UNIVERSITY OF TRENTO

DEPARTMENT OF INFORMATION ENGINEERING AND COMPUTER SCIENCE

Ph.D. Thesis

Personal Context Recognition from Sensors

Ph.D. candidate:
Wanyi Zhang

Supervisor:
Prof. Fausto Giunchiglia

Co-Supervisor:
Prof. Andrea Passerini

Final Exam Year 2021

Acknowledgements

First of all, I want to express my appreciation and thanks to my advisors Prof. Fausto Giunchiglia and Prof. Andrea Passerini. Thanks for giving me such a great opportunity to study here. You have helped me a lot, encouraged a lot and trust me a lot during the past four years. Without your support and encouragement, I can not finish my Ph.D. Thanks for having me work in the WeNet Project. I appreciate that. It allows me to meet more people with passion and to discover a bigger world. Besides, I am also deeply grateful to all the other professors and all my colleagues from KnowDive who helped me and supported me. Last but not least, I want to express my gratefulness to my beloved parents my family for their unhesitating support. This Ph.D is dedicated to all of you.

Wanyi Zhang
March 2022

Abstract

Machine learning has become one of the most emerging topics in a lot of research areas, such as pervasive and ubiquitous computing. Such computing applications always rely on the supervised learning approach to recognize user's context before a suitable level of services are provided. However, since more and more users are involved in modern applications, the monitored data cannot be guaranteed to be always true due to wrong information. This may cause the mislabeling in machine learning and so affects the prediction. The goal of this Ph.D. thesis is to improve the data quality and solve the mislabeling problem caused by considering non-expert users. To achieve this goal, we propose a novel algorithm, called Skeptical Learning, aiming at interacting with the users and filtering out anomalies when an invalid input is monitored. This algorithm guarantees the machine to use the pre-known knowledge to check the availability of its own prediction as well as the label provided by the users. This thesis clarifies how we design this algorithm and makes three main contributions: *(i.)* we study the predictability of human behavior through the notion of personal context; *(ii.)*we design and develop Skeptical Learning as a paradigm to deal with the unreliability of users when providing non-confidential labels that describe their personal context; *(iii.)* we introduce an MCS platform where we implement Skeptical Learning on top of it to solve unreliable labels issue. Our evaluations have shown that Skeptical Learning could be widely used in pervasive and ubiquitous computing applications to better understand the quality of the data relying on the machine knowledge, and thus prevent mislabeling problem due to non-expert information.

Keywords: Machine learning, ubiquitous and mobile computing, fixing mislabeling, mobile information processing system

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Context

With the development of mobile devices, many different sensors are embedded in smartphones and variant applications are installed. Smartphones are becoming a more and more important part of people's daily life. Applications on smartphones, such as personal assistant, smart home monitor and healthcare monitor, are designed to make people's life more convenient and easier. There are tons of applications working on people's phone nowadays. Here we give some examples of a few categories. For example, these map applications, e.g., Google map[1] and Baidu map[2], can give the user suggested paths to his destination. The user could select one of them basing on what he need. For example, the user may choose the one with the shortest distance because he does not want to walk too much, or the one that will take the least time because he wants to arrive there earlier, or the one with a set of location points included because these are the places that he wants to pass by. There are some applications or systems for health-care purpose. For instance, TrackYourTinnitus (TYT), TrackYourHearing (TYH), TrackYourDiabetes (TYD), and TrackYourStress (TYS) keep track of the progress of users' hearing loss, diabetes, or stress level, respectively [56]. This helps the users to be more aware of symptom changes in specific contexts. Another type of the applications focuses on smart environment. These applications allow the user to control and monitor the devices in a certain space. For example, work in [73] described a smartphone application that can be connected

[1]https://maps.google.com

[2]https://map.baidu.com

with the home gateway remotely, thus the user can control and monitor the devices in home, and manage schedules. Some of the applications provide certain services at certain time wherever the user is and whatever the user is doing. As you may notice, these applications may interrupter the user at an inappropriate time. It can be smarter if they can offer the user appropriate assistance at a right moment. This requires the application or system to understand the user's current context before providing further services.

Automatic recognition of personal context is one of the approaches to realize the context-aware applications or systems, which "use context to provide relevant information and/or services to the user, where relevancy depends on the user's task" according to the definition of context-awareness in [21]. For instance, if an application, working on user's phone as his personal smart assistant, is aware that the user is alone at home according to the noise detection, and is in an important online meeting according to his agenda, it should keep the smartphone in a quiet mode to avoid potential interruption of incoming call or message. When it detects that the user has been sit for a long time, it should remind the user to have a break, as a real assistant will do. Even better, if this application also have access to the information from third part, such as the user's smart refrigerator, and detects that there is no enough food for today, it could remind the user and suggest him to order a take-away lunch.

1.2 The Problem

Another example of a context-aware application is when the application detects that the user is in an environment which need to be quiet, such as a Church or a library, it should not push information or messages to the user with a ringer and turn the phone to a quiet mode automatically. A simple solution is to keep track of the user's location by reading the GPS data from the phone. But it is not always a good solution when the location is not detectable or multiple activities could happen in the same location. One more complicate example is that when the smartphone detects that the user is in his office area, it should push message quietly if the user is in a meeting, but it should also allow the message to come in with a ringtone if the user is just having a rest or chatting with his colleagues. Although the term of *context recognition* in [60, 82] means *activity recognition*, above examples shows that personal context recognition is more than only location recognition or activity recognition. The model of personal context should be designed or chosen wisely depending on the need of the task.

Machine learning algorithms are good tools to solve the recognition problem, especially supervised machine learning. The machine can learn from a labeled data set, and when it is given a new input, it can output a label as its prediction. The accuracy of prediction highly depends on the quality of the data set. However, a data set with high-quality labels are rare to meet in the real world. The implicit assumption made in (even recent) mainstream machine learning is that annotators are *experts*. While, with the fast paced development of mobile devices and the evolution of data collection method, more and more normal users are involved. Normal-user involvement generates a new set of challenges. Among them, the biggest one is the data quality, due to the inaccurate input from *non-expert* users. This is a phenomenon that relate to the people *response biases*, e.g., conditioning, memory bias, and sometimes also unwillingness to report, *cognitive bias*, e.g., the inadequate recall of respondents when providing an annotation, and *carelessness*, namely not putting enough attention in providing the answer, e.g., because of hurriedness [39].

The main problems that we identify in this these are listed below:

- An appropriate context model is needed for the personal context recognition task.
- The quality of the annotations provided by the normal user is usually not high. Users may make mistakes when they give their labels. And the mislabeling issue has impact on the performance of the subsequent machine learning algorithms.
- A system or a platform is needed to collect high-quality data by interacting with the users in real time.

1.3 The Solution

A good designed context model is necessary and fundamental to context recognition tasks. We propose to use the personal context model which is introduced in [37]. This model takes into account different dimensions of the environment, namely time, location, activity being carried out, and social ties. It enables downstream context recognition tools to leverage the correlations between these dimensions. We then investigate the role played by this four dimensions (or modalities) of the context model on the predictability of individuals' behaviors. A set of data analysis was taken on a data set that generated in the SmartUnitn Project. This project was designed to understand the university students' behaviour, and the annotations of the students' context was collected together with the sensor data from their smartphone. The results of the analysis firstly show that multi-modality model is necessary since it provides more information than the single-modality ones. It further highlights the fundamental importance of the jointly interplay of different contextual modalities in behavior analysis. And in turn, it proves the correctness of selecting multi-modality personal context model. Secondly, the results support the argument that subjective location is far more informative than objective location for predicting behavior. Last buy not least, the results of analysis present the diversity of individuals, and it shows that individuals are more easily identified by rarer, rather than frequent, context annotations.

To solve the data quality issue caused by non-expert users, we propose a general algorithm for Skeptical Learning, which interacts with the user and challenges him when a potential inaccurate of his input is detected. The key idea is that the machine uses the knowledge it has available to check the correctness of its own prediction and of the label provided by the user. By keeping track of the sequence of wrong and right answers, the machine builds a measure of confidence towards itself and the user, which is then used, in the case of a contradiction, to decide what is actually the case. In this context, by *available knowledge* we mean both the knowledge inductively built out of the previous learning activity and the knowledge which may come from third parties or may be built-in as *a prior* knowledge.

To solve the last problem that is identified in this thesis, we propose a general Mobile Crowd Sensing (MCS) platform that, together with the i-Log application, can be used to collect sensor data and user's annotations. And On top of the platform, it runs the Skeptical Learning algorithm that we proposed to deals with the unreliability of non-expert users when providing labels.

To be corresponding to the problems, we conclude the main contributions of this thesis as following:

- We propose to use a multi-modality personal context model. And we evaluate our model by analysing data collected in SmartUnitn project.
- We propose the Skeptical learning algorithm to deal with the mislabeling issue.
- We propose the architecture of a MCS platform which runs the Skeptical learning

on the top of it to improve the quality of the data collected from the users.

1.4 The Structure of the Thesis

In Chapter 2 the state of the art on five related research topics are introduced. We firstly introduce personal context models are designed in other work, and the advantages and disadvantages of these models. Secondly since in our work the context is mainly about the student's life, we introduce some work which study university students' life in different ways. In most of traditional machine learning works, that make use of user's annotation to train machine, one big challenge is how to deal with the label noise, so we introduce how label noise issue is solved in other work. We finally introduce the state of the art work on mobile crowd sensing systems.

Chapter 3 introduces the personal context model that we use in our work. We also introduce different representation of the personal context, namely endurant and perdurant context. These concepts are then used to design the personal context annotation. The main citation of this chapter is [37]. The personal context model we use was firstly proposed by their work. In this chapter, based on that work, we introduce the concepts of objective, subjective and machine context additionally. We apply subjective context when design the annotations, which is from human perception and allows users to give annotations of their subjective point of view. It was later proven that subjectivity is necessary for framing behavior from the subject's perspective, and it has a substantial effect on predictability and regularity of behavior in practice. Then we introduce the SmartUnitn projects as use cases since the personal context model was applied in these experiments. The goal of this series of experiments is to understand how university students manage their time and how their time management ability affects their academic performance. We also introduce the design of the experiments, including the population, the duration, the data collection of these two experiments, and the data sets generated. The data sets are also the ones we use in the subsequent sections.

In Chapter 4, we firstly introduce the personal context model aiming at capturing the array of relevant subjective experiences characterizing a specific behavior of an individual. And then we study the predictability of human behavior through the notion of personal context. We investigate the role played by four contextual dimensions (or modalities), namely time, location, activity being carried out, and social ties, on the predictability of individuals' behaviors. This chapter is based on the following publication:

- **Wanyi Zhang**, Qiang Shen, Stefano Teso, Bruno Lepri, Andrea Passerini, Ivano Bison, and Fausto Giunchiglia. *Putting Human Behavior Predictability in Context.[J]* EPJ Data Science, 2021. (Accepted)

In Chapter 5, we introduce the annotation of personal context and the quality issue. We propose a methodology to evaluate the annotations provided by users. This chapter is based on the following publication:

- Fausto Giunchiglia, Mattia Zeni, Enrico Bignotti, **Wanyi Zhang**. *Assessing annotation consistency in the wild[C]* 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops). IEEE, 2018: 561-566.

In Chapter 6, we firstly study the related work about label noise problem which is caused by non-expert annotator. Then we introduce the Skeptical Learning algorithm that we propose to solve the mislabeling issue. And finally we give a detailed description of the experiment we have carried out, which highlights the advantages of an interactive, skeptical approach to learning over state-of-the-art but non-skeptical machine learning alternatives. This chapter is based on the following publications:

- Mattia Zeni, **Wanyi Zhang**, Enrico Bignotti, Andrea Passerini, and Fausto Giunchiglia. *Fixing mislabeling by human annotators leveraging conflict resolution and prior knowledge[J].* Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2019, 3(1): 1-23.

- **Wanyi Zhang**. *Personal Context Recognition via Skeptical Learning[C].* IJCAI. 2019: 6482-6483.

- **Wanyi Zhang**, Andrea Passerini, and Fausto Giunchiglia. *Dealing with Mislabeling via Interactive Machine Learning[J].* KI-Künstliche Intelligenz, 2020: 1-8.

In Chapter 7 , we focus on data quality issues in Mobile Crowd Sensing (MCS) area. The main contribution is that we propose a general MCS platform for integrating at scale sensor data and user's general data in the form of labels, together with the statics knowledge of the world. This chapter is based on the following publication:

- **Wanyi Zhang**, Mattia Zeni, Andrea Passerini, and Fausto Giunchiglia. *Skeptical Learning – an Algorithm and a Platform for Dealing with Mislabeling in Personal Context Recognition.[J]*. Algorithms, 2022. (Accepted)

Finally, in Chapter 8 we present the conclusions and future work respectively.

Chapter 2

State of the Art

To solve the first problem that has been identified in the introduction chapter, i.e., an appropriate context model shall be chose for the personal context recognition task. Thus we firstly investigate the existing personal context models. Since our work mainly focuses on a university student's life scenario, and the experiment of the proposed skeptical learning is carried out on the data set of students' university life, therefore we investigate secondly the existing studies on students' life. As for solving the problem of mislabelling issues, we introduce some existing approaches dealing with label noise issues. Last but not least, our work is within the research area of participatory sensing and mobile crowd sensing, thus we introduce some existing mobile crowd sensing systems.

2.1 Personal Context model

A context model defines how context data are structured. A good context model should capture all kinds of situational information relevant to the application at hand [21] and use the right level of abstraction [6]. Ontology is a widely accepted tool for formalizing context information [59], and several context ontologies have been proposed. Typical examples include CONON [103] and CaCONT [106]. CONON focuses on modeling locations by providing an upper ontology and lower domain-specific ontologies organized into a hierarchy. CaCONT defines several types of entities, and provides different levels of abstraction for specifying location of entities, e.g., GPS and location hierarchies. Focusing on semantic information of place, the work in [109] proposed a place-oriented ontology model representing different levels of place and related activities and improve the performance of place recognition. In [54], they proposed an ontology model involving social situation and the interaction between people.

These models, however, suffer from two main limitations. First, in order to support context recognition, the model should account for subjectivity of context descriptions. For instance, the *objective* location "hospital" plays different roles for different people: for patients it is a "place for recovering", while for nurses it is a "work place". This makes all the difference for personal assistants because the services that a user needs strongly depend on his or her subjective viewpoint. Most context models ignore this fact, with few exceptions, cf. [53]. Second, arguably answers to four basic questions – "what time is it?", "where are you?", "what are you doing?", and "who are you with?" – are necessary to define human contexts. Correlations between these aspects are also fundamental in recognition and reasoning: if the user is in her room, a personal assistant should be more likely to guess that she is "studying" or "resting", rather than "swimming". In stark contrast, most models are restricted to one or few of the above four aspects and therefore fail to capture important correlations, like those between activity and location or between time and social context.

2.2 University Student's Life Study

The main population of our experiments is students, since they are a relevant sample of population for sociological studies, especially with respect to their time management ability, which includes setting goals and priorities and plays a crucial role in improving students' performance [68]. Another area that focuses on students, due to their wide adoption of smartphones and tech-savviness [9], is computational social sciences, which is based on approaches for extracting and analyzing behaviors using smartphone data such as proximity, location, and call logs [26]. These data are combined with surveys, which may be administered via smartphones, for socio-psychological metrics such as personality traits, daily mood, or sleep quality [9]. Within the work in computational social sciences, the Student Life study [101] is the closest one to ours. For this study, smartphones were used to assess the impact of workload on stress, sleep, activity, mood, sociability, mental well-being and academic performance of a class of 48 students (38 males and 10 females) of a computer science class across a 10-week term at Dartmouth College. Moreover, the SmartGPA study [102] used the data from [101] to show that there is evidence of a link between the students' GPA and their behavioral patterns. In this work, regression analyses were used to develop a behavioral slope and behavioral breakpoints in order to identify changes in a student's behavior on a weekly basis. The temporal granularity and the predictive model does not consider raw data, since the pre-built classifiers feed into a regression model, e.g., the accelerometer. Differently from our work, these studies do not have approaches in place for assessing the quality of the students' annotations. In fact, they rely on heuristics, e.g., spending at least 20 minutes in a library means that the student is studying, and classifiers, which may be inaccurate, e.g., conversation classifiers cannot distinguish a person speaking in a TV from one physically in the room [45].

2.3 Label Noise

While traditional approaches to concept learning assume perfectly labeled training sets, most recent supervised learning techniques can tolerate a small fraction of mislabelled training instances (see for instance [31]). A common solution consists in designing learning models which are robust to (some) label noise [30]. In particular, by averaging predictions of multiple learners, ensemble methods usually perform well in terms of noise robustness [22, 77]. In this line of thought, the robustness of random forests, the ensemble method used in this paper, has recently been shown both theoretically and empirically [33]. Nonetheless, label noise badly affects the performance of learning algorithms [67]. Our approach diverges from existing solutions since it involves an interactive error correction phase. This process allows tolerating a much larger amount of noise, achieving substantial improvements over the previous work.

The field of statistical relational learning [5] deals with the integration of symbolic and sub-symbolic approaches to learning. Frameworks like Markov Logics [80], Semantic-Based Regularization [23] or Learning Modulo Theories [96] combine logical rules or other types of constraints with learnable weights to encourage predictions consistent with the available knowledge. Our main difference is that we use knowledge in an interactive way to identify potential errors in user feeback, and activate a conflict resolution phase to solve such controversies.

While many machine learning approaches assume an expert user, it is not the case in other areas of research, e.g., mobile crowdsensing [43], where users collect and share sensed data and their annotations via their smartphones. A relevant issue here is assessing the quality of users' annotations (see [78] for a comprehensive review). However, the focus in these works is on gathering reliable information about locations or events of *common interest* among a set of users. The key difference from this work is that we focus on personal data, e.g., personal context and activities, to be used by the user herself. Thus, the quality of a user's annotation cannot be evaluated by comparing it with other annotations from the crowd (which would be very hard if not impossible, since we deal with personal data) but, rather, by comparing it with the machine's knowledge.

2.4 Mobile Crowd Sensing System

In recent years, Mobile Crowd Sensing (MCS) has become an emerging sensing paradigm where, as from [43], ordinary citizens contribute data sensed or generated from their mobile devices. Such data are then aggregated for crowd intelligence extraction and people-centric service delivery. Human involvement is one of the most important characteristics of MCS, especially in applications where participatory sensing applies. One such example are the applications that are designed for environmental monitoring. Work in [29] apply the MCS-RF model to infer the real-time and fine-grained PM2.5 throughout Beijing. In their application, Users can upload images with a time stamp and GPS information. Users' input is used along with other official data sources, such as meteorological data and traffic data published by relevant agencies to do the environmental monitoring. Another kind of applications are those designed to monitor the use's health status. For example, the application in [75] is designed to monitor the user's daily activities and emotions for health caring purposes. It collects sensor data and meanwhile asks for the user's self-report observations. The TrackYourTinnitus (TYT) project tracks user's tinnitus by his answers to questionnaires and the sensor data collected by his phone. Similar to that, TrackYourHearing (TYH), TrackYourDiabetes (TYD), and TrackYourStress (TYS) keep track of the progress of users' hearing loss, diabetes, or stress level, respectively [56]. This helps the users to be more aware of symptom changes in specific contexts. Finally, there are other applications that collect urban data, such as the reports of damaged infrastructures, with the final goal of monitoring urban safety [114, 62]. In this case the user is asked to report observations by answering questionnaires or uploading images.

In MCS systems, a major problem is that of the quality of the data. In [49], the authors deal with this problem by computing the reputation score of the device as a reflection of the trustworthiness of the contributed data. Dually, in [107], the reputation system focuses on the reputation of the human participants. Differently from this work we propose a hybrid approach where we evaluate both users and the machine, we deal with personal data, and our metrics leading to reputation are the result of a machine learning activity.

Chapter 3

Data Gathering and Annotation

3.1 Introduction

In this chapter we introduce the definition of personal context model and the annotations of context. This work is based on the paper [37] and paper [91]

A good designed context model is necessary and fundamental to context recognition tasks. The term "context" refers to any kind of information that is necessary to describe the situation that an individual is in [21]. But human individual has a limited view of their surroundings all the time in their life. Work in [34] gives the definition of personal context, i.e., "a theory of the world which encodes an individual's subjective perspective about it". Personal context modeling should take this relation between the user and the context into consider.

Most existing work focus on a controlled, predefined environment in a closed domain, and suffer from two main limitations, i.e., they do not taking subjectivity context into account and fail to capture important correlations, like those between activity and location or between time and social context.

We propose a personal context model based on [34]. it takes into account different dimensions of the environment and enables downstream context recognition tools to leverage the correlations between these dimensions. We also explain the notions of *endurant* and *perdurent* context, *objective* and *subjective* context. The notions are then used to create an ontology accounting for the way how perception guides humans to aggregate their description of their surrounding environment. We apply and test our approach by collaborating with sociology experts and taking part to the SmartUnitn project, which aims at recognizing behavioral patterns of students to see how their life style affects their academic performance. The collaboration with sociology experts provides us with useful methodological considerations on how to adapt our ontology to be usable in real life. Basing on these considerations we provide three list of context annotations which can be used for tracking user's activity, location and social relations, in accordance with sociological methodologies such as time use surveys [12].

3.2 Personal Context Model

To illustrate the personal context, we firstly consider a scenario that occurrences in a Ph.D student's daily life. It is Tuesday morning, and Ph.D student Wanyi is in a meeting with her supervisor Prof. Fausto and her colleague Andrea. The topic is to discuss about the paper they will submit. The meeting is taking place in office 230, with a desk in the middle and a white board on the wall. Wanyi is talking about her idea and writing on the board to explain it in detail.

Figure 3.1 present this scenario as a knowledge graph. Each node represents an entity, e.g., the person and the room, with its respective attributes and their attribute values. For instance, the attributes of Fausto in Figure 3.1 are "Class", "Name", and "Role", and their corresponding attribute values are "Person", "Fausto", and "Professor". Edges represent relations between entities, e.g., "Office room" has two relations: "HasActivity" for "Meeting", and "In" for "Board" and "Desk".

We formalize the relation between context, which is a partial representation of the real world, and the person, which is centered on from Figure 3.1, as

$$MyWorld = \langle me, Context \rangle$$

where:

- *me* is the person on which the context is centered, and it is an entity with its attributions and relations.
- *Context* is the real world context of the person, aggregating different elements surrounding the user. Notice that, it is not the *global* view of the environment, but *local* view of the environment encompassing a user centered subset.

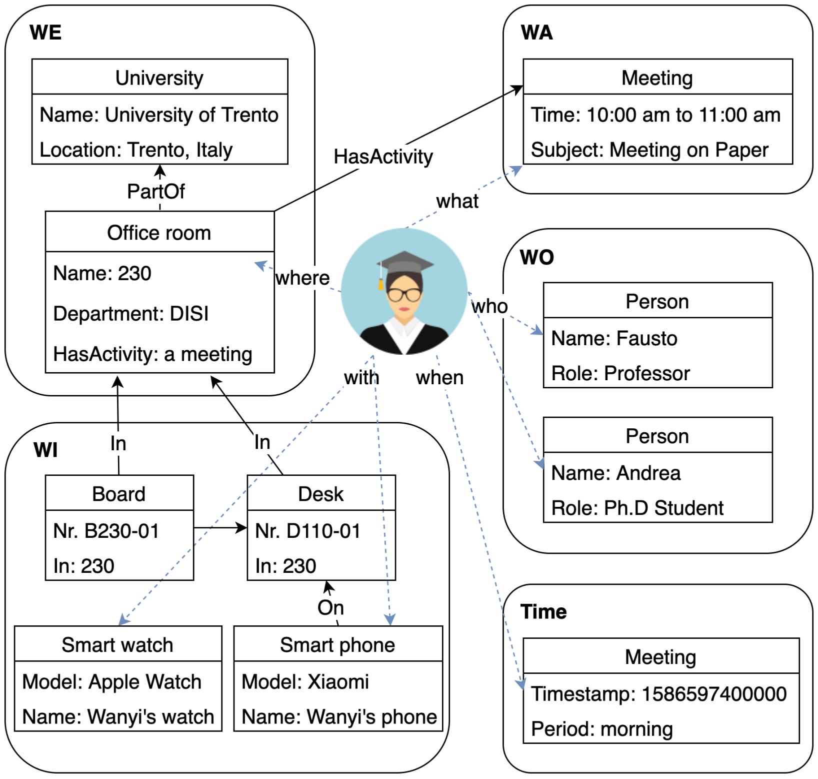


Figure 3.1: The five dimensions of the context, centered on the user.

In Figure 3.1, the blue dashed arrows represent the relations between *me* and *Context*. They link different elements of the context directly to the person, e.g., the smartphone that the person has, and the office room that the person is in.

Context is a theory of the world that encodes an individual' subjective perspective about it [34]. Individuals have a limited and partial view of the world at all times in their everyday life. For instance, consider the example we present in Figure 3.1 that Wanyi is in a meeting with her professor and colleague in an office. Despite all the commonalities, each person in the room has a different context because they focus on different elements of their personal experience (Wanyi focuses on telling they her idea while Fausto and Andrea focus on listening to her) and ignore others (like the sound of the projector, the weather outside, and so on.). Given the diversity and complexity of individual experiences, formalizing the notion of context in its entirety is essentially impossible. For this reason, simpler but useful application-specific solutions are necessary.

Previous work has observed that reasoning in terms of questions like "what time is it?", "where are you?", "what are you doing?", "who are you with?", "what are you with?" is fundamental for describing and collecting the behavior of individuals [34]. Motivated by this observation and our previous work [36, 37, 69] , we designed an ontology-based context model organized according to the aforementioned dimensions of the world: time, location, activity, social relations and object. Formally, context is defined as a tuple:

$$Context = \langle \text{TIME}, \text{WE}, \text{WA}, \text{WO}, \text{WI} \rangle$$

where:

- **TIME** captures the exact time of context, e.g., "morning". We refer to it as the *temporal context*. Informally, it answers the question "When does this context occur?".
- **WE** captures the exact location of context, e.g., "office room". We refer to it as the *location context*. Informally, it answers the question "Where are you?".
- **WA** captures the activity of context, e.g., "studying". We refer to it as the *activity context*. Informally, it answers the question "What are you doing?".
- **WO** captures the social relations of context, e.g., "friend". We refer to it as the *social context*. Informally, it answers the question "Who is with you?".
- **WI** captures the materiality of context, e.g., "smartphone". We refer to it as the *object context*. Informally, it answers the question "What are you with?".

3.3 Endurant and Perdurant Context

Context aggregates different dimensions of a person's environment, addition to that, context also accounts for the fact that it aggregates based on points of view, i.e., humans fundamentally use two elements to drive their representation: location and activity. We explain this with the notions of *endurant* and *perdurant* context.

According to [32], endurants are always wholly present at anytime they are present. Endurants are entities that are in time but lack temporal parts (all their parts flow with them in time), e.g., buildings and locations. While perdurants extend in time by accumulating different temporal parts. At any time they are present, they are present partially. Some of

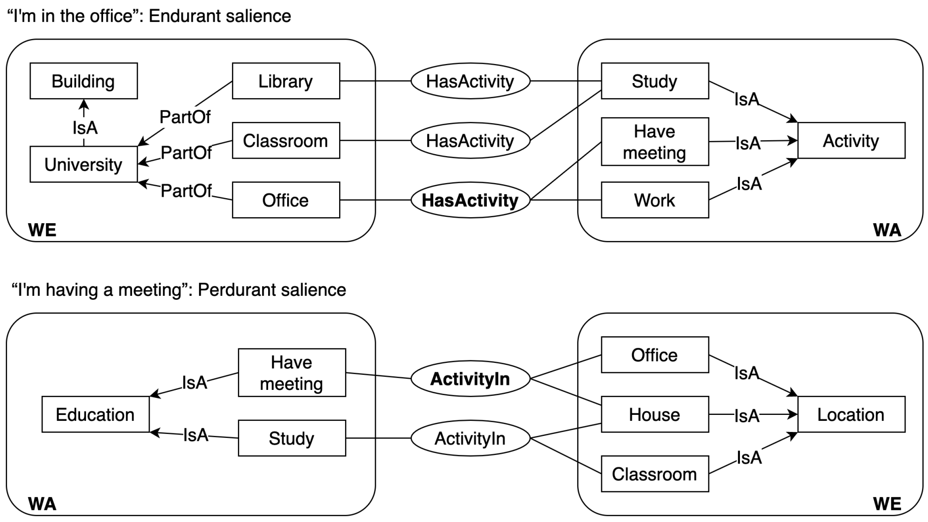


Figure 3.2: The difference between the notions of endurant and perdurant context.

their proper parts (previous or future phases) may not be present. Perdurant are entities that happen in time and can have temporal parts, all their parts are fixed in time, e.g., events and activities.

Therefore, the context can provide different representation of the same state of affairs depending on which element is more important. For instance, consider the scenario described in Section 3.2. In an endurant context, Wanyi could say "I'm in the office", implying a certain level of granularity within the building (in fact, saying "I'm at the university" would work too). In a perdurant context, Wanyi could say "I'm having a meeting", while some other activities may be happening at the same time, e.g., somebody is knocking at the door or Fausto and Andrea are discussion. The state of the world is the same, but the representation is different.

Figure 3.2 extends the scenario described in Section 3.2 by taking a subset of the ontology based on [38]. This work proposed an ontology unifying human perception and knowledge representation, thus corroborating how contexts allow for different perceptions of the environment based on location or activity. Notice that Figure 3.2 is at the level of the entity class from Figure 3.1, and focuses only on WA and WE for clarity's sake.

The two possible representation are as follows:

- **Endurant context:** "I'm in the office". In this representation, the office room is more relevant, while the activities that are performed are not fixed, either woking

Level	TIME	WE	WA	WO
Objective Context	2020-04-11 11:30am	Via Sommarive, 9, 38123 Povo TN	Talk	Prof. Fausto
Subjective Context	Morning	Office	Have meeting	Supervisor
Machine Context	1586597400000	46°04'01.9"N 11°09'02.4"E	Accelerometer: 0g,0g,0g	"Fausto" is in contact list

Table 3.1: An example of our three-partitioned context model.

or having a meeting could be possible.

- **Perdurant context:** "I'm having a meeting". In this representation, having meeting is more relevant among all the activities, while the possible locations are less relevant, and there may be different locations.

Notice that the relations in bold between locations and activities, i.e., "HasActivity" and "ActivityIn", are not simply inverse functions. In other words, $HasActivity = Activity^{-1}$ does not necessarily hold. For example, in the endurant context in Figure 3.2, "HasActivity" maps "Office" to both "Have meeting" and "Work", while in the perdurant context, "ActivityIn" maps "Have meeting" to not only "Office" but also "House". This shows that relations do not always have one to one mappings and the structure changes depending on the viewpoint.

These phenomena affect the identification process of the activity, because depending on which context is active, the elements to be identified and expected will change, along with possible services. For example, if a user is describing his context with endurant, it means the location is more relevant for him, and location-based services, e.g., sharing location with friends, may be more needed for him.

3.4 Objective, Subjective and Machine Context

The example in Figure 3.1 is presented in objective terms, that is, facts are stated as if they were independent of personal conscious experiences. However, each person interprets the world and her surroundings from her personal privileged point of view, which accounts for her personal knowledge, mental characteristics, states, etc. For instance, while in Figure 3.1 "Fausto" has an objective role of professor, for other people "Fausto" plays the roles of a “supervisor”, a “friend”, or a “father” subjectively. The subjective context which is related to personal consciousness, knowledge, etc. can provide more information for applications such as personal assistant in order to give more intelligent services.

Notice that a person’s view of her context is radically different from what her handheld personal assistant observes. In fact, machines interpret the world via sensors, while humans do not only interpret the world via their perceptions but with their knowledge as well. For instance, while a machine views location (e.g., a building) as a set of coordinates, humans interpret it based on its *function* (e.g., whether the building is their home or office).

