Alma Mater Studiorum – Università di Bologna

DOTTORATO DI RICERCA IN

Computer Science and Engineering

Ciclo XXXI

Settore Concorsuale: 01/B1

Settore Scientifico Disciplinare: INF/01

DELIVERING IOT SERVICES IN SMART CITIES AND ENVIRONMENTAL MONITORING THROUGH COLLECTIVE AWARENESS, MOBILE CROWDSENSING AND OPEN DATA

Presentata da: Federico Montori

Coordinatore Dottorato Prof. Paolo Ciaccia

Supervisore Prof. Luciano Bononi

Esame finale anno 2019

Abstract

The Internet of Things (IoT) is the paradigm that allows us to interact with the real world by means of networking-enabled devices and convert physical phenomena into valuable digital knowledge. The number of connected objects nowadays consistently overtook the number of people in the world and novel IoT applications permeate several areas of our lives, among which home automation, industry 4.0, healthcare, Smart Cities and environmental monitoring. Such a rapidly evolving field leveraged the explosion of a number of technologies, standards and platforms. Consequently, different IoT ecosystems behave as closed islands and do not interoperate with each other, thus the potential of the number of connected objects in the world is far from being totally unleashed. Typically, research efforts in tackling such challenge tend to propose a new IoT interoperability platform or standard, however, such solutions find obstacles in keeping up the pace at which the field is evolving and interested parts hardly adapt.

Our work is different, in that it originates from the following observation: in use cases that depend on common phenomena such as Smart Cities or environmental monitoring either a lot of useful data for applications is already in place somewhere or devices capable of collecting such data are already deployed. Specifically, for such scenarios, we propose and study the use of Collective Awareness Paradigms (CAP), which offload data collection to a crowd of participants. We bring three main contributions: (1) we study the feasibility of using Open Data coming from heterogeneous sources, focusing particularly on crowdsourced and user-contributed data that has the drawback of being incomplete, partially annotated and imprecise and we then propose a State-of-the-Art algorithm and framework that automatically classifies and annotates raw crowdsourced sensor data; (2) we design a data collection framework that uses Mobile Crowdsensing (MCS) and puts the participants and the stakeholders in a coordinated interaction in order to regulate the data collection process according to the common needs, furthermore, we design a distributed data collection algorithm that prevents the users from collecting too much or too less data, which would hinder the extraction of knowledge that reflects the reality; (3) we design a Service Oriented Architecture (SOA) that constitutes a unique interface to the raw data collected through CAPs through their aggregation into ad-hoc services that can be created, instantiated and destroyed by the end users through a customized language that we designed, moreover, we provide a prototype implementation.

Our work is a novel effort in such direction and it is a significant step forward in tackling the challenge of interoperability for IoT applications in the contexts of Smart Cities and environmental monitoring for the common welfare.

Glossary of Acronyms

Here are defined (in alphabetical order) the acronyms used throughout the dissertation in order to facilitate the reader.

1NN	One-Nearest-Neighbor
3GPP	3rd Generation Partnership Project
ANOVA	ANalysis Of VAriance
AO-F	Asymptotic Opportunistic algorithm for Fairness
AO-JFS	Asymptotic Opportunistic algorithm for Joined Fairness and Satisfaction Index
AO-S	Asymptotic Opportunistic algorithm for Satisfaction Index
AOB	Asymptotically Optimal Backoff
AP	Access Point
API	Application Program Interface
BLE	Bluetooth Low Energy
BNF	Backus-Naur Form
BOPF	Bag-of-Pattern Features
BOS	Bag-of-Summaries
BOSS	Bag-of-SFA-Symbols
BOW	Bag-of-Words
BT	Bluetooth
BSSID	Basic Service Set IDentifier
CANOVA	Classwise ANalysis Of VAriance
CAP	Collective Awareness Paradigm
CBM	Community-Based Monitoring
CBOS	Classwise Bag-of-Summaries
CCU	Central Coordination Unit
CDF	Cumulative Distribution Function
CDMA	Code Division Multiple Access
CIoT	Cellular IoT
CoAP	Constrained Application Protocol
CS	Citizen Science
CSI	Custom Service Instance
CSMA/CA	Carrier Sense Multiple Access for Collision Avoidance
CST	Custom Service Template
DDL-NLP	Dictionary Damerau-Levenshtein with Natural Language Processing
DI	Deviation Index
DLL	ISO/OSI Data Link layer
DSA	Datastream Annotator

DSM	Datastream Manager
DTW	Dynamic Time Warping
ED	Euclidean Distance
FDMA	Frequency Division Multiple Access
GSM	Global System for Mobile Communications
GZI	Grid Zone Identifier
HAS	Home Automation Systems
HTTP	Hyper Text Transfer Protocol
IoT	Internet of Things
IETF	Internet Engineering Task Force
JFI	Jain's Fairness Index
LAN	Local Area Network
LPWAN	Low Power Wide Area Network
LTE	Long-Term Evolution
MAC	Medium Access Control sublayer
MCS	Mobile Crowdsensing
MGRS	Military Grid Reference System
ML	Machine Learning
MoCroSS	Mobile Crowdsensing module for SenSquare
MTC	Machine Type Communication
NFC	Near-Field Communication
OCS	Opportunistic Crowdsensing
OFDMA	Orthogonal Frequency Division Multiple Access
PAA	Piecewise Aggregate Approximation
PCS	Participatory Crowdsensing
PDF	Probability Density Function
PHY	ISO/OSI Physical Layer
PoC	Proof of Concept
REST	REpresentational State Transfer
RFID	Radio Frequency IDentifier
RMS	Root Mean Squared error
RSSI	Received Signal Strength Indication
SAX	Symbolic Aggregation approXimation
SI	Satisfaction Index
SINR	Signal and Interference to Noise Ratio
SOA	Service-Oriented Architecture
SOSA	Sensor, Observation, Sample, and Actuator
SSID	Service Set IDentifier
SSN	Semantic Sensor Network
TDOA	Time Difference Of Arrival
TDMA	Time Division Multiple Access

TKSE	Top-k Sequential Ensemble
TSC	Time Series Classification
UMTS	Universal Mobile Telecommunications System
WBAN	Wireless Body Area Network
WLAN	Wireless Local Area Network
WoT	Web of Things
WPAN	Wireless Personal Area Network
WWAN	Wireless Wide Area Network
WSN	Wireless Sensor Network

Contents

List of Figures

List of Tables

Part I

Background

Chapter 1

Introduction

The Internet of Things (IoT) is everywhere and it is nowadays permeating nearly each aspect of our life, just as it was predicted few years ago [45]. In fact, it is fostering novel applications in more and more areas, among which healthcare, Smart Cities, environmental monitoring and smart houses, all of them with their own different requirements. This makes the IoT a macro field in research that cannot be studied as a whole. The number of connected devices is of the order of magnitude of tens of billions, the number of personal mobile devices that can connect to the Internet is soon predicted to overcome the human world population and the amount of data generated about the environment that surrounds us is growing exponentially. The goal of the IoT is to transform such machine-interpretable data in actual human-understandable knowledge for the common benefit, mapping uniquely the real world into the digital world by means of sensors and actuators. The IoT, given its high potential and the number of applications is one of the most studied fields in research, indeed, it has still a number of research challenges that capture the focus of a plethora of researchers worldwide. One of the most notable challenges is the interoperability. The proliferation of a variety of standards and technologies has led to "IoT islands", closed ecosystems that do not interoperate with each other; hence new applications often choose to rely on brand new (expensive) deployments and designs rather than reuse what is available, also due to the lack of Open Source in the community. This is a well-known issue among researchers in the field of IoT, in fact, many interoperability-based frameworks, standards and solutions have been proposed throughout the years. However, many of such solutions fail in providing a unique approach, either due to the high number of such proposals, or due to the fact that, according to them, the customer is supposed to stick to a number of standards. Unfortunately, the pace at which the field is evolving imposes rapid choices to the industry, which typically relies on ad-hoc proprietary solutions. In contrast to this, in this dissertation we brought a number of contributions that are founded on two main observations: (i) Especially

within the scope of Smart Cities and environmental monitoring there are a lot of Open Data repositories that provide data produced by IoT devices and, in general, data about phenomena of common interest are already available somewhere and (ii) Devices capable of providing data are, in many cases, already in place. These observations are crucial in a world in which, typically, the information available to individuals that extract it from their own ecosystem is limited. In fact, if a piece of information is missing, it is likely being produced by some other entity. Making both parts aware is a key feature of Collective Awareness Paradigms (CAP), i.e. paradigms such as MCS or crowdsourcing, that leverage collaboration among parties by offloading data collection tasks to a crowd of participants or making use of what is already available. How then can we take advantage of it and transform it into knowledge? This leads us to the definition of our main macro contributions (1 and 2 are methodological innovations and 3 is a system implementation):

1. Regarding observation (i), Open Data is available in different formats and many non-official sources do not have a semantic connotation, thus they need a way (e.g. machine learning) to be interpreted. Considering that a lot of such data is incomplete (often even missing a data class due to a poor annotation) we need an automatic way to classify such data into a schema that is well-known and reausable for the purpose of the applications. Therefore, our first methodological macro contribution is **an algorithm to classify unannotated datastreams**, a topic that we cover throughout Chapter 4.

2. For observation (ii), we use the paradigm of Mobile Crowdsensing (MCS) – i.e. monitoring an environment through the automated participation of users through their own mobile devices and participating in a data collection campaign – in order to provide useful data for scenarios like Smart Cities and environmental monitoring. This opens up a plethora of new possibilities, but there are challenges to deal with (data quality, coverage, energy efficiency, budget constraints etc.). In our case, we focused on automatically balancing the amount of data collected in opportunistic MCS for a better energy efficiency and less resource waste. Therefore, our second methodological macro innovation is **a distributed algorithm for balancing the amount of data produced in opportunistic MCS scenarios**, which is presented, together with an experimental MCS platform, in Chapter 5.

