regression. *IEEE Transactions on Automation Science and Engineering*, 10(3):829–835, 2013.