To model context precisely and completely, in addition to considering four dimensions as discussed in Section 3.2, we also model three perspectives: objective context, subjective context and machine context. Table 3.1 shows the above example viewed through three types of perspective. The objective context captures the fact that at the University of Trento, Italy, at 11:30 AM, a person is talking to Fausto. When moving from objective to subjective, things change dramatically. From the perspective of the machine, the temporal context “11:30 AM” is viewed as a timestamp “1586597400000”, and in subjective terms it becomes “morning”; similarly, “University of Trento” becomes coordinates “Office” from a subjective perspective and “46°04’N,11°09’E” for the machine perspective. For the activity context, the activity of talking can be subjectively annotated as “Have meeting” by user, but it can be described as “connecting WIFI of classroom, sensors such as gyroscope, accelerometer are sensed as static”. For the social context, “Fausto” is described as “Supervisor” subjectively by the user and the machine senses “Fausto” is in the contact list of the user.

3.5 Personal Context Annotation

In Section 3.2, we present the definition of personal context model which is a tuple with different dimensions as components. Now we model each dimension as an ontology based on the general ontology in [38] unifying human perception and knowledge representation. Ontology can act as a hierarchy of labels to be used as annotations, and it contains more information than isolated labels. However, ontology is hard to use for users in real life since it can become arbitrarily complex and general. They must adapt to be usable by specific application or scenario.

Thus we present them to the users as time diaries, which is widely used in sociology to analyze human behavior where respondents report activities performed, location visited, and people encountered during their day [95].

Time use surveys are particularly relevant approaches, since they are widely used to investigate a specific aspect of people's time management, e.g., working, academic performance, and so on [12]. In fact, we based our modelling for activities on several time use surveys, especially the American Time Use Survey (ATUS) [90].

We apply and test our methodology in the SmartUnitn project that was introduced in Section 3. The interaction with the sociology experts in the project leads to an adaptation of our ontology to the experiment accounting based on the following methodological considerations:

- **Perdurannt context:** Since activities are the main focus of this experiment, the context to be mapped to the annotation is a perdurant context. It allows us to reflect the relevance of activities, since events are the aggregating elements for perdurant context.

- **Subjective context:** We model each dimension from human perception and allow the user to give their subjective annotations.

- **From ontology to annotation lists:** Following the sociology experts inputs, to make the ontology usable it has to be adapted to a list of annotations, without any sort of hierarchy. In fact, a simpler, leaner presentation is more likely to elicit and engage the users' answers, coupled with a controlled vocabulary for reducing possible ambiguities. In order to capture the most salient triple of location, activity and social relations [47], the annotations act as a list of possible answer for the corresponding questions, i.e. "Where are you?" (locations), "What are you doing?" (activities) and "Who is with you?" (social relations).

- **No WI context annotations:** In the case of this experiment, object context (WI) was not collected as it is hard to track without disrupting the user's routines. And the sociology experts do not deem the WI context relevant. Thus no object context annotation is required.

- **No Time context annotations:** We simplify the annotations of time to "morning", "afternoon", "evening" and "night", and it is easily inferred from the timestamp.

Thus we do not bother users to give annotations of temporal context, though it is an important dimension.

- **Order of the questions:** According to the sociology experts, and in general for time use surveys [47], activities are more relevant than locations and social relation in the experiment. Thus, the ordering of the three question reflects this hierarchy: activities first, locations second and then social relations.

- **No locations and activities constraints:** In activity recognition, locations can often act as constraints for the activities performed there [79]; for instance, when in bathrooms, people take a shower instead of cooking. However, from a sociological point of view, constraints may lead to a loss of valuable sociological data, e.g., students studying in places not explicitly designed for it, such as workplaces, bars or gyms. As a result, no constraints are imposed between the locations and activities annotation lists.

- **Adding "Other":** In time use surveys, the answer "Other" is a standard option with possible variations, e.g., the "n.e.c." field (i.e., Not Elsewhere Classified) in the ATUS [90]. Methodologically speaking, this means that the possible activity, location or social relation is outside the research scope of the sociologist, so it does not matter; "Other" covers such cases [12]. Ontologically speaking, "Other" acts as an element of openness, i.e., as a placeholder node in the ontology to accommodate and expand new pieces of information to be added in time to an ontology.

The mapping from three different dimensions of context to the annotation lists will be introduced in the following subsections.

3.5.1 Activities

Figure 3.3 shows the mapping of activities, i.e., the WA context, from the perdurant context and the question about activities. Here the annotations are adapted by the first tier of activities, especially for "Relax", which maps to 4 annotations, i.e., "Hobbies", "Cultural Activity", "Other Free Time", and "Social Life". This coarseness in the mapping is due to the fact that, in order to capture high level patterns, activities are required to be very general. Furthermore, more detailed activities, as underlined by the sociology

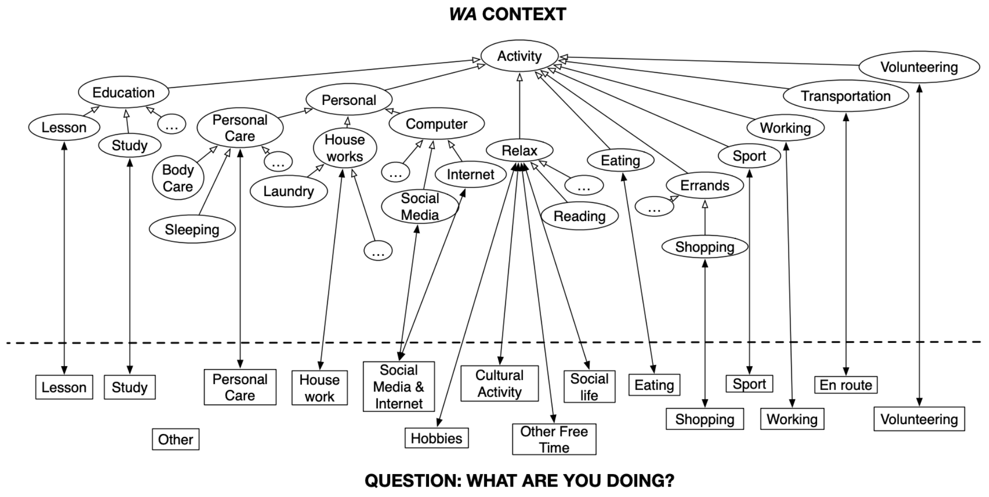


Figure 3.3: The mapping from the WA context to the activities annotation list.

experts, would cause more cognitive load in terms of memory for students and force them to answer more questions to reach an unnecessary fine grained level of detail.

3.5.2 Locations

Figure 3.4 shows the mapping from the locations, i.e., the WE context, of the perdurant context to the question about locations. Here the mapping is almost one to one with the lowest tier, except for "Other Universityplace" and "Other Home", since they group more specific types of buildings.

Notice that, even though "En route" is an activity, it refers to actual locations. So, if a student chooses it, then, instead of the options in Figure 3.4, a list of means of transportation is provided and the question is "How are you travelling?". The possible means of transportation are listed exactly as suggested by the sociology experts, i.e., "By Foot", "By Bus", "By Train", "By Car", "By Motorbike", and "By Bike". See the questionnaires that are introduced in Figure 3.6 and Figure 3.7

3.5.3 Social Relations

In the case of social relations, unlike locations and activities, the mapping is one to one, since they are a simple list in our current version of the WO context, as shown in Figure 3.5.

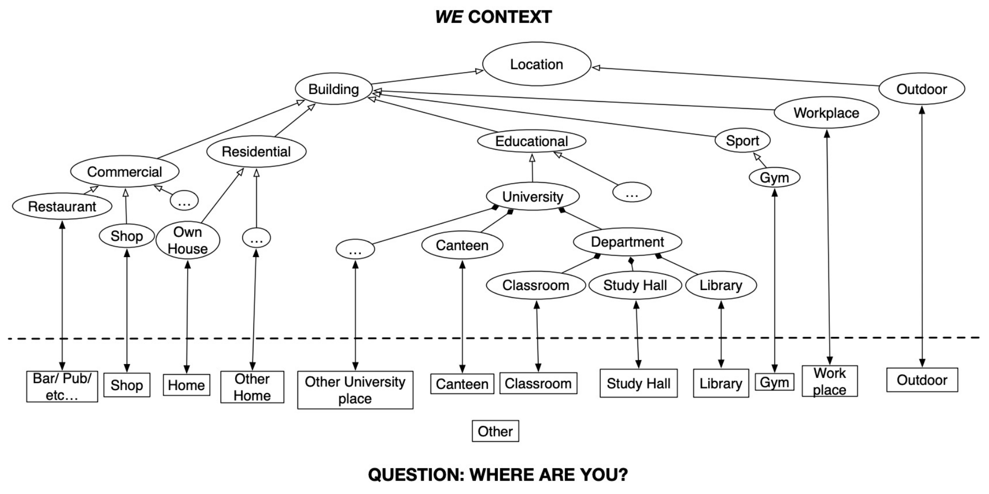


Figure 3.4: The mapping from the WE context to the locations annotation list.

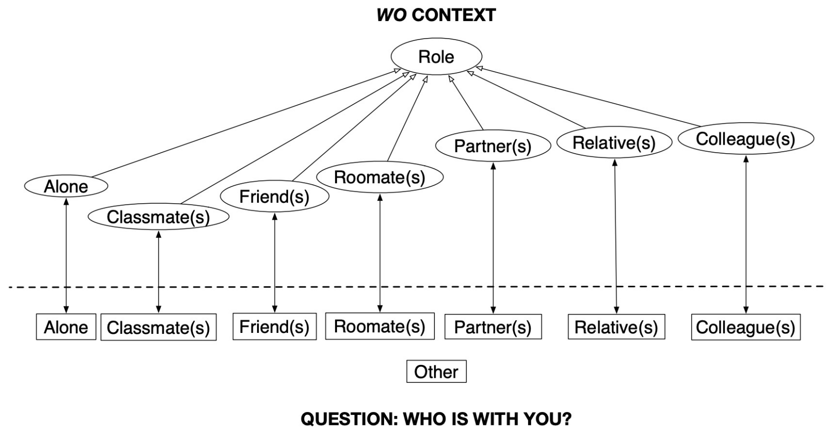


Figure 3.5: The mapping from the WO context to the social relations annotation list.

The three lists of annotations composes the questionnaire to be administered to the users. Figure 3.6 shows the one we used in SmartUnitn One Project. Each list of answers is the mapped set of annotations from Figure 3.3, Figure 3.4 and Figure 3.5, answering questions "What are you doing?", "Where are you?" and "Who is with you?" respectively. The link between the fourth question "How are you travelling?" and the "En route" activity is shown via an asterisk at the end of the latter. Similar to Figure 3.6, Figure 3.7 shows the one that we used in SmartUnitn Two Project. One more question was asked in that questionnaire, i.e., "What is your mood?", since in the second project experiment, we also wanted to understand the relationship between student's emotion and behaviours.

3.6 Use Case: Student's University Life

Academic performance is important to university students. Obtaining a university degree will help the student to get a good job. On the other hand, students' drop-out is a waste of economic and social resources within a society. How university students allocate their time between academic activity and other daily activity is their time management ability. Researches have shown that students' time management ability is important aspect that effects the students' academic performance. It is especially true for students in universities who have no parental support or teacher supervision.

To understand the students' time allocation, sociological researchers usually use surveys, which ask the users to report their total time amount and how they spend for given activities. However, traditional tools like survey suffers from the data quality issues due to the reliability of the users.

We ran the SmartUnitn One and Two experiments in 2016 and 2018 respectively to collect time use data from students in the University of Trento. The final goal of these experiments is to understand the how the students spend their time and how their time management ability affects their academic performance. Instead of using traditional survey, we designed and developed an App i-Log installed in students' mobile phone to collect students' answers and collect data of sensors embedded in the phone. We apply the personal context model in these two experiments when the students give annotations of their own life. Notice that, the data sets generated in these two experiments are the ones that we use for the experiments and analysis in the next few chapters.

3.6.1 SmartUnitn One

The population of the experiment

The first iteration of experiment is SmartUnitn One. It was carried out in the University of Trento in the late 2016. Generally speaking, the sample of SmartUnitn One consists of university students who enrolled in the academic year of 2015-2016. The selection of the population took account of the distribution in terms of gender, department and their economic status. To get available academic performance data, it requires the students taking part in the project must attend the lessons regularly during the experiment. Further more, since the i-Log is an App developed based on Android system, it requires the students to own an Android smartphone with 5.0.2 or higher operating system.

The length of the experiment

SmartUnitn One experiment had two phases. Each phase last for one week. The first phase starts from 29/11/2016. In this first week, the students responded to the time diaries and annotated their activities, locations and social relations by answering the questions. Meanwhile the i-Log App ran in the background collecting sensor data. The second phase starts from 06/12/2016. Different from the previous week, time diaries were not sent to the students, and the students only had to keep the App running in the background without answering the questions.

The frequency of data collection

Except the third-party data which can be obtained from the administrative office (e.g., GPA and ECTS), the rest types of data, smartphone-based data and survey-based data, can be collected by the functionalities provided by i-Log. And for different data, we set different collection frequency.

- Survey-based data. This data was the students' response to the time diaries sent by i-Log. In SmartUnitn One experiment, the time diaries were sent to the students only in the first week. Each time diary would be sent every 30 minutes. If the student did not answer it in time, it would be stores in a queue. After 5 time diaries stacked in the queue, the least one would be expired and its answer would be treated as null.

- Smartphone-based data. This data was collected from the sensors embedded in the smartphone. For different sensors, the data collection frequency was different. See the details in the Table 3.2:

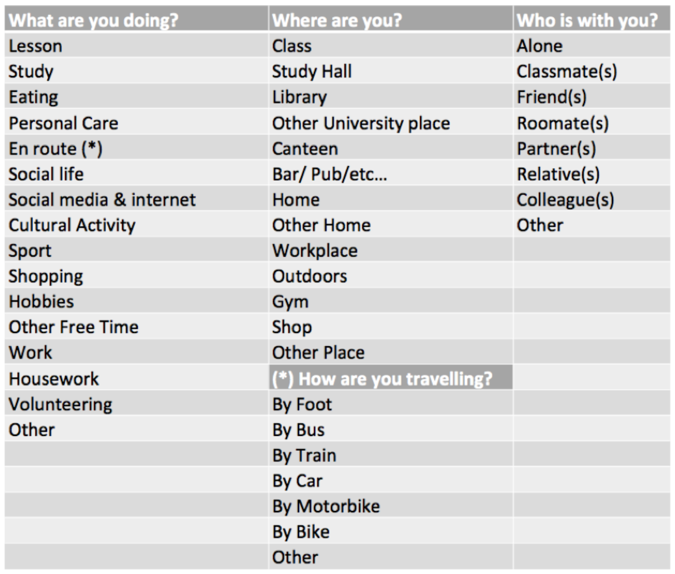
The time diary and annotation list

During the SmartUnitn One experiment, we use the time diaries to ask the students what context they were in at that moment. The questions that have been asked in the time diary

Table 3.2: Collection frequency of sensor data.

Sensor	Frequency	Sensor	Frequency	Sensor	Frequency
Acceleration	20Hz	Screen Status	On change	Proximity	On change
Linear Acceleration	20Hz	Flight Mode	On change	Incoming Calls	On change
Gyroscope	20Hz	Audio Mode	On change	Outgoing Calls	On change
Gravity	20Hz	Battery Charge	On change	Incoming Sms	On change
Rotation Vector	20Hz	Battery Level	On change	Outgoing Sms	On change
Magnetic Field	20Hz	Doze Modality	On change	Notifications	On change
Orientation	20Hz	Headset plugged in	On change	Bluetooth Device Available	Once every minute
Temperature	20Hz	Music Playback	On change	Bluetooth Device Available (Low Energy)	Once every minute
Atmospheric Pressure	20Hz	WIFI Networks Available	Once every minute	Running Application	Once every 5 seconds
Humidity	20Hz	WIFI Network Connected to	On change	Location	Once every minute

are show in Figure 3.6.



What are you doing?	Where are you?	Who is with you?
Lesson	Class	Alone
Study	Study Hall	Classmate(s)
Eating	Library	Friend(s)
Personal Care	Other University place	Roomate(s)
En route (*)	Canteen	Partner(s)
Social life	Bar/ Pub/etc...	Relative(s)
Social media & internet	Home	Colleague(s)
Cultural Activity	Other Home	Other
Sport	Workplace	
Shopping	Outdoors	
Hobbies	Gym	
Other Free Time	Shop	
Work	Other Place	
Housework	(*) How are you travelling?	
Volunteering	By Foot	
Other	By Bus	
	By Train	
	By Car	
	By Motorbike	
	By Bike	
	Other	

Figure 3.6: The questionnaire in SmartUnitn One.

3.6.2 Data and Tools

According to the final goal we set for the SmartUnitn projects, we define the following three types of data that we need to collect:

- Smartphone-based data: we collect these data from the sensors in accordance with the smartphones' specifications.

- Survey-based data: to understand the students' time allocation, we still need their annotations on their activities. Psychological questionnaires will be sent to the students and their answer labels will be sent back and collected.

- Third-party data: to understand the relationship between students' time management ability and their academic performance, we also need their academic data, such as GPA and ECTS from the administrative offices of the University of Trento.

To collect these data, a technology tool called i-Log [110] was used. It is a mobile phone App that can be installed in the students' phones and it provides two functionalities:

- **Data collection**: i-Log is designed to collect data from multiple sensors on phone simultaneously, from hardware (e.g., GPS, accelerometer, gyroscope and so on) to software (e.g., the applications running on the devices). A dedicated back-end infrastructure manages the tasks of synchronizing and storing the streams of data from the smartphones.

- **Time diaries**: i-Log could also push time dairies to the App users. Each time diary is composed of questions of activities, locations and social relations of students. The time diaries would be sent every 30 minutes, and the student user's answers could detail how he/she allocated his/her time during the day. Every time diary could be answered within 150 minutes, with a maximum of 5 time diaries stacked in a queue. And once a time diary get expired, its answer would be treated as null. To avoid bothering the students, these time diaries would appear on their phones as silent notifications. Due to the minor difference between SmartUnitn One and Two, the questions in time diaries are a little bit different, and they will be introduced in the next sections.

3.6.3 SmartUnitn Two

The population of the experiment

SmartUnitn Two was the second iteration of the experiments. It was carried out in 2018. The population design of this experiment is similar to the previous one. But this time we wanted to involve more participants so that we sent invitations to the entire student population of the University of Trento. The population excluded 1) the students who did not have a smartphone with Android system 5.0 or higher, 2) the students who did not participate in regular university life, and 3) the students who were born after 1993 to limit the dynamics of students out-of-school.

The length of the experiment

SmartUnitn Two experiment also had two phases. In this experiment, we tried to collect more data by involving more students and lengthening the experiment duration. The first phase started from 07/05/2018 and last for 2 weeks. In these weeks, the students need to let i-Log running on their phone and responded to the time diaries. The second phase starts from 21/05/2018 and last for another 2 weeks. In the rest weeks, to avoid bothering the students too much, only 5 time diaries would be sent everyday.

The frequency of data collection

The frequency of data collection in SmartUnitn Two was similar to SmartUnitn One. The smartphone-base data from sensors were collected by the same frequency as shown in Tabel 3.2. And the settings of sending time diaries in the first phases of both the experiments were the same, which is that the students had to answer the questions every 30 minutes. The only difference was that in the second phase of SmartUnitn Two, the students were still sent the time diaries but with lower frequency, one time diary in every 2 hours

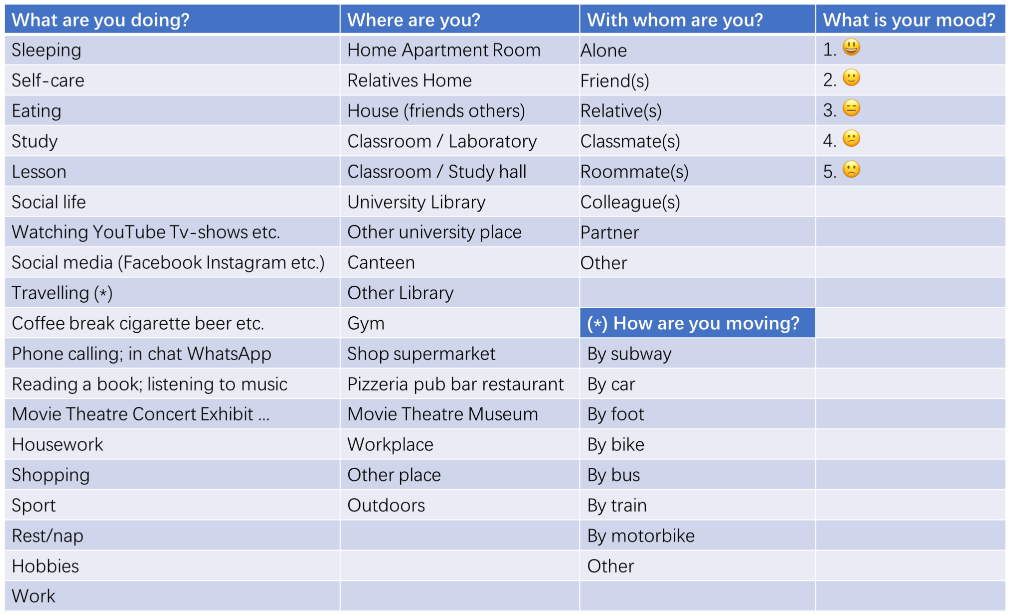
The time diary and annotation list

In the SmartUnitn Two, we updated the design of the questionnaires. We asked not only the student's context at that moment, but also their emotion. Figure 3.7 shows the questions that have been asked in SmartUnitn Two:

3.6.4 Date Sets

Data Set of SmartUnitn One

In SmartUnitn One, to initialize the experiment, 312 students enrolled in the first academic year in any of the bachelor courses active in 2016 at our University were contacted through a web survey to ask for their participation. From this initial population, 104



What are you doing?	Where are you?	With whom are you?	What is your mood?
Sleeping	Home Apartment Room	Alone	1. 😃
Self-care	Relatives Home	Friend(s)	2. 🙂
Eating	House (friends others)	Relative(s)	3. 😑
Study	Classroom / Laboratory	Classmate(s)	4. 🙁
Lesson	Classroom / Study hall	Roommate(s)	5. ☹
Social life	University Library	Colleague(s)	
Watching YouTube Tv-shows etc.	Other university place	Partner	
Social media (Facebook Instagram etc.)	Canteen	Other	
Travelling (*)	Other Library		
Coffee break cigarette beer etc.	Gym	**(*) How are you moving?**	
Phone calling; in chat WhatsApp	Shop supermarket	By subway	
Reading a book; listening to music	Pizzeria pub bar restaurant	By car	
Movie Theatre Concert Exhibit ...	Movie Theatre Museum	By foot	
Housework	Workplace	By bike	
Shopping	Other place	By bus	
Sport	Outdoors	By train	
Rest/nap		By motorbike	
Hobbies		Other	
Work			

Figure 3.7: The questionnaire in SmartUnitn Two.

students fulfilled three specific criteria: *i)* to have filled three university surveys in order to obtain their socio-demographics, shown in Table 3.3, and other characteristics, e.g., psychological and time use related; *ii)* to attend lessons during the period of our project in order to describe their daily behavior during the university experience, and *iii)* to have a smartphone with an Android version 5.0.2 or higher.

Overall, 75 students accepted to participate, but three of them declined during the project due to unexpected technological incompatibility with the application. In the end, the final sample consisted of 72 students to reflect the general population of freshman year students of our University in terms of gender and departments.

Table 3.3: Socio-demographics of students from the experiment

Gender		**Departments**		**Scholarship**		**Age**	
Male	**Female**	**Scientific**	**Humanities**	**True**	**False**	**Min**	**Max**
61.1%	39.9%	56.9%	43.1%	37.5%	62.5%	19	22

The students were asked to attend an introductory presentation where they were presented with the aims of the project and how to use the application. Those who decided to participate after this presentation were presented with a consent form to sign, after which they installed the application on their smartphones. Users were informed about all aspects of the management of their personal information concerning privacy, from data collection to storage to processing. Furthermore, before starting the data collection, we obtained approval from the ethical committee of our university.

The resulting data set contains 20 billion sensor values plus user annotations, resulting in a total size of 110GB. In addition, it is merged with both the pre and post project surveys collecting socio-demographic characteristics of students, their time use habits, some psychological traits measured by validated scales (i.e., pure procrastination scale and goal orientation scale), and academic performance data from the administrative office from our university. All the data were collected in compliance with the latest regulations in terms of privacy and are stored entirely anonymously.

Data Set of SmartUnitn Two

In SmartUnitn Two experiment, we sent e-mails to invite participation in the data collection to all 12000 regularly enrolled students at the University of Trento. The e-mail clearly explained that students could choose to participate in the study for two or four weeks, and that in the first two weeks they would receive a notification every half hour, while in the second two weeks every two hours. There were 273 students registered and logged in i-Log after the invitation. Remove the students who have just downloaded the App without answering it and those who abandoned the experiment during the App test (between 7th to 8th), there were 237 students remained. The total events recorded during the whole experiment were 130489. 110702 of those were detected in the first phase, and 19787 of those were detected in the second phase.

However, not all those 237 students performed correctly. Only 22 students answered all the time diaries in the first phase and 23 did in the second phase. If we relax the restriction of a "valid" participant of the first phase using the following standards, there were 184 valid participants in the first phase: 1)failure to reply at most 7 times for a continuous period of more than 10 hours, 2) completed the questionnaire for at least 13 days, and 3) provided a number of valid answers greater than 300.

3.7 Summary

In this chapter we introduced a personal context model based on [34], which is divided into five dimensions of the environment. We also introduced the notions of endurant and perdurant context, objective and subjective context. The notions are then used to create an ontology accounting for the way how perception guides humans to aggregate their description of their surrounding environment. Last but not least, we made some adaption of our ontology and generated three list of context annotations in open domains. The personal context model introduced in this chapter was applied in the SmartUnitn Projects where the questionnaire about the students' context was designed. As we mentioned in the Section 1.3, a good designed context model is necessary and fundamental to context recognition tasks. This is the context model that we proposed to use in the personal context recognition task. This model is further validated in the next chapter.

Then we introduce the SmartUnitn One and Two experiments, which belongs to the same family of projects, including the population of the experiments, the length of the experiments, the frenquency of data collection and the time diaries and the annotation list that were used in the experiments. There are three different types of data. One is smartphone-based data which come from the sensors embedded. Another one is survey-based data which is the users' annotations of their context during the experiment. And the last one is third-party data which is the users' academic performance. To collect the data, we used the i-Log application and the time diaries. We also introduce the generated data sets, which will be then used in the coming chapters.

Chapter 4

Predictability of the Personal Context

4.1 Introduction

The work in this chapter is based one of our publication [113].

Context prediction are usually used to recognize the current context. To do context recognition, we firstly try to understand how predictable are the various aspects of human behaviors. In the last decade, several works have investigated the role of randomness in human behavior and how predictable are various aspects of human activities such as mobility [8, 42, 94, 61, 93, 2, 16, 1], social interactions [27, 25, 65, 83], shopping [58, 19], and online [92] behaviors.

Research studies have also highlighted how similar mechanisms seem to govern different human activities. For example, people show a finite number of favourite places [2] and friends [65]. In a similar way, some individuals tend to explore and change favourite places [70] over time, as they do with friendships [65] and mobile phone apps [20], while others tend to maintain stable their behavior.

However, existing studies on human dynamics have been often limited to a single or to the combination of few behavioral dimensions (e.g. mobility and social interactions) [42, 94, 11, 24, 97]. Moreover, these studies have adopted the perspective of an outside observer who is unaware of the motivations behind the activities of a given individual.

In this chapter, we propose a different angle for analyzing the predictability of human behavior. In particular, our study revolves around the observation that, in typical circumstances, human behavior is deliberated based on an individual's own perception of the situation s/he is involved in, as captured by the notion of *personal context* which was introduced in Chapter 3. We analyze *regularity* and *diversity* in behavior through the joint interplay of four modalities of personal context (i.e. *time*, *location*, *activity*, and *social ties*) widely used in context-aware and ubiquitous computing communities [21, 99, 6, 37, 100].

In particular, we perform a rigorous statistical analysis of the effects of these four modalities of personal context on the predictability of human behavior using a month of collected mobile phone sensor readings and self-reported annotations about time, location, activity and social ties from more than 200 volunteers, which was introduced in Chapter 3. Our analysis leverages information theoretic techniques introduced by studies on human mobility [94, 76, 92] to characterize the predictability of individual behavior for single modalities and extends them to study correlations across distinct modalities. In addition, we look at behavior diversity across individuals through the lens of the four identified contextual modalities.

Our analyses and findings offer several pieces of evidence in support of the role played by the investigated contextual modalities. As a first step, we have estimated the performance of an *ideal*, *optimal* classifier for independently predicting each modality (i.e. time, location, activity, and social ties) for each individual in the data set. This showed that an optimal classifier with access to the previous annotations *for the same user and contextual modality*, but not their chronological order, cannot do better than 45% to 65% accuracy. In other words, ignoring correlations across time and between contextual modalities entails a large *irreducible error* of 35% to 55%, depending on the target modality. Disclosing the order of past annotations (again, available for the target modality only) makes the optimal classifier performs much better and the irreducible error decreases to 10% to 15%. However, supplying the optimal classifier with information about the other modalities (e.g. providing time, activity, and social ties while predicting location) but not their order decreases the irreducible error even more, below 5%. This shows that taking inter-modality correlations into account makes a substantial difference in the predictability of an individual behavior and supports the idea that inter-modality correlations may be more important than short- and long-term correlations over time. These results, which hold for *optimal* classifiers, were shown to carry over to practical classifiers (namely, Random Forests) in a location recognition experiment. This experiment also shows that some locations that are hard or impossible to predict using sensor data suddenly become easy to predict when information from time, activity, and social ties is taken into account. This further highlights the fundamental importance of the jointly interplay of different contextual modalities in behavior analysis.

Then, the analysis was extended to determine the impact of subjectivity on predictability of behavior. Consistently with our finding that activity and location are strongly tied, we compared the impact of injecting objective versus subjective location information on the performance of optimal classifiers for activity and social ties. Here, subjective location was implemented using self-reported annotations, while objective location was derived from GPS measurements. Our results support the argument that subjective location is far more informative than objective location for predicting behavior.

In a final experiment, we investigated the role played by the identified four contextual modalities in studying behavior diversity across individuals. The goal of this experiment was to determine whether common or uncommon behaviors are what distinguishes different individuals. The results clearly show that, first of all, the context distribution is heavy tailed, and therefore that contextual modalities offer support for analyzing "rare" behaviors, and second that annotations in the tail of the context distribution are much more effective than those in the head at identifying individuals. This was verified in a practical identity recognition experiment.

4.2 An Overview of the Dataset

In this work, we use the personal context model that was introduced in Chapter 3. We focus on fore modalities of the context, i.e., temporal context, activity context, location context and social context, which are widely used in ubiquitous computing communities for capturing and describing situations occurring in daily life [21, 99, 6, 37, 100]. The context can be represent as a tuple: $Context = \langle \text{TIME, WE, WA, WO} \rangle$. Thus the example we gave in Chapter 3, i.e., Ph.D student Wanyi is having meeting with her supervisor Prof. Fausto in a office, can be represented as $\langle$"morning", "office", "have meeting", "supervisor"$\rangle$

4.2.1 Data PreProcessing

The data we use in this work is generated in the project SmartUnitn Two, which was introduce in Chapter 3. As previously said, the data collection was split in two phases, each two weeks long. In the first phase (7th to 24th of May) questionnaires were submitted to the volunteers every 30 minutes, while in the second one (25th of May to 7th of June) every 2 hours, to lessen the cognitive load. In this second stage, the volunteers were also specifically requested to leave the app running at all times.

Before analyzing the predictability of personal context, we firstly clean the raw annotations and give an overview of the dataset.

In a first step, a simple criterion was used to identify valid (that is, "trustworthy") study participants. A participant was deemed valid if s/he failed to reply no more than 7 times within any 10-hour window, completed all questionnaires for at least 13 days, and provided at least 300 valid answers. All of these conditions must hold for a study participant to be deemed valid. A total of 184 study participants were marked as valid. The records of all invalid study participants were discarded. The next step was to delete events with invalid or missing values (like empty string labels) and records spuriously occurring outside the date window, i.e., before the 8th of May or after the 5th of June. Finally, in order for all statistics of the data in the two phases to be directly comparable, the records obtained from the second phase were replicated four times.