3. Finally, in order to address the problem of lack of interoperability and reausability as well as gathering our scientific contributions into a big picture, we carried out the implementation of a platform that fills the aforementioned gaps and allows the creation and customization of IoT services based on data produced through CAPs. The third contribution is **SenSquare: an interoperability framework for Smart Cities and environmental monitor-**

ing. This includes other prototypes that are outlined throughout the thesis and we developed as part of our work: the Crowdroid app, the MoCroSS framework, the INFORM platform and the RouteX framework.

Hereby is reported more in detail the structure of the single sections. The whole Chapter 2 will be focused on the research landscape around the relevant areas in order to give a context to our research work. In particular, the preface to the chapter will give an introduction to what IoT is and under which perspective we tackled the challenges; Section 2.1 will focus on Machine-to-Machine (M2M) communication technologies, a pillar in the field of IoT about which we recently published a survey paper [150] and two performance studies [152][153]; in Section 2.2 we will give an extensive definition of the CAPs and their categorization [154], focusing primarily on crowdsourcing and crowdsensing; Section 2.3 will deal with the paradigm of Collaborative IoT and Open Data, as well as recent works dealing with open and user-contributed data and why it matters [146]; Section 2.4 deals with the concept of MCS, its main areas of research and its challenges, while Section 2.5 deals more in detail with a well-known issue that we call the *"Curse of Sensing"*, that is, the inability of MCS applications to deal with data that is too sparse or too dense and, thus, it leads to biased results [154]. Chapter 3 wraps up the State-of-the-Art and, on top of the research challenges, outlines our research question, already stated above and further expanded, to which we respond with our contributions, detailed in Part II.

In Part II, Chapter 4 outlines our contributions in the field of Open Data, focusing on datasets contributed by users. In particular, Section 4.1 is about the general paradigm of crowdsourced IoT data streams, together with our findings and our proposals [146]; Section 4.2 explains in detail our Top-K Sequential Ensemble (TKSE) classification algorithm, that aims to automatically infer the data class of unannotated and uncategorized raw datastreams [155]; Section 4.3 outlines briefly a framework that we designed in order to automatically annotate and give a semantic description to poorly annotated datastreams according to ontologies [78]. Chapter 5 outlines our contributions in the field of MCS, oriented to applications especially for Smart Cities and environmental monitoring. In particular, in Section 5.1 we propose MoCroSS, a framework for opportunistic MCS applications in which stakeholders can issue campaigns and participants can choose which task to contribute in [149]. In Section 5.2 we tackle the challenge given by the *"Curse of Sensing"* that can occur in scenarios like the one in Section 5.1. Specifically, we design a distributed algorithm that tunes the number of observations required by the users, pushing users to contribute more when data is sparse or less when data is dense [147]. In Chapter 6, in order to give a common scenario to our contributions, we prototyped the SenSquare platform [148] , a community-

sized platform that implements data collection both from MCS scenarios (using the framework we designed) and Open Data repositories, automatically classified and unified. With such a platform, through an easy and visual language that we designed, users can create, share and instantiate personal aggregated services that can provide compound and dedicated information.

Of course, this topic is vast and many future extensions are foreseen. In Chapter 7 we propose a set of future works to enhance the State-of-the-Art and foster scenarios in which contribution and collectiveness are a central enabler and we wrap up the dissertation summarizing our contributions.

Chapter 2

State-of-the-Art

The IoT is nowadays a key component of our life and permeates each of its aspects. Our world is surrounded by smart devices and their connections, the goal of which is the common benefit, the improvement of the conditions of individuals and companies. As a consequence, the demand for smart things and environments rose exponentially over the last decade and the amount of money invested in such business is gargantuan. The term IoT was first coined by Kevin Ashton – executive director of the Auto-ID Center – in 1999 [12]. The original definition of the IoT was significantly more restricted compared to the current use; in fact, it used to be about the possibility to map objects in the real world to data through the use of identifiers. By then, the majority of such technologies resided in the Radio Frequency IDentifiers (RFID), small and cheap electronic tags that could communicate a small amount of information about the identified object in the real world. The amount of scientific works in the field of IoT is proportional to the amount of definitions that have been given to it throughout the years [13]. For such reason, and because the topic of IoT is so well-known that a basic knowledge of what it is by the reader is assumed, we do not even report such definitions, rather we outline here our own vision on the IoT. The definition in our point of view can be summarized by Figure 2.1: in brief, the IoT is a paradigm in which real world concepts and objects (the "things") can be mapped in a 1-to-1 correspondence to data in the digital world. This happens through a perception layer typically formed by sensors, which can perform observations in the real world and deliver them to a hub through communication technologies, most of them specifically designed for the IoT. IoT data is not human-interpretable as it is, thus the IoT circle is complete when algorithms for data processing (data mining, machine learning, statistical methods, etc.) transform IoT data into valuable knowledge for the consumers. In addition to this basic view, we can add other components, such as the actuators (the dual of sensors) which are devoted to act on the real world on top of certain data and commands from the digital world. The context, a set of

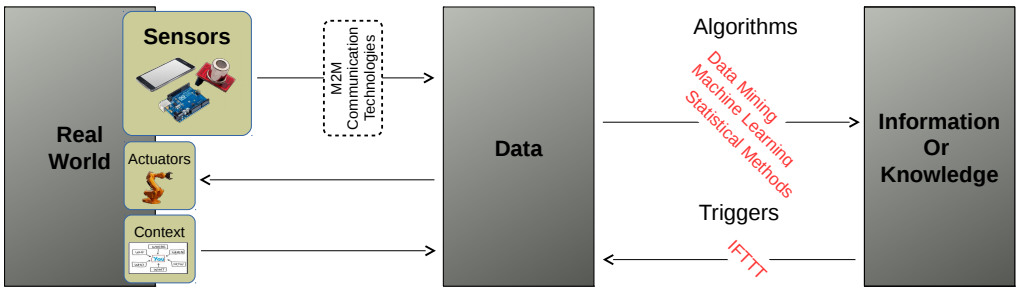


Figure 2.1: Our vision on the IoT: the real world mapped to data, mapped in turn to knowledge.

meta-information around the raw data itself, is often provided together with the actual observation. Finally, there are some mechanisms that can trigger certain behaviors on top of the acquired knowledge: the figure reports the example of If This Then That (IFTTT), a platform in which various IoT services can be linked together to set off a reaction by the system when a certain combination of events occur. The IoT world is certainly vast, however, we believe that our concise definition is exhaustive enough to convey what we describe in this dissertation.

The IoT became a major trend in research and industry between 2008 and 2009, when the number of connected device surpassed the number of humans on the planet [63]. Nowadays, pretty much every IoT-related paper indicates how impressively the IoT is growing. To give an idea, three major forecasting reports are cited and periodically updated: Ericsson Mobility Report [40], Machina Research IoT Forecast [181] and Cisco Visual Networking Index (VNI) [45]. The forecasts are skyrocketing: the global IoT market is expected to generate soon a revenue of 4.3 trillions of dollars, by 2021 we expect to witness a total of 27.1 billions of connected devices, which means an average of 3.5 devices per capita (12.9 if we only consider North America). In 2018 we experienced 7.9 billions of mobile broadband subscriptions (which surpassed the world population) with an impressive growth happened recently in India, China, Indonesia, Bangladesh and Pakistan. We are also experiencing the usage of the IoT in more and more fields of application (which would be detailed in Section 2.1) such as home automation, Industry 4.0, environmental monitoring, Smart Cities, healthcare and many others.

The numbers of IoT undeniably dragged the interest of an uncountable amount of researchers and companies. For such reason, before outlining our contributions anticipated in the introduction, we will give an extensive literature review on the IoT, with a focus on the fields of interest for our research work. In particular, we will start with the communication technologies for the IoT and how they are im-

plied in several use cases around the world (Section 2.1), then we will introduce the Collective Awareness Paradigms (CAP), the main pillar on which our research work is founded (Section 2.2), then we will discuss the role of Open Data in current IoT applications and ecosystems (Section 2.3) and, finally, we will explore the current State-Of-The-Art for Mobile Crowdsensing (MCS) (Section 2.4) focusing particularly on the issue known as the *"Curse of Sensing"* (Section 2.5). All these literature reviews are supported by the related works of each of our publications plus two surveys, each of them revised in order to be up to date.

2.1 Machine-to-Machine (M2M) communications for the IoT

At the same time, heterogeneous IoT deployments might prioritize different qualitative or quantitative metrics that are required by the applications on top.

The Machine-to-Machine (M2M) communication technologies play the crucial role to enable wireless data exchange among the IoT devices and the gateway, and then from the gateway to a remote repository via the Internet. The aim of this section is to review the State-of-the-Art of the M2M wireless technologies for the IoT by classifying the existing solutions according to a multi-layer taxonomy that allows clarifying the technical features of each approach. Open issues and future research directions are discussed as well. This work is a reduced version of our recent survey paper [150], to which we redirect the interested reader for a deeper analysis of the topic. Despite the overwhelming number of survey papers on IoT, our work can be considered a missing piece of the puzzle , since:

- it focuses on the existing wireless technologies and on the PHY/MAC layers, hence it differs from generic surveys like [10], [226], [65], [7], [198], [204], [178], [141], [125] and [8], which describe the IoT protocols at each layer of the network stack, thus giving a *broad* vision of the IoT;
- at the same time, it is not restricted to any specific stack or infrastructure like [208], [193], [203], [115] and [172], rather it provides an *in-depth* review of the existing solutions, considering both open standards and proprietary solutions, short-range, long-range and cellular-based solutions.

Three main contributions are provided. First, we introduce a novel multi-layer taxonomy, which allows classifying the existing M2M wireless technologies. In particular, we aim to analyze requirements that assure efficiency and suitability involving M2M communication technologies (Section 2.1.1) as well as the axes upon which we intend to pursue our categorization (Section 2.1.2). We then outline the common use cases for IoT scenarios with a particular focus on the weight, for each use case, attributed to the different requirements (Section 2.1.3). Based on the classification criteria defined above, we biefly review the existing technologies, distinguishing between short-range (Section 2.1.4) and long-range (Section 2.1.5) solutions. Finally, we discuss the mapping between the enabling M2M communication technologies and the IoT use cases (Section 2.1.6).

2.1.1 M2M Requirements

Here we report a list of features for M2M technologies universally considered to be strong requirements, to which all the technologies presented in this paper

adhere in different measures.

Low power consumption Low power consumption is clearly one of the key features that devices must satisfy, since, in several cases, networked sensors and actuators need to be powered by means of batteries, due to their extremely distributed physical topology, as the availability of power sources is usually limited or absent and, especially in wide area deployments, the replacement of batteries is time consuming and implies substantial costs in the long run. Network activity is the main source of energy depletion, since connectivity has been shown to be more energy-consuming than computation by two to three orders of magnitude [189]. Hence, whenever a scenario hosts a number of devices with limited or no access to constant power sources, energy-saving optimizations take place both at the PHY and the MAC layer. Solutions like duty cycling, a technique that allows the device to turn on and off its radio interface, and energy harvesting can be adopted in order to maximize the battery duration [158][39]. Such algorithms always imply a "deep sleep" time window, in which the radio interface is turned off and the power consumption is close to null. The frequency of the wakeup periods depends on the use case, however, the technology is responsible for part of the preprocessing duration. There are several other methods that can be adopted in order to increase the energy efficiency of M2M communication. According to [176], they can be divided in five main categories, i.e.: radio optimization, data reduction, sleep schemes, energy-oriented routing and battery repletion. We redirect the readers to [176] for further details on the topic.

Low Cost Due to the high number of devices in an IoT ecosystem, end devices necessarily need to satisfy a low cost per unit, minimizing the amount of hardware and, as a consequence, making the device extremely specialized on its task. Furthermore, low cost and low power solutions are highly linked; in fact, manual battery replacement is a costly process, especially when repeated for a huge number of units. The cost factor highly impacts the choices made at the MAC layer, especially in the channel access techniques. For instance, in contention-free environments, TDMA is the most viable option, since CDMA-based approaches are not suitable for low power and low cost deployments, primarily due to their complexity. Furthermore, pure FDMA approaches are not used in M2M application due to the high cost of the high-performing frequency filters in the radio hardware of each unit. An exception is given by OFDMA-based systems, due to their easy and low cost implementation of the FFT in chips as well as the lack of necessity for filters for each sub-channel [120].

Scalability With the advent of massive IoT deployment for new use cases, scalability is a necessary feature. Typically, a high number of nodes brings issues regarding collisions, load balancing, deployment cost and data fusion; for such reasons, a high scalability always implies reconfiguration to be efficient as well as support for a high number of devices per gateway. Scalability also impacts the channel access method, since in dynamic scenarios – i.e. with a non-static number of participants and with dynamically entering and leaving nodes – contention-based methods face an increase of collisions, whereas contention-free ones need to deal with a time-consuming reconfiguration [242].

Reliability Reliability is a strict requirement for many use cases. There are several ways of estimating reliability in networks, which, in general, include the probability that a certain node in the network will get the message upon the failure of a certain set of links [177][197]. Now, as lack of reliability depends primarily on link failures and lack of controlling mechanisms that would put a burden onto the data packets, network topology (see also Section 2.1.2) and management have a central role in addressing it. The failure of a communication link is a damage to the system reliability that can be alleviated by the usage of mesh redundant topologies. Networks organized in plain stars, a common topology used in long range deployments, support reduced reliability, in fact a single link failure results in a single node exclusion. In some use cases this is tolerable, however, in many situations, node or gateway redundancy has to be supported, which results in a cost growth. Lastly, tree networks are, reliability-wise, the worst topologies as any link failure results in the exclusion of the whole subtree.

Low Latency Low latency is often a highly desirable feature and it is unavoidably bound to other aspects that can influence it. There are physical deployment dependencies such as the link strength between the endpoints and the number of hops in an average communication path as well as the number of nodes in the network. PHY layer mechanisms such as spread spectrum techniques, modulation and coding schemes, frequency and spatial diversity also greatly affect latency [220]. The choice of the MAC layer channel access method (i.e. contention-free vs. contention-based) in relation with the network topology is also crucial, as it can introduce unexpected delays [172]. In general, contention-based protocols used in MTC communications suffer from idle listening and dramatically high delays for large networks. This is the case of CSMA/CA, which is widely used

in some technologies due to its possibility to scale efficiently with no need for reconfiguration in small networks. Contention-free protocols are more suitable for large networks, since they offer algorithms capable of exploiting well the available resources without waste, although they do not scale efficiently due to the need for global reconfiguration anytime a node joins or leaves the network. This is the case of TDMA networks, which are largely used in different adaptations in IoT.

Enhanced Communication Range A wider range of communication means a wider area deployment, which is the current trend in future generation IoT deployments targeting the market of monitoring and public welfare. For many use cases, such feature is a must-have, being aware that the nominal range is often not enough in order to calculate how wide a deployment can be. Indoor scenarios, obstacles and the spatial coexistence with other technologies often put the range in correlation with the spectrum frequency bands and modulation encoding schemes. The 2.4 GHz frequency bands, besides being designed for relatively consistent data transfer, has a list of non-negligible drawbacks for IoT long range scenarios. Due to its nature, it supports more easily a high data rate, however it suffers more from obstacles, indoor deployments and it requires more power in order to be pushed to long distances. Furthermore, the recent overcrowding of such frequency bands does not help in scenarios with high network population. For such reasons, technologies deployed in sub-GHz bands are gaining more and more interest in IoT [212]. Almost all the long-range technologies exploit either unlicensed bands like the 868 MHz, or the licensed bands around 800 MHz, in coexistence with other cellular technologies such as LTE, UMTS and GSM. Furthermore, enhanced range is typically chosen in contrast with the power consumption at the price of a reduced data rate. Many future generation applications require very low consumption and not much data rate, for which arising narrowband long-range solutions designed for wide area deployments appear to be convenient [58].

Security Security is also a challenging issue due to the nature of M2M deployments, which makes them vulnerable to attacks such as denial of service (DoS) and might compromise confidentiality, authentication, integrity, authorization, and availability. In fact, many aspects of M2M solutions unfortunately open up new vectors for DoS. An example on how dangerous a lack of security can be in a crowd on small devices is given by the Mirai botnet, which in September 2016 used more than 400,000 devices to perform DDoS attacks generating more than enough traffic to knock several services offline [107] Although it is important to mention security, it is being discussed in the present section mostly as an open

issue. Furthermore, many other works address specifically the problem [69][80].

2.1.2 Technology Classification

In M2M we consider of paramount importance the differences brought by the range and the data rate of each communication technology, as well as the topology adopted in their deployment. Since such characteristics determine the suitability of the technologies for specific purposes and the network size, we chose to classify each technology using these discriminants. As they are orthogonal, we believe that their combination gives an efficient way to categorize each technology.

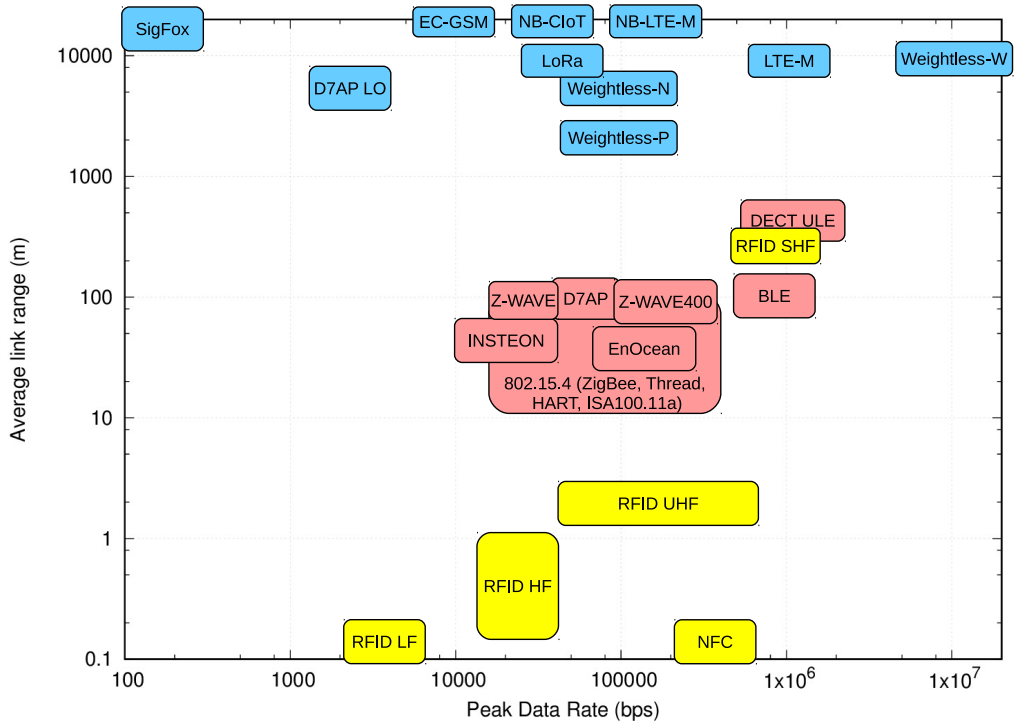


Figure 2.2: Diagram showing at a glance all the technologies included in the present review work cathegorized by range and data rate. Proximity technologies are identified in yellow boxes, capillary technologies in red and LPWAN technologies in blue [155].