4.2.2 Number of Annotations per Participant

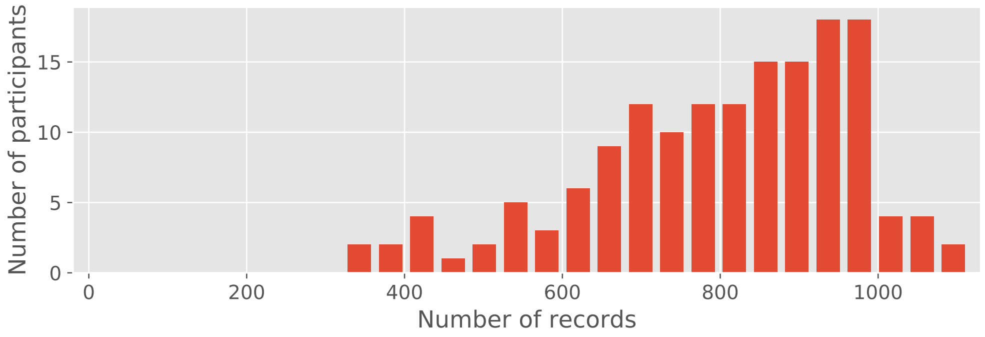


Figure 4.1: Number of annotations per participant in the final data set.

After the processing, there are 156 study participants with 124,963 records in total in the data set. The processed dataset consists of several time series $\vec{x}^{u,m}$, one time series for each valid study participant $u \in \{1, \dots, 156\}$ and contextual modality $m \in \{\text{TIME}, \text{WE}, \text{WA}, \text{WO}\}$. Each time series can be viewed as a vector $\vec{x}^{u,m} = (x_1^{u,m}, \dots, x_T^{u,m})$, where T is the number of questionnaires administered to a study participant during the collection procedure and every $x_t^{u,m}$, with $t = 1, \dots, T$, indicates the annotation for modality m reported by study participant u at time t. The number of annotations per study participant, reported in Figure 4.1, shows that most participants have in-between 400 and 1000 records.

4.2.3 Distribution of Annotations

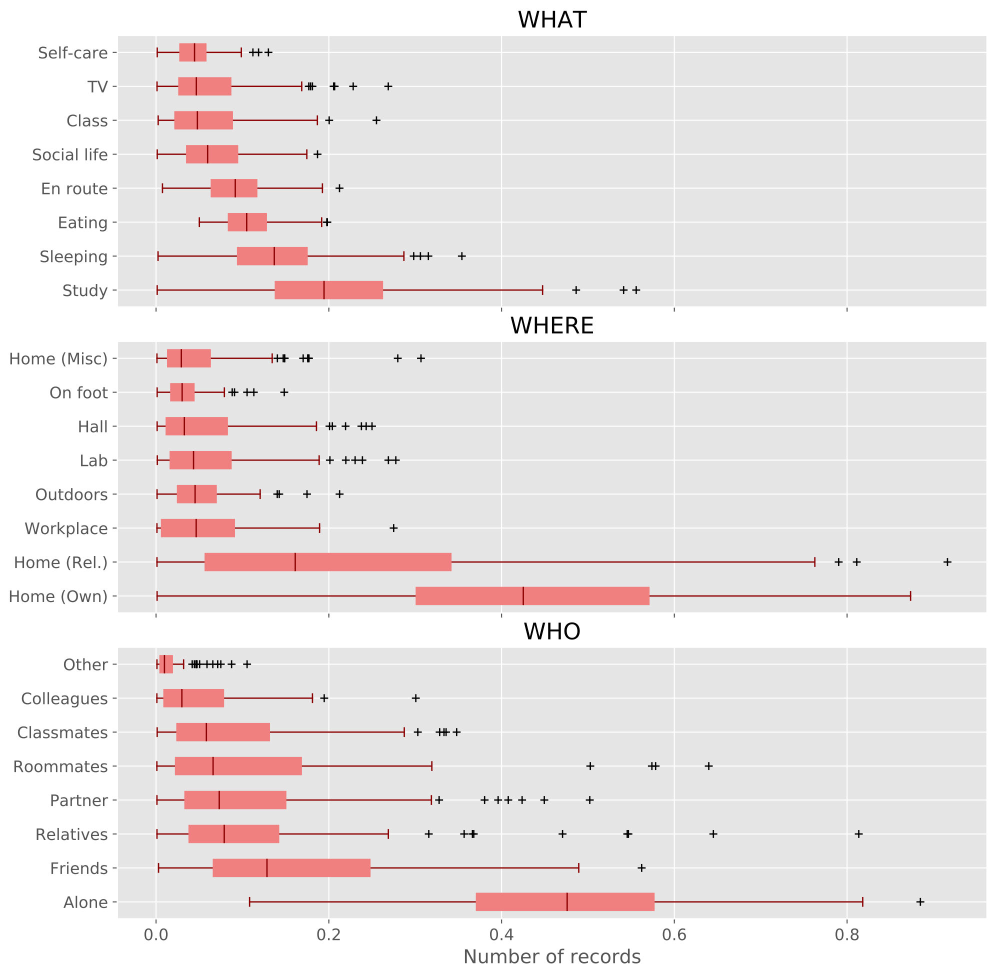


Figure 4.2: Value distribution of different aspects. From top to bottom: WA, WE, and WO.

Figure 4.2 reports the distribution of annotations in the data. The plot shows that, for all contextual modalities, few values take up most of the mass. Only the eight most frequent annotations are shown for each aspect. Roughly speaking, this means that study participants spend most of their time performing four basic activities (namely studying, sleeping, eating, and moving between locations, which account for about 55% of the records), mostly stay at home (either their home or their relatives, more than 50%), and mostly by themselves or with their friends (almost 50% and 16%, respectively). The boxes extend from the 1st to the 3rd quartiles, while the bars extend to ± 1.5 inter-quartile range from the median. Study participants with very high annotation frequency (i.e. outliers) are denoted by crosses.

TIME is special in that its annotations are extremely regular and mostly determined by the experimental setup rather than by individual preferences. This is especially true for nocturnal annotations, as the user can set i-Log to "sleep mode" so that it will automatically reply to the questionnaires accordingly during the night. For this reason, TIME is omitted from the figure.

The profile transpiring from the data reflects the source demographics.[1] The concentration of mass on few preferred values is consistent with previous studies on mobility [94, 2].

4.2.4 Regular and Irregular Users

To get an intuition of the regularity in the behavior of volunteers, we selected two participants with the highest and lowest annotation diversity and visualize their annotations in Figure 4.3. For each participant, sub figures from left to right represent his annotations for What (activity), Where(location), and Who(social relation). Each row in sub figures is a day, and columns are hours. Weekends are indicated on the y axis. Colors indicate different values. The values were grouped by similarity, for interpretability. Missing values are in blue.

Figure 4.3 shows that the most regular study participant has a distinctly simpler behavior than the other one, as expected. The figure also shows that even the behavior of the more regular volunteer is still quite irregular and displays substantial variability across days and across weeks.

[1]Considering that the volunteers are university students, the self-reported amount of studying is likely to be a (slight) over-estimate.

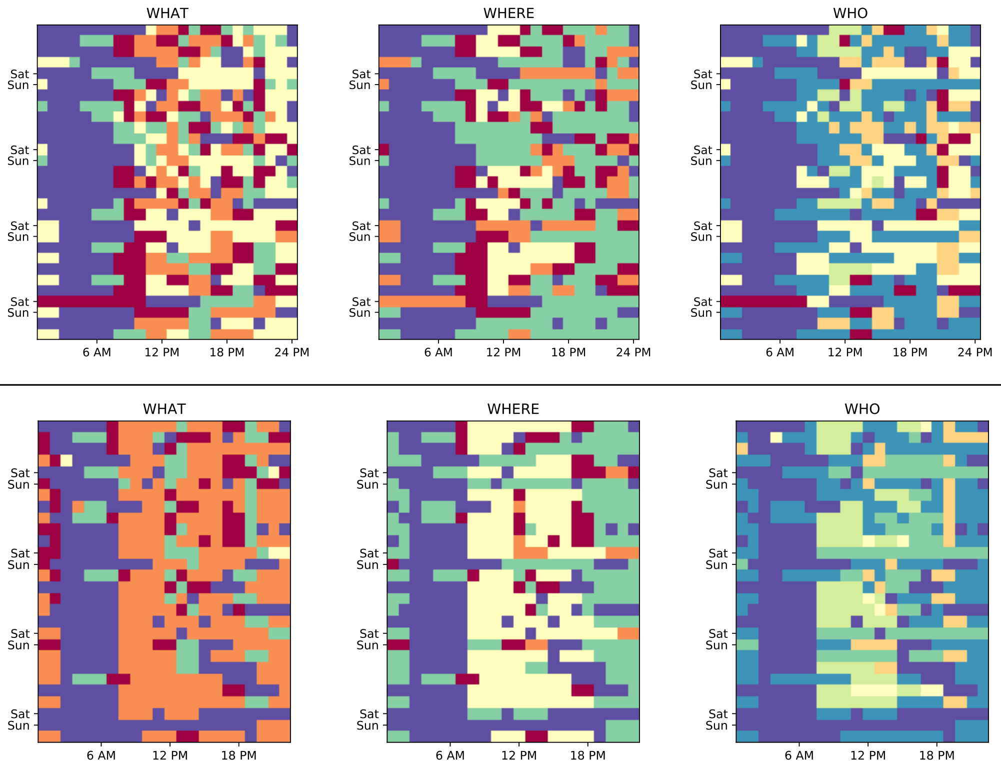


Figure 4.3: Annotations of an irregular (top) and a regular (bottom) study participant.

4.3 Entropy

We are interested in understanding to what degree individual modalities are predictable and whether some modalities are inherently more predictable than others. In line with previous work [42, 94, 76, 92], we answer these questions using *entropy* and *predictability*.

Entropy measures the number of bits necessary to encode a random source: an entropy of b bits indicates that, on average, an individual who chooses her/his next value (i.e. location, activity, or social tie) randomly according to the ground-truth distribution will be found in 2^b distinct states with high probability [15]. Hence, higher entropy implies higher uncertainty. In order to evaluate the contribution of different factors, consistently with previous studies [94, 92], we estimated three forms of entropy:

(1) The *random entropy*, defined as:

$$H_{\text{rand}}(X^{u,m}) = \log_2 N^{u,m}$$

where $X^{u,m}$ is a random variable that represents the value of modality m for individual u and $N^{u,m}$ is the number of distinct values observed for that modality and individual in the full data set. The random entropy assumes that the study participant is equally likely to choose any of the values that s/he has annotated.

(2) The *time-uncorrelated* or *flat entropy*, defined as:

$$H_{\text{flat}}(X^{u,m}) = -\sum_x \Pr(X^{u,m} = x) \log_2 \Pr(X^{u,m} = x)$$

where the sum runs over all the possible values for modality m and $\Pr(X^{u,m} = x)$ denotes the empirical probability that individual u reported value x for modality m, as estimated from the data. The flat entropy is more informed than the random entropy as it takes the full value distribution into account.

(3) The *true entropy*, defined as the limit of the joint entropy:

$$H_{\text{time}}(X^{u,m}) = \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} H(X_1^{u,m}, \ldots, X_t^{u,m})$$

Here $X_t^{u,m}$ is a random variable that captures the value provided by individual u for modality m at time t, and the joint entropy $H(X_1^{u,m}, \ldots, X_t^{u,m})$ measures the disorder of t random variables:

$$\begin{aligned} -\sum_{x_1,\ldots,x_t} & \Pr(X_1^{u,m} = x_1, \ldots, X_t^{u,m} = x_t) \\ & \times \ \log_2 \Pr(X_1^{u,m} = x_1, \ldots, X_t^{u,m} = x_t) \end{aligned}$$

Compared to the flat entropy, the true entropy takes correlations over time, including short- and long-range correlations, into account. The true entropy is estimated from the data using the Lempel-Ziv estimator [55].

4.4 Predictability

While entropy measures uncertainty, it only gives indirect information about how "easy to guess" a random source is. This is better captured by the notion of *predictability*, which was introduced to assess regularity of human mobility [94]. Formally, the predictability $\Pi(X) \in [0, 1]$ of a random variable X is the accuracy of an *optimal* classifier for X, that is, the probability that this classifier outputs the correct value. As a consequence, if the predictability of a random variable is 0.8, then no classifier can have an accuracy higher than 80% – or, in other words, all classifiers must be mistaken 20% of the time. This means that predictability measures the *irreducible error* intrinsic in a random source. A notable property of the predictability Π is that, thanks to Fano's inequality [15], it can be derived directly from the entropy H by solving the equation:

$$H = -(\Pi \log_2 \Pi + (1-\Pi)\log_2(1-\Pi)) + (1-\Pi)\log_2(N-1) \tag{4.1}$$

Here N is the number of distinct values that X can take. Please see [94] for a detailed derivation. For our goals, it suffices to know that, very intuitively, lower entropy entails higher predictability. In order to measure the effect of annotation distribution and correlations over time, the predictability of each individual u and modality m was obtained by solving Eq. 4.1 using the random, flat, and true entropy. The resulting values are indicated as $\Pi^{u,m}_{\text{rand}}$, $\Pi^{u,m}_{\text{flat}}$, and $\Pi^{u,m}_{\text{time}}$, respectively.

4.4.1 Intra-modal Entropy and Predictability

Figure 4.4 illustrates the distribution of entropy (left) and predictability (right) for each modality. From top to bottom, they represent for WA, WE, and WO. The bar height indicates the number of participants. The histograms show that while all modalities are to some extent regular, some are more regular than others. This is partly due to the fact that the theoretical maximum of the entropy is $\log_2 N^m$, and it is controlled by the number of possible values N^m for modality m. Hence, modalities with more states, like activity and location, are inherently more uncertain and less predictable than modalities with fewer states. In our setting, the theoretical maximum of the entropy (represented in the entropy plots by a green line) is about 4.4 for location, 4.3 for activity, and 3 for social tie.

The plots show that entropy is largely determined by distributional information, and short- and long-range correlations always impact the measured entropy: random entropy (blue) is always much higher than flat entropy (red), which is itself much higher than true entropy (purple). These changes in uncertainty demonstrate that taking annotation distribution and time correlations into account can substantially lower uncertainty and increase predictability. The same effect can be observed for all modalities, with some differences. For all entropy measures, the WA modality has the highest entropy, followed by WE and WO. However, the difference between modalities is more pronounced for the

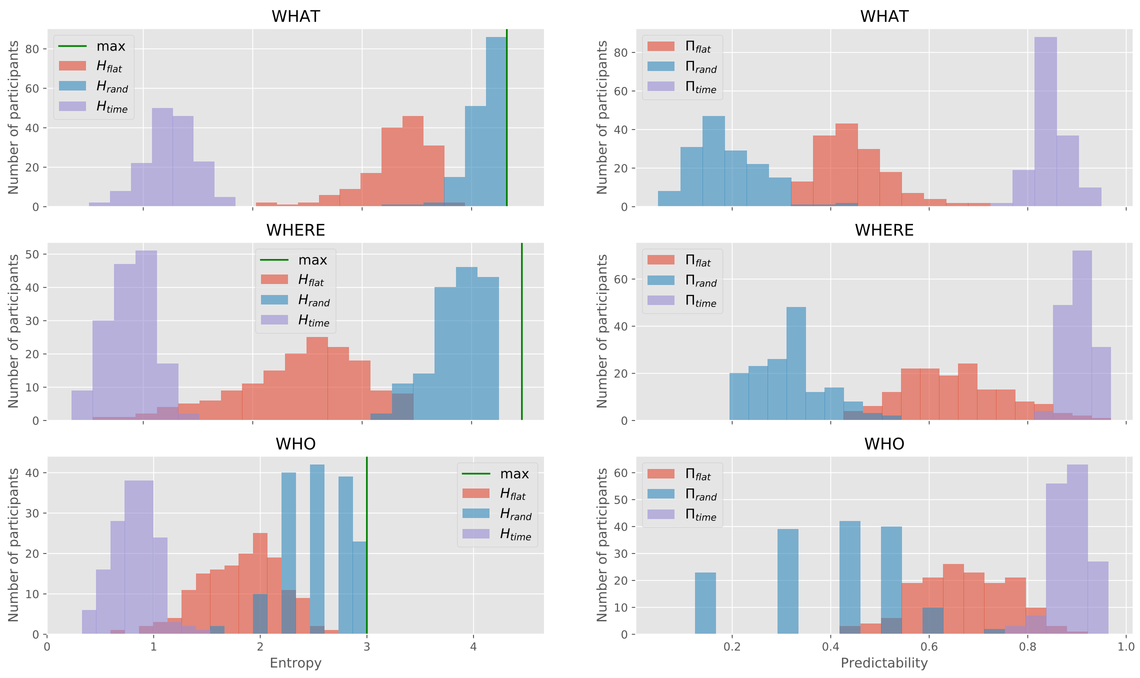


Figure 4.4: Empirical distribution of entropy (left) and corresponding predictability (right).

random and flat entropy, while it is limited for the true entropy, confirming the usefulness of taking time correlations into account.

Modality	H_{rand}	H_{flat}	H_{time}	Π_{rand}	Π_{flat}	Π_{time}
WHAT	4.11 ± 0.15	3.32 ± 0.31	1.25 ± 0.23	0.20 ± 0.07	0.45 ± 0.07	0.85 ± 0.03
WHERE	3.88 ± 0.24	2.43 ± 0.57	0.87 ± 0.22	0.32 ± 0.07	0.65 ± 0.10	0.90 ± 0.03
WHO	2.58 ± 0.30	1.82 ± 0.37	0.82 ± 0.20	0.40 ± 0.15	0.67 ± 0.10	0.89 ± 0.04

Table 4.1: Empirical entropy (left) and predictability (right) averaged over all study participants and standard deviation thereof.

Figure 4.4 (right) shows predictability of each modality for the different types of entropy. Comparing these histograms with those on the left makes it clear that increasing the amount of information dramatically increases predictability, as expected. Table 4.1 reports means and standard deviations of empirical entropy and predictability of each modality and type of entropy. The predictability for the true entropy Π_{time} (and hence maximal prediction accuracy) is 85% for activity, 89% for social tie, and 90% for location. This entails that irreducible error, even when taking all the available information into account, is about 10%–15% across modalities. The irreducible error for the flat entropy is even larger, 35%–55%.

The standard deviation of predictability – that is, the spread of the histogram – does considerably shrink as more information is taken into consideration. This points at the fact that, as more information is considered, all participants appear to act more predictably. It is worth noting that, however, the standard deviation of the true entropy is non-zero, hinting at the fact that some participants are *intrinsically* less predictable than others. This partially motivates our study of behavior diversity across individuals, presented later on.

4.4.2 Inter-modal Entropy and Predictability

So far, we have studied individual modalities taken in isolation. This approach is simplistic in that it ignores correlations between modalities, which we hypothesize to be very significant. In the following, we study the effect of inter-modal correlations on predictability.

This is achieved by estimating the conditional entropy $H(X^{u,m}|X^{u,m'})$, which quantifies the number of bits b needed to encode a random source $X^{u,m}$ assuming that $X^{u,m'}$ is known (with $m' \neq m$). Intuitively, the more $X^{u,m'}$ influences or determines $X^{u,m}$, the lower the conditional entropy [15]. The conditional entropy is defined as:

$$H(X^{u,m}|X^{u,m'}) = \textstyle\sum_{x'} \Pr(X^{u,m'} = x')H(X^{u,m}|X^{u,m'} = x')$$

where $H(X^{u,m}|X^{u,m'} = x')$ is the entropy of $X^{u,m}$ estimated only on those records that satisfy $X^{u,m'} = x'$. An issue with conditioning is that it is incompatible with the full entropy H_{time}, as it breaks time correlations: two non-consecutive records may appear to be consecutive in the conditional data set simply because they satisfy the same condition $X^{u,m'} = x'$ and none of the records in-between them does. This means that the conditional and unconditional entropy cannot be compared directly.[2] For this reason, in the following we use the flat, time-uncorrelated entropy H_{flat} in all computations.

The reduction in flat entropy due to conditioning, averaged over all study participants, is illustrated in Figure 4.5 (top). From left to right, the target modality is WA, WE,

[2]A naïve comparison shows that the conditional full entropy appears to be larger than the unconditional full entropy, which is clearly impossible.

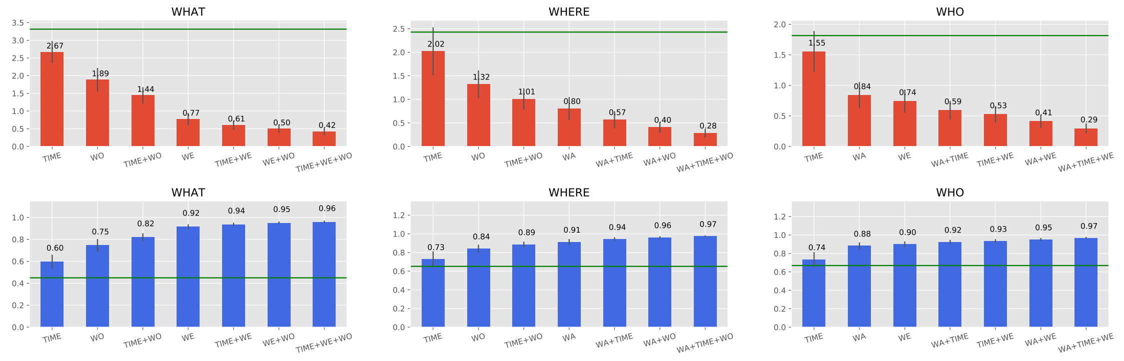


Figure 4.5: Entropy (top) and predictability (bottom) of each modality after conditioning on all subsets of other dimensions, averaged over all study participants.

and WO, respectively. The green line represents the entropy prior to conditioning (as reported in Table 4.1), while the red bars represent the conditional entropy. The change in predictability is reported below in the same figure.

The plots show very clearly that *in all cases, inter-modal information substantially reduces uncertainty and improves predictability.*[3] Indeed, conditioning any modality on the rest of the context (including TIME, rightmost bar in the plots) reduces entropy by more than 80% and increases predictability by at least 30%. More in detail, upon conditioning on the full context model, the entropy drops from 3.32 to 0.42 for WA, from 2.43 to 0.28 for WE, and from 1.82 to 0.29 for WO. At the same time, the predictability goes from 0.45 to 0.96 for WA, from 0.65 to 0.97 for WE, and from 0.67 to 0.97 for WO, cf. Table 4.1. This shows that the potential gain in accuracy from using multi-modal contextual dimensions is extremely large for all the modalities. The results for predictability make this point even clearer, as conditioning gives an impressive reduction of the *irreducible error* (that is, $1 - \Pi$). In particular, the irreducible error of WA sees a huge drop from 55% to 4%, that of WE from 35% to 3%, and that of WO from 33% to 3%. This is consistent with our argument that time, location, activity, and social ties strongly influence each other, and provides empirical evidence in favor of our approach of taking into consideration all the four contextual dimensions.

The magnitude of entropy reduction is largely independent of the target modality: conditioning reduces entropy of WA by 84%, of WE by 86%, and of WO by 81%, and

[3]The conditional entropy is – by definition – never larger than the unconditional entropy, that is, $H(X \mid X') \leq H(X)$, regardless of the relation between X and X'. Still, if X' is independent of X, then conditioning has no effect on entropy. This is clearly not the case in our plots.

increases predictability by 160%, 133%, and 131%, respectively. At the same time, some modalities appear to carry more information than others: while conditioning on TIME shrinks entropy by only 15-20%, conditioning on WO, WA, and WE reduces entropy by 45%, 54–67%, and 59–77%, respectively. The four modalities can be ordered by average impact as $TIME \prec WO \prec WA \prec WE$, meaning that $TIME$ is the least informative modality and location the most informative one in the setting under investigation in this study. The largest impact is observed when conditioning activity on location or vice versa, although conditioning on multiple modalities makes this effect more noticeable.

Comparing these results, which refer to flat entropy and predictability and that therefore ignore correlations over time, with those for full entropy supports the idea that inter-modal correlations are more influential than pure temporal correlations. Indeed, the full entropy of WA, WE, and WO reported in Table 4.1 are 1.25, 0.87, and 0.82, respectively, while the flat entropy obtained upon conditioning on the rest of the context is much lower, namely 0.42, 0.28, and 0.29, respectively.

4.5 Location Prediction

The above analysis shows that taking multiple contextual modalities into account helps to identify regularities in the behavior of individuals. Along this line, we also expect that some activities, locations, or social relationships cannot be predicted unless information from other modalities is available. Furthermore, while predictability measures the performance of an *optimal* classifier, it is important to study whether improvements in predictability due to conditioning affect the performance of real classifiers *in practice*.

To investigate this issue, we carried out a practical location prediction experiment. Specifically, we measured the difference in prediction performance between a prototypical statistical classifier [48] that predicts location from sensor measurements and that of analogous classifiers that were additionally given annotations about activity and/or social ties. As for the classifier, we opted for Random Forests due to their performance and reliability [7].

We trained one Random Forest classifier for each participant u. Each Random Forest takes as inputs the sensor measurements s_t^u of user u at time t – and optionally the annotations for the activity $x_t^{u,\mathrm{WA}}$ and social ties $x_t^{u,\mathrm{WO}}$ – and predicts the corresponding location $x_t^{u,\mathrm{WE}}$. For simplicity, the sensor measurements s_t^u were restricted to features

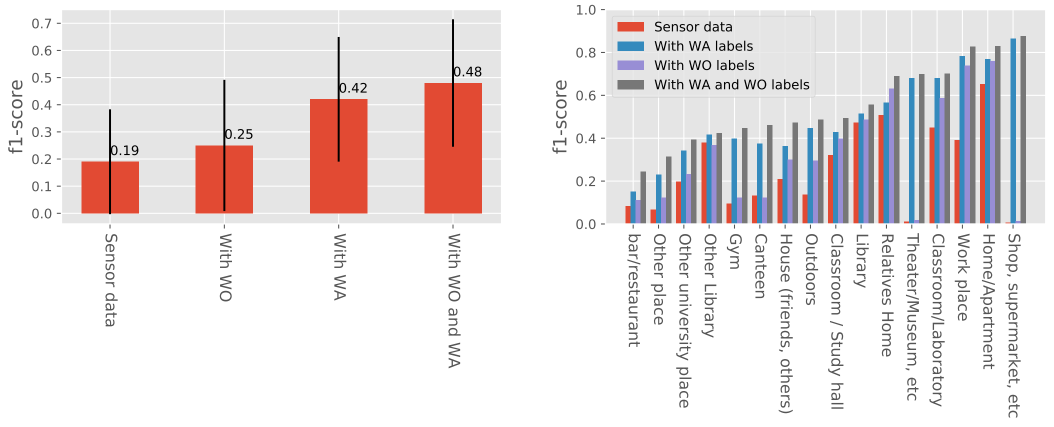


Figure 4.6: Left: Macro mean F_1 scores achieved by a Random Forest classifier for location using sensors only, sensors with WO, sensors with WA, and sensors with WO and WA. Right: per-label F_1 score achieved by the same classifier for individual locations.

derived from GPS information, and in particular to longitude, latitude, and total distance traveled by the subject since the last questionnaire. This simple setup is sufficient for location prediction, and readings from the other sensors were found empirically to not be very relevant for the task at hand.

Prediction performance was evaluated using a 5-fold cross validation procedure. Namely, for each study participant, her/his records were randomly partitioned into 5 folds: one fold was used for performance evaluation while the remaining four were used for training the classifier. This step was repeated five times by varying the test fold. The performance of the Random Forest was taken to be the average over the five repeats.

For each user, we evaluated the impact of inter-modal annotations by comparing the performance of four classifiers: a baseline Random Forest that uses only GPS-derived inputs s_t^u and three Random Forests – with the very same depth – that were given also annotations for WA and/or WO as inputs. All hyper-parameters were kept to their default values[4]. except for forest depth, which was selected on a separate validation set to optimize the performance of the *baseline* Random Forest. In order to account for annotations skew (i.e. some locations are naturally more frequent than others), performance was measured using the macro F_1 score. The latter is simply the F_1 score of individual locations averaged over all locations.

The overall macro F_1 scores averaged across study participants, as well as a break-

[4]As given in the `scikit-learn` package, version 0.24 [71]

down of the F_1 scores for individual locations, are reported in Figure 4.6. The plots show that GPS information can predict reasonably well several locations (red bars), like "Home", "Relative's home", and "Library", among others, on which the baseline Random Forest achieves 40% F_1 score. We conjecture this to be partially due to the fact these locations are very specific – in our data, the home of most users is unique and often easily identified from even few examples – and partially due to the abundance of annotations for these locations, cf. Figure 4.2. GPS information, however, is clearly insufficient for locations like "Shop/Supermarket/etc.", "Theater/Museum/etc.", "Gym", which are far more generic. Here the baseline Random Forest performs very poorly. This can be explained by two facts. First, these locations are composed of multiple objective locations (e.g. different shops, some of which possibly never observed during training), and therefore they are harder to predict based on GPS data alone. Second, the number of annotations for these locations is much lower.

Performance dramatically improves once WA and WO are supplied as inputs. In particular, the overall F_1 score increases by about 30%. Moreover, while knowledge of either WO or WA always helps recognition performance, supplying both improves performance even more, as expected. We also note that WA is more useful than WO in general. These observations are consistent with the results for the optimal classifier.

One question is whether these results are influenced by the performance of particularly easy to predict classes. We assessed this possibility by computing a variant of the macro F_1 that considers the *median* (rather than the mean) performance over classes, and as a result is naturally insensitive to classes that perform exceptionally well or exceptionally badly. The results are as follows: the macro mean F_1 for the four cases (sensors only, sensors with WO, sensors with WA, and sensors with WO and WA) is 0.19, 0.25, 0.42 and 0.47 respectively, whereas the macro median F_1 is 0.09, 0.13, 0.43 and 0.47. The more significant difference between macro mean and median F_1 appears when no activity information is present: the *baseline* drops by about 10% and the "with WO" Random Forest by 13%. However, the latter can be almost entirely explained by the former: adding social information contributes roughly $+5\%$ to both macro mean and median F_1 (from 0.19 to 0.25 and from 0.09 to 0.13, respectively). Summarizing, this shows that the macro mean F_1 overestimates the quality *of the sensor-only baseline* by about 10%. This probably occurs because all on-the-way locations like driving and walking are very hard to predict from sensors only (they individually achieve less than 8% F_1), meaning that the macro median F_1 tends to consider the higher-performing classes as outliers and ignores them. Most importantly, the contribution of inter-modal information to predictive performance is confirmed even by this more strict metric.

An important finding of this experiment is that some locations that were completely unpredictable from GPS data alone, are much easier to recognize if WA and WO annotations are supplied as inputs. The two most impressive examples are "Shop/Supermarket/etc." and "Theater/Museum/etc.", in which the correlation between location and activity boosts the F_1 score from less than 5% to more than 70%. This very encouraging result offers further support for the jointly leverage of different contextual modalities, as some locations that are essentially impossible to recognize suddenly become essentially trivial to recognize when rich contextual information is provided.

4.6 Subjectivity and Predictability

Here, we investigate whether subjective annotations are more relevant than objective ones for determining predictability of behavior.

In particular, we compared the reduction in entropy due to conditioning on subjective location (namely, the WE annotations) to that due to conditioning on objective location, interpreted here in terms of GPS coordinates and related information. As in the location recognition experiment, we defined objective location using longitude, latitude, and total distance travelled since the last questionnaire. Computing the conditional entropy for continuous variables – in our case, the GPS coordinates – is not statistically straightforward. In order to avoid issues, we discretized the GPS information using a simple binning procedure. In particular, we allocated $k = 3$ equal size bins for each of the three dimensions (longitude, latitude, amount travelled), for a total of 27 values for the objective data. This is done by using the KBinsDiscretizer class provided by scikit-learn [71] using the "quantile" strategy, which ensures that all bins contain roughly the same number of points. The number of bins roughly matches the number of subjective values (i.e. locations), which are 22. Since the variance of the conditional entropy estimator depends strongly on the number of alternative values, our choice of having roughly the same number of values for both subjective and objective data discourages the estimator from having dramatically different variances for the two cases.