Range and Data Rate M2M communication technologies are used in network types that span, depending on their communication range, from Wireless Body Area Network (WBAN) to Wireless Personal Area Network (WPAN), to Wireless

Local Area Network (WLAN) to even Wireless Wide Area Network (WWAN). According to this, we separate IoT communication technologies in Proximity, Short Range and Long Range. Proximity technologies, such as RFID and NFC, have typically a range of very few meters and are used for identification purposes or small data transfers. Although they are the main pillars on which IoT rose, we do not extensively deal with them in this section as we do not consider them as strictly M2M technologies.Short Range technologies, often referred to as "Capillary" and outlined in Section 2.1.4, have a communication range of some meters up to a maximum of a hundred and are typically suitable for WBANs, WPANs and WLANs. For such reason, their deployment is typically restricted to a certain limited area (e.g. a room, a small building, a house). Finally, Long Range technologies, considered the rising star in the future IoT, are suitable for big WLANs and WWANs, covering areas of few kilometers. This means that a single network is able to serve a big building, a factory or even a rural area, depending on the amount of direct LoS links. Figure 2.2 gathers nearly all the technologies that we address, using spatial range as discriminant and putting it in orthogonal relation with data rate.

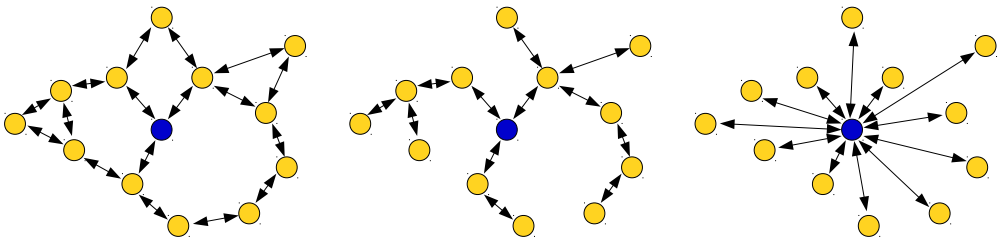


Figure 2.3: Schemes showing the differences among topologies. (a) Mesh topology, (b) Hierarchical tree topology, (c) Star topology [155].

Topology Network topology is also a determining feature in relation with the purpose of a certain deployment. A small recall to the existing network topologies is shown in Figure 2.3. The star topology is the most common network type, in which a central node acts as the sink, while the peripheral nodes are connected to it via a direct link without being connected to each other. In general, the sink is the gateway to the outer world or it is connected directly to such gateway. The mesh topology is the dual of the star network, where nodes are connected to each other in a multi-hop fashion with only few of them connected to the sink. In the hierarchical tree topology connections are designed as in a tree, in which the root is the sink and peripheral nodes are connected in layers via direct links. Choosing one of such deployments determines a different priority given to a number

of aspects and features for which the topology is responsible [196]: reliability, scalability, energy efficiency and latency are among them.

2.1.3 Use Cases

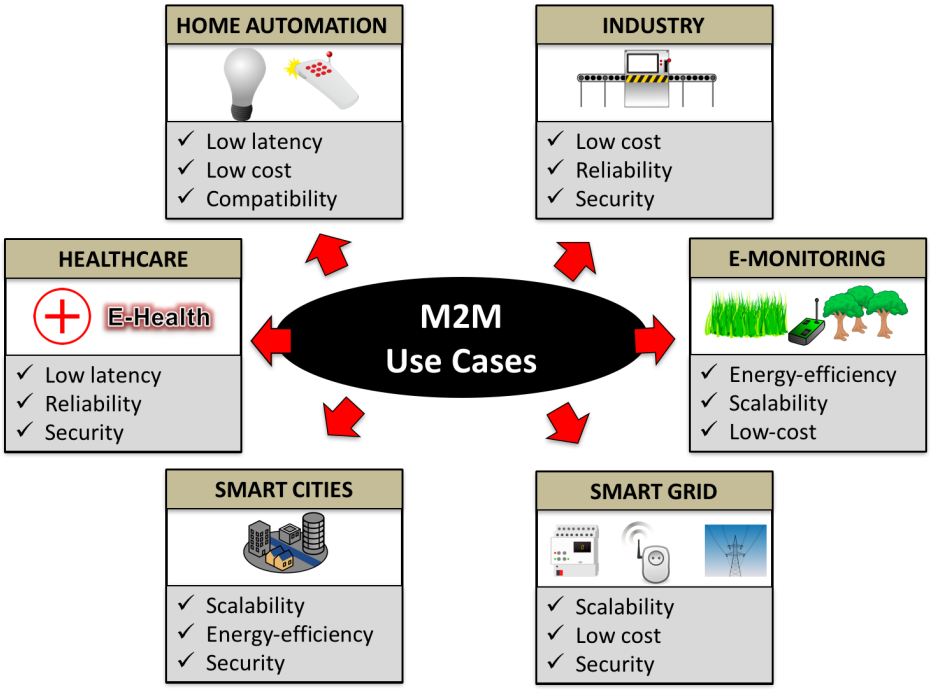


Figure 2.4: The M2M use cases and main requirements [155].

Use cases determine what is required and what is optional when choosing a specific communication technology for a deployment. Such differences can involve the deployment size, the required latency, the required reliability, the amount of data to be shared, the availability of power sources, the monetary resources, the security requirements, the compatibility, the business models and, clearly, the purpose [71]. Below we briefly mention M2M use cases and their dependence on certain technical requirements. They are also summarized briefly in Table 2.1, and illustrated in Figure 2.4.

Home Automation A common citizen, who deals with problems related to home automation and everyday life monitoring purposes, rarely would care about a scalable network or a wide deployment. Conversely, features such as compatibility with preexisting infrastructures and cost would be much more preferred. Low latency is also something appealing in home automation scenarios, since the

interaction between sensors and actuators is commonly required "here and now" [205].

Industry Industrial scenarios, concerning automation and process control, are a completely different reality as they prioritize cost, low latency and reliability over all the other possible metrics [213], giving in some cases secondary importance to scalability and compatibility depending on the factory/installment physical size and location. Required data rate may vary significantly from case to case, while the security is also a central issue, since a malign agent can have devastating consequences [185].

Healthcare Healthcare scenarios highly prioritize the qualitative metrics such as reliability, the low latency and the security [30], while most of the others, such as the cost and the power consumption are (or should be) of secondary importance. The scalability strongly depends on the installment size which may span from very small (a specialized hospital ward) to very wide (remote patient monitoring). Data rate is also highly variable, since it might be high, like in real-time health status and predictive information, or low, like in periodic monitoring.

Environmental Monitoring Environmental monitoring normally implies huge deployment zones and prioritize scalability. The end nodes are only committed to report periodically data and usually the network involves no actuator, thus, with few exceptions, the use case normally tolerates delays as well as data unreliability, simply by adding more sensing instances. For such reasons the end devices must be extremely cost-effective and, due to the deployment size which implies a significant maintenance cost, they must observe a high energy efficiency [116].

Smart Cities Smart city scenarios are rather complex deployments, in which all the mentioned metrics are quite important, as any application relying on such deployments requires the synergy of several IoT entities on a city-scale (e.g. bike sharing applications) thus, information must cover long distances. Since actuators are part of the network, data integrity and reliability is necessary as well. Cost is another key issue, which can be partially covered whenever the new deployment can coexist and cooperate with legacy systems [236].

Smart Grid Finally, the Smart Grid is another scenario for which IoT technologies and standards are of paramount importance and, since the continuous energy supply is the main concern of customers, reliability, cost effectiveness and security are the key concept for such systems [136].

Table 2.1: M2M common use cases and requirements, for each of which the average estimated importance (from low to high) is stated.

Use Case	Scalability	Data rate	Reliability	Low Latency	Low Power	Cost	Security	Compatibility
Home Automation [205]	Low	Medium	Medium	High	Medium	Medium	Medium	High
Industry [213]	Medium	Medium	High	High	Medium	High	High	Medium
Environmental Monitoring [116]	High	Low	Low	Low	High	High	Medium	Low
Smart City [236]	High	Medium	High	Medium	High	High	High	High
Healthcare [30]	*Variable*	*Variable*	High	High	Low	Low	High	Low
Smart Grid [136]	High	High	High	High	Low	High	High	High

2.1.4 Short Range Communication Technologies (Capillary)

Capillary technologies are M2M technologies enabling a communication range spanning from few to above a hundred meters. In most cases, technologies in this category are used to design Wireless Sensor Networks (WSNs), consisting of a set of devices with different tasks, committed to sense or act in the real world, connected through peer-to-peer links and sticking to a set of constraint [9]. These networks are suitable for deployment in spatially limited environments, usually within a range of around a hundred meters (it can be more for multi-hop networks), where the interactions between the entities are contextually not separable and require simple and secure communication links [82]. This is the case of home automation scenarios, industrial process control, object identification, body activity monitoring, indoor localization and many others. In a current work we explored the suitability of legacy technologies (i.e. WiFi) for IoT home automation scenarios, a use case for which such technologies are still suitable, but have their undeniable drwabacks [152]. Most of the communication technologies used in such contexts are outlined here and exhaustively reviewed in [150]. More in detail, we covered popular technologies used for WPANs like IEEE 802.15.1 Bluetooth Low Energy [67], IEEE 802.15.4 [90] together with its major implementations in the upper layers of the stack (Thread 6LoWPAN [209], ZigBee [243], WirelessHART [106] and ISA 100.11a [93]) and Z-Wave [234] as well as less known and proprietary solutions such as INSTEON [91], EnOcean [138], the DASH7 Alliance Protocol (D7AP) [52] and DECT ULE [29]. The relevant features of each technoloy are resumed in Table 2.2.

Table 2.2: Capillary IoT technologies. Data was cross-checked with [104].

Name	Spectrum	Bandwidth	Peak DR	Range	Topology	PHY Modulation	MAC Access
BLE	2.4 GHz	2 MHz	1 Mbps	100 m	Star	GFSK (FHSS)	TDMA
Thread 6LowPAN	2.4 GHz	5 MHz	250 kbps	$10-75$ m	Mesh	OQPSK (DSSS)	CSMA/CA
ZigBee	2.4 GHz	2 MHz	250 kbps	$10-75$ m	All	OQPSK (DSSS)	S-CSMA/CA
ZigBee	915 MHz	1.2 MHz	40 kbps	$10-75$ m	All	BPSK (DSSS)	S-CSMA/CA
ZigBee	868 MHz	600 kHz	20 kbps	$10-75$ m	All	BPSK (DSSS)	S-CSMA/CA
WirelessHART	2.4 GHz	3 MHz	250 kbps	$30-90$ m	Mesh	OQPSK (DSSS)	TDMA
ISA 100.11a	2.4 GHz	5 MHz	250 kbps	$30-90$ m	Mesh	OQPSK (DSSS)	TDMA
Z-Wave	868/908 MHz	200 kHz	$9.6-40$ kbps	$30-100$ m	Mesh	FSK	TDMA
Z-Wave 400	2.4 GHz	-	200 kbps	$30-100$ m	Mesh	FSK	TDMA
INSTEON	908 MHz	-	38.4 kbps	45 m	Mesh	FSK	TDMA
EnOcean	868/315 MHz	62.5 kHz	125 kbps	30 m	Mesh	ASK, FSK	TDMA
D7AP Hi-Rate	433/868/915 MHz	200 KHz	166.67 kbps	10 m	Tree	GFSK	CSMA/CA
D7AP	433/868/915 MHz	200 KHz	55.55 kbps	100 m	Tree	GFSK	CSMA/CA
DECT ULE	1.8/1.9 GHz	1.728 MHz	1152 kbps	$70-300$ m	Star	GFSK	TDMA

2.1.5 Long Range Communication Technologies (LPWAN)

Nowadays, the common interest in IoT technologies is shifting from capillary scenarios, in which object clusters are enclosed in a LAN (or a PAN), to wide area scenarios, already envisioned as a key component of the future 5G deployments [16][178][141][8] and now starting to hit the market. Several companies already working on proprietary IoT wireless protocols for the purpose of home automation and monitoring scenarios are now focusing more and more on wide area technologies, such as the Wavenis technology [49]. The architectures for long range technologies follow the principles of the cellular deployments, therefore mesh networks are not an option, since the high capacity of the gateway and the wide communication range make any node capable to reach the gateway in one hop. Existing cellular networks, based on 2G, 3G and 4G technologies, already meet some of the MTC requirements, while some others, such as low power and low battery consumption, are still a challenge. Several solutions have been proposed and can be subdivided into two main categories: proprietary LPWAN solutions, deployed in unlicensed spectrum bands, and solutions integrated with the existing cellular infrastructure, sharing licensed bands with the current cellular deployment. We will refer to the latter solutions as Cellular IoT (CIoT).

Proprietary LPWAN

The Low Power Wide Area Network (LPWAN) architectures aim to exploit IoT over a wide area deploying the connections of small devices in unlicensed spectrum bands [178]. This enables stringent requirements, such as a low per-device cost, a long battery life, a low deployment cost, a high coverage (which is granted by the long range transmission) in all scenarios (e.g. indoor and outdoor) and a high scalability. Proprietary LPWAN technologies also can rely on immediate deployment, since they do not need to coexist with legacy cellular standards due to the different frequency bands. They are also considered a hot research theme, since LPWAN connected objects are expected to be 3.6 billions by 2024, according to Machina Research forecasts [135], an impressive slice of the market. They are currently competing with 3GPP cellular technologies operating in licensed bands, outlined in Section 2.1.5, which, however, are 1 to 3 years away from providing a competitive solution and a significant deployment [128]. In [150] we reviewed thoroughly the currently used LPWAN technologies: SigFox[1], LoRa[2] [127], Weightless[3] [68] and Ingenu's Machine Network[4] [50]. The relevant

[1] http://www.sigfox.com
[2] https://www.lora-alliance.org/
[3] http://www.weightless.org/
[4] http://www.ingenu.com/

features of each technoloy are resumed in Table 2.3.

Table 2.3: LPWAN technologies operating in unlicensed bands. Data was cross-checked with [104].

Name	Spectrum	Bandwidth	Peak DR UL	Peak DR DL	Range	PHY Modulation	MAC Access
D7AP Lo-Rate	433/868/915 MHz	25 kHz	9.6 kbps	9.6 kbps	∼5 km	GFSK	CSMA/CA
SigFox	868/915 MHz	192 kHz	∼100 bps	∼100 bps	>20 km	GFSK/DBPSK (UNB)	ALOHA
Ingenu MN	2.4 GHz	1 MHz	∼30 kbps	∼30 kbps	∼15 km	FSK, PSK (DSSS)	RPMA
LoRa	868/915 MHz	125 kHz	∼50 kbps	∼50 kbps	∼11 km	CSS	ALOHA
Weightless-N	868 MHz	200 Hz (?)	∼100 kbps	-	∼5 km	DBPSK (UNB)	S-ALOHA
Weightless-P	868 MHz	12.5 kHz	∼100 kbps	100 kbps	∼2 km	GMSK, OQPSK (UNB)	FDMA,TDMA
Weightless-W	470/790 MHz	6 − 8 MHz	∼10 Mbps	∼10 Mbps	∼10 km	DBPSK/QPSK 16-QAM (DSSS)	FDMA,TDMA

Cellular IoT (CIoT)

CIoT technologies represent the second facet of long range M2M technologies; their distinction lies in their deployment in licensed bands alongside with existing cellular technologies, whereas proprietary LPWAN technologies use unlicensed spectrum. The need for such technologies is quite evident from recent performance evaluations; for instance, in one of our recent works we perform accurate performance and simulation tests to assess the current unsuitability of LTE for the busrty traffic typical of the IoT (in such case, the simulations have been performed with respect to the vehicular infrastructure) [153]. The term CIoT was first approved by 3GPP in GERAN and 3GPP is now seeking for new proposals with regards to the following aspects [72]: improved indoor coverage (where RF signal penetration is limited), support for a massive number of low throughput devices in limited bandwidth and delay sensitivity These technologies are currently under rollout, thus there is no operating instance. In [150] we reviewed thoroughly the CIoT proposals: EC-GSM [5], LTE-M [4] (also referred to as LTE Cat-M1, LTE Cat-M or eMTC), NB-LTE-M [159] (also known as LTE Cat-NB1 or, more commonly, simply as NB-IoT) and Clean Slate NB-IoT [233]. The relevant features of each technology are resumed in Table 2.4.

Table 2.4: Cellular IoT technologies operating in licensed bands. Data was cross-checked with [104].

Name	Spectrum	Bandwidth	Peak DR UL	Peak DR DL	Range	Modulation	Access
EC-GSM	700/900 MHz	200 kHz	∼10 kbps	∼10 kbps	∼15 km	GMSK	TDMA
LTE-M	700/900 MHz	1.4 MHz	∼1 Mbps	∼1 Mbps	∼11 km	QPSK, 16-QAM, 64-QAM	OFDMA
NB-LTE-M	700/900 MHz	200 kHz	∼144 kbps	∼200 kbps	∼15 km	QPSK, 16-QAM, 64-QAM	OFDMA
NB-CIoT	800/900 MHz	180 kHz	∼36 kbps	∼45 kbps	∼15 km	BPSK, QPSK, 16-QAM	OFDMA

2.1.6 Discussion

In this Section, we examine horizontally the technologies that we presented in Sections 2.1.4 and 2.1.5, focusing primarily on the metrics and the use cases we

introduced in Sections 2.1.1, 2.1.2 and 2.1.3. In [150] we also report the major research challenges specific to the design of M2M technologies and we redirect there the interested reader leaving such matter apart for the purpose of the present document.

Scenario Specific Discussion

We now discuss scenario specific possibilities using the technologies presented so far and related to the use cases introduced in Section 2.1.3.

Clearly, short range communication is more suited for networks that do not need to span across considerable distances. Rather, their characteristics make them useful for networks in need of local control, which may rely on other technologies to bring the data at longer distances through the Internet. Long range communication technologies enable M2M devices to communicate at longer distances, enabling novel possibilities for services requiring communication over different places located farther apart.

Concerning *Home Automation* scenarios, short range technologies are certainly those which are better suited and more widespread in the current deployments [235]. While intra-network communication may leverage specific technologies tailored for the specific device and communication requirements, such as Zigbee and Z-Wave, the use of a user device for interaction requires a shared technology, like BLE.Typically, a bridge device, generally main powered, acts as a central gateway which is equipped with multiple technologies (i.e. the ones suited for the intra-network communication and the ones for communicating with the user device or with the home router), which makes the communication possible. The main research challenge here resides on making the communication efficient between different technologies, which is typically realized in the gateway through a middleware which handles the heterogeneity between the connections, a challenge tackled in the Fog Computing paradigm. In contrast, long range technologies are not the best suitable option for *Home Automation* due to the limited space in which the network is deployed. However, they may still be viable for specific scenarios, such as connecting parts of the building that are either far apart from each other or need different features not offered by short range technologies in order to overcome obstacle shadowing (e.g. more transmitting power or lower frequencies).

Industry 4.0 nowadays heavily relies on short range communication technologies, mainly due to energy efficiency and reliability. Among the possible scenarios which *Industry 4.0* face, such as *Predictive Analytics* and *Machine Internal Control*, all of them need long operational life, and resilience to malfunctions. For such reason, in the vast majority of deployments, TDMA-based protocols (such as WirelessHART and ISA 100.10a) are chosen over others, due to their efficiency

in time and the fact that industrial scenarios are rarely subject to topology change. BLE has been taken into account as well due to recent developments in its mesh real-time variant [164]. Although *Industry 4.0* does not normally rely on long range technologies, since the majority of the nodes tend to be close to each other in the network, long range technologies may be used for scenarios in which different buildings have to be connected or separate entities can be cut off from the network. In fact, the use of unlicensed spectrum, as in LPWAN, has reliability issues, due to the lack of guarantee of service availability, mainly because of duty cycling and Listen-Before-Talk (LBT) regulations. The coexistence problems introduced doubts on cellular solutions as well [200].