A comparison of conditional entropy of WA and WO obtained by conditioning using

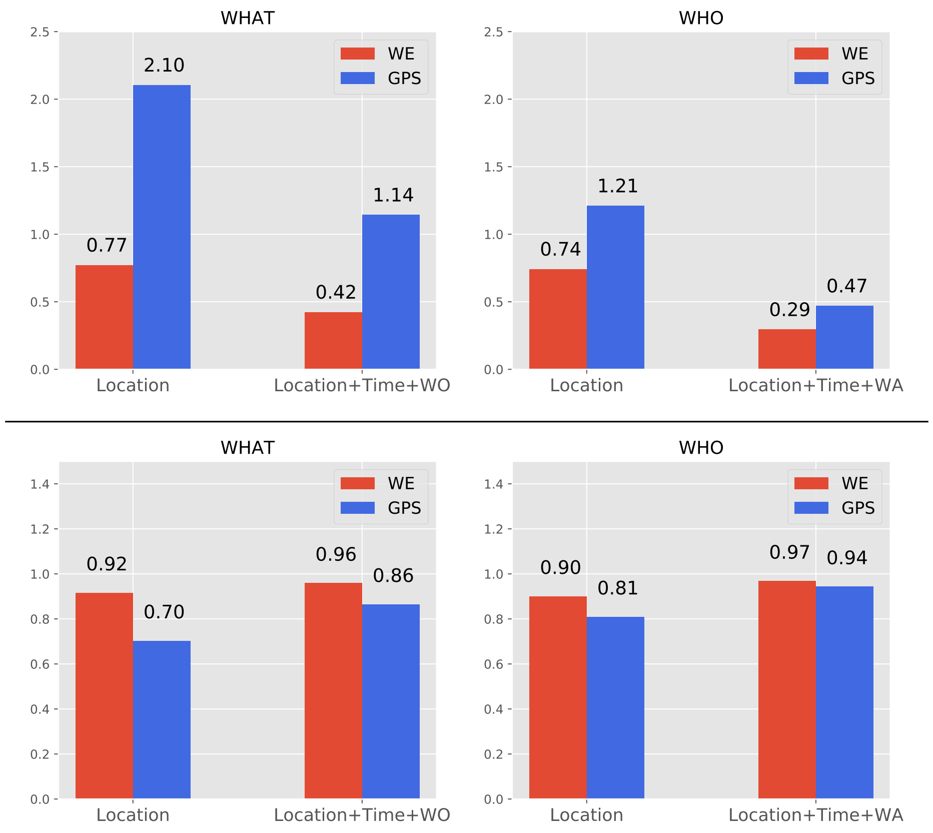


Figure 4.7: Entropy (up) and predictability (down) of modality WA and WO after conditioning on subjective labels (red) and objective labels (blue).

subjective (red) versus objective (blue) location is reported in Figure 4.7. The two left bars in each plot refer to conditioning the target modality using location only, while the two right bars indicate conditioning on all other modalities. There is a very clear difference between self-reported locations (WE) and GPS data: while knowing the GPS coordinates and traveled distance of the study participant reduces entropy in all cases, the reduction is far more modest than that obtained by conditioning on subjective location. The impact on predictability is analogous: GPS information provide a substantial boost to predictability (cf. Table 4.1), from 45% to 70% for WA and from 67% to 81% for WO. This is compatible with the results obtained above for inter-modal correlations. The improvement is however always inferior to the one induced by subjective location: for WA, predictability is 70% when supplying objective location but goes up to 92% when supplying subjective annotations. For WO, the difference is less pronounced:

81% (objective location) against 90% (subjective). This is, again, likely due to the strong connection between activity and location. The situation is roughly unchanged if we condition the target modality on the rest of the context, namely location (either subjective or objective), time, and the remaining modality. These results show that subjectivity, besides being necessary for framing behavior from the subject's perspective, has a substantial effect on predictability and regularity of behavior in practice.

4.7 Diversity of Personal Behavior

4.7.1 Motivation

In the last experiment we studied the diversity of personal behavior. The motivation underlying this experiment is to provide some evidence of the intrinsic diversity, both objective and subjective, of the personal context of an individual. It is a widespread intuition that most of the time people behave similarly to each other. Indeed, everybody sleeps, eats, works, and socializes, and these activities take up most of our time. So, at a high level, everybody behaves the same during these high-frequency (subjective) activities. Our intuition is that individual differences manifest themselves in infrequent behaviors – for instance, while most people only go to the cinema in the evening, a cinephile has no issue going to a matinée.

A prerequisite to this argument is that rare behaviors occur often enough to be statistically meaningful. To determine whether this is the case, we checked whether the empirical distribution of context annotations is heavy-tailed. This was achieved by fitting three candidate distributions, a power law distribution, a log-normal distribution, and an exponential distribution to the data.[5]. It is apparent from the plot shown in Figure 4.8 that the log-normal distribution (with $\mu = -8.2$, $\sigma = 1.6$) offers a much better fit of the behavior of individuals than the exponential model, which is not heavy-tailed. This supports the idea that individual behavior described using the four identified contextual modalities is heavy-tailed, as expected.

Inspired by some studies on the uniqueness of mobility [18, 81] and apps usage [84] behaviors, we investigate whether annotations in the tail of the context distribution are indicative of personal identity, that is, whether it is easier to identify individuals using

[5]Using the `powerlaw` package [3].

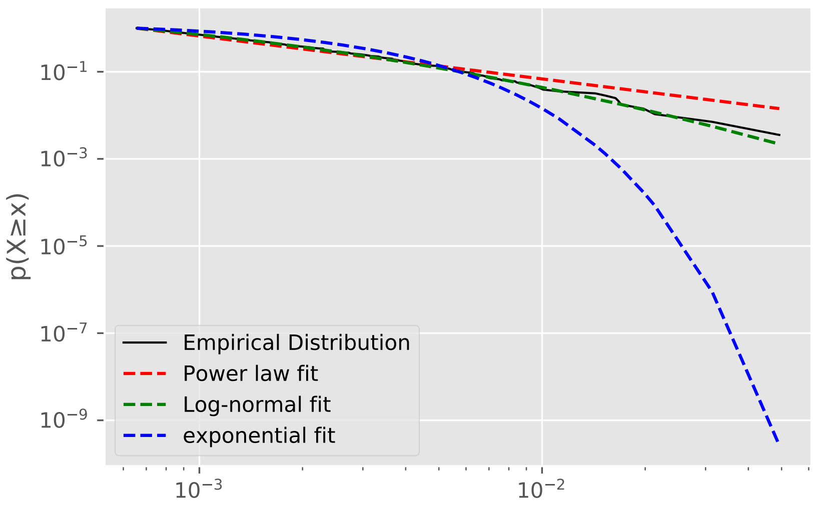


Figure 4.8: Comparison of power-law distribution and exponential distribution fit on the empirical distribution.

annotations from the tail or from the "head" of the distribution. For instance, in our university setting we expect common (head) annotations like ⟨morning, classroom, lesson, classmates⟩ to convey very little information about individual identity, as most university students attend lectures in the morning, and rarer (tail) annotations like ⟨morning, workplace, work, alone⟩ to be far more informative.

4.7.2 Experiment and Results

We designed a classification task in which the goal was to predict the identity of individuals based on context annotations only. All records in our data set were annotated with the ID of the subject they were generated by. The head and tail of the distribution were then defined using an arbitrary threshold $\tau \geq 0$: annotations that appear with frequency below τ were taken to fall in the tail and the others in the head. Next, we trained two Support Vector Machine (SVM) classifiers [13] separately on the tail data and on the head data, and compared their performance. Both models received annotations for all modalities as inputs. As above, performance was measured in terms of F_1 score (the higher the better) in a 10-fold cross validation setup. Notice that the number of personal IDs is 156, which is fairly large and renders the classification task highly non-trivial. For reference, the expected F_1 score of a random classifier is $1/156$ (indicated in cyan in the plots).

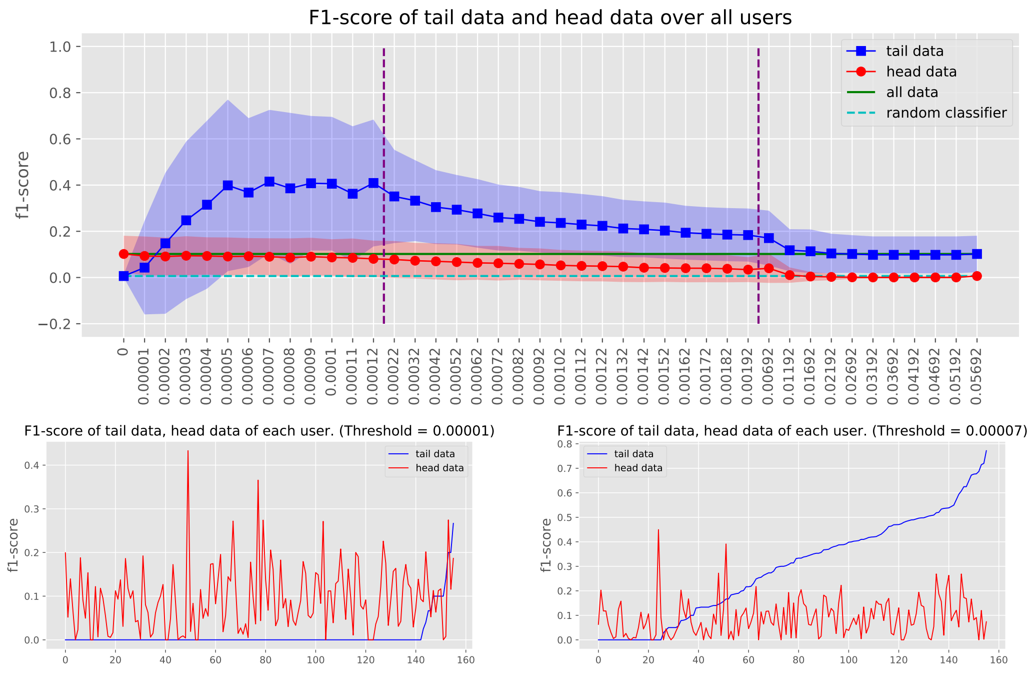


Figure 4.9: Top: average F_1 score over all study participants for classifiers trained on tail (blue) and head (red) data for increasing values of the threshold τ. The performance of a classifier trained on all data (green) and of a random baseline (cyan, dashed) are also reported, for comparison. Bottom: F_1 score of each individual for thresholds $\tau = 0.00001$ (left) and $\tau = 0.00007$ (right). The x-axis represents the 156 participants. Individuals are sorted by increasing F_1 score of the tail classifier.

The results can be viewed in Figure 4.9. The top plot shows the F_1 score of the two classifiers as the threshold τ is increased. Recall that a lower threshold entails that fewer annotations fall in the tail and more in the head. The threshold ranges from 0 (left of the plot), in which case no annotation falls in the tail, to the smallest value for which all data fall in the tail, which is ≈ 0.57 (right of the plot). Broadly speaking, the tail classifier always outperforms the head classifier by a large margin, while the head classifier never performs better than a classifier trained on both head and tail annotations (the green line in the figure). In order to better analyze the plot, we split it into three regions, highlighted by the purple lines (notice that the sticks on the x-axis are non-uniform.) In the leftmost region, the tail classifier does outperform the head classifier as soon as there are enough annotations in the tail, and it stabilizes at around 40% F_1 score for τ from about 0.00005 to 0.00012. Here the tail is maximally informative, presumably because it only contains rare and informative context annotations. As the threshold increases and less “rare” annotations fall in the tail (middle region), the tail classifier drops off in performance but it still outperforms the “all” and the head classifiers. The head classifier also performs worse and worse, as more annotations move from the head to the tail. In the rightmost region, the tail converges to the full data set and hence the tail classifier converges to the performance of the “all” classifier.

A break-down of performance for different study participants is reported in Figure 4.9 (bottom) for the two thresholds corresponding to the minimum ($\tau = 0.00001$) and maximum ($\tau = 0.00007$) of F_1 respectively. Individuals are sorted on the x-axis according to the F_1 score of the tail classifier, for readability. In the left figure, when $\tau = 0.00001$, the size of tail data is extremely small and only less than 20 users have annotations. This explains clearly why the performance of the tail classifier drops when the threshold is too small. On the other hand, for $\tau = 0.00007$ (right figure) the overwhelming majority of individuals is more likely to be identified correctly by looking at their infrequent behaviors – with less than 10 exceptions. This provides evidence in support of the fact that the tail of the distribution conveys much more information about personal identity than the head. The “exceptional” participants themselves can also be easily explained. These individuals are hard to classify because their behavior is slightly more regular than that of the other volunteers, meaning that their most of their annotations occur more frequently and therefore are more likely to fall in the head of the distribution. Indeed, we verified that this issue disappears once the threshold is increased slightly (data not shown). A proper solution for this issue would be to choose the threshold τ on a subject-by-subject basis. This is however orthogonal to our goals, and beyond the scope of this paper.

4.8 Summary

In this work, we have studied the predictability of human behavior through the notion of personal context. Our study captures a rich, multi-faceted picture of individual behavior by looking at four orthogonal but interrelated dimensions – namely time, location, activity, and social ties – viewed from the subject’s own perspective. An empirical analysis on a large data set of daily behaviors shows the benefit of this choice: the different contextual modalities and their subjective description are shown to provide important cues about the predictability of individual behavior. Motivated by this, we also applied our contextual modalities to study behavioral diversity. The obtained results highlight that individuals are more easily identified from rarer, rather than more frequent, subjective context annotations.

This work can be extended in several directions. First and foremost, while our results are promising, we plan to further validate them in more settings and in specific applications. To this end, we are currently working on collecting a much larger data set, with students from several universities in four different countries, which will serve as a basis for a thorough investigation of the results presented here.

This work also highlights an interesting conundrum. Our results suggest that subjective annotations are very useful for predicting certain contextual modalities. However, these subjective annotations, obtained by filling questionnaires, have some degree of error related to, for example, the list of alternatives that are allowed to the respondent, e.g. the list of activities, places, or people; the memory effect of the respondent when s/he does not respond immediately; the social desirability effect that may prevent the study participant from reporting certain (socially disapproved) activities; and unreported activities when the participant perceives this as an intrusion into her/his privacy. Moreover, in practical applications, collecting self-reported annotations is not always an option. This means that in some settings and scenarios one has to compute predictions from sensor measurements only, which is likely to incur a substantial performance penalty. Going forward, one option is to replace the ground-truth self-reported annotations with predictions. This makes especially sense in a multi-task prediction pipeline in which all contextual modalities are predicted *jointly* from sensor measurements. This way, the predictor can leverage inter-modal correlations, which are key for inferring some locations and activities and for avoiding inconsistencies. This prediction pipeline would be fully operationalizable even in the absence of subjective annotations, so long as an initial training set is available. The downside is that replacing annotations with predictions does introduce noise into the system. Finding a complete solution to this problem is an interesting avenue for future work.

Chapter 5

Personal Context Annotations Quality

5.1 Introduction

The work in this chapter is based on the work we published in [40]. Understanding human behaviour and human context in real life scenarios can not be done by sensor data alone without human knowledge. One solution is to involve human by asking users to give annotations to report their daily behavior. Users' annotations will then be used to train machines together with sensor data. This helps the machine to understand user's behavior or context automatically at appropriate moments before providing further services. This solution is widely used in some research areas such as participatory sensing [50] and mobile crowd sensing [43].

A relevant issue is the quality of the data provided by users [72, 52]. Majority of the existing studies focus more on the quality of sensor data, e.g., lack of sensor calibration, environmental noise, and data redundancy [63]. And there is still a lack of systematic approach concerning users' annotations [44], especially the annotations in the wild which is not a controlled environment. It is vital to have high-quality annotations for learning and understanding users' behavior and context without relying on time-consuming manual validation by experts, which is used in some controlled environment [46, 51]. In fact, even experts are unlikely to be able to mirror the experience of users, except in some simple tasks, e.g., traveling [10]

In this chapter, we propose a methodology to evaluate users' annotation by using the entropy from information theory, which was introduced in Section 4.3. We firstly generate annotation clusters for each user basing on the coordination of the locations.

The main idea is that, if an annotation cluster has higher number of unique annotations in it, it will be harder to infer which location this cluster is actually referring to. In other words, a cluster with higher entropy is more difficult to be predicted. And a user with more consistent clusters completes the annotation task more consistently.

The evaluation of the annotation process is performed on the data set from SmartUnitn One project which was introduced in Chapter 3. In this project, the annotations are built semantically depending on users' understanding of their context. While the project collected the users' annotations of different dimensions of context, we focus on the annotations that are provided to describe their locations and traveling habits. Furthermore, we propose to exploit the ontological information that is used in building the annotation labels to group semantically the labels to improve the results. The results show that our student users are consistent with respect to the random baseline. and the results can be improved by exploiting the semantics of annotations.

5.2 Annotation Consistency

It is not easy to evaluate the quality of user's annotation in the wild. More exist work focus on the evaluation of the sensor data. In fact, annotations can not be evaluated without the corresponding sensory data, which makes developing a general approach and a challenging task. To the best of our knowledge, there are still no systemic approached for assessing the quality of annotations in the wild in the literature.

We propose a methodology to assess the quality of user's annotation, and validate our solution on the data set generated from project SmartUnitn One, which was introduced in Chapter 3. To evaluate the annotations quality, we focus on two specific annotation types, i.e., locations and movement. Notice that, though movement is not one of the dimensions of the personal context that we proposed (Chapter 3), when a user answers the question "What are you doing?" with the label "En route", s/he will be asked with "How are you traveling?" instead of "Where are you?". Thus instead of generating a location annotation, s/he will generate a movement annotation to indicate his/her transportation means using one of the following labels: "by foot", "by bus", "by train", "by car", "by motorbike" and "by bike".

The main reason for choosing these two types of annotation is that both location and movement are relatively easy to recognize from the point of view of sensing strategies, and there is no need to rely on complex combinations of sensor data to identify them. Another reason for locations is that locations are endurants, which are not likely to change its function or position during the time of the experiment. This is important for this work since we do not have external ground truth to compare the annotation with, e.g., students' home addresses, which are not provided for privacy concerns.

5.2.1 Annotation Clustering

We firstly formalize a annotation A as a tuple $\langle L, LOC \rangle$, where the L is a label that the user provides to answer the questions "Where are you?" or "How are you traveling?", and the LOC is the physical location collected by the smartphone sensor at the time when the question is generated. The LOC is represented as a triple $\langle latitude, longitude, altitude \rangle$. It can be collected by the GPS or calculated by the network Wi-Fi connection.

Secondly, for each user, we cluster his annotations by using DBSCAN algorithm [28] with their physical location regardless of the collection time. DBSCAN is a density-based clustering algorithm that, given a set of points in some space, it groups the points that are closely packed together (points with many nearby neighbors), and marks the points as outliers which lie alone in low-density regions (whose nearest neighbors are too far away). The purpose of clustering all the annotations by their physical location coordinates is to generate a series of proxy locations or movements for every user. More formally, a cluster of annotations is defined as a set of labels $CL = [L_1, L_2, \ldots, L_N]$, where L_i is the i-th label, N is the total number of the labels in this cluster. Let M to be the number of total unique labels, then $M \leq N$. An alternative way to see a cluster is to group all the instances of the same M labels in the cluster with the number of their occurrence O, so that $CL = [L_1 : O_1, L_2 : O_2, \ldots, L_M : O_M]$. CL is a vector of pairs L_i, O_i, which means label L_i happens O_i times in this cluster. For instance, there are two clusters for the same user: $CL_1 = [Home : 132, Office : 1, Library : 1]$, and $CL_2 = [Home : 48, Office : 40, Library : 46]$. We can infer that, the user intends to annotate the first cluster CL_1 as his own home, while "Office" and "Library" are obviously outliers due to, e.g., uncertainly in the measurement accuracy or the wideness of the window for collecting sensor data. On the other hand, for the second cluster CL_2, it is not clear which location this cluster is referring to. Finally, annotations from a user U can be represented as a set of Q clusters: $U = [CL_1, CL_2, \ldots, CL_Q]$.

5.2.2 Definition of Consistency

As we can see in the examples in last subsection, cluster in which most space points are annotated with the same label, e.g, CL_1, is a consistent cluster, while a cluster with an even distribution of labels, e.g., CL_2, is an inconsistent cluster. Intuitively, a user with more consistent clusters is a relative consistent user. We base our intuition of consistency on the entropy $H(X)$ as defined in information theory [89]. We shortly introduced three types of entropy in Section 4.3. The entropy we use here in this section is the flat entropy, which takes the full value distribution into account but ignore the information of collecting time. The formal definition is:

$$H(X) = E[-lnP(X)] = -\sum_{i=1}^{n} P(x_i) \log_2 P(x_i)$$

where $H(X)$ is the entropy of a discrete random variable X with n possible values $x_1, x_2, \ldots, x_n$ and probability mass function $P(X)$.

The entropy is a measure of unpredictability of the information represented as a number between 0 to 1, where the lower the entropy is, the higher the predictability of the variable will be. If all the labels in a cluster are the same, which means the annotation task is consistent, then this cluster is highly predictable. On the other hand, if all the labels in this cluster is different, then this cluster is highly unpredictable. Since the consistency should intuitively be better when high, while entropy behaves in the opposite way, we decide to define the consistency of a cluster CL as:

$$C(CL) = 1 - H(CL)$$

where, recalling the definition, $CL = [L_1 : O_1, L_2 : O_2, \ldots, L_M : O_M]$. Then the consistency formula becomes:

$$C(CL) = 1 + \sum_{i=1}^{M} P(L_i) \log_2 P(x_i)$$

where

$$P(L_i) = \frac{O_i}{\sum_{j=1}^{M} O_j}$$

is the probability of the occurrence of the i-th label L_i in the cluster CL. After this consideration, a cluster can be represented as $CL = C(CL) : NA$, which is composed by the consistency value $C(CL)$ and the total number of annotations in this cluster $NA = \sum_{i=1}^{M} O_i$.

Now, in order to compute the consistency value of a user $\overline{C}_U$, recalling that $U = [CL_1, CL_2, \ldots, CL_Q]$, we compute the average of weighted consistency value of each cluster of this user.

$$\overline{C}_U = \frac{1}{\sum_{i=1}^{Q} NA_i} \sum_{i=1}^{Q} C(CL_i) NA_i \tag{5.1}$$

5.3 Assessing Location and Movement Consistency

In order to evaluate the consistency of the students' annotations generated in the SmartUnitn One project, we compute the consistency values $\overline{C}_U$ using Equation 5.1 for each of the 72 students in the dataset. To achieve better evaluation, we decided to use four different annotation sets for each user: i) Labels, ii) Semantic labels, iii) Random Baseline, and iv) Random Stratified Baseline. The result is shown in Figure 5.1

Labels are the annotations that the students provided during the project. The consistency of a student calculated with label annotation is represented as a round dot in the plot. The mean consistency value is 0.69 and the standard deviation is 0.12.

As for *Semantic labels*, they are semantically related labels that are grouped basing on the semantics of the ontology concepts as in Section 3.5.3 where we introduced the personal context annotation. This step followed two different levels of abstractions:

1. **Being in buildings vs. Travelling:** We firstly divide the annotations according to whether they refer to actual buildings or whether they refer to any traveling. Thus dividing them is to group all the following annotations as "traveling": "outdoors", "by foot", "by bus", "by train", "by car", "by motorbike" and "by bike", and leave the rest as they are.

2. **Home vs. University:** We further divide the annotations according to whether they refer to educational, residential or other buildings. We focus on the first two types of buildings since they are the two main contexts that students spend most of their daily lives on. For education buildings, we group "class", "study hall", "library", and "other university place". And for the residential buildings, we group "home" and "other private house".

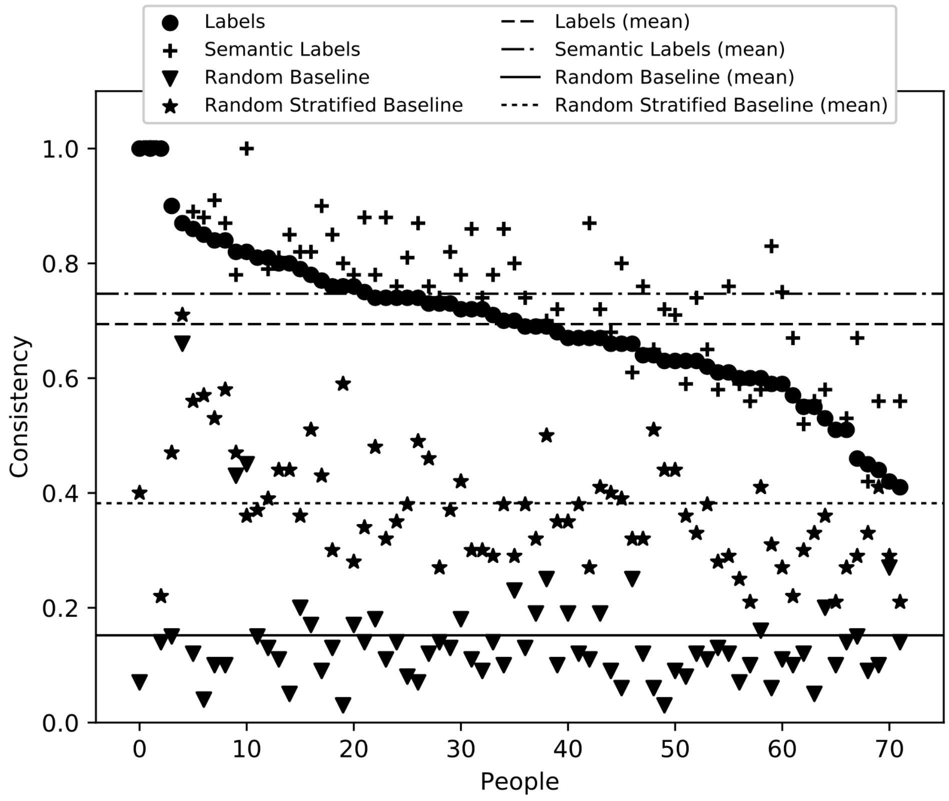


Figure 5.1: Consistency of four annotation sets. Consistency values on the Y axis and different student on the X axis.

These two categories allow us to smooth the distribution of closely relates spatial elements, e.g., study hall and libraries that belong to the same building and semantically refer to the educational context, and distinguishing when a user is moving. The final result obtained by adding this semantic step is that the mean consistency value is 0.74 and the standard deviation is 0.13, which further improves the initial result by students.

To properly assess the consistency value as defined in this work, it must be compared with a baseline value. For this reason, we present two different baseline, as shown in Figure 5.1. The first baseline is the *Random Baseline*, which simulates an entirely random behavior of each user. More specifically, we consider the same number of labels of each user annotation from the original dataset. This random baseline annotation set is generated by simply replacing each original annotation with a randomly chosen label from the 19 possible labels in the time diary. As expected, the consistency value is lower for all users, showing a final value of 0.15 (standard deviation is 0.13).

The second baseline, called *Random Stratified Baseline*, relied on stratified random sampling [87]. It randomly provides labels considering their original distribution for every user. The mean consistency value for this baseline is 0.38, and the standard deviation is 0.12.

The results of the evaluation show that the users annotated their own data almost five times more consistently that the entirely random baseline, and 50% more consistently than the stratified random baseline. It also shows that exploiting semantics in the whole process from the building of annotations to the analysis can improve the result by additional 7.2%. It is important to notice that, concerning standard approaches which rely on constrained environments and human experts intervention, the labels were collected at a significantly low rate (1 label every 30 minutes). Given that the labels were collected in a real-world scenario with such a low frequency, the results are even more significant.

5.4 Summary

In this chapter, we introduced a methodology that evaluates the users' annotation quality basing on the computation of entropy from information theory. It was then applied on the data set which is generated in the SmartUnitn One project. We mainly focused on the annotations of locations and movement among all modalities. The results show that the students were fairly consistent respect to the random baseline in giving annotations during the project experiment. And by considering the semantics of labels, the results can be improved.

Chapter 6

Skeptical Learning

6.1 Introduction

The work in this chapter is based on our work in [111] and [112].

Nowadays, our smartphones are no longer just miniature computers to send emails or access the Web, but they are becoming more and more like assistants for our everyday life. The critical factor for the change of roles that machines play in our lives is knowledge; in fact, machines need to have access to our knowledge and not just convey it. Knowledge is however diverse and heterogeneous, and what a person knows may differ radically from another, and this requires humans to help machines in understanding the world in the same way that they do. In recent approaches in computer science such as (supervised) machine learning [86] or mobile crowd sensing [43], human knowledge is usually provided as annotations or labeling of data collected with sensors or other means. Involving human knowledge as we mentioned in the chapter 5 will help the machine to better understand the user's current context. This type of human contribution is at the heart of the "human in the loop" paradigm which leverages both human and machine intelligence to create machine learning models by involving humans in training, tuning and testing data for a particular machine learning algorithm. The final goal is to use humans to improve the quality of the results of the machine.

The performance of supervised learning algorithms crucially depends on the quality of the labeling of the data they are trained on. A perfectly labeled training set however is a condition rarely met in real-world scenarios. Most modern supervised learning approaches can tolerate a small fraction of mislabelled training instances. The implicit assumption made in (even recent) mainstream machine learning and also in [31] is that annotators are *experts*. However, with the use of machine learning becoming viral, more and more applications are being developed where the tagging is provided by *non-expert users*.

Normal users (non-expert users) can make mistakes and more often than not unknowingly or unwittingly, due to their biases, which alter their perception and judgment of reality. Research in the Social Sciences provides evidence of the unreliability of people when required to compile self-reports such as time diaries [95] describing their behavior. For instance, respondents have difficulties recalling sequence and relevance of distant activities, i.e., memory bias and cognitive load [98], or may feel discouraged or non-cooperative because they do not understand the instructions or other reasons, i.e., unwillingness to report [14], or change their reported behavior to one that is considered socially desirable, i.e., conditioning [104].

We propose Skeptical Learning algorithm (SKEL) to deal with the untrustfulness and unreliability of human annotators. The main idea is that the machine exploits its available knowledge to assess the correctness of both its prediction and of the user. The machine can obtain a confidence measurement about not only itself but also the user by monitoring the sequence of wrong and correct answers. When a contradiction arises, namely the user's label and the prediction label generated by the machine are not compatible, these confidences label are then used by the machine to make a estimation whether it is the user gives an incorrect label or it is the machine not well-trained gives a wrong prediction. The machine will be further trained in the previous case and the user will be challenged to confirm his label in the later case.

In this chapter, we also introduce an evolved SKEL algorithm, in which the predictor is implemented as a hierarchy of classifiers matching prior semantic knowledge. The experiments and evaluation have been substantially extended by comparing our algorithm with three alternative strategies for dealing with conflicts.

We evaluate the SKEL algorithm on the data set generated in SmartUnitn One project. The results highlight the advantages of this interactive, skeptical approach to learning over state-of-the-art but non-skeptical machine learning alternatives. Results show that a SKEL architecture improves over an approach relying only on the noise-robustness of the underlying learning algorithm, as is customary in the machine learning community. Questioning user labeling and exploiting semantic knowledge for conflict resolution are crucial aspects underlying the observed increase in performance. The results provided also show the extreme variability of the students' behaviors and, therefore, the need for a person-centric customized solution to the problem of mislabeling.

6.2 Method

To solve the mislabeling issue of human annotation, we propose a method which is composed of a set of three basic ingredients, namely:

- A reference architecture, described in Section 6.2.1, which integrates the two main components used in supervised learning, namely the machine learning (ML) algorithms and the user providing feedback with two extra components, namely a knowledge component which stores the prior knowledge and the skeptical learning algorithm. The key idea is that the algorithm component, based on its internal strategy, involves the user and the ML algorithms in the most suitable way aimed at solving the mislabeling problem. This strategy may or may not involve using the prior knowledge or asking the user for different tasks, for instance, asking her for a second opinion on a previous labeling, with the goal of solving a contradiction;
- The main skeptical learning algorithm, described in Section 6.2.2, which exploits the critical idea that this algorithm can be in one of three states, namely: *TrainMode* where the goal is to accumulate enough knowledge about the user dynamics, *RefineMode* where, being confident of the previously acquired knowledge, SKEL starts checking the user provided label quality and, finally, *RegimeMode* where SKEL deploys an active learning strategy where the user is only occasionally asked for feedback as the way to check that SKEL and the user herself are aligned in their ability to interpret the world;
- The conflict management algorithm described in Section 6.2.3, which solves possible labeling conflict by optionally involving the prior knowledge and an *oracle*, for instance, the user herself on a second opinion, who is called upon making the final decision about what is the case.