Healthcare is a broad scenario that makes large use of short range communication technologies. Apart from hospital devices, which form networks on their own, more recent wearable computing devices also leverage these technology, for continuous monitoring of the vital signs of human beings. These devices need a gateway to report data to the user, being it the user's smartphone, hence generally using BLE, or a different gateway, hence using 802.15.4 [103]. Usually networks are composed by a reduced number of devices, hence the challenges are rather on the upper layer optimization, reducing communication between the end devices and the gateway to reduce battery consumption. For *Healthcare*, long range technologies are mainly used to report patient monitoring data to a central aggregator. This is particularly useful for recent scenarios such as those in which, instead of monitoring patients in hospitals, the monitoring takes place remotely, however, for many of the long range techologies, the reliability of the connection is not always granted. In fact, practical studies have been conducted, stressing the current unsuitability of LPWAN technologies for critical monitoring use cases [168].

Environmental monitoring usually requires to span over large distances. Hence, short range communication technologies are not the most suitable option, although, using multi-hop short range communication technologies may still be viable, clearly with increased battery consumption due to the increased volume of communications. Long range technologies are much more suitable for *Environmental monitoring*; standards like LoRa and SigFox are already used depending on the scenario requirements and, in the future, cellular technologies are also desirable. Energy efficiency is the most important focus here, in contrast with reliability, as a longer battery duration turns out in a huge monetary saving. In particular, NB-LTE-M and LoRa appear to be suitable options, with more than $10\ \mathrm{km}$ range outdoors. NB-CIoT is another alternative too, although it slightly penalizes the data rate, favoring the number of devices supported per BS.

In *Smart cities and Smart buildings* there are many different use cases, such as the *Smart grid*. Clearly, there is and there will be a merge of different telecommunication technologies, therefore, the main challenge is making those interactions efficient and resilient to different problems. Energy efficient routing algorithms

and software optimization such as caching, along with self healing capabilities for both the devices and the bridge are needed. A specific technology is hard to predict, as each of those is built according to specific constraints and can suit better a specific use case compared to others. Again, the interaction between different networks and at different layers of the network architecture is the key challenge and, in the commonly shared future IoT vision, such ecosystems will necessarily make extensive use of long-range technologies as well. Finally, as already pointed out, *Smart cities and Smart buildings* is a wide use case, in which both short range and long range technologies are used. Depending on the size of the city, and on the layer of optimization, different standards may be well suited. For instance, the authors of [113] compare the coverage of GPRS, NB-IoT, LoRa, and SigFox technologies via a simulation study over a realistic, large-scale city scenario; the experimental results show that the NB-IoT technology provides the largest coverage, however they also reveal the need of additional measurements and research studies in order to identify the best trade-off in presence of multiple requirements (e.g. scalability and deployment costs on dense populated urban areas).

Current M2M Deployments

In this Section, we discuss the existing deployments of M2M technologies worldwide, by identifying current trends and future initiatives. We mainly focus on LPWAN-based deployments, since most of short range and capillary technologies constitute consolidated approaches and are less preferable for large-scale installations, particularly when these are sparse. This is not surprising, due to the new requirements that characterize use cases like smart cities, healthcare and remote monitoring, in which end devices are expected to be arbitrarily deployed and moved anywhere without connectivity consequences [225]. To this end, proprietary LPWAN technologies are already hitting the market in several countries, while the efforts to bring CIoT technologies to an active state on the market are still at their beginning. In fact, apart from few testbeds aimed to compare CIoT technologies under similar environmental circumstances, the actual studies are still limited to analytics [144][66] and simulations [167]. Technologies like SigFox and LoRa are still under rollout worldwide, however, they have been adopted as a local network in different measures. SigFox, at the time of writing, covers officially 20 countries in Europe, 10 in Asia, 11 in South America, 2 in North America, 4 in Oceania and 3 in Africa [3], although the numbers are changing incredibly fast. It was first deployed to cover nationally France in 2014 and it fastly reached coverage in 5 countries in 2015. LoRa is a big competitor to SigFox and slightly more common. It is operating actively in 43 countries through 76 different network operators giving a public network access [129]. Although SigFox and LoRa tend to be concurrent deployments, they have different features and,

in a sense, they are complementary, thus coexisting deployments can serve easily different types of market and use cases [141], e.g. LoRa grants more payload length, more latency performance and more deployment flexibility thanks to the hierarchical network topology, whereas SigFox offers more coverage (only three SigFox base stations can offer coverage to the whole Belgium).

The other big competitor in the area is LTE-M together with its complementary NB-IoT (or NB-LTE-M), although it comes somewhat late in the big LPWAN party, as currently (to the best of our knowledge) it has no active and publicly available deployment. Nevertheless, its backward compatibility with the current cellular deployments is a strong point that will give to this technology a central role within the future IoT traffic in the 5G. Moreover, during 2017 and 2018 its rise has been quite impressive, with 41 launches by 23 mobile IoT commercial operators in 26 countries as of 21 February 2018 and currently under rollout [2].

2.2 Collective Awareness Paradigms

A fundamental building block of the present dissertation is given by the Collective Awareness Paradigms (CAP), a set of methodologies and systems that leverage the power of the collaboration in information acquisition, data collection, task execution and many other fields in which a hard, complex and resource-consumptive task is offloaded to a multitude of workers. This results in a minimal effort for the individual, a benefit for both the executors and the issuers and, whether the available resources are well managed, a massive economic saving. In this section we introduce various CAPs that have been extensively studied in literature and, due to the lack of a proper definition and separation of CAPs in literature, we accurately define, as we did in a recent work [154]. In particular, we identify the key features to classify the most used CAPs in IoT (Crowdsourcing and Crowdsensing, for which we provide detailed definitions in Section 2.2.1). To the best of our knowledge, this section and its related survey that we produced [154] are the first ones that distill the features that could be used to classify such applications, as, currently, works in the literature assume this to be known a priori or devote very little attention to the classification aspect, which is imperative in order to understand and solve real word problems with the right solution. In Section 2.2.1 we provide the definition of every CAP that we found in literature, being it related with IoT or not, in Section 2.2.2 we provide a minimal set of features on top of which applications and contexts can be classified in the proper CAP, finally, in Section 2.2.3 we provide a couple of examples that aim to stress the differences between different CAPs, i.e. the same problem solved through different CAP-based applications.

2.2.1 The Wisdom of the Crowds: CAP Definitions

We define Collective Awareness Paradigms (CAP) – there is no global consensus so far on a term that comprehends all this types of application – as paradigms that leverage the power of offloading tasks, as part of a campaign, to a crowd of individuals. The purpose is to collect data from crowds (large group of people), analyze and use such information for the benefit of the crowd itself [194]. CAPs were introduced in works like [137], where they are referred to as "collective intelligence", based upon the fact that the aggregation of different points of view or observations leads to better decisions, a concept that has also been referred to as "crowd wisdom" [206]. In this section, we aim to give an extensive description of such paradigms and applications, inspecting their definitions in literature in-order to identify features that can be used to draw a clear boundary between various CAPs. Furthermore, we need to clearly separate software-based paradigms, such as Crowdsensing and Crowdsourcing, from others where a dedicated platform is

not strictly necessary, such as Community-Based Monitoring (CBM) and Citizen Science.

Crowdsourcing

The CAP **Crowdsourcing**, coined in 2005 by Jeff Howe [84], defines a paradigm for which a specific service, information or task is offloaded to a crowd of individuals, often connected by a common interest/goal as in an online community. The most comprehensive definition of crowdsourcing has been given in [62]: *"Crowdsourcing is a type of participative online activity in which an individual, an institution, a non-profit organization, or company proposes to a group of individuals of varying knowledge, heterogeneity, and number, via a flexible open call, the voluntary undertaking of a task. The undertaking of the task, of variable complexity and modularity, and in which the crowd should participate bringing their work, money, knowledge and/or experience, always entails mutual benefit. The user will receive the satisfaction of a given type of need, be it economic, social recognition, self-esteem, or the development of individual skills, while the crowdsourcer will obtain and utilize to their advantage what the user has brought to the venture, whose form will depend on the type of activity undertaken."* As recalled, the process of transferring the task to the crowd happens through an open call – we will refer to the terms "open call" and "campaign" interchangeably throughout the paper – and it can be supported by different motivations: crowdsourcing has been successfully implemented and adopted by platforms supported by monetary incentives, such as Amazon Mechanical Turk[5] and MicroWorkers[6], or fostered by a community interest, such as Wikipedia[7].

Mobile Crowdsensing (MCS)

Mobile Crowdsensing (MCS) is another CAP which has been coined in [64] and referred to in literature previously as "community sensing" [194], "mobile crowdsourcing" and "people-centric sensing". According to the original definition, a number of individuals, forming the crowd, is committed to perform observations of real world phenomena of common interest through the use of mobile phones, given their capacity to sense the environment and other phenomena in the community, e.g. finding the total number of people in a restaurant given their GPS position, reporting anomalies in the traffic such as car accidents, reporting the geo-located presence of a particular bird though pictures. The definition has been

[5] https://www.mturk.com/
[6] https://microworkers.com/
[7] https://www.wikipedia.org/

extended from smartphones to any connected mobile device capable of observing phenomena and performing computation [73]. In [64] mobile crowdsensing was first classified into participatory and opportunistic, a separation that has been pointed out in other subsequent works. In particular:

- **Participatory Crowdsensing (PCS)** is a paradigm in which the user is actively involved, often through the use of a front-end application, and intentionally reports observations through a specific action.
- **Opportunistic Crowdsensing (OCS)** is perceived as the dual of participatory, where the user involvement is minimized (or, in some cases, none) and, often, an application is running in background performing sensing and monitoring tasks and performing decisions on where and when to sense and send on behalf of the user.

In recent years, a multitude of many other contrasting definitions and nomenclatures for MCS have emerged [73][111][41]. Nonetheless, their definition tends to be uncertain and their separation not well defined. For example, in [64] the main discriminant is the spectrum of user involvement and the two paradigms, namely PCS and OCS, are put at the opposite ends of the scale, a concept recalled in [73]. Differently, in [134] and [111] the separation line occurs towards user awareness instead, and the PCS and OCS are depicted as complementary. Another sharp line is drawn in [41], where the sensing automation is the key parameter and, moreover, MCS is defined as a subset of crowdsourcing in which mobile phones are required. Differently, in other works, like [73], MCS is considered as an extension of crowdsourcing, as it does not fit the original definition of crowdsourcing in all cases. Some of these premises suggest that participatory and opportunistic crowdsensing are orthogonal sets of MCS, instead, they intersect to a great extent with each other as well as with crowdsourcing. Due to the emerging of such different definitions in the literature, the classification of software-based CAP applications – we consider Crowdsourcing and MCS (in particular OCS and PCS) as software-based – is fuzzy. Hence, in the next section we develop and identify a set of features that can be used to draw a clear boundary between various CAPs.

Community-Based Monitoring (CBM)

Another CAP that closely relates to our work is **Community-Based Monitoring (CBM)**, which is defined as *"a process where concerned citizens, government agencies, industry, academia, community groups, and local institutions collaborate to monitor, track and respond to issues of common community environmental concern"* [221]. Through such process, citizens and institutions collaborate with the aim of solving issues related to the environment in which participants

are committed to collect information through eye-witnessing and may or may not take active part in the decisions deriving from the outcomes of the campaign. The literature about human collaborative actions taken upon CBM and IoT concepts is vast, indeed, environmental CBM has been deployed in several projects and it is categorized on top of both the capabilities and the awareness that are granted to participants [114]. More in detail, We refer to "consultative CBM" whenever citizens are participating in collecting data and measurements without being necessarily involved in observing the results neither in decisions taken upon them. We name as "collaborative CBM" the paradigm in which participants are still the primary source of information, however they can get access to the outcomes and can take decisions on future directions. Collaborative CBM can be categorized further and presents a more complex structure of user pool: it can include citizens, stakeholders, producers and consumers. As an example, it can be pushed to "transformative CBM", in which the actual demand and the goals of each campaign come directly from the end users, the citizens in most cases. Hence, it is clear how consultative CBM, being driven by the government or a certified institution, has a clear goal and is able to provide long-term datasets. Nevertheless, it is dramatically linked both to the issuer's resources and to appropriate incentive techniques. On the other hand, collaborative CBM presents an intrinsic advantage for the participant, thus it needs less explicit incentives to reach a satisfactory coverage. However, the power given to both malign and inexpert users might be dangerous for the data credibility [47]. An example of one of such campaigns is given by the Louisiana Bucket Brigade[8], an environmental health and justice organization collecting participants' reports and initiatives concerning petrochemical pollution through eye-witnesses. CBM differentiates from Crowdsourcing and MCS in that it is solely oriented to the observation of phenomena in the scope of environmental and urban monitoring and does not require necessarily a software platform (if it does, it then becomes an instance of Crowdsourcing or MCS).

Citizen Science (CS)

Citizen Science (CS) is a CAP that aims to involve citizens as volunteers in the conduction of a task finalized to the accomplishment of scientific research. With "citizens" we mean amateurs, people without the total knowledge of the field that the research deals with. It has been defined in the mid nineties [92] and, since then, it relies more and more on the support of technologies. In particular, CS plays a fundamental role in Public Data Archiving (PDA) for scientific experiments, in order to build open access datasets useful to researchers [165]. Clearly, CS is a powerful and convenient tool, as it grants a consistent amount of data

[8] `http://www.labucketbrigade.org/`

without the need to pay experts for its collection, however, on the other hand, it suffers from two main weaknesses: it needs to sufficiently foster the participation of volunteers (which is not always granted) and it can provide, due to the inexperience of volunteers, massive amounts of low quality and high biased data, which can be even damaging to potential results [108]. CS is exploited for the most part in the field of ecology and conservation biology, where finding volunteers driven by passion is easier. Over the recent years CS has acquired a highly technological connotation to the point that its blend with software-based CAPs is highly evident. This has been shown by several research groups among which the Cornell Laboratory of Ornithology (CLO) [24], and some well-known Crowdsourcing and Crowdsensing projects gathering electronic records of specimen, like eBird[9] and iNaturalist[10]. In spite of the reciprocal interest between the communities of engineers and biologists, the interactions have always been sporadic; over the last few years the area of Conservation Technology, a step forward in the collaboration between the scientific communities, has risen [21].

2.2.2 Classification of Collective Awareness Paradigms

Below we define in detail the features that we used to classify CAPs. We employ definitions similar to the ones used in the literature (where available) to describe the features. We recall that, for the purpose of this dissertation as well as a guideline for the whole computer science community, we solely refer to CAPs that are software-based, namely Crowdsourcing and Crowdsensing, while others are left apart even though they can blend to an extent. Below we enlist the features used for the comparison.

- *Participant:* We define participant as any actor belonging to the crowd in the CAP ecosystem that is able to be issued with, accept and perform tasks. Whenever a participant contributes actively through specific actions (e.g. providing content, activate sensors, go to a specific location, etc.) and/or performs decisions, we say that there is a degree of **User Involvement**.

- *Campaign and Task:* We define a task as a sensing activity that can be delivered to all or a defined group of participants. A task may or may not have a time limit and it is defined upon a specific type of observation. A set of tasks are generally combined into a campaign that is owned and managed by an individual, organization or government.

[9] `https://ebird.org/home`
[10] `https://www.inaturalist.org/`

- *Sampling:* We define sampling as a combination of sensing cycle and transmission cycle. Sensing cycle refers to the frequency at which measurements (termed observations) about a phenomenon are performed (either by the sensor or by the participant). The transmission cycle refers to the frequency at which the observations are reported. When the whole sampling process or part of it is performed by a sensor or a software automatically (i.e. without user intervention), we say that there is **Sampling Automation**.
- *Location and Time:* Information about location and time are key features of CAP applications. Most CAP applications require tasks to make observations at a certain location during a defined period of time (e.g. report pollution level in the city center of Melbourne, Australia during New Year's night). The information on location and time can be obtained manually or automatically. If they are required, we say that the applications support **Spatio-Temporal awareness**.

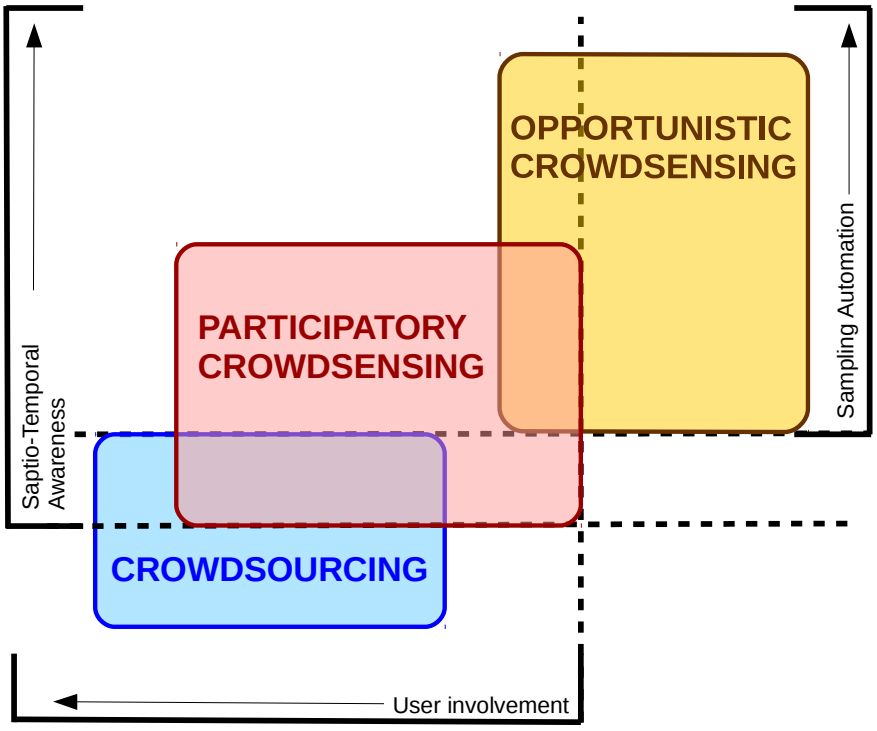


Figure 2.5: Classification of Collective Awareness Paradigms [154].

In Figure 2.5 we present the classification of CAPs (Crowdsourcing, OCS and PCS) based on the aforementioned features. From the figure, it is evident that

MCS (which comprehends PCS and OCS) builds on the fundamentals of Crowdsourcing, while having its own unique features. For instance, Crowdsourcing and PCS applications always require user involvement to perform a task while OCS requires little to none. Similarly, the capability to infer location automatically and dependence on time to perform a specific task is a requirement for PCS and OCS, as opposed to Crowdsourcing. Finally, OCS applications require sampling automation while crowdsourcing and PCS applications, due the high level of user involvement, can cope with little or no sampling automation (e.g. user performing activities such as identifying images with a car in CAPTCHA[11]). Based on the identified features used to classify CAPs, we propose the following definition to clearly define MCS applications: *"a paradigm through which a number of individuals, called **participants**, are committed to perform **tasks** – as part of a **campaign** – involving sampling of real world phenomena of common interest through the use of portable, connected and **Spatio-Temporal Aware** mobile devices in order to enable its mapping through information aggregation"* .