We describe these three main components in the remainder of this section.

6.2.1 The SKEL Architecture

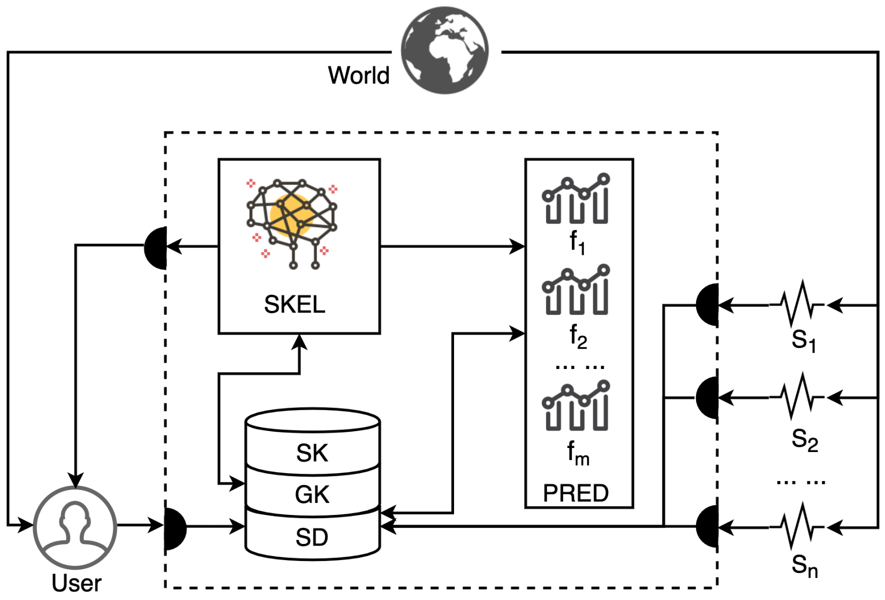


Figure 6.1: The SKEL architecture.

Figure 6.1 shows the multi-layer architecture of the Skeptical Learning. The fundamental intuition is that the human annotator(s) and the machine learning algorithm(s) are treated as *interpretation channels* which provide their *fallible* perspective on what is the case in the real world. In Figure 6.1, the *first* layer, inside the dashed box, is the HW/SW *Machine*, e.g., a smartphone with its software, implementing the overall SKEL functionality while the *third* and the last layer is the world, which is only indirectly accessible to the Machine. In the *second* layer, Machine and World are connected via, on one side, a set of n sensors $S_1, ..., S_n$ (as they exist, e.g., in a smartphone or a smartwatch) which generate a set of streams $\{x_t\}_{S_i}$, with x_t the value collected at time t by the sensor S_i and $\{x_t\}_{S_i}$ the stream of data generated by S_i and, on the other side, the Machine *reference user*, namely the person who is the direct beneficiary of its services.

The reference user, on request (outgoing arrow on the left of the dashed box) provides a label y_t which is interpreted as the value at time t of a specific property P_j, thus generating, by giving answers to different questions, a set of label streams $\{\{y_t\}_{P_j}\}_{\text{u}}$. For instance, the label "University" which answers the question "Where are you?" (ingoing arrow on the bottom left of the dashed box) is interpreted as $In_t(\text{User}, \text{University})$. It is important to underline how the answer by the user is provided based on her perception of the world and without any clue of the sensor values nor of how the Machine uses her input. The mapping from sensor values to user-provided labels is produced by a set of machine learning algorithms, whose existence may not be even known by the user, as described below. Note how this design decision allows for full freedom in the choice of the machine learning algorithms which can, therefore, be tuned to the specific learning problem.

As from Figure 6.1, the Machine consists of three components:

1. The SKEL main algorithm, hence SKEL implementing the skeptical learning strategy as described in the next sections;

2. A *Predictor* PRED consists of an *ensemble* (see, e.g., [74]) of m learning algorithms $f_1, .., f_m$, each taking in input an array of data $\mathbf{x}_t$ (the concatenation of all sensor readings at a certain time) and producing a score $f_k(\mathbf{x}_t, y)$ for all possible labels $y \in \mathcal{Y}_j$ of a given property P_j. These scores are then aggregated by PRED into a single label y_t which, similarly to the user, is the value that a certain property P_j takes at time t (from the point of view of PRED). Different properties can be managed using different ensembles (one per property) or having the ensemble score a combination of labels, one for each of the properties. PRED, similarly to the user, provides its internally produced labels on request by SKEL, where requests are represented in Figure 6.1 as right outbound arrows from SKEL. As implied in Figure 6.1, both the user and the predictor do not answer back to SKEL but rather store their answers in the knowledge component SD. This process allows for the possibly asynchronous interaction of SKEL with the user and PRED, which leaves room for independent loosely connected reasoning strategies;

3. A *Knowledge component* which stores whatever prior knowledge the machine has accumulated in time. It consists of three sub-components:

   (a) The Stream Data storage SD, with SD being

   $$\text{SD} = < \{x_t\}_{S_i}, \{\{y_t\}_{P_j}\}_{\text{p}}, \{\{y_t\}_{P_j}\}_{\text{u}} >$$

   where $\{x_t\}_{S_i}$ is the set of data streams coming from the sensors and $\{\{y_t\}_{P_j}\}_{\text{p}}$ and $\{\{y_t\}_{P_j}\}_{\text{u}}$ are the multiple streams, one per property P_j, provided by the predictor and the user, respectively, as answers to the SKEL queries;

(b) the component called GK, for *Ground Knowledge*, contains factual knowledge about the world (in the actual implementation it is stored as a knowledge graph), which can be generated in one of three possible ways: (i) it was provided to the machine as a priori knowledge, (ii) it was independently provided by third parties or (iii) it was previously generated by SKEL, as described below. GK is where the Machine cumulates the knowledge learned in time. One example of GK is the knowledge about specific locations, or events, or people, for instance, the fact that Fausto's office is part of the Department building, which in turn is a part of the University premises;

(c) the component called SK, for *Schematic Knowledge* which contains general knowledge, for instance in the form of a hierarchy of concepts stated in a certain language, see, e.g., [35]. One trivial example of SK content is an axiom stating that *resting* is a more general concept than *sleeping*. Another example is general statements about locations and sub-locations, for instance, the fact that Department buildings are always inside the Univesity premises. In this paper, we assume that SK is static and not growing and that it is known by the Machine because it is provided as a priori knowledge.

GK and SK are pivotal for the correct interpretation of the Machine and user's' answers. As described in detail in Section 6.2.3, they are quite useful in the conflict resolution phase as, whenever this is the case, they allow to infer that the user and the Machine meant the same thing even though they used different labels. Thus, for instance, the SK can be used to infer that the user is *resting* from her assertion that she is *sleeping* while the GK can be used to infer that she is *in the University* from her assertion that she is *in her office*, which is part of the University. The key observation is that the GK and the SK, which provide a model-driven view of the world, are unavoidable components whenever there is an interest in making the machine capable of fully understanding the user input, in its intended semantics and, vice-versa, in enabling the user in providing semantics to (i.e., in fully understanding) the output of the Machine internal data-driven machine learning algorithms.

6.2.2 The SKEl Main Algorithm

In this section, without loss of generality, we assume that there is a single property of interest P, e.g., the location of the user at a particular time. We represent by $\mathcal{Y}$ the set of possible values for this property.

Algorithm 1 Skeptical Supervised Learning (SKEL)

```
 1: procedure SKEL($\theta$)
 2:     init $\mathbf{c}^{\mathrm{u}} = 1, \mathbf{c}^{\mathrm{p}} = 0$
 3:     while TRAINMODE($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \theta$) do
 4:         $\mathbf{x}_t$ = SENSORREADING()
 5:         $y_t$ = ASKUSER()
 6:         $\hat{y}_t$ = PRED($\mathbf{x}_t$)
 7:         TRAIN($\mathbf{x}_t, y_t$)
 8:         UPDATE($\mathbf{c}^{\mathrm{p}}, \hat{y}_t, y_t$)
 9:     while REFINEMODE($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \theta$) do
10:         $\mathbf{x}_t$ = SENSORREADING()
11:         $y_t$ = ASKUSER()
12:         $\hat{y}_t$ = PRED($\mathbf{x}_t$)
13:         SOLVECONFLICT($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \mathbf{x}_t, \hat{y}_t, y_t$)
14:     while True do
15:         $\mathbf{x}_t$ = SENSORREADING()
16:         $\hat{y}_t$ = PRED($\mathbf{x}_t$)
17:         if CONF($\mathbf{x}_t, \hat{y}_t, c^{\mathrm{p}}_{\hat{y}_t}$) $\leq \theta$ then
18:             $y_t$ = ASKUSER($\hat{y}_t$)
19:             SOLVECONFLICT($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \mathbf{x}_t, \hat{y}_t, y_t$)
```

SKEL is an algorithm which takes in input a continuous stream of sensor data as they are stored in SD (see Figure 6.1). The pseudocode of SKEL is reported in Algorithm 1. The algorithm can be in one of three modalities which, for simplicity, we assume are activated sequentially, namely: *Train* mode as in typical supervised learning, *Refine* mode which checks the quality of the user answers and, under certain conditions, it challenges them, and the *Regime* mode where it starts being autonomous and only queries the user for particularly ambiguous instances.

The algorithm takes as input a confidence threshold θ. It starts by setting the user confidence to one and the predictor confidence to zero for all classes ($|\mathbf{c}^{\mathrm{u}}| = |\mathbf{c}^{\mathrm{p}}| = |\mathcal{Y}|$). Then the training phase begins. The algorithms collect sensor readings ($\mathbf{x}_t$) to be used as input for the predictor. In our current implementation, the prediction procedure PRED simply returns the highest scoring prediction:

$$\textsc{Pred}(\mathbf{x}) := \operatorname*{argmax}_{y \in \mathcal{Y}} \frac{1}{m} \sum_{j=1}^{m} f_j(\mathbf{x}, y) \tag{6.1}$$

where the score of a prediction y is the average score given to that label by the learners in the ensemble (in the first iteration, when no training has not been performed yet, $\textsc{Pred}(\mathbf{x})$ returns a randomly chosen label). The system then asks the user for a label (y_t) to be used as ground truth. The input-output pair is used to train the predictor using the TRAIN procedure. This procedure can either perform batch learning, in which the predictor is retrained from scratch using all input-label pairs stored in memory, or do an online learning step [88], where the novel input-output pair is used to refine the current predictor. The choice between these learning modalities depends on the specific implementation and constraints (e.g., storage capacity), see Section 6.3 for the details on the actual implementation. After training, the confidence of the predictor is updated using the UPDATE procedure, receiving as input the ground truth label and the predicted one *before* the training step. When the predictor is confident enough to start challenging the user on the correctness of a certain labeling, the training stage is stopped. The confidence in a prediction y for an input $\mathbf{x}$ is obtained from the product of the score of the prediction times the confidence c^{p}_y the predictor has in predicting that label:

$$\textsc{Conf}(\mathbf{x}, y, c^{\text{p}}_y) := c^{\text{p}}_y \cdot \frac{1}{m} \sum_{j=1}^{m} f_j(\mathbf{x}, y) \tag{6.2}$$

The system remains in training mode as long as the expected probability of contradicting the user does not exceed the threshold:

$$\begin{aligned} &\textsc{trainMode}(\mathbf{c}^{\text{p}}, \mathbf{c}^{\text{u}}, \theta) := \\ &E[\mathbb{1}(\textsc{Conf}(\mathbf{x}, \hat{y}, c^{\text{p}}_{\hat{y}}) > c^{\text{u}}_y \cdot \theta)] \leq \theta/2 \end{aligned} \tag{6.3}$$

where $\hat{y} = \textsc{Pred}(\mathbf{x})$ is the predicted label for input $\mathbf{x}$, y is the label provided by the user for that input, $\mathbb{1}(\varphi)$ evaluates to one if φ is true and zero otherwise, and the expectation is taken over all inputs seen so far. The user is contradicted when the confidence in the predicted label exceeds a factor θ of the confidence of the user in her own label.

Once the system enters the refinement mode, it keeps asking the user for labels, but it starts to compare them with its predictions. The SOLVECONFLICT procedure deals with this comparison, and will be described in detail later on in this section. The refinement stage is stopped when the predictor is confident enough to stop asking for feedback to the user on every input, but selectively query the user on “difficult” cases. In general, it should be the user who decides when to switch mode, trading off system maturity and cognitive load. A simple fully automated option, similar to the one used for the train mode consists of staying in refine mode as long as the expected probability of querying the user exceeding the threshold:

$$\begin{aligned}&\textsc{refineMode}(\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \theta) := \\ &E[\mathbb{1}(\textsc{Conf}(\mathbf{x}, \hat{y}, c^{\mathrm{p}}_{\hat{y}}) \leq \theta)] \geq \theta/2\end{aligned} \tag{6.4}$$

again with expectation taken over all inputs are seen so far.

When leaving the refine mode, the system enters the regime one, where it remains indefinitely. In this mode, the system stops asking feedback for all inputs, and an active learning strategy [85] begins. The system queries the user only if the confidence in a certain prediction is below the “safety” threshold θ. If the system decides to query the user, it includes the tentative label in the query, and then behaves as in refinement mode, calling SOLVECONFLICT to deal with the comparison between the predicted and the user labels.

6.2.3 The Conflict Management Algorithm

Algorithm 2 shows the SOLVECONFLICT procedure. The procedure takes as input the predictor and user confidence vectors $\mathbf{c}^{\mathrm{p}}$ and $\mathbf{c}^{\mathrm{u}}$, an input $\mathbf{x}$ with its predicted label ($\hat{y}$) and the label given by the user (y). It firstly compares the two labels according to the ISCOMPATIBLE procedure. In the simplest case, this outputs true if the two labels are identical, and false otherwise. In more complex scenarios, this procedure can use existing knowledge, as stored in the SK or in the GK, to decide whether two distinct labels are compatible, e.g., if the concept denoted by one is a generalization of the concept denoted by the other. In case the labels are compatible, a consensus label is taken as the ground truth, and the predictor and user confidences are updated accordingly. A natural choice for the consensus (and the one we use in our experiments) is being conservative and choosing the least general generalization of the two concepts.

A labeling conflict arises in the case of the two labels are not compatible. In case

Algorithm 2 Procedure for solving labeling conflicts.

```
1:  procedure SOLVECONFLICT($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \mathbf{x}, \hat{y}, y$)
2:      if ISCOMPATIBLE($\hat{y}, y$) then
3:          $y^* = $ CONSENSUS($\hat{y}, y$)
4:          UPDATE($\mathbf{c}^{\mathrm{p}}, \hat{y}, y^*$)
5:          UPDATE($\mathbf{c}^{\mathrm{u}}, y, y^*$)
6:      else if CONF($\mathbf{x}, \hat{y}, c^{\mathrm{p}}_{\hat{y}}$) $\leq c^{\mathrm{u}}_{y} \cdot \theta$ then
7:          TRAIN($f, \mathbf{x}, y$)
8:          UPDATE($\mathbf{c}^{\mathrm{p}}, \hat{y}, y$)
9:      else
10:         $y^* = $ ASKUSER($\hat{y}, y$)
11:         if $not$ ISCOMPATIBLE($\hat{y}, y^*$) then
12:             TRAIN($\mathbf{x}_t, y^*$)
13:         UPDATE($\mathbf{c}^{\mathrm{p}}, \hat{y}, y^*$)
14:         UPDATE($\mathbf{c}^{\mathrm{u}}, y, y^*$)
```

the confidence of the prediction is not large enough to contradict the user, the user label is taken as ground truth, the predictor is retrained with this additional feedback, and its confidence is updated accordingly. Otherwise, the system queries the user providing the two conflicting labels as input, asking her to solve the conflict. The user is free to stick to her own label, change her mind and opt for the label suggested by the predictor, or provide a third label as a compromise. As we are assuming a non-adversarial the setting, the system eventually trusts the newly provided label (even if unchanged) which becomes the ground truth. At this point, a compatibility check is made in order to verify whether a retrain step is needed, and the predictor and user confidences are updated. Notice how this is only an initial choice which can be extended in many different ways. What is needed at this point in the algorithm is a trusted *oracle* which will give its final word on the dispute. This oracle can be the user herself asked for a second opinion, as discussed above, or some trusted learning algorithm, as in the experiments below, or some kind of social opinion by, e.g., some friends or people nearby, or any combination of these possibilities.

The UPDATE procedure takes as input a confidence vector, a tentative label ($\hat{y}$) and a ground truth label (y^*), and updates the confidence vector according to the relationship between the two labels. The new confidence vector is as a label-wise running average accuracy over the current and past predictions, for a certain window size d.

6.3 Experiments and Results

6.3.1 Data Processing and Setup

We validate SKEL algorithm on the data collected in the SmartUnitn Project. The evaluation on this dataset required a pre-processing stage that generated a set of 122 feature vectors for each user, using all the available sensors inputs (the number depends on the smartphone, and consequently on the user). The features can be described based on the type of the data they refer to:

- *Periodic data* that include hardware sensors recording data every 1/20 seconds, mostly three-dimensional coordinates. To aggregate this type of data the *norm* of the axes for each row falling in the time slot is computed. This avoids discrepancies deriving from different smartphone positions/orientations at the time of the reading. Furthermore, unreasonable spikes in the reading are discarded. Finally, *mean* and *variance* are calculated from the resulting vector of norms;

- *On change data*, that represent discrete events generated by the device software. Features computed for this category include the *time* in milliseconds a particular state persists within the time slot, e.g., for how long a screen is on and off respectively. Additionally, a value denoting the *presence* of events and one keeping track of the *number of events* seen are recorded;

- Other types of data, including any sensor generating data that does not belong to the above categories. The most important features are the ones about the user location. These consist of a set of features indicating: i) proximity to one of the ten most common locations visited by the students during the experiment (one flag per location). Most common locations were computed using the DBSCAN clustering algorithm [28] using a maximum distance of $12.5m$ for considering two points part of the same neighborhood. To avoid single users with a high number of diverse locations to skew the results in their favor, the clustering was run an additional time on the clustering resulting from the first run, this time with a distance of $10m$. Among the resulting super-clusters, the 10 most common were eventually selected; ii) proximity to the home of the user. The home of the user was identified by running the DBSCAN algorithm with a distance of $108.27m$

(the average accuracy of the considered points) on those location points that were collected by the application in two situations: between 5:30AM and 7:30AM (the best moment according to [57]) and right before the GPS stops collecting data, past midnight, when it is reasonable to assume that the user just reached home to go to sleep. Among the resulting clusters, we chose the largest cluster's centroid to be the home location. From this point, the feature we considered is the distance calculated in meters from the current position; iii) none of the above. Additionally, the distance between the first and the last detected points and the total distance travelled within the time interval are recorded. For the data relative to proximity networks (WiFi and Bluetooth) information such as *number of unique devices* in range (if at all) is recorded. Finally, for the running applications, for each window the features represent *which categories of applications run for how much time in milliseconds*, (eg. Social : 10000, Tools : 25000) and *the number of events* recorded.

The features were calculated using a window size of 30 minutes, which is the time between two consecutive annotations. The analysis focuses on the locations the participants visited during the two weeks of the experiment. We decided to focus on locations because it is easier to verify the correctness of such data with respect to, for example, activities. To this aim, we created an additional element, *the oracle*, which provides ground truth labels independently of both the predictor and the user annotations. The oracle relies on information regarding the location of the University buildings, and identifies the home of a user by clustering the locations she labels as home via DBSCAN [28] and choosing the cluster where she spends most of the time during the night. Note that SKEL has no access to this information. The oracle is used for the evaluation of the performance of the system in predicting actual labels, and to simulate a non-adversarial, collaborative user as detailed in the next section. By comparing the labels provided by the users with the ones generated by the oracle, we discovered that 65 out of 72 participants provided more than 10% wrong labels, while 35 of them provided 30% wrong ones. In order to highlight the potential advantage of the SKEL framework in highly noisy scenarios, we focused on the subset of users with more than 20% labeling errors (42 users). Note that this proportion of errors is substantially higher than the one usually expected in machine learning applications. We implemented the predictor as a random forest classifier [7] (with batch training), which is known to be robust to labeling noise [33], in order to evaluate the ability of SKEL to improve over an already noise-robust baseline. For simplicity, we used an infinite window ($d = \infty$) for the confidence update, also given the relatively short duration of the experiment. The confidence parameter θ was set to 0.2 in order to achieve a reasonable trade-off between accurate training and reasonable cognitive effort for the user, considering the complexity of the learning task.

6.3.2 SKEL with No Prior Knowledge

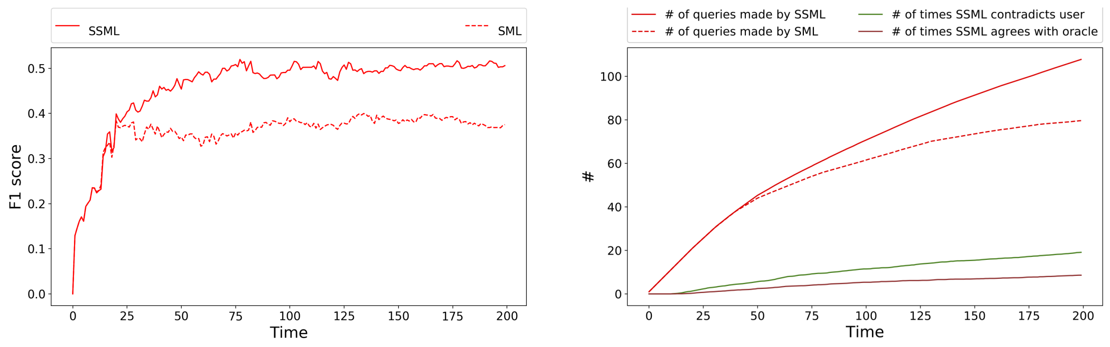


Figure 6.2: Results of SKEL algorithm and non-skeptical algorithm with no prior knowledge

The first experiment we performed is aimed at evaluating the robustness of SKEL to labeling noise, without using semantic information on user labels, as obtained from the prior knowledge stored within the system. We collapsed the original labels into three classes, *Home*, *University* and *Others*. In this setting, the oracle can provide accurate ground truth values for all labels, as Others is recognized as not being Home and University. The procedure ISCOMPATIBLE inside Algorithm 2 in this case simply returns true if the two labels are identical and false otherwise. Given that we are working on previously collected data, we cannot query users in real-time in case a conflict arises. To simulate a collaborative, non-adversarial user, we replace the ASKUSER procedure in Algorithm 2 with a call to the oracle, i.e., the user returns the ground truth label when her initial label is contested.

Figure 6.2 reports the results of SKEL as compared to a solution never contradicting the user (obtained by replacing SOLVECONFLICT with a train and update step, as happens in the training phase). Here we refer to the SKEL as SSML, standing for SKEL Supervised

Mahcine learning, and we refer to this baseline simply as SML, standing for (non-skeptical) Supervised Machine Learning. Results were averaged over all users with a number of training samples greater than 200. On the left of Figure 6.2, it reports the f_1 score of the SKEL (solid red) and SML (dashed red) predictors for increasing number of iterations. The score was computed on a fixed test set, namely the latest 15% of the all data available for each user, which was not used for training. This provides an estimate of the performance of the algorithms when doing predictions on future data. The results indicate that with a slight increase of queries to the user, the system achieves a 34.2% relative improvement in performance (from $f_1 = 0.38$ to $f_1 = 0.51$). On the right of Figure 6.2, it reports the number of queries made to the user by SKEL and SML respectively (red solid and red dashed). It also reports, for the SKEL case, the number of times the user was contradicted (green) and in how many of these cases the user (as simulated by the oracle) ended up agreeing with the predictor (brown). It is interesting to notice that most of the times in which SKEL contradicts the user, the oracle agrees with it.

When looking at the performance of individual users, we observe very different behaviors. We range from a very accurately modeled user ($f_1 = 0.96$ at the last iteration) to one where the predictor ends up always predicting the same label ($f_1 = 0.12$). Note that in the former case the conflict management strategy is responsible for the performance, as the non-skeptical alternative learns a degenerate model predicting the same label most of the times. In some cases, SKEL also learns a degenerate model. This happens because a long stream of annotations with the same label forces the model to focus on it with very high confidence, move to the regime mode and stop interacting with the user. Smarter initialization strategies could help to avoid this behavior. Our aim here is not to fine-tune the system for this specific problem, but rather highlight the potential of challenging user feedback in general interactive learning scenarios. A more detailed analysis of the behavior of the system with various ”prototypical” users is provided in Section 6.3.5.

6.3.3 SKEL with Prior Knowledge

The second experiment is aimed at evaluating the impact of semantics on the overall performance of the SKEL, in particular at the Schematic Knowledge level (see Figure 6.1). To do so, we created a set of labels including all the thirteen classes in the original dataset, plus three novel labels representing superclasses, as shown in Table 6.1. The latter were

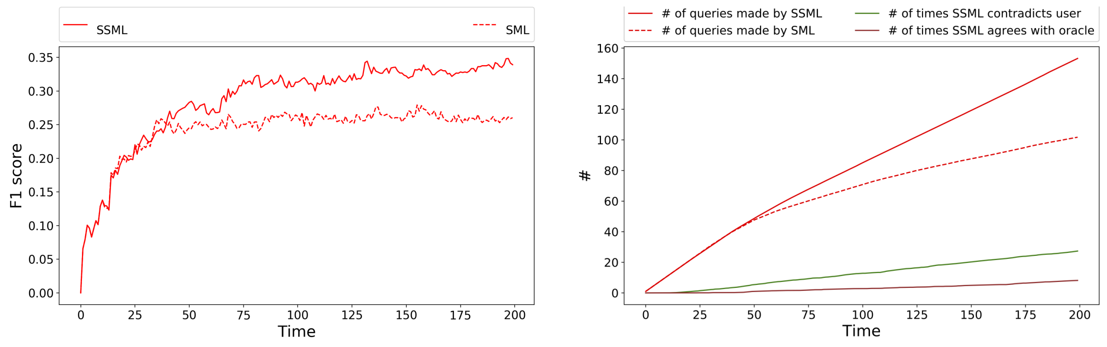


Figure 6.3: Results of SKEL and non-skeptical algorithm with prior knowledge.

Table 6.1: Table showing the location labels that the users in the experiment could select and the mapping with the three superclasses we defined.

Bar	Gym	Shop	Outdoors	Workplace	Other Home	Home	Class	Canteen	Study hall	Library
Others						Home	University			

the same as the ones used in the first experiment, namely *Home*, *University* and *Others*. Semantics was introduced in terms of part-of relationships, such as *Class* and *Laboratory* being part-of *University*. In this setting, ISCOMPATIBLE returns true if the two labels belong to the same super-class (or if they are the same), false otherwise. The CONSENSUS procedure is implemented as a conservative choice that always returns the super-class, e.g, *University*, *Home* or *Others*. Concerning the oracle (and thus, the replacement of the ASKUSER procedure within Algorithm 2 as well as the one used to generate test labels), the ground truth label is the user provided one if compatible with the one detected by the oracle, otherwise it is the label of the oracle.

Figures 6.3 report the results of this experiment, for f_1 score and number of queries respectively. The overall performance in this setting is lower than in the non-semantic case, despite a higher number of queries to the user, because the larger number of classes substantially increases the complexity of the labeling task. Nonetheless, the trend is very similar, with a relative performance improvement of 30.7% (from $f_1 = 0.26$ to $f_1 = 0.34$).

6.3.4 Objective vs. Subjective Labelling

Our algorithm aims to learn to predict the ground truth labels, despite being fed with user-provided labels in the first place. We can think of the former, the ones provided by the oracle, as *objective labels* and of the latter, the ones provided by the user when not acting as an oracle, as *subjective labels*. In this section, we investigate the performance of the SKEL algorithm with respect to these two types of labels.

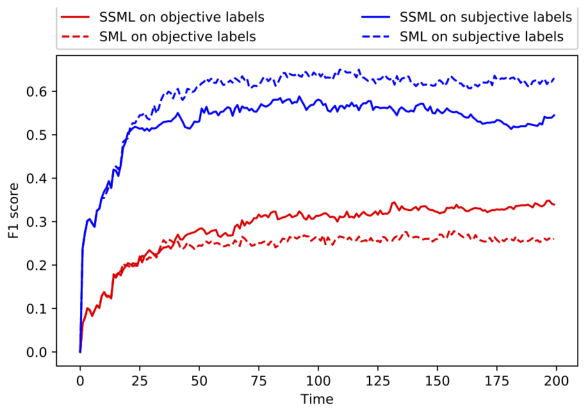
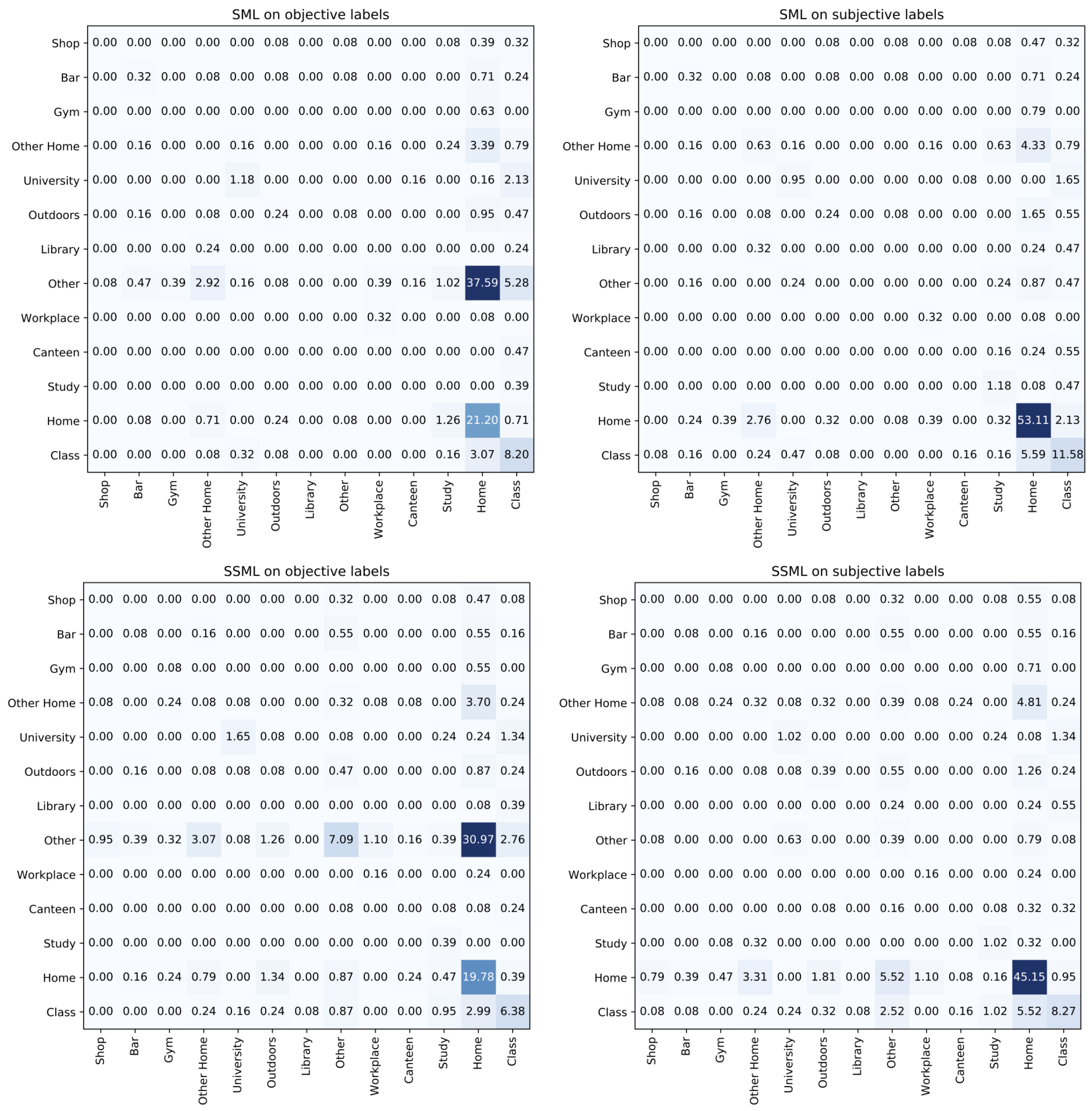


Figure 6.4: Average f_1 score of the SKEL and SML algorithms on all the users in the experiment.

Figure 6.4 shows the average f_1 scores in four different settings, in the scenario including semantic information (the one described in Section 6.3.3). The solid red curve and the dashed red curve show the same results as in Figure 6.3. They represent f_1 scores of the SKEL and SML algorithms, computed on the test set using objective labels as the ground truth. The solid blue curve and the dashed blue curve represent again f_1 scores of the SKEL and SML algorithm respectively but computed on the test set using subjective rather than objective labels as the ground truth.

These results provide some interesting insights into the performance of the algorithm. Overall, the performance evaluated on subjective labels (blue curves) is higher than the one evaluated on objective labels (red curves). This means that learning subjective labels is easier than learning objective labels; this is is an expected result, given that the algorithm often receives subjective labels as feedback. When comparing the performance of the SKEL algorithm (solid curves) with respect to the SML one (dashed curves), we observe opposite behaviors depending on whether we evaluate them in terms of objective or subjective labels. The SKEL algorithm is effective in improving prediction quality on objective labels (solid red curve vs. red dashed curve), as discussed in Sections 6.3.2, 6.3.3. Conversely, the conflict management phase of the SKEL algorithm is harmful when the evaluation is made in terms of subjective labels (solid blue curve vs. blue dashed curve). This is also an expected result and indicates that challenging the user is pointless if the final goal is adapting to her subjective viewpoint rather than trying to make it match any type of objective knowledge. The decision of which choice to make must be made in a more global setting *i)* as a function of the specific type of knowledge, *ii)* the final goal to be achieved, and *iii)* what the user wants to do with the machine learning results. Thus, for instance, a certain place could objectively be closer to the user than another one, but subjectively farther, given the user's specific knowledge of the paths to both places (including taking some shortcuts not known to the general public).

Figure 6.5 reports the confusion matrices for all settings. The first row refers to SML and the second to SKEL, while the first and second columns refer respectively to the objective and subjective labels. Results are normalized over the sum of all the elements in the matrix. The fact that all matrices are quite sparse explains the low f_1 score of the overall results presented in Figure 6.4. Overall, the *Home* and *Other* labels are the most common. Comparing the matrices on objective and subjective labels, we can see that true positives on *Home* on subjective labels are significantly higher than on objective ones. In the latter case, there are many cases in which the algorithm predicts *Home*, but the actual label was *Other*. This is a clear example of misleading feedback from the user, who always answers *Home* when being in some *Other* location. By looking at the first column, we see that our SKEL algorithm is capable of partially fixing this problem, bringing true positives for the *Other* label from 0.0% to 7.09%. This can explain the gap between SKEL and SML on objective labels (red curves) in Figure 6.4.



SML on objective labels

	Shop	Bar	Gym	Other Home	University	Outdoors	Library	Other	Workplace	Canteen	Study	Home	Class
Shop	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.08	0.00	0.00	0.08	0.39	0.32
Bar	0.00	0.32	0.00	0.08	0.00	0.08	0.00	0.08	0.00	0.00	0.00	0.71	0.24
Gym	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.63	0.00
Other Home	0.00	0.16	0.00	0.00	0.16	0.00	0.00	0.00	0.16	0.00	0.24	3.39	0.79
University	0.00	0.00	0.00	0.00	1.18	0.00	0.00	0.00	0.00	0.16	0.00	0.16	2.13
Outdoors	0.00	0.16	0.00	0.08	0.00	0.24	0.00	0.08	0.00	0.00	0.00	0.95	0.47
Library	0.00	0.00	0.00	0.24	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.24
Other	0.08	0.47	0.39	2.92	0.16	0.08	0.00	0.00	0.39	0.16	1.02	37.59	5.28
Workplace	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.32	0.00	0.00	0.08	0.00
Canteen	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.47
Study	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.39
Home	0.00	0.08	0.00	0.71	0.00	0.24	0.00	0.08	0.00	0.00	1.26	21.20	0.71
Class	0.00	0.00	0.00	0.08	0.32	0.08	0.00	0.00	0.00	0.00	0.16	3.07	8.20

SML on subjective labels

	Shop	Bar	Gym	Other Home	University	Outdoors	Library	Other	Workplace	Canteen	Study	Home	Class
Shop	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.08	0.00	0.08	0.08	0.47	0.32
Bar	0.00	0.32	0.00	0.08	0.00	0.08	0.00	0.08	0.00	0.00	0.00	0.71	0.24
Gym	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.79	0.00
Other Home	0.00	0.16	0.00	0.63	0.16	0.00	0.00	0.00	0.16	0.00	0.63	4.33	0.79
University	0.00	0.00	0.00	0.00	0.95	0.00	0.00	0.00	0.00	0.08	0.00	0.00	1.65
Outdoors	0.00	0.16	0.00	0.08	0.00	0.24	0.00	0.08	0.00	0.00	0.00	1.65	0.55
Library	0.00	0.00	0.00	0.32	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.24	0.47
Other	0.00	0.16	0.00	0.00	0.24	0.00	0.00	0.00	0.00	0.00	0.24	0.87	0.47
Workplace	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.32	0.00	0.00	0.08	0.00
Canteen	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.16	0.24	0.55
Study	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.18	0.08	0.47
Home	0.00	0.24	0.39	2.76	0.00	0.32	0.00	0.08	0.39	0.00	0.32	53.11	2.13
Class	0.08	0.16	0.00	0.24	0.47	0.08	0.00	0.00	0.00	0.16	0.16	5.59	11.58

SSML on objective labels

	Shop	Bar	Gym	Other Home	University	Outdoors	Library	Other	Workplace	Canteen	Study	Home	Class
Shop	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.32	0.00	0.00	0.08	0.47	0.08
Bar	0.00	0.08	0.00	0.16	0.00	0.00	0.00	0.55	0.00	0.00	0.00	0.55	0.16
Gym	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.55	0.00
Other Home	0.08	0.00	0.24	0.08	0.08	0.00	0.00	0.32	0.08	0.08	0.00	3.70	0.24
University	0.00	0.00	0.00	0.00	1.65	0.08	0.00	0.08	0.00	0.00	0.24	0.24	1.34
Outdoors	0.00	0.16	0.00	0.08	0.08	0.08	0.00	0.47	0.00	0.00	0.00	0.87	0.24
Library	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.39
Other	0.95	0.39	0.32	3.07	0.08	1.26	0.00	7.09	1.10	0.16	0.39	30.97	2.76
Workplace	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.16	0.00	0.00	0.24	0.00
Canteen	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.00	0.08	0.08	0.24
Study	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.39	0.00	0.00
Home	0.00	0.16	0.24	0.79	0.00	1.34	0.00	0.87	0.00	0.24	0.47	19.78	0.39
Class	0.00	0.00	0.00	0.24	0.16	0.24	0.08	0.87	0.00	0.00	0.95	2.99	6.38

SSML on subjective labels

	Shop	Bar	Gym	Other Home	University	Outdoors	Library	Other	Workplace	Canteen	Study	Home	Class
Shop	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.32	0.00	0.00	0.08	0.55	0.08
Bar	0.00	0.08	0.00	0.16	0.00	0.00	0.00	0.55	0.00	0.00	0.00	0.55	0.16
Gym	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.71	0.00
Other Home	0.08	0.08	0.24	0.32	0.08	0.32	0.00	0.39	0.08	0.24	0.00	4.81	0.24
University	0.00	0.00	0.00	0.00	1.02	0.00	0.00	0.00	0.00	0.00	0.24	0.08	1.34
Outdoors	0.00	0.16	0.00	0.08	0.08	0.39	0.00	0.55	0.00	0.00	0.00	1.26	0.24
Library	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.24	0.00	0.00	0.00	0.24	0.55
Other	0.08	0.00	0.00	0.00	0.63	0.00	0.00	0.39	0.00	0.00	0.00	0.79	0.08
Workplace	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.16	0.00	0.00	0.24	0.00
Canteen	0.00	0.00	0.00	0.00	0.00	0.08	0.00	0.16	0.00	0.00	0.08	0.32	0.32
Study	0.00	0.00	0.08	0.32	0.00	0.00	0.00	0.00	0.00	0.00	1.02	0.32	0.00
Home	0.79	0.39	0.47	3.31	0.00	1.81	0.00	5.52	1.10	0.08	0.16	45.15	0.95
Class	0.08	0.08	0.00	0.24	0.24	0.32	0.08	2.52	0.00	0.16	1.02	5.52	8.27

Figure 6.5: Confusions matrices of the SKEL and SML algorithms computed at the last iteration reported in Figure 6.4.

6.3.5 Variability of Users

In this section, we present a breakdown of the performance of the SKEL algorithm with respect to a recognizable pattern of user behaviour. We analysed the performance graphs of the different users and identified four patterns as highly common. These performance patterns can be related to some distinct behavioural patterns, which we discuss in the following. Figure 6.6 reports the results for these prototypical users, where each row refers to a specific user. Left figures report f_1 scores in different settings, which are the very same as in Figure 6.4 (i.e., objective vs subjective labels, SKEL vs SML). Right figures report information on the number of queries and agreement with the user and with the oracle, same as in Figures 6.2(b) and 6.3(b).

Inattentive user The results of the first row in Figure 6.6 show that the highest score is achieved by the SKEL algorithm evaluated on objective labels. All other settings achieve substantially lower performance. This behavior can be explained in terms of an *inattentive* user, who often provides subjective labels which are different from the objective ones (difference between red and blue curves), and is largely inconsistent in the initial feedback, which makes predicting subjective labels harder than objective ones (solid red vs. dashed blue), even if most of the feedback is subjective anyhow. The inconsistency of the user is also reflected in the right graph of the first row of Figure 6.6, showing a rather large fraction of times in which the user is contradicted (because there is no agreement) and the predictor agrees with the oracle. This is the type of user for which the SKEL algorithm is most beneficial. Note that the fact that SKEL manages to correct user inconsistencies indicates that the system reaches a confidence which is sufficient to start challenging the user, i.e., the user is a "detectably" inconsistent one.

Predictable user The second case is a particularly interesting one. For the first 20 iterations, both SKEL and SML are completely incapable of predicting the user labels. After this initial step, the algorithm learns to predict subjective labels with a high accuracy (solid red against solid blue). This happens because the user is consistent in providing feedback, but her subjective labels are largely different from the objective ones. At a certain point, the system starts challenging the user and soon afterward (around iteration 190), the system learns to predict objective labels with an higher accuracy with respect to subjective ones. We refer to this user

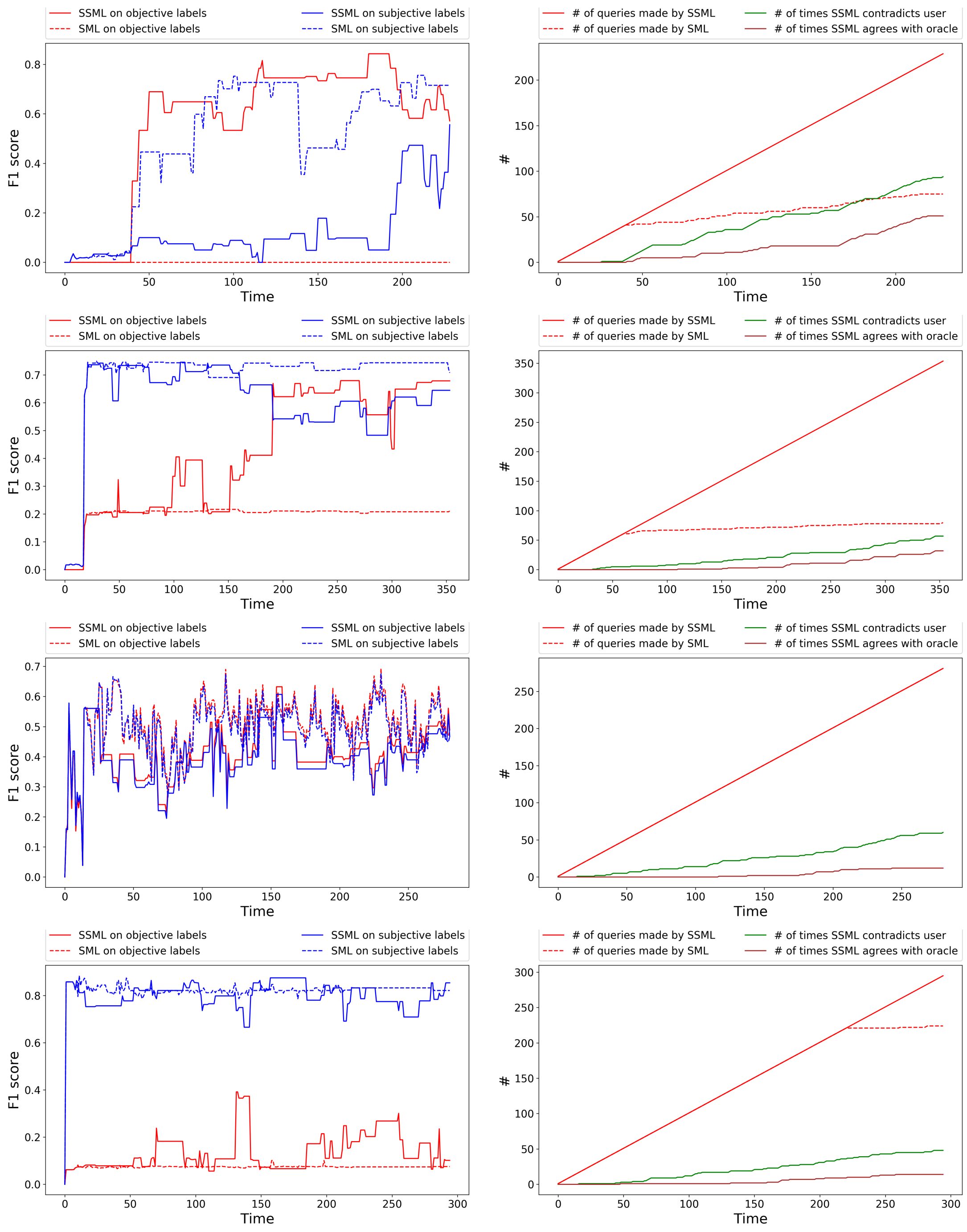


Figure 6.6: Results for four different prototypical users: the first row shows an *inattentive user*; the second one a *predictable user*; the third one a *reliable user* and the last one a *tricksy user*. The images on the left report the f_1 scores in different settings while the ones on the right report information about the number of queries and agreement with the user and the oracle. The Time axes represent the number of iterations the algorithm is going through.

as “predictable”. Albeit subjective labels are mostly different from objective ones for this user, both of them can be predicted with high accuracy, once the system receives the appropriate feedback. This is confirmed by the high number of times in which predictor and oracle agree, as shown in the right figure. A predictable user is thus another case in which the benefits of SKEL are substantial, even if it takes some time for the system to figure out the discrepancy between subjective and objective labels.

Reliable user The third row of Figure 6.6 shows the results of a user for which the performance of the SKEL and SML algorithms are basically the same. This is because the user is already reliable in providing initial feedback, as can be seen by the substantial overlap between the red and blue curves. Indeed, the user is contradicted only occasionally (green curve in right figure), and even rarer are the cases in which the oracle agrees with the predictor against the subjective label of the user (brown curve). This is a user for whom SKEL is not helpful, but also not harmful, as it ends up asking about the same number of queries as SML (solid and dashed red curves in the right figure). Both systems never exit the refinement phase and keep asking for feedback, because the location of this specific user keeps changing and, most importantly, almost all the labels in the label space are selected.

Tricksy user The last one is a case in which the SKEL algorithm completely fails to predict user behavior. By looking at the difference between red and blue curves, it is apparent that the problem is in the difference between subjective and objective labeling. The algorithm keeps learning the former, even when given the chance to question user labeling. The right figure shows that this chance is rarely taken by the algorithm, and almost never leads to discovering the cases in which subjective and objective labels disagree. The user here succeeds in fooling the system by convincing it of the correctness of her own feedback, namely the fact of being at home way more often than what was actually the case. Note that additional prior knowledge would substantially help the system in figuring out that something strange is going on. Indeed, GPS information clearly shows that “home” is in too many places for this user. We kept this information hidden from the system in this study because we used it to implement the oracle. However, in a real setting with actual users as oracles, this information would be part of the prior knowledge of the

Table 6.2: Table showing the means of transportation considered in the experiment and the mapping with the three superclasses used.

Label	On foot	Bus	Train	Car	Motorbike	Bike
Superclass	On foot	In vehicle				On bicycle

system and contribute to its capacity to identify inconsistencies in user feedback.

Additional types of users could be identified, e.g., an "average" user behaving similarly to Figure 6.4, or an "unpredictable" one for whom the system was incapable of learning neither subjective nor objective labels. Overall, these results highlight how the performance of SKEL is strongly affected by the behavior of the user, and call for additional work to deal with the difficult cases. For instance, finding a way to make the system aware of cases when the user is not being helpful would allow it to start searching for some additional source of information, e.g., nearby users, in order to compensate for the lack of reliable feedback.

6.3.6 SKEL in Few-Shot Learning

In order to evaluate the generality of the skeptical learning algorithm, we ran a second test aimed at recognizing the means of transport of users during their movements. We decided to focus specifically on movement activities since as for the location experiment, we could simulate a collaborative user with an oracle replacing the ASKUSER procedure in Algorithm 2, as will be shown in the following. During the data collection process, the user could indicate one of six possible means of transportation if in movement, as shown in Table 6.2. We created a hierarchy of labels by introducing three novel labels representing superclasses, namely *On foot*, *In vehicle* and *On bicycle*. In this setting, ISCOMPATIBLE works as for the experiment presented in Section 6.3.4, it returns true if the two labels belong to the same super-class or if they are the same, false otherwise. The CONSENSUS procedure is again implemented as a conservative choice that always returns the super-class, e.g., *On foot*, *In vehicle*, and *On bicycle*. We created an oracle for the prediction of means of transportation leveraging on the Google data that users were asked to provide at the end of the data collection experiment. By default, Google keeps track of the users' movements and detects how the user was moving at fixed time intervals. At every detection, a list of possible labels (namely *On foot*, *In vehicle*, *On bicycle*,

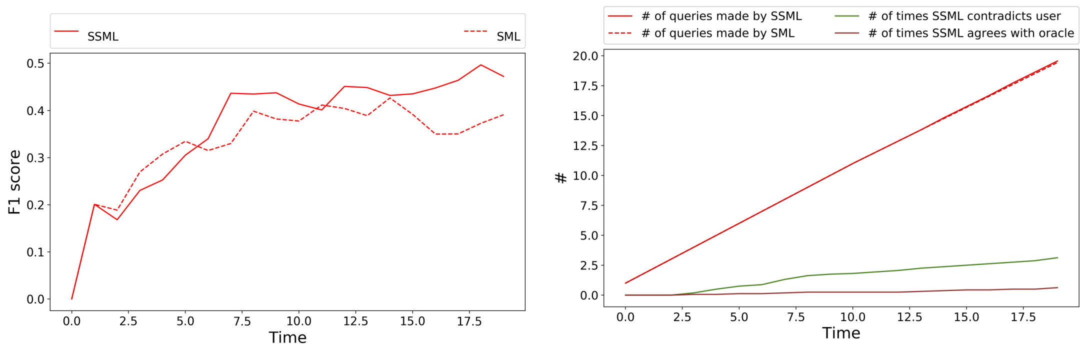


Figure 6.7: Results of SKEL and non-skeptical algorithm on the task of predicting the means of transportation.

Still, *Tilting*, *Exiting vehicle* and *Unknown*) is provided with an attached confidence value, from 0 to 100. In processing Google data, we ignored the labels *Unknown*, *Still*, *Tilting* and *Exiting vehicle*, which have no correspondence in the set of candidate user answers, and focused on *On foot*, *In vehicle* and *On bicycle*. Google data, when available, have a frequency of a label per minute on average. On the other hand, users provide labels once every 30 minutes. We combined Google provided labels into oracle labels by taking the most common label over the 30-minute interval. As for the prediction of location with prior knowledge, the oracle only provides labels for the superclasses of the hierarchy (see Table 6.2). As for that setting, the ground truth label is the user provided one if compatible with the one detected by the oracle; otherwise, it is the label of the oracle. Out of 72 users participating in the experiment, only 31 of them provided Google data. Furthermore, movement data are sparser than location ones, with just 15 iterations labelled as moving activities on average for each user over the two weeks of the experiment. This is in line with the expected behavior of an average student that passes most of her time either at home or at the University, commuting only few times a day for short trips. This challenging setting is known as few-shot learning in the machine learning literature and allows us to test the ability of SKEL to achieve improvements even with just a handful of labeled examples.

Results are shown in Figure 6.7. As for the previous settings, the left graph reports the f1 score of the SKEL and SML predictors for increasing number of iterations, while the right graph reports the number of queries made to the user by SKEL and SML, the number of times SKEL contradicts the user and in how many of these cases, the user (as simulated by the oracle) ends up agreeing with the predictor. While having

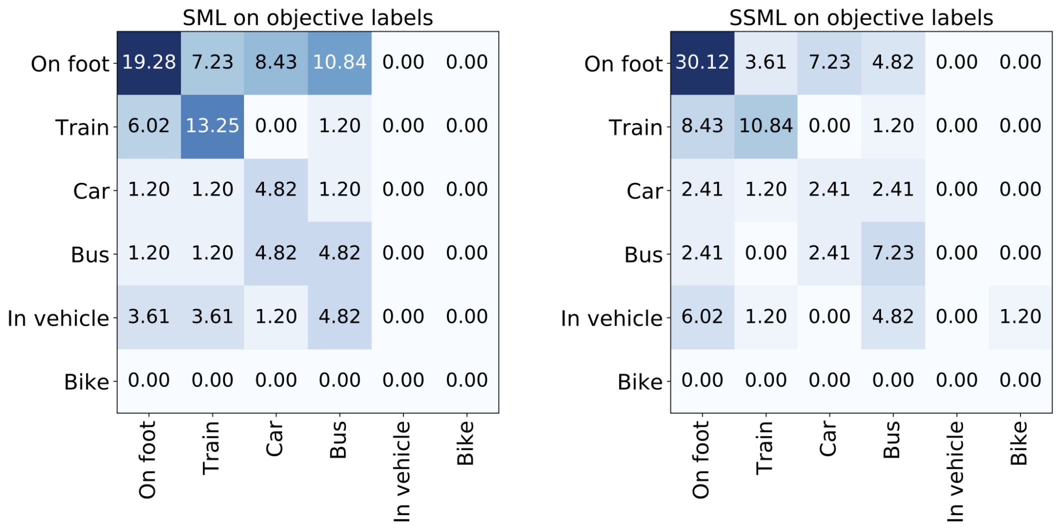


Figure 6.8: Confusions matrices of the non-skeptical algorithm (left) and SKEL (right) algorithms computed at the last iteration reported in Figure 6.7.

larger oscillations and closer curves because of the minimal number of examples, the f1 prediction results show a clear trend towards an advantage of SKEL over SML. The difference in the number of iterations (i.e., training instances) between these graphs and those in Figures 6.2 and 6.3 can give an idea of the amount of supervision available in this setting and confirm the capability of SKEL to exploit even very limited feedback to improve prediction performance. Figure 6.8 reports confusion matrices of the SML (left) and SKEL (right) algorithms at the last iteration. The advantage of SKEL is due to its ability to better predict the *On foot* and *Bus* labels, at the cost of a moderate decrease in the accuracy of the *Train* and *Car* ones.

6.4 Envolved Skeptical Learning with Prior Knowledge

In Section 6.2 we have introduced the skeptical learning architecture and the main SKEL algorithm. In this section, we will introduce an evolved skeptical learning, where the predictor is implemented as a hierarchy of classifiers matching the prior knowledge. The evolved algorithm will be compared with three alternatives which have different strategies dealing with the conflict. And the extended evaluation and new results will be introduced in next section.

6.4.1 The Prior Knowledge

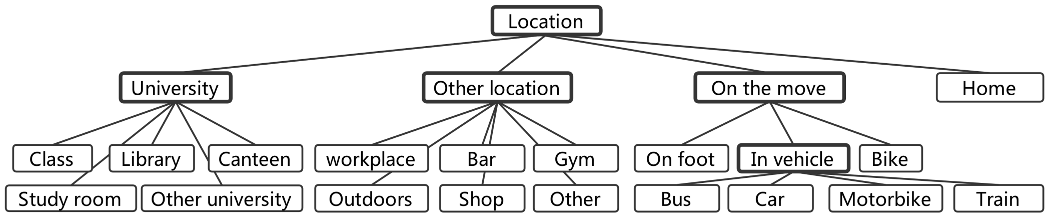


Figure 6.9: Ontology of the labels used in the experiment. Bold contours correspond to classifiers in the PRED procedure.

We have introduces two components GK (Ground Knowledge) and SK (Schematic Knowledge) in the SKEL architecture in Section 6.2.1. Both of the components provide the system prior knowledge that can be used in the conflict resolution phase. They allow to infer that the machine and the user meant the same thing with different labels. The difference is that GK contains the factual knowledge, for example, the fact that Fausto's office is part of the Department building, while SK contains general knowledge in concept level, for example, resting is a more general concept than sleeping. The GK component, stored as a knowledge graph, can be updated and growing as it is where the machine accumulates the knowledge learned in time. The SK component, however, containing general knowledge about the world, is a static and not growing component.

A detailed description of SK is outside the main focus of our work; the interested reader can read [35] for a detailed account of the approach and the resource. Here is is worth noticing that SK is a hierarchy, more precisely a multi-rooted DAG, where each node is labeled with a concept and where a child-parent link codifies a subsumption axiom between a more specific and a more general concept. This hierarchy has more than 100 thousand nodes and codifies a few million subsumption axioms. One trivial example of SK content is a subsumption axiom stating that $vehicle$ is a more general concept than bus, in DL formulas $bus \sqsubseteq vehicle$. Another example is any general statement about locations and sub-locations, for instance, the fact that Department buildings are always inside (they are partOf) the University premises. The hierarchy in Figure 6.9, which has been exploited in the evaluation, is a very minor portion of the SK hierarchy itself. From a technical point of view, it is worth noticing how the hierarchy in Figure 6.9 is (partially) a *partOf* hierarchy rather than an *is-a* hierarchy, the latter being directly codified into a set of subsumption axioms. PartOf relations on locations can be codified as subsumption relations by seeing locations as sets of points. Under this assumption, the points in a part are always a subset of the points in the whole. The SK encompasses both *isa*-like and *partOf*-like relations (the first being called *hyponym*, the second, *meronym*) [35].

6.4.2 Evolved Algorithm

Algorithm 3 Evolved Skeptical Learning (SKEL)

```
1: procedure SKEL($\theta$)
2:     init $\mathbf{c}^{\mathrm{u}} = 1, \mathbf{c}^{\mathrm{p}} = 0$
3:     while TRAINMODE($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \theta$) do
4:         $\mathbf{x}_t = \text{SENSORREADING}()$
5:         $y_t = \text{ASKUSER}()$
6:         $\hat{y}_t = \text{PRED}(\mathbf{x}_t)$
7:         $\text{TRAIN}(\mathbf{x}_t, y_t)$
8:         $\text{UPDATE}(\mathbf{c}^{\mathrm{p}}, \hat{y}_t, y_t)$
9:     while REFINEMODE($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \theta$) do
10:        $\mathbf{x}_t = \text{SENSORREADING}()$
11:        $y_t = \text{ASKUSER}()$
12:        $\hat{y}_t = \text{PRED}(\mathbf{x}_t)$
13:        $\text{SOLVECONFLICT}(\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \mathbf{x}_t, \hat{y}_t, y_t)$
14:    while True do
15:        $\mathbf{x}_t = \text{SENSORREADING}()$
16:        $\hat{y}_t = \text{PRED}(\mathbf{x}_t)$
17:        if $\min_{\hat{y}'_t \in \text{SMERS}(\hat{y}_t)} \text{CONF}(\mathbf{x}_t, \hat{y}'_t, c^{\mathrm{p}}_{\hat{y}'_t}) \leq \theta$ then
18:            $y_t = \text{ASKUSER}()$
19:            $\text{SOLVECONFLICT}(\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \mathbf{x}_t, \hat{y}_t, y_t)$
```

How the prior knowledge would be used in the conflict management phase has been introduced in Section 6.2.3. The main innovation of the evolved skeptical learning algorithm is that the system is given semantic knowledge as the prior knowledge. And this knowledge is used not only to solve the conflict of labels, but also to build a set of classifiers matching the hierarchy of ontology as the predictor. The pseudocode of evolved SKEL algorithm is reported in Algorithm 3. We will introduce the main changes of this algorithm comparing to the previous one in the next subsections.

Algorithm 4 Hierarchical predictor

```
1: procedure PRED($\mathbf{x}$)
2:     init $y = y_{root}$
3:     while $not$ IS_LEAF($y$) do
4:         $y = \text{argmax}_{y' \in \text{CHILDREN}(y)} f_y(\mathbf{x}, y')$
5:     return $y$
```

6.4.2.1 The Predictor

The prediction procedure PRED is implemented as a hierarchy of classifiers matching the SK ontology for the property to be predicted. There is one multi-class classifier for each internal node in the ontology (bold contour nodes in Figure 6.9), discriminating between its children. Prediction starts from the root classifier and progresses down in the hierarchy following the highest scoring class at each node, until a leaf node is reached which is the class eventually predicted. In the first iteration, when no training has been performed yet, each classifier returns a random label. See algorithm 4 for the pseudocode of PRED, where $f_y(\mathbf{x}, y')$ is the classifier at node y, and y_{root} is the most general value for the property (e.g., a generic *Location* in Figure 6.9).

Note that, thanks to the fact that the transitive closure over the SK axioms is pre-computed, the system can infer, at run time, all the labels which subsume that provided by the user, i.e., all those from the root to the user label. Each classifier in the path is thus retrained with the addition of its corresponding input-output pair during a TRAIN procedure.

6.4.2.2 The Confidence Computing

After training, the confidence of the predictor is modified via the UPDATE procedure which takes as input the ground truth label and the one predicted *before* the training step. The UPDATE procedure takes as input a confidence vector, a tentative label ($\hat{y}_t$) and a ground truth label (y_t), and updates the confidence vector according to the relationship between the two labels. The new confidence vector computed is as a label-wise running average accuracy over the current and past predictions, for a certain window size d. Notice that, similar to the TRAIN procedure, the confidence updates are applied not only to the predicted and ground truth label pair, but also to all implied label pairs according to the SK, i.e., those from the root to the predicted (respectively) ground truth label.

However, difference from equation 6.2, where the score of a prediction y is the average score given to that label by the classifiers in the ensemble, here we use the score given by the parent classifier only:

$$\textsc{Conf}(\mathbf{x}, y, c^{\mathrm{p}}_y) := c^{\mathrm{p}}_y \cdot f_{\textsc{parent}(y)}(\mathbf{x}, y) \tag{6.5}$$

Same to the strategy used in Section 6.2.2, the system stays in training mode till the expected probability of contradicting the user becomes bigger than the threshold (see the Equation 6.3), and the system stays in refine mode as long as the expected probability of querying the user exceeds the threshold (see the Equation 6.4).

However, since we are using the semantic knowledge and the predictor is implemented according to the hierarchy in Figure 6.9, when a predicted label is compared with a user provided label, care must be taken in making a sensible comparison. If the two labels belong to different branches of the hierarchy (e.g., *Train* and *On foot*), they cannot be directly compared as confidences are normalized across siblings. Therefore, the system recovers all the labels in the hierarchy up to the first common root, i.e., *the least common subsumer* [4] and compares them instead of the original ones. Thus, in the previous example, *Train* implies *In vehicle* which is then compared to *On foot* as they are both children of *On the move*. Therefore, the labels to be compared in Equation 6.3 are obtained as $(\hat{y}, y) = \textsc{LCS\_Children}(\textsc{Pred}(\mathbf{x}), y_u)$, where $\textsc{Pred}(\mathbf{x})$, the predicted label for input $\mathbf{x}$, y_u is the label provided by the user for that input, and the $\textsc{LCS\_Children}$ procedure outputs a pair of implied predicted/user labels which are children of the least common subsumer.

When the system enters the regime mode, the system stops asking feedback for all inputs and will start an active learning strategy. Notice that, see line 17 in Algorithm 3 and line 17 in Algorithm 1, instead of simply comparing the confidence the prediction label $\hat{y}_t$ with threshold, it takes a conservation approach and considers the smallest confidence of all the prediction's subsumers. For instance, if the prediction is *Train*, it will check the confidence in *Train* of classifier *In vehicle*, the confidence in *In vehicle* of classifier *On the move*, and the confidence in *On the move* of classifier *Location*. And if the smallest of those three confidence is not exceeds the threshold, the active learning strategy will start.

Algorithm 5 Procedure for solving labeling conflicts

```
 1: procedure SOLVECONFLICT($\mathbf{c}^{\mathrm{p}}, \mathbf{c}^{\mathrm{u}}, \mathbf{x}, \hat{y}, y$)
 2:     if ISCOMPATIBLE$(\hat{y}, y)$ then
 3:         $y^* =$ CONSENSUS$(\hat{y}, y)$
 4:         UPDATE$(\mathbf{c}^{\mathrm{p}}, \hat{y}, y^*)$
 5:         UPDATE$(\mathbf{c}^{\mathrm{u}}, y, y^*)$
 6:     else
 7:         $(\hat{y}', y') =$ LCS_CHILDREN$(\hat{y}, y)$
 8:         if CONF$(\mathbf{x}, \hat{y}', c^{\mathrm{p}}_{\hat{y}'}) \leq c^{\mathrm{u}}_{y'} \cdot \theta$ then
 9:             TRAIN$(f, \mathbf{x}, y)$
10:             UPDATE$(\mathbf{c}^{\mathrm{p}}, \hat{y}, y)$
11:         else
12:             $y^* =$ CHALLENGEUSER$(\hat{y})$
13:             if $not$ ISCOMPATIBLE$(\hat{y}, y^*)$ then
14:                 TRAIN$(\mathbf{x}_t, y^*)$
15:             UPDATE$(\mathbf{c}^{\mathrm{p}}, \hat{y}, y^*)$
16:             UPDATE$(\mathbf{c}^{\mathrm{u}}, y, y^*)$
```

6.4.2.3 The Conflict Management Algorithm

The evolved SOLVECONFLICT procedure is described in Algorithm 5. SOLVECONFLICT takes as input the predictor and the user confidence vectors $\mathbf{c}^{\mathrm{p}}$ and $\mathbf{c}^{\mathrm{u}}$, the input $\mathbf{x}$ with its predicted label ($\hat{y}$) and the label given by the user (y). The first step is to compare the two labels according to the ISCOMPATIBLE procedure. As the SK encodes a subsumption hierarchy for the property of interest, the procedure returns true if the two labels are the same or if one subsumes the other. When the two labels are compatible, a consensus label is taken as the ground truth, and the predictor and user confidences are updated accordingly. As a natural choice for the consensus, the system chooses the more general among the two labels, this being the choice also used in the experiments. The motivation is that both the user and the system are taken to be truthful and, therefore, the system chooses the label which carries more meaning.

If the two labels are not compatible, a conflict management phase starts. In particular, when the confidence of the prediction is not large enough, the user label is taken as ground truth, the predictor is retrained with this additional feedback, and its confidence is updated accordingly. Otherwise, the system contradicts the user, advocating its own prediction as the right one[1]. The user is now responsible for solving the conflict. She can decide to stick to her own label, realize that the machine is right and converge on the predicted one, or provide a third label as a compromise. Note that the user can, and often will because of imperfect memories, make a prudent choice and return an intermediate node of the label hierarchy rather than a leaf. As we are assuming a non-adversarial setting, and we aim at providing a support to the user rather than a replacement for her, eventually the system will trust the latest provided label (even if unchanged), which in turn will become the ground truth. As a last step, a compatibility check is performed to verify whether there is a need for retrain and the predictor and user confidences are updated.

6.5 Experiment and Results of evolved SKEL

6.5.1 Comparing SKEL with Three Alternative Strategies

As discussed in Section 6.4.2.1, the PRED procedure of SKEL consists of a hierarchy of multiclass classifiers, one for each internal node in the hierarchy. Each classifier implemented a random forest [7], which is known to be robust to labeling noise [33], to evaluate the ability of SKEL to improve over an already noise-robust baseline. The confidence parameter θ of SKEL has been set to 0.2, which has resulted in a reasonable trade-off between accurate training and cognitive effort for the user. For simplicity, we have used an infinite queue ($d = \infty$) for the confidence update due to the relatively short duration of the experiment. Being the experiment built on previously collected data, we could not query users in real-time in case of conflicts. To simulate a collaborative, non-adversarial user, we assumed that the user returns the ground truth label when her initial label is contested.

The evaluation of SKEL is done by comparing it with three alternative algorithms:

- NONSKEL, that never contradicts the user (obtained by replacing SOLVECONFLICT with a train and update step, as happens in the training phase);

[1] In order to support its argument, the machine could provide some sort of explainable critique to the user feedback, in terms of counter-examples or evidence of inconsistencies with respect to the SK. This is a promising direction for future research.

- IGNORE, that simply ignores any example for which a conflict arises (obtained by removing everything from the ELSE onwards in Algorithm 2);

- BOTHER, that always contradicts the user (obtained by calling CHALLENGEUSER after all ASKUSER calls, and removing SOLVECONFLICT).

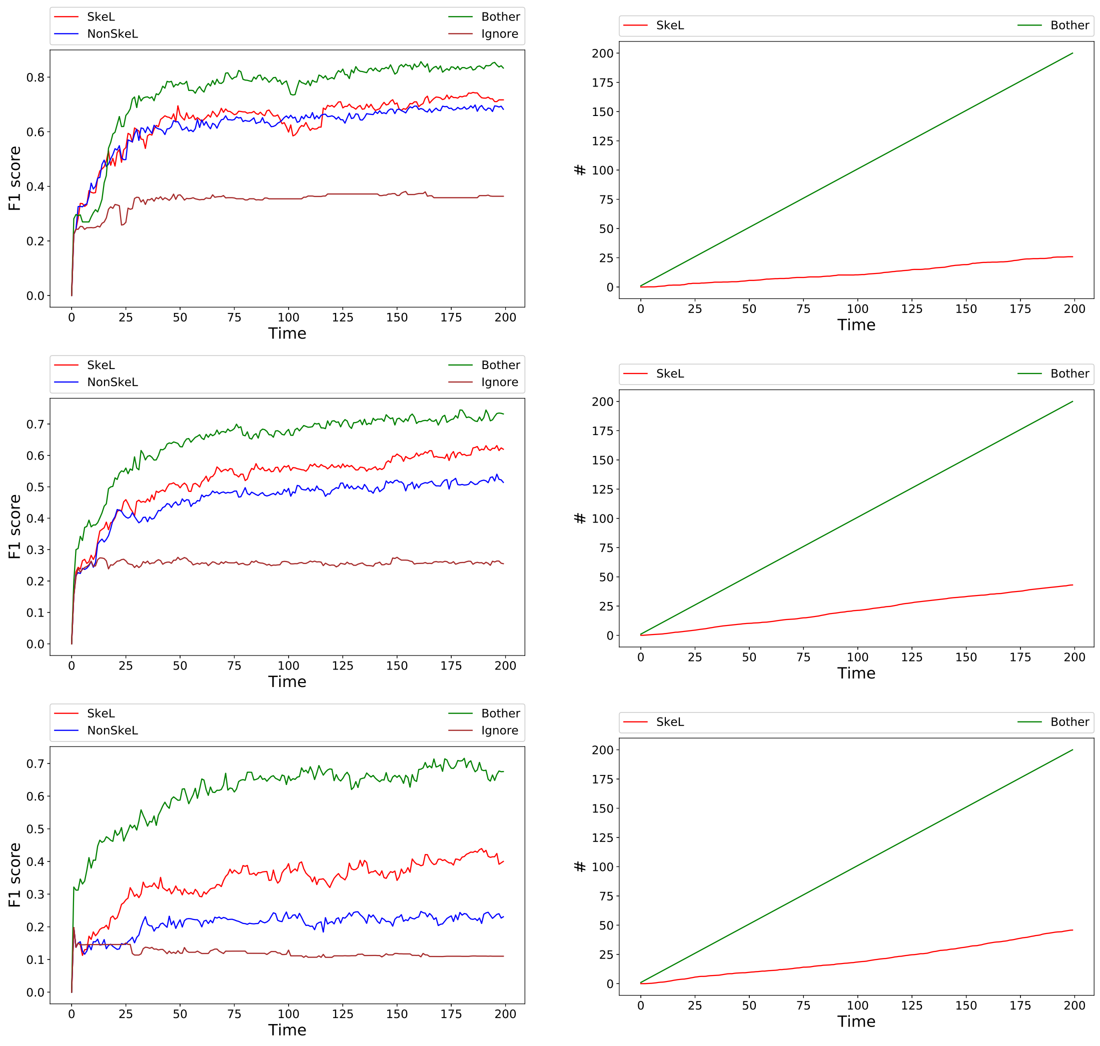


Figure 6.10: Results averaged over users with at most 10% (top row), from 10% to 25% (middle row) and more than 25% (bottom row) labelling errors. Left colume: F_1 scores on left-out data. Right column: number of times user is contradicted.

As presented in the previous section, a surprisingly high proportion of labels present inconsistencies. To estimate the effect of this large and very diverse proportion of labelling errors on the performance of the system, we divided the set of users in the three groups reported in the table. Figure 6.10 reports the results of SKEL and of the three alternatives for an increasing number of iterations. Each row represents the results for a different group of users: at most 10% labelling errors (top), 10% to 25% (middle), more than 25% (bottom). The left column reports f_1 scores averaged over all users in the corresponding group with a number of training samples greater than 200. The score for each user is computed on a fixed test set, namely the latest 15% of the all data available for that user, which was not used for training. This score provides an estimate of the performance of the algorithms when making predictions on future data. Note that we consider a label as correctly predicted if it is compatible with the ground-truth label, because this is the only type of reliable supervision we have access to. Results clearly indicate that our skeptical algorithm (red curve) consistently outperforms a non-skeptical alternative (blue curve). As expected, the advantage is moderate for users with a relatively small fraction of labelling errors (top figure), and grows with the unreliability of the users, reaching a gap of 0.20 for users with more than 25% labelling errors (bottom figure). Ignoring conflicting cases (brown curve) is clearly not an option, as it achieves the worst performance in all cases. On the other hand, having always access to correct supervision, BOTHER (green curve) clearly achieves the highest performance. However SKEL is capable of getting reasonably close to this upper bound when enough iterations are provided, at a fraction of the cost in terms of user effort. The right column reports the number of times the user is contradicted for SKEL (red curve) and for BOTHER (green curve), for which they are simply the number of iterations. SKEL clearly contradicts more when facing increasingly unreliable users. However, the cost remains substantially lower than the one of BOTHER, going from 13% (top figure) to 23% (bottom figure).

Figure 6.11 reports the confusion matrices of the *Location* classifier (the root of the hierarchy in Figure 6.9) for the last time instant of the different algorithms shown in Figure 6.10. Results are normalized over the sum of all the elements in the matrix. In each matrix, rows are ground truth labels, columns are predicted labels. Matrix entries are coloured, with darker shades corresponding to larger entries. Each row represents the results for a different group of users who have at most 10%, from 10% to 25% and more than 25% labelling errors respectively. The matrices in the first column report results of the SKEL algorithm, the second column report results of the NONSKEL one, the third column report results of the BOTHER one, and the last column report results of IGNORE

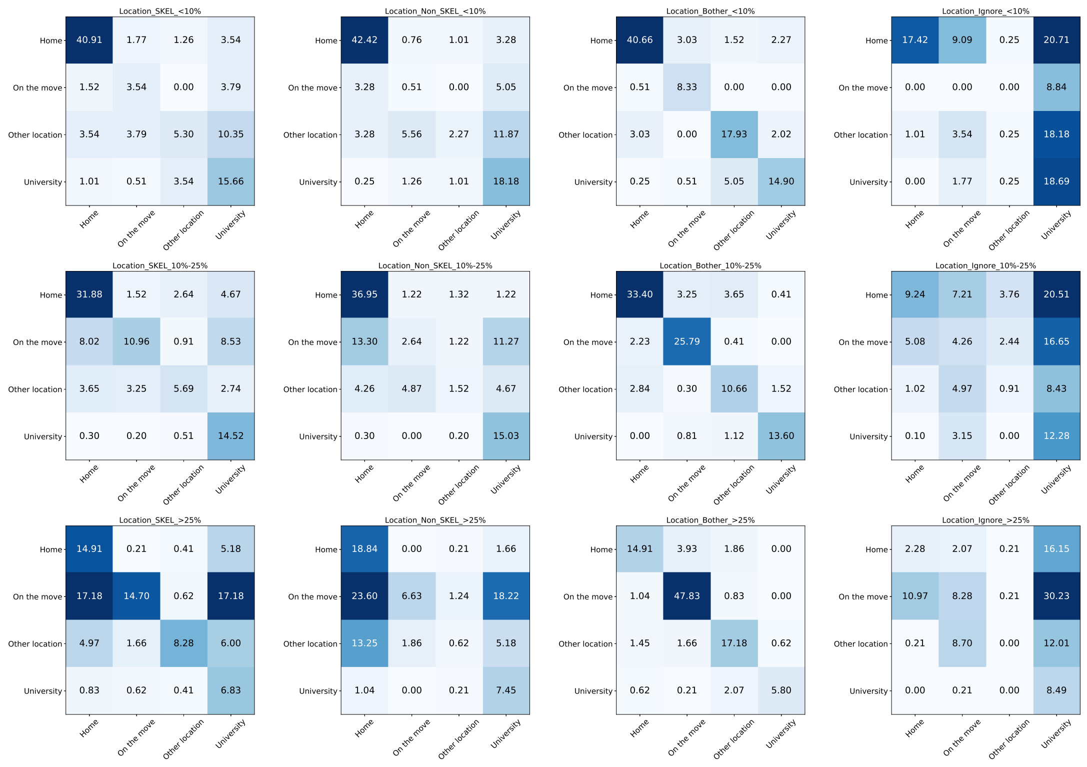


Figure 6.11: Confusion matrices for the evaluation on oracle labels at the last iteration of the algorithms. Oracle labels are on the rows, predicted labels are on the columns. All the matrices refer to the *Location* classifier and are normalized over all entries.

one.

Clearly, BOTHER has the best performance, as can be seen by the dark colouring of the main diagonal (corresponding to correct predictions), and shows the upper limit one can reach by having always access to the ground truth during training. While not as accurate, SKEL has a similar qualitative behavior, with on-diagonal entries typically larger than off-diagonal ones. The bad performance of IGNORE and NONSKEL are due to the fact that they tend to overpredict certain classes (as shown by the dark columns), *University* and *On the Move* for IGNORE, *University* and *Home* for NONSKEL.

The first row represents the results on the users with fewer labeling errors. Given that these users tend to provide high-quality labels, SKEL is only slightly better than NONSKEL. This is consistent with the F_1 score in the first row of Figure 6.10, where the rad curve and blue curve are close to each other. However, SKEL manages to substantially improve the recognition ability for *On the move* and *Other location*, the classes for which the user feedback is least frequent, at the cost of a (smaller) decrease in accuracy for the most popular ones, *University* and *Home*. This behaviour is even more evident when considering users with a higher fraction of errors (middle row, from 10% to 25% labelling errors), where the fraction of times *On the move* and *Other location* get predicted and the prediction is correct increases from 2.64% to 10.96% and from 1.52% to 5.69% respectively. The third row reports results for the least reliable users, with more than 25% labeling errors. While the distance to the "ideal" (for the machine) setting represented by BOTHER increases, the advantage of SKEL over NONSKEL (and IGNORE) also widens. The tendency of the user to always reply with *Home* and *University* does affect SKEL, which starts to show a similar behaviour (dark columns in the matrix). On the other hand, despite the substantial unreliability of the use feedback, it still manages to recover a large fraction of the *On the Move* (from 6.63% to 14.70%) and the *Other location* (from 0.62% to 8.28%) classes, both virtually lost by NONSKEL.

6.5.2 Variability of Users

Similar to the analysis on users' performance in Section 6.3.5, in this section, we investigate the performance of the SKEL algorithm with respect to two types of labels. One is *objective label* provided by the oracle, and the other one is *subjective label* provided by the user. By analysing performance graphs of every single user, we can identify four different patterns that are related to distinct behavior patterns. Figure 6.12 shows the results for these four prototypical users. Each row refers to a specific user. Left figures report f_1 scores with respect to the *objective labels* and the *subjective labels*. Figures on the right column report the number of queries by SKEL, the number of times when SKEL challenges the user, and the number of times when SKEL agrees with the oracle label. It is clearly to see that these prototypical performance of users match the types we find in Section 6.3.5, namely *Inattentive user*, *Predictable user*, *Reliable user* and *Tricksy user*.

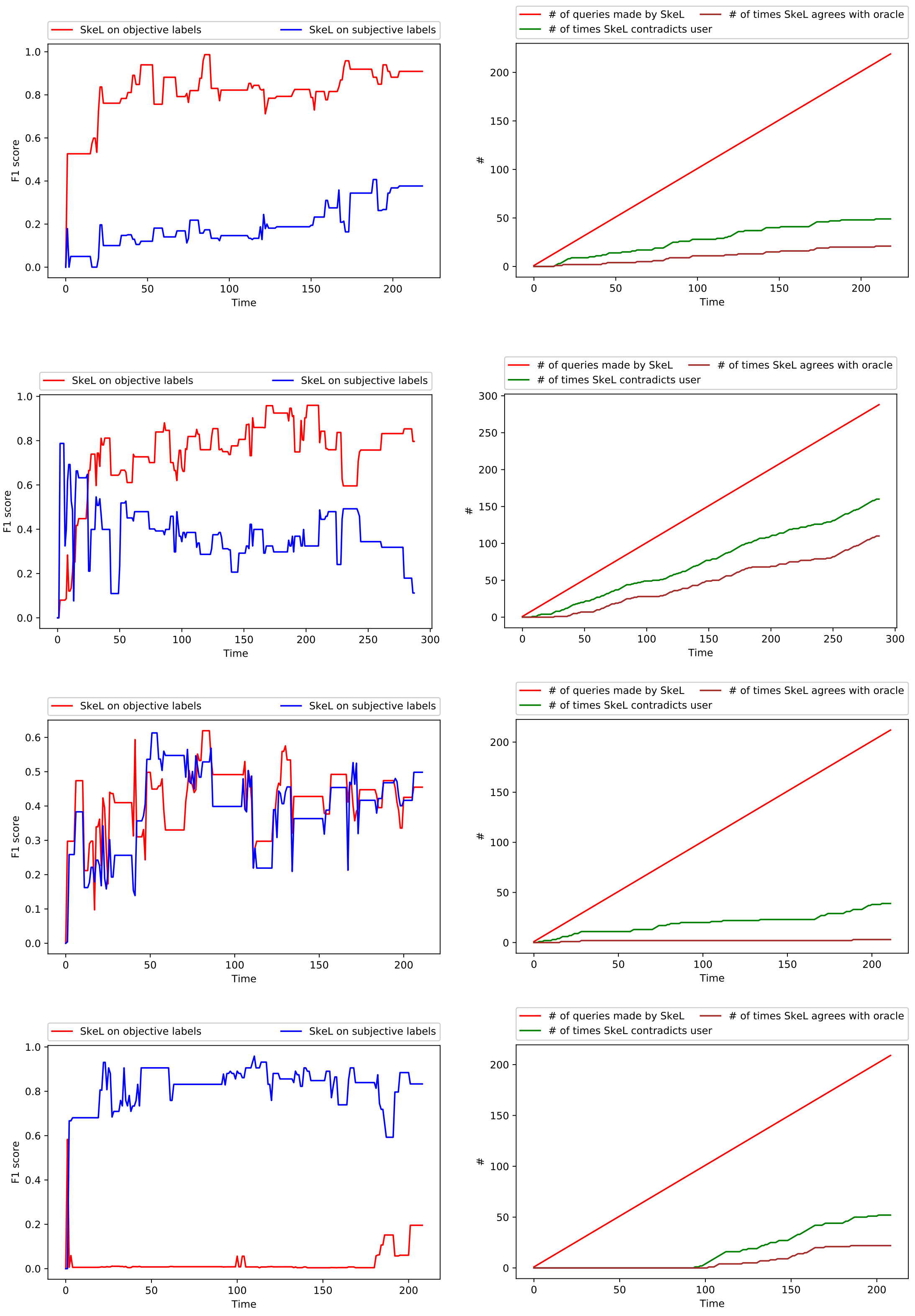


Figure 6.12: Results for four different prototypical users, namely an *inattentive user*, a *predictable user*, a *reliable user*, and a *tricksy user* (from the top to the bottom).

6.6 Summary

In this chapter we introduced Skeptical Learning (SKEL) as a paradigm for dealing with the unreliability of users when providing labels that describe their personal context. The fundamental idea is to use the available knowledge when deciding what is more reliable between the output of the machine learning algorithms and the user input, and to engage in a conflict resolution phase when a controversy arises. We also introduced an envolved skelptical learning algorithm, which implemented the predictor as a hierarchy of classifiers matching prior semantic knowledge. Experimental results show the pervasiveness of mislabelling when dealing with feedback from non-expert users, and the effectiveness of Skeptical Learning in addressing the problem as compared to existing approaches to deal with noisy labels.

Chapter 7

The Skeptical Learning Platform

7.1 Introduction

Mobile crowd sensing (MCS), with reped development of technologies in mobile devices, has become an emerging sensing paradigm in recent years, where ordinary citizens can contribute data sensed or generated from their mobile devices. The goal of MCS is to extract crowd intelligence from the information provided by users and their smart phones. And the crowd intelligence can be used to provide better people-centric service to users in return. For example, [108] proposed an application that monitors the noise level in the city by collecting and analyzing the microphone data from users' phone. Similar to that, the application in [66] can monitor the traffic congestion from the user's coordination data. These applications can then provide useful environment information for the crowds and society. Another kind of application is concerning the user's own benefits. For example, the application in [75] is designed to monitor the user's daily activities and emotions for health caring purposes. It collects sensor data and meanwhile asks for the user's self-report observations.

Human involvement is one of the most important characteristics of MCS. Most participants of MCS applications are norm (non-expert) users, which allows systems to collect human annotations and information as much as possible. However, as we mentioned in Chapter 6, the biggest challenge is was brought due to this characteristic. Normal users might provide incorrect annotations, intentionally or unintentionally, as input to the system. These incorrect annotations could impact the performance of downstream machine learning algorithms. Controlling data quality is much easier in

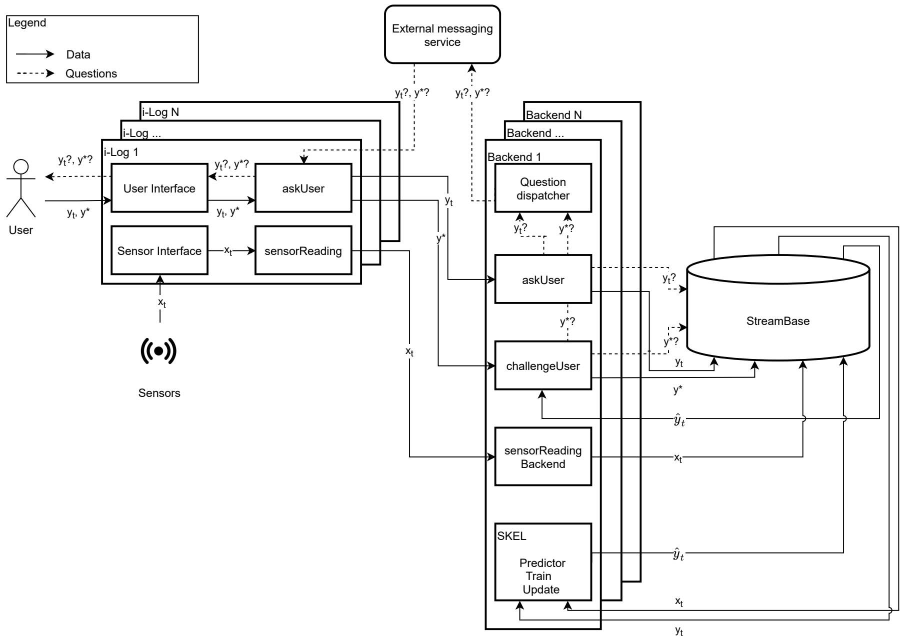


Figure 7.1: The platform architecture.

traditional sensing systems which rely on high-quality devices and annotations from trained experts. But to the best our knowledge, there is no existing MCS system that can deal with the mislabeling issue caused by normal users.

In this chapter, we propose a general MCS platform for integrating at scale sensor data and label data generated by users, together with the static knowledge of the world. The former informs SKEL main algorithm about the world evolution, while the latter codifies the prior knowledge which will be used in the conflict resolution phase. On the top of the platform, it runs the SKEL algorithm that we proposed in last chapter to deal with the unreliability from users.

7.2 The architecture of the platform

The SKEL platform is depicted in Figure 7.1. To better understand it, the reader should assume that all processes between components run asynchronously. This platform combines two parts: the i-Log front-end interacting with the world in terms of user and device (left) and a back-end implementing, among others, the SKEL logic (right). There is an instance of i-Log, and corresponding back-end, per user while there is a single storage for all the users. We analyse below these three components.

7.3 Front End

One of the core functionalities provided by the platform is the ability to acquire knowledge about the world via both sensors and users, and to make these data available for later processing and data analysis. Since the concept behind SKEL is to empower ordinary citizens, this task becomes more challenging with respect to solutions in other scenarios. As from Figure 7.1, i-Log has two main input elements: the user on one side, that provides annotations in form of answers y_t, y^* to questions $y_t?, y^*?$. The former is a type of question sent at fixed time interval, which answers are mainly used as standard annotations, while the latter is sent on demand, when the system decided to challenge the user. On the other side there is the world, that is captured through sensors embedded in the smartphone and wearables, generating sensor readings x_t. i-Log is composed by four high level components: (1) *askUser*, (2) *sensorReading*, (3) *user interface* and (4) *sensor interface*. Let us analyze these components in turn.

7.3.1 askUser

The main functionalities of *askUser* are related to the collection of feedback from the user in form of answers to questions. Its responsibilities are end-to-end, meaning that it has to deliver the questions $y_t?, y^*?$ to the user (dashed lines in Figure 7.1), but also to deliver the answers y_t, y^* to the back-end system (continuous lines in Figure 7.1). The questions are received from the back-end through an external messaging system such as Google's messaging service Firebase[1], or Baidu Messaging[2], among others. The advantages of using such external services can be summarized in four elements: (1) battery optimization, (2) time to deliver, (3) size of the message and (4) caching. Concerning battery optimization, Google Firebase has a functionality built in that is responsible for delivering messages in an efficient way in ”windows”. It basically consists in a mechanism that groups together messages coming from different applications and delivers all of them together, i.e., to wake up the phone only once when in sleep mode.

[1]https://firebase.google.com
[2]https://push.baidu.com/

This leads to important savings in terms of battery life on the device side, since the phone doesn't need to listen to incoming messages continuously. At the same time, messages are usually delivered in a short time frame, in most cases within few seconds which is perfect even for real-time use cases.

Another important aspect to keep under control on mobile devices is the network cost, especially in some countries. If it is true that nowadays it is not uncommon to see data plans for tens of GB, this is not true in all countries around the world. For this reason, the transmission of data should always be optimized and compressed. Firebase helps in this regard since every message has a size limit of $4KB$, which forces the developer to transmit only the essential information. Moreover, sending all messages together and compressed reduces the amount of bytes that need to be transferred.

Finally, the last advantage of using an external delivering message system is that in most situations, i.e., with Firebase and Baidu, a caching system is provided. In fact, smartphones are characterized by intermittent network connections, that can leave the device disconnected from the internet multiple times a day for an arbitrary amount of time. Consider for example when the user enters a building with no signal, or when the devices switches between cellular networks. In such situations it's important to have a caching system in place when delivering messages, since if the message is delivered at the wrong moment in time, when the phone is disconnected, it could get lost. Services like Firebase provide a robust caching mechanism that can be customized on a per-message basis. We can specify the amount of minutes according to which the message should be delivered, as soon as the device is connected. If this time interval expires and the message has not been delivered, the message itself is discarded. This mechanism is perfect in the situations mentioned above, when the phone disconnects for short periods of time, but it applies also when the user turns off the phone for hours, i.e., during the night.

It is important to underline that this component works in an asynchronous fashion: the questions are received as soon as they are generated by the back-end, but the answers are collected asynchronously, since they involve an action from the user. Finally, this component collects two types of answers, y_t that are timediaries asked at fixed time intervals by the *askUser* back-end component, and y^* which instead are answers to a challenge request by SKEL.

7.3.2 sensorReading

The main functionality of *sensorReading* is to collect sensor readings x_t from the edge device and synchronize them with the back-end system. Similarly to the *askUser* component, it also works asynchronously: the front-end collects sensor readings from the device continuously, without any input from the back-end, and synchronizes them only at specific moments in time, to make them available to the back-end and SKEL. This is done for different reasons, the most important ones being the battery life of the edge device and the network costs for the user. In fact, depending on the MCS task, x_t can be big in size, considering that it can collect up to 30 different sensor streams simultaneously, with frequencies up to $100Hz$. The list of the supported sensor streams and the sampling rate has been introduced in Table 3.2 of Chapter 3 when we introduce the frequency of smartphone-based data.

7.3.3 User interface

User interface is the component that the user interacts with directly on her smartphone when it comes to collecting the user answers to *askUser* and *challengeUser* in SKEL. Its main responsibilities are to display a user interface tailored for the specific device that contains the corresponding questions $y_t?, y^*?$. In the current implementation, there are different question formats that are usable by the system. They all have in common the question field, which is text based. Some of them have additional graphical elements that can support the question, such as images, among others. Finally, the answer field is diverse for each type. The most relevant question formats are: text question with multiple choice answer (with (1) single and (2) multiple selection allowed), (3) text question with free text input allowed, (4) text question supported by a map view and a pointer displayed on it with multiple choice answer, (5) text question supported by a map view and a path displayed on it with multiple choice answer and (6) a text question with the possibility for the user to take a picture. An example of the usage of the latter format is presented in [62].

An important feature of the user interface is that it keeps track of is the time required by the user to answer each question, i.e., the delay between when a question is received and when it is answered, plus the time taken to provide the answer, namely the elapsed time between when the user opens a question and when he presses the finish button.

These two parameters are important because they allow to filter out answers during the analysis. In [41] linked these time measures to biases in the respondents, memory and carelessness.

Finally, this component is also in charge of implementing the logic of dependency between questions and answers. This feature is crucial in that it allows to customize the next question based on the previous answer given by the user. For example, if the user replies to the question ”What are you doing?” with ”I'm travelling”, then the next question will be ”Where are you going?” rather than ”Where are you?”. These types of dependency must be setup in the back-end while generating the sequence of questions.

7.3.4 Sensor interface

This is the component that interacts directly with the hardware of the edge device, e.g., the accelerometer and the gyroscope. Its main goal is to abstract the hardware from the higher level processes and to generate the streams of sensors data x_t. A second main functionality of this component is also to generate streams of what we call ”*software sensors*”. Software sensors are software modules that generate data generated by monitoring the various software modules running on the device. One example of software sensor is the one that monitors the status (ON/OFF) of the screen, another one detects the name of the application that is running in foreground, every 5 seconds, among others.

The *Sensor interface* component is of primary importance to enable a systematic sensor data collection on different edge devices. This applies to both hardware and software sensors. In fact, in each MCS task we will have multiple smartphone models, with different operating system's versions, since the user uses her own personal smartphone. These aspects affect the sensor data collection in two ways: (1) different smartphones have different sensors and (2) even if the sensor set is the same, those sensors will generate data differently on each device. The *Sensor interface* component addresses both. First of all, for each device it enables only the sensor streams that are present on the smartphone, automatically, disabling the other ones. Secondly, it collects data in such a way so that the generated streams are consistent, with similar collection frequency, amplitude, precision, and so on.

7.4 Back End

The back-end, as shown in Figure 7.1, works paired with the front-end part to provide an end-to-end user experience. This is why each user has an instance of the back-end, as well as the front-end. In fact, each user has different parameters that customize the dynamics in the back-end, to deliver a tailored user experience. Three out of five components map directly between front-end and back-end, (1) *askUser* (and (2) *challengeUser*), (3) *sensorReading*, while the other two (4) *Question Dispatcher* and (5) *SKEL* are only available in the back-end.

7.4.1 askUser

askUser performs two tasks: (1) it accepts answers y_t from the device, processing and storing them in the StreamBase database (see the next subsection), while, at the same time (2) triggering questions $y_t?$ at fixed time intervals, tailored on the user. The time interval can vary because different users behave differently: some of them are more reliable in replying and have less variance in their activities throughout the day, while others may have a lot of variance or may be unreliable. For the former ones the system requires less annotations/ answers and the time interval can be kept in the order of 30-60 minutes, while for the latter more annotations are required. The content of each question is configurable in the system and is usually experiment-based. Since every MCS task has a different goal, the questions may differ every time. An example of such questions is presented in Figure 3.6 and Figure 3.7 in Chapter 3.

7.4.2 challengeUser

challengeUser similarly to *askUser*, accepts answers y^* from mobile devices and generates questions $y^*?$ to be sent to the the user. Differently from *askUser*, the questions are not generated at fixed time intervals but rather on demand, whenever a conflict in the SKEL component arises. When this happens, this component takes the prediction $\hat{y}_t$ from StreamBase which is generated by SKEL. Then, asynchronously this component detects a new question and triggers question $y^*?$ to the user.

7.4.3 sensorReading

sensorReading is a component responsible for accepting sensor data x_t from the mobile device. The operations is performs are (1) pre-processing the data, (2) normalizing them and finally (3) pushing such data in StreamBase. Due to the size of the data, this component is the one in the back-end that needs the biggest resources.

7.4.4 Question Dispatcher

Question Dispatcher is the component that is responsible for efficiently delivering questions $y_t?, y^*?$ to the mobile devices. To do so it relies on external services depending on the need: the system can be easily configured in this regard. Using external services removes an important overhead on our side and allows to send content to mobile devices efficiently as previously explained in the front-end part.

7.4.5 SKEL

SKEL is the architectural component that embeds an implementation of the SKEL algorithm. It takes sensor data x_t and user's answers y_t as input to train the predictor. When new sensor data is generated by the edge devices and synchronized with the server, it generates a prediction label $\hat{y}_t$. SKEL can detect a conflict by comparing user's answer and predictor's label. When it happens, and the predictor's confidence is high enough, it triggers *challengeUser* to generate an additional question to be sent to the user. The user's answer y^* is asynchronously collected and considered as ground truth. It is used to retrain the predictor and to update the confidence value of user and predictor accordingly.

7.5 Stream Base

StreamBase is the storage solution that is in charge of storing all the information collected from smart phones. As can be seen in Figure 7.1 it is part of the back-end system but it is logically separated from it. In fact, while the back-end has an instance per user, we can consider the database as a unique entity where all the data is stored, logically separated for privacy reasons. The technology at the core of it is a NoSQL database called Apache

Cassandra[3]. The reason why we decided to adopt it is that the amount of information and the growth rate can be significant even with a limited amount of users for our use-cases. For this reason, standard solutions based on SQL could not satisfy the requirements in terms of latency and scalability. In the current configuration, Cassandra is distributed on multiple nodes in the cloud, thus allowing us to handle huge bursts in the number of requests. Three main types of information are stored in Cassandra, all as streams of data with an attached timestamp: (1) sensor values x_t, (2) questions $y_x?$ and (3) answers y_t. Let us analyze these types of data in turn.

Sensor values are the biggest in size, mainly due to their frequency. Depending on the configuration of the MCS task, we can have the inertial sensors generating values add up to $100Hz$. All such data is stored in the database as time series. There are two columns common to every other sensor, timestamp and day. The former is used to allow temporal queries, while the latter is used as partition key in Cassandra to balance the data in the different distributed nodes composing the cluster. The remaining columns are different for every sensor. For example, in Table 7.1 we present the structure of the inertial sensors used to identify the movement of the smartphone, the accelerometer.

Table 7.1: Accelerometer sensor data stored in StreamBase.

Day	**Timestamp**	**X**	**Y**	**Z**
20200118	20200118100500	9.18	0.00	0.01
20200120	20200120125603	0.89	6.18	4.04
20200120	20200120120500	2.74	2.01	9.20
20200120	20200120131836	2.94	0.32	15.86

The second type of data stored are the questions that are asked to the user. It is important to keep track of them, with the proper provenance information in order to reconstruct the flow from the system to the user. In this regard, the questions IDs are essential to the SKEL algorithm. An additional field present is the status of the question, which can be either DELIVERED, SENT or DISCARDED. Apart from these we have the ”question” field which contains the text that was shown to the user, and ”timestamp” which is the time when the question was generated in the back-end. An example of question stored in the database can be seen in Table 7.2.

The last type of data in the StreamBase storage system are the answers to the questions

[3]https://cassandra.apache.org/

Table 7.2: Questions stored in StreamBase.

Day	Question	Timestamp	Id	Status
20200118	{”q”: ”What are...”}	20200118100500	aDFQivswqA	delivered
20200119	{”q”: ”What are...”}	20200119200500	ELYs/3HeJY	delivered
20200120	{”q”: ”Were you...”}	20200120125603	pmQXTjxrLA	delivered
20200120	{”q”: ”What are...”}	20200120120500	tLN6iIQpdz	delivered
20200120	{”q”: ”Were you...”}	20200120131836	Yw6q7KXw3c	sent

generated by the askUser and challengeUser components. An example of such data can be seen in Table 7.3. The ”day” field is common to the other types of data. We then have ”answer” and ”payload” that respectively contain the text response and any additional response element that the user provided, i.e., a picture, information on a map, etc. We then have the ”question id” which links the answer to the corresponding question as presented in table 7.2. Finally, we have multiple columns that define time variables: ”question timestamp” is the same as in the question table, which refers to the time at which the question was generated and sent to the device; ”notification timestamp” refers to the time when the front-end received the question (usually within few seconds if the phone is turned on and connected) and finally ”answer timestamp” which is the time when the user answers the question. Two additional fields are present, delta and duration which map to the answering behaviour of the users as explained in the front-end part previously in the paper.

Obviously, most of the data in the storage system that refer to the user is sensitive information. However, in the contest of GDPR and personal data, none of them is considered personal identifier, except for the user email that we store for technical purposes. In fact, the email is used to authenticate all the requests coming from the front-end applications. In the i-Log application the user is required to login with Google Identity[4] anywhere in the world except for China (where Baidu Login[5] is used instead.). At every request, a token is sent and the server uses it to extract the user email, that is then used to understand under which user to store the data in the main database. To comply with the regulations, we split the email identifier from the main storage system through an intermediate table in MySQL. This table has only two columns, personal identifier (email) and a uuid. Whenever a user registers in the system, a random uuid is

[4]https://developers.google.com/identity
[5]https://login.bce.baidu.com/

Table 7.3: Answers stored in StreamBase.

Day	**Answer**	**Payload**	**Question Timestamp**	**Notification Timestamp**	**...**
20200118	{”a”: ”Eating”}	{}	20200118100500	20200118100509	
20200119	{”a”: ”Sport”}	{}	20200119200500	20200119200505	
20200120	{”a”: ”Yes”}	{}	20200120125603	20200120125612	
20200120	{”a”: ”TV”}	{}	20200120120500	20200120120601	
20200120	{”a”: ”No”}	{}	20200120131836	20200120132834	

Answer Timestamp	**Question Id**	**Delta**	**Duration**		
20200118101009	aDFQivswqA	300	65		
20200119201505	ELYs/3HeJY	600	5		
20200120131612	pmQXTjxrLA	1200	98		
20200120120701	tLN6iIQpdz	60	23		
20200120132849	Yw6q7KXw3c	15	304		

generated and a row is added to this table. All the data in the StreamBase system based on Cassandra is then stored according to that uuid. Whenever the need to link the data to the email address expires, the corresponding line in the intermediate table is removed and the data in Cassandra immediately becomes anonymous and cannot be linked back to the user who generated it.

7.6 Summary

In this chapter we introduced a general mobile crowd sensing system platform that, together with the i-Log application, can be used to collect sensor data and user’s annotations. On top of the platform it runs SKEL, a machine learning algorithm introduced in Chapter 6 to deals with the unreliability of non-expert users when providing labels. And The main goal of this work is to build a MCS platform with skeptical learning algorithm running on the system to solve the mislabeling issues in mobile crowd sensing area. In this chapter, we detailed the architecture of our proposed platform, described each components of front-end and back-end, and introduced how the StreamBase part works as a storage.

Chapter 8

Conclusion

8.1 Main Conclusion

Automatically personal context recognition is the main step to achieve context-aware application implementation. We firstly identified three main problems or challenges, i.e., 1) an appropriate context model is needed for the recognition of personal context; 2) normal user involvement brings challenge on controlling the quality of collected data, and high-quality data is fundamental for the subsequent machine learning algorithms; 3) a system or a platform is needed to collect high-quality data by interacting with the users in real time. To solve these problem that we identified, 1) we proposed to use a multi-modality personal context model, and we evaluate our model by analysing the data sets generated in the SmartUnitn Projects; 2) we proposed the Skeptical Learning algorithm to deal with the mislabeling issues caused by normal users; 3) we also proposed an general architecture of a MCS platform which runs the SKEL algorithm on the top of it to improve the quality of data collected. These main contributions were then introduced and described in details in the next chapters.

Therefore, we firstly introduced the personal context model that we chose and applied in our work in Chapter 3. This context model has multiple modalities, i.e., time, location, activity, and social relations. We also introduced the concepts of objective, subjective context. We apply the subjective context when design the annotation list given to the users, which is from human perception and allows users to give annotations of their subjective point of view. It was later proven in the next chapter that subjectivity is necessary for framing behavior from the subject's perspective, and it has a substantial effect on predictability and regularity of behavior in practice.

In Chapter 4 we have studied the predictability of human behavior through the notion of personal context. Our study captures a rich, multi-faceted picture of individual behavior by looking at four orthogonal but interrelated dimensions – namely time, location, activity, and social ties – viewed from the subject's own perspective. An empirical analysis on a large data set of daily behaviors shows the benefit of this choice: the different contextual modalities and their subjective description are shown to provide important cues about the predictability of individual behavior. Motivated by this, we also applied our contextual modalities to study behavioral diversity. The obtained results highlight that individuals are more easily identified from rarer, rather than more frequent, subjective context annotations.

This work also highlights an interesting problem. Our results suggest that subjective annotations are very useful for predicting certain contextual modalities. However, these subjective annotations, obtained by filling questionnaires, have some degree of error due to different reasons. This also proves that the noise-label issue is hardly to avoid.

Then, in Chapter 5 we proposed a methodology to evaluate the quality of annotations collected from the users by checking their consistence in giving labels. We applied the methodology to a data set generated in SmartUnitn One Project, and analysed user's annotations of movement and locations. The result showed that the students involved in this project were fairly consistent, and the results can be improved by using semantic knowledge.

In Chapter 6 we introduced Skeptical Learning as a paradigm for dealing with the unreliability of users when providing labels that describe their personal context. The fundamental idea is to use the available knowledge when deciding what is more reliable between the output of the machine learning algorithms and the user input, and to engage in a conflict resolution phase when a controversy arises. Experimental results show the pervasiveness of mislabelling when dealing with feedback from non-expert users, and the effectiveness of Skeptical Learning in addressing the problem as compared to existing approaches to deal with noisy labels.

To solve the last problem we identified, we proposed a general MCS platform in Chapter 7. On the top of the platform, it runs the Skeptical Learning algorithm that we proposed in Chapter 6. The main goal is to build a platform that could collect sensor data from the devices and the label data from the users, and meanwhile, it could detect the correctness of user's label and interact with the user if the label conflict arise. There are three main components of this platform, i.e., the front-end, the back-end and the StreamBase. We introduced the details of each component in this chapter.

8.2 Prospects for Future Work

In this final section of our final chapter we would like to discuss prospects for the future research. There are some directions to extend the scope of this thesis. Firstly, the implementation of the proposed Skeptical Learning platform shall be done to interact with users in real time. Though in the experiments that were carried out in this thesis, we manage to simulate the interaction with the user if the Skeptical Learning algorithm detect the possible incorrect label, but when we ran the SmartUnitn Projects using i-Log to collect users' labels, it could not actually interact with the user immediately. We proposed the Skeptical Learning platform architecture in Chapter 7, but the platform and system needs to be implemented to be used in real world This is an ongoing work that we have been doing.

As we introduced in Chapter 6, the machine has access to the knowledge component. However, currently the knowledge that can be used by the machine is general knowledge of the world and is pre-given. Therefore, the second potential direction is that the machine could use information or knowledge from other sources. For example, allowing for the possibility to ask third parties (e.g., a friend or the crowd) whenever a conflict arises, allowing for multiple machine learning algorithms and for various inputs from the crowd while providing a uniform way to measure their truthfulness, dealing with adversarial learning [17, 64] and adversarial label contamination [105]. These dimensions will enable a much more powerful role of semantics in the future work, leading to building *evolvable* knowledge, contrary to its static nature in this work, that *adapts* and *evolves* in order to accommodate for the ever coming new knowledge, as it is the case in our everyday lives.

The last direction is to apply Skeptical learning in other domains or scenarios. In our work, we have been focus on studying the students' behaviours and recognizing their personal context. Thus in all of our experiments, we applied the Skeptical Learning algorithms on the context recognition task. However, the idea of keeping the machine skeptical about users' input and enabling the machine to challenge the user when conflict happens can be used in many other tasks, other domains and other scenarios. The difference is we need to design new predictors and conflict management algorithms for different tasks.

Bibliography

[1] L. Alessandretti, U. Aslak, and S. Lehmann. The scales of human mobility. *Nature*, 587(7834):402–407, 2020.

[2] L. Alessandretti, P. Sapiezynski, V. Sekara, S. Lehmann, and A. Baronchelli. Evidence for a conserved quantity in human mobility. *Nature Human Behaviour*, 2(7):485–491, 2018.

[3] J. Alstott and D. P. Bullmore. powerlaw: a python package for analysis of heavy-tailed distributions. *PloS one*, 9(1), 2014.

[4] F. Baader, D. Calvanese, D. McGuinness, P. Patel-Schneider, and D. Nardi. *The description logic handbook: Theory, implementation and applications*. Cambridge university press, 2003.

[5] G. H. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data (Neural Information Processing)*. The MIT Press, 2007.

[6] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2):161–180, 2010.

[7] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[8] D. Brockmann, L. Hufnagel, and T. Geisel. The scaling laws of human travel. *Nature*, 439(7075):462–465, 2006.

[9] S. Centellegher, M. De Nadai, M. Caraviello, C. Leonardi, M. Vescovi, Y. Ramadian, N. Oliver, F. Pianesi, A. Pentland, F. Antonelli, et al. The mobile territorial

lab: a multilayered and dynamic view on parents' daily lives. *EPJ Data Science*, 5(1):3, 2016.

[10] Y.-J. Chang, G. Paruthi, H.-Y. Wu, H.-Y. Lin, and M. W. Newman. An investigation of using mobile and situated crowdsourcing to collect annotated travel activity data in real-word settings. *International Journal of Human-Computer Studies*, 102:81–102, 2017.

[11] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1082–1090, 2011.

[12] B. J. Claessens, W. Van Eerde, C. G. Rutte, and R. A. Roe. A review of the time management literature. *Personnel review*, 2007.

[13] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[14] L. Corti. Using diaries in social research. *Social research update*, 2(2), 1993.

[15] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[16] A. Cuttone, S. Lehmann, and M. C. Gonzalez. Understanding predictability and exploration in human mobility. *EPJ Data Science*, 7(2), 2018.

[17] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma. Adversarial classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 99–108, New York, NY, USA, 2004. ACM.

[18] Y.-A. De Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3:1376, 2013.

[19] Y.-A. de Montjoye, L. Radaelli, V. K. Singh, and A. Pentland. Unique in the shopping mall: On the re-identifiability of credit card metadata. *Science*, 347(6221):536–539, 2015.

[20] M. De Nadai, A. Cardoso, A. Lima, B. Lepri, and N. Oliver. Strategies and limitations in app usage and human mobility. *Scientific Reports*, 9(1):1–9, 2019.

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

[22] T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, Aug 2000.

[23] M. Diligenti, M. Gori, and C. Saccà. Semantic-based regularization for learning and inference. *Artificial Intelligence*, 244:143 – 165, 2017. Combining Constraint Solving with Mining and Learning.

[24] T. M. T. Do and D. Gatica-Perez. Contextual conditional models for smartphone-based human mobility prediction. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 163–172, 2012.

[25] N. Eagle, A. Pentland, and D. Lazer. Inferring friendship network structure by using mobile phone data. *Proceedings of the National Academy of Sciences*, 106(36):15274–15278, 2009.

[26] N. Eagle and A. S. Pentland. Reality mining: sensing complex social systems. *Personal and ubiquitous computing*, 10(4):255–268, 2006.

[27] N. Eagle and A. S. Pentland. Eigenbehaviors: Identifying structure in routine. *Behavioral Ecology and Sociobiology*, 63(7):1057–1066, 2009.

[28] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[29] C. Feng, Y. Tian, X. Gong, X. Que, and W. Wang. Mcs-rf: mobile crowdsensing–based air quality estimation with random forest. *International Journal of Distributed Sensor Networks*, 14(10):1550147718804702, 2018.

[30] A. Folleco, T. M. Khoshgoftaar, J. V. Hulse, and L. Bullard. Identifying learners robust to low quality data. In *2008 IEEE International Conference on Information Reuse and Integration*, pages 190–195, July 2008.

[31] B. Frénay, A. Kabán, et al. A comprehensive introduction to label noise. In *ESANN*, 2014.

[32] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. Sweetening ontologies with dolce. In *International Conference on Knowledge Engineering and Knowledge Management*, pages 166–181. Springer, 2002.

[33] A. Ghosh, N. Manwani, and P. S. Sastry. On the robustness of decision tree learning under label noise. In J. Kim, K. Shim, L. Cao, J.-G. Lee, X. Lin, and Y.-S. Moon, editors, *Advances in Knowledge Discovery and Data Mining*, pages 685–697, Cham, 2017. Springer International Publishing.

[34] F. Giunchiglia. Contextual reasoning. *Epistemologia, special issue on I Linguaggi e le Macchine*, 16:345–364, 1993.

[35] F. Giunchiglia, K. Batsuren, and G. Bella. Understanding and exploiting language diversity. In *IJCAI*, pages 4009–4017, 2017.

[36] F. Giunchiglia, E. Bignotti, and M. Zeni. Human-like context sensing for robot surveillance. *International Journal of Semantic Computing*, 12(01):129–148, 2017.

[37] F. Giunchiglia, E. Bignotti, and M. Zeni. Personal context modelling and annotation. In *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 117–122. IEEE, 2017.

[38] F. Giunchiglia and M. Fumagalli. Concepts as (recognition) abilities. In *FOIS*, pages 153–166, 2016.

[39] F. Giunchiglia, M. Zeni, and E. Bignotti. Personal context recognition via reliable human-machine collaboration. In *Pervasive Computing and Communications Workshops (PerCom Workshops), 2018 IEEE International Conference on*, page in print. IEEE, 2018.

[40] F. Giunchiglia, M. Zeni, E. Bignotti, and W. Zhang. Assessing annotation consistency in the wild. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 561–566. IEEE, 2018.

[41] F. Giunchiglia, M. Zeni, E. Gobbi, E. Bignotti, and I. Bison. Mobile social media and academic performance. In *International Conference on Social Informatics*, pages 3–13. Springer, 2017.

[42] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453(7196):779, 2008.

[43] B. Guo, Z. Yu, X. Zhou, and D. Zhang. From participatory sensing to mobile crowd sensing. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2014 IEEE International Conference on*, pages 593–598. IEEE, 2014.

[44] J. Hamm, B. Stone, M. Belkin, and S. Dennis. Automatic annotation of daily activity from smartphone-based multisensory streams. In *International Conference on Mobile Computing, Applications, and Services*, pages 328–342. Springer, 2012.

[45] G. M. Harari, S. R. Müller, M. S. Aung, and P. J. Rentfrow. Smartphone sensing methods for studying behavior in everyday life. *Current Opinion in Behavioral Sciences*, 18:83–90, 2017.

[46] Y. Hattori, S. Inoue, and G. Hirakawa. A large scale gathering system for activity data with mobile sensors. In *2011 15th annual international symposium on wearable computers*, pages 97–100. IEEE, 2011.

[47] M. Hellgren. Extracting more knowledge from time diaries? *Social Indicators Research*, 119(3):1517–1534, 2014.

[48] T. K. Ho. The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, 20(8):832–844, 1998.

[49] K. L. Huang, S. S. Kanhere, and W. Hu. Are you contributing trustworthy data? the case for a reputation system in participatory sensing. In *Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems*, pages 14–22, 2010.

[50] S. S. Kanhere. Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces. In *International Conference on Distributed Computing and Internet Technology*, pages 19–26. Springer, 2013.

[51] N. Kern, B. Schiele, and A. Schmidt. Recognizing context for annotating a live life recording. *Personal and Ubiquitous Computing*, 11(4):251–263, 2007.

[52] W. Z. Khan, Y. Xiang, M. Y. Aalsalem, and Q. Arshad. Mobile phone sensing systems: A survey. *IEEE Communications Surveys & Tutorials*, 15(1):402–427, 2012.

[53] M. M. Kokar, C. J. Matheus, and K. Baclawski. Ontology-based situation awareness. *Information fusion*, 10(1):83–98, 2009.

[54] I. Kola, C. M. Jonker, and M. B. van Riemsdijk. Who's that?-social situation awareness for behaviour support agents. In *International Workshop on Engineering Multi-Agent Systems*, pages 127–151. Springer, 2019.

[55] I. Kontoyiannis, P. H. Algoet, Y. M. Suhov, and A. J. Wyner. Nonparametric entropy estimation for stationary processes and random fields, with applications to english text. *IEEE Transactions on Information Theory*, 44(3):1319–1327, 1998.

[56] R. Kraft, W. Schlee, M. Stach, M. Reichert, B. Langguth, H. Baumeister, T. Probst, R. Hannemann, and R. Pryss. Combining mobile crowdsensing and ecological momentary assessments in the healthcare domain. *Frontiers in neuroscience*, 14:164, 2020.

[57] J. Krumm. Inference attacks on location tracks. In *International Conference on Pervasive Computing*, pages 127–143. Springer, 2007.

[58] C. Krumme, A. Llorente, M. Cebrian, A. Pentland, and E. Moro. The predictability of consumer visitation patterns. *Scientific Reports*, 3(1645), 2013.

[59] R. Krummenacher and T. Strang. Ontology-based context modeling. In *Proceedings*, 2007.

[60] Y.-S. Lee and S.-B. Cho. Mobile context inference using two-layered bayesian networks for smartphones. *Expert Systems with Applications*, 40(11):4333–4345, 2013.

[61] M. Lin, W.-J. Hsu, and Z. Lee. Predictability of individuals' mobility with high-resolution positioning data. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 381–390, 2012.

[62] E. Maddalena, L.-D. Ibáñez, E. Simperl, R. Gomer, M. Zeni, D. Song, and F. Giunchiglia. Hybrid human machine workflows for mobility management. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 102–109. ACM, 2019.

[63] A. Melis, S. Mirri, C. Prandi, M. Prandini, P. Salomoni, and F. Callegati. Crowdsensing for smart mobility through a service-oriented architecture. In *2016 IEEE International Smart Cities Conference (ISC2)*, pages 1–2. IEEE, 2016.

[64] D. Miller, X. Hu, Z. Qiu, and G. Kesidis. Adversarial learning: A critical review and active learning study. *CoRR*, abs/1705.09823, 2017.

[65] G. Miritello, R. Lara, M. Cebrian, and E. Moro. Limited communication capacity unveils strategies for human interaction. *Scientific Reports*, 3(1):1–7, 2013.

[66] P. Mohan, V. N. Padmanabhan, and R. Ramjee. Nericell: rich monitoring of road and traffic conditions using mobile smartphones. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 323–336, 2008.

[67] D. F. Nettleton, A. Orriols-Puig, and A. Fornells. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review*, 33(4):275–306, Apr 2010.

[68] S. A. Nonis, M. J. Philhours, and G. I. Hudson. Where does the time go? a diary approach to business and marketing students' time use. *Journal of Marketing Education*, 28(2):121–134, 2006.

[69] N. Osman, C. Sierra, R. Chenu-Abente, Q. Shen, and F. Giunchiglia. Open social systems. In *17th European Conference on Multi-Agent Systems (EUMAS)*, Thessaloniki, Greece, 2020.

[70] L. Pappalardo, F. Simini, S. Rinzivillo, D. Pedreschi, F. Giannotti, and A.-L. Barabasi. Returners and explorers dichotomy in human mobility. *Nature Communications*, 6(1):1–8, 2015.

[71] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos,

D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[72] D. Peng, F. Wu, and G. Chen. Pay as how well you do: A quality based incentive mechanism for crowdsensing. In *Proceedings of the 16th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 177–186, 2015.

[73] R. Piyare and S. R. Lee. Smart home-control and monitoring system using smart phone. *ICCA, ASTL*, 24:83–86, 2013.

[74] R. Polikar. Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, 6(3):21–45, 2006.

[75] R. Pryss, M. Reichert, B. Langguth, and W. Schlee. Mobile crowd sensing services for tinnitus assessment, therapy, and research. In *2015 IEEE International Conference on Mobile Services*, pages 352–359. IEEE, 2015.

[76] S.-M. Qin, H. Verkasalo, M. Mohtaschemi, T. Hartonen, and M. Alava. Patterns, entropy, and predictability of human mobility and life. *PloS one*, 7(12), 2012.

[77] G. Rätsch, B. Schölkopf, A. J. Smola, S. Mika, T. Onoda, and K.-R. Müller. Robust ensemble learning for data mining. In T. Terano, H. Liu, and A. L. P. Chen, editors, *Knowledge Discovery and Data Mining. Current Issues and New Applications*, pages 341–344, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[78] F. Restuccia, N. Ghosh, S. Bhattacharjee, S. K. Das, and T. Melodia. Quality of information in mobile crowdsensing: Survey and research challenges. *ACM Transactions on Sensor Networks (TOSN)*, 13(4):34, 2017.

[79] D. Riboni and C. Bettini. Cosar: hybrid reasoning for context-aware activity recognition. *Personal and Ubiquitous Computing*, 15(3):271–289, 2011.

[80] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1):107–136, Feb 2006.

[81] L. Rossi, J. Walker, and M. Musolesi. Spatio-temporal techniques for user identification by means of gps mobility data. *EPJ Data Science*, 4(11), 2015.

[82] S. Saeedi, A. Moussa, and N. El-Sheimy. Context-aware personal navigation using embedded sensor fusion in smartphones. *Sensors*, 14(4):5742–5767, 2014.

[83] J. Saramäki, E. A. Leicht, E. Lopéz, S. G. B. Roberts, F. Reed-Tsochas, and R. I. M. Dunbar. Persistence of social signatures in human communication. *Proceedings of the National Academy of Sciences*, 11(3):942–947, 2014.

[84] V. Sekara, L. Alessandretti, E. Mones, and H. Jonsson. Temporal and cultural limits of privacy in smartphone app usage. *Scientific Reports*, 11(3861), 2021.

[85] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[86] B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[87] M. Shahrokh Esfahani and E. R. Dougherty. Effect of separate sampling on classification accuracy. *Bioinformatics*, 30(2):242–250, 2014.

[88] S. Shalev-Shwartz. Online learning and online convex optimization. *Found. Trends Mach. Learn.*, 4(2):107–194, Feb. 2012.

[89] C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE mobile computing and communications review*, 5(1):3–55, 2001.

[90] K. J. Shelley. Developing the american time use survey activity classification system. *Monthly Lab. Rev.*, 128:3, 2005.

[91] Q. Shen, S. Teso, W. Zhang, H. Xu, and F. Giunchiglia. Multi-modal subjective context modelling and recognition. *arXiv preprint arXiv:2011.09671*, 2020.

[92] R. Sinatra and M. Szell. Entropy and the predictability of online life. *Entropy*, 16(1):543–556, 2014.

[93] G. Smith, R. Wieser, J. Goulding, and D. Barrack. A refined limit on the predictability of human mobility. In *Proceedings of the 2014 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 88–94, 2014.

[94] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[95] P. A. Sorokin and C. Q. Berger. *Time-budgets of human behavior*, volume 2. Harvard University, 1939.

[96] S. Teso, R. Sebastiani, and A. Passerini. Structured learning modulo theories. *Artificial Intelligence*, 244:166 – 187, 2017. Combining Constraint Solving with Mining and Learning.

[97] J. L. Toole, C. Herrera-Yaqüe, C. M. Schneider, and M. C. González. Coupling human mobility and social ties. *Journal of The Royal Society Interface*, 12(105):20141128, 2015.

[98] R. Tourangeau, L. J. Rips, and K. Rasinski. *The psychology of survey response*. Cambridge University Press, 2000.

[99] M. H. Tran, J. Han, and A. Colman. Social context: Supporting interaction awareness in ubiquitous environments. In *In Proceedings of the 6th International Conference Mobile and Ubiquitous Systems: Networking & Services, MobiQuitous 2009*, pages 1–10. IEEE, 2009.

[100] Y. Vaizman, K. Ellis, and G. Lanckriet. Recognizing detailed human context in the wild from smartphones and smartwatches. *IEEE Pervasive Computing*, 16(4):62–74, 2017.

[101] R. Wang, F. Chen, Z. Chen, T. Li, G. Harari, S. Tignor, X. Zhou, D. Ben-Zeev, and A. T. Campbell. Studentlife: assessing mental health, academic performance and behavioral trends of college students using smartphones. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 3–14. ACM, 2014.

[102] R. Wang, G. Harari, P. Hao, X. Zhou, and A. T. Campbell. Smartgpa: how smartphones can assess and predict academic performance of college students. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 295–306. ACM, 2015.

[103] X. H. Wang, D. Q. Zhang, T. Gu, and H. K. Pung. Ontology based context modeling and reasoning using owl. In *IEEE annual conference on pervasive computing and communications workshops, 2004. Proceedings of the second*, pages 18–22. Ieee, 2004.

[104] B. T. West and J. Sinibaldi. The quality of paradata: A literature review. *Improving Surveys with Paradata*, pages 339–359, 2013.

[105] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, and F. Roli. Support vector machines under adversarial label contamination. *Neurocomputing*, 160:53 – 62, 2015.

[106] N. Xu, W. S. Zhang, H. D. Yang, X. G. Zhang, and X. Xing. Cacont: A ontology-based model for context modeling and reasoning. In *Applied Mechanics and Materials*, volume 347, pages 2304–2310. Trans Tech Publ, 2013.

[107] H. Yang, J. Zhang, and P. Roe. Using reputation management in participatory sensing for data classification. *Procedia Computer Science*, 5:190–197, 2011.

[108] M. Zappatore, A. Longo, and M. A. Bochicchio. Using mobile crowd sensing for noise monitoring in smart cities. In *2016 international multidisciplinary conference on computer and energy science (Splitech)*, pages 1–6. IEEE, 2016.

[109] L. Zavala, P. K. Murukannaiah, N. Poosamani, T. Finin, A. Joshi, I. Rhee, and M. P. Singh. Platys: From position to place-oriented mobile computing. *Ai Magazine*, 36(2):50–62, 2015.

[110] M. Zeni, I. Zaihrayeu, and F. Giunchiglia. Multi-device activity logging. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, pages 299–302. ACM, 2014.

[111] M. Zeni, W. Zhang, E. Bignotti, A. Passerini, and F. Giunchiglia. Fixing mislabeling by human annotators leveraging conflict resolution and prior knowledge. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(1):1–23, 2019.

[112] W. Zhang, A. Passerini, and F. Giunchiglia. Dealing with mislabeling via interactive machine learning. *Künstliche Intelligenz*, pages 1–8, 2020.

[113] W. Zhang, Q. Shen, S. Teso, B. Lepri, A. Passerini, I. Bison, and F. Giunchiglia. Putting human behavior predictability in context. *EPJ Data Science*, 10(1):42, 2021.

[114] X. Zhao, N. Wang, R. Han, B. Xie, Y. Yu, M. Li, and J. Ou. Urban infrastructure safety system based on mobile crowdsensing. *International journal of disaster risk reduction*, 27:427–438, 2018.