2.2.3 Applications

Here we describe two application scenarios and use these to drive and further exemplify the differences between Crowdsourcing, PCS and OCS.

Smart Transportation One of the most common examples is given by the traffic navigator applications Google Maps[12] and Waze[13], in particular in their way of predicting traffic intensity and jams. As a matter of fact, Google Maps performs an opportunistic analysis on the GPS fingerprints of its users, estimating the amount of vehicles located within the same road segment and their average speed; in light of this it is an OCS application. On the other hand, Waze works upon the active participation of its users, who are able to actively post notifications about car accidents, traffic jams, road structure changes and many other phenomena; therefore, it is a PCS application. Furthermore, organizations such as VicRoad[14] in the city of Melbourne, Australia provide web-based form for citizens to report road hazards, which is a classic example of Crowdsourcing (no Spatio-Temporal Awareness of the device), while solutions such as Nericell [145] achieve the same goal via OCS (minimal user involvement, Spatio-Temporal Awareness).

[11] `https://captchas.net/`

[12] `https://www.google.com/maps/dir/`

[13] `https://www.waze.com/en/`

[14] `https://www.vicroads.vic.gov.au/traffic-and-road-use/report-a-road-issue`

Smart Parking Another application to demonstrate the differences between Crowdsourcing and MCS (both PCS and OCS) is the Smart Parking. Smart Parking takes advantage of IoT and mobile devices to facilitate contextual functionalities such as finding car parks based on the availability, distance and context [229]. For example, a Crowdsourcing application will require users to report availability of parking spots at a given location via physical observation of the space. A participatory MCS application would require the users to perform observation on the parking availability on top of their location, as in [42]. An opportunistic MCS such as ParkNet [139] will enable automatic detection of parking availability and report this to a parking reservation system. Moreover, the advent of IoT could further transform OCS application such as ParkNet with ability to fetch parking information by communicating directly with the parking space monitoring sensors.

A visual illustration of the both smart transportation and smart parking scenario is depicted in Figure 2.2.3.

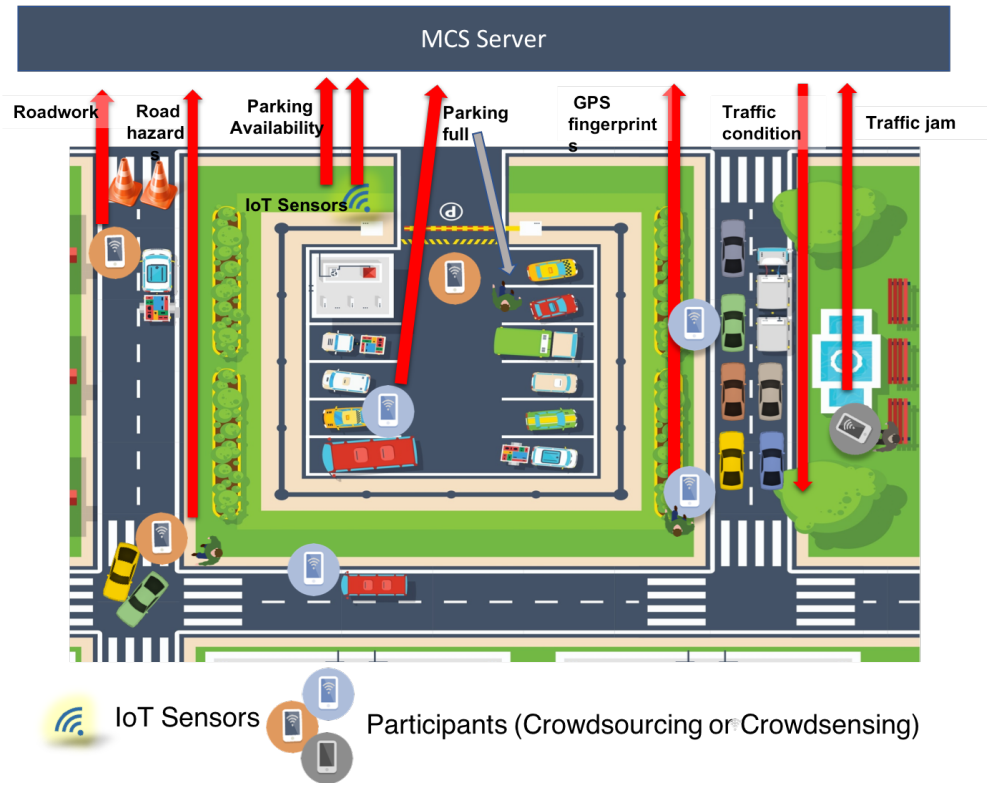


Figure 2.6: Smart City, Smart Transportation and Parking [154].

2.3 Collaborative IoT and Open Data

Regardless of the different perspectives in predicting the near future in the IoT, it is clear that the number of devices is growing, the data is becoming more and more heterogeneous, and one of the main challenges is how to handle such an amount of data and how to give a meaning to it. At the same time, makers worldwide build their own IoT in-home networks, which provide a low cost and customized environment that suit their needs. Platforms like Arduino[15], (and all its derivatives) and Raspberry Pi[16] have demonstrated their ease of use and they fit most of the needs of citizens willing to build their own network. However, certain types of sensor can be too expensive, or cannot be deployed due to restrictions or physical space limitations, or simply because users might not have the right skills to make use of a personal IoT ecosystem by themselves. For this reason, a collaborative approach to IoT is seen as a useful solution that facilitates the access to critical data. Open Data, described in this section, is one of the most powerful solutions that creates cooperation between end users and an information reuse through the massive generation and sharing of scattered IoT data. A fundamental requirement in successfully re-purposing such open IoT data, in order to enable interoperability as envisioned by Semantic Web 3.0, is to be able to automatically characterize its metadata i.e. information such as observation type (e.g. temperature, humidity), unit of observation (e.g. Celsius, Fahrenheit), location etc. However, as validated by a recent study in the literature [199], most publicly available IoT data, due to their crowdsourced nature, lack availability of such accurate metadata and, in most cases, even the observation type is unclear (i.e. what is actually being measured).

This section addresses the current situation in the world of IoT architectures for use cases such as Smart Cities and environmental monitoring, why Open Data is a viable option for bridging gaps in the architectures' interoperability and the challenges that such approach introduces.

2.3.1 Data Silos

The Internet of Things (IoT) is one of the research and industrial fields that faced the most rapid growth in the recent years, mostly thanks to the proliferation of new technologies associated with the ease of installation and use. This development has created a wide variety of standards and solutions at each layer of the network and application stack, leading to an heterogeneous environment both in

[15] `http://www.arduino.cc/`
[16] `https://www.raspberrypi.org/`

terms of communication technologies (as outlined in Section 2.1) and data storage. Since the beginning of its diffusion, the potential of the IoT has been explored in various fields of application and its major usefulness has been claimed to be in service composition and interoperability [13]. The requirements when designing collaborative IoT-related automation systems are varying due to the heterogeneity of the platforms and the hardware components as well as the network interfaces. This resulted in a sparse set of technologies and terminologies used in several scenarios determining a lack of interoperability among systems. Therefore, IoT ecosystems behave as disconnected network islands, in which it is easy to build networks with homogeneous devices, however it is hard to integrate data provided by other sources. It is actually difficult to talk about an "Internet of Things" when what is out there it is more a set of "Intranet of Things".

Projects for IoT Frameworks

The common approach to the problem of unifying entities within an ecosystem is typically architectural and leads to a difficult reuse of the components among different solutions [109]. To face these issues the European Commission supported initiatives like IoT-A[17], which aimed to release an architectural reference model, and FI-WARE[18], which also helped architects in establishing a unified vision and nomenclature and now had become an implementation-driven open community. FI-WARE also provided a sandbox, in which partners could upload their Open Data, although it is not of broad use nowadays. Such solutions, unfortunately, did not solve the problem introduced by architectures, in fact different approaches still tend to create separate ecosystems which are hard to unify. Another project that had significant audience is the Global Sensor Network (GSN) [6] which consists of a middleware that implements virtual sensor abstraction together with a multitude of functionalities to declare and deploy a virtual sensor, discover it and retrieve the values through powerful aggregate queries that combine primitive data types. On the same concepts, more recently, the project OpenIoT [202] had the goal of constructing a unique and interoperable IoT ecosystem leveraging the concept of the Semantic Web applied to the world of IoT; for such purpose they used ontologies like the Semantic Sensor Network (SSN), created specifically for describing sensors and sensor data. Projects that leverage interoperability and automation in the IoT world have been indeed extremely appealing to the European Commission, as a matter of fact, some of them reach an outstanding size in terms of number of partners and fundings. This is the case of the Euro-

[17] `http://www.iot-a.eu/public/front-page`
[18] `https://www.fiware.org`

pean project Arrowhead[19][55], a project with more than 80 partners, to which we brought significant contributions, outlined in detail in the Appendix.

Commercial IoT Frameworks

Commercial solutions aim to constitute a living ecosystem in which entities are "plugged" and interoperable, participating for the benefit of the whole system and fully compliant with the other actors within the same environment. Most of the times, such frameworks, some of which are deeply investigated in [57], provide efficient software adapters for legacy systems. Such types of frameworks are often self contained and tend to create a cluster of devices which need to be framework-compatible in order to interoperate. An example is Cumulocity[20], a platform providing an unified service oriented HTTP REST interface to devices. Another project attracting interest in recent years is AllJoyn[21], developed by the Allseen Alliance. Such a framework again forces devices to either implement an attachment to a software bus between applications, which is indeed the AllJoyn core, or connect to an AllJoyn router using a thin library. Either way, the communication introduces very low overhead and grants integration to even constrained devices; however, the protocol used is highly customized and makes AllJoyn a quite isolated ecosystem. Another example is Xively[22], which, again, allows devices to obtain interoperability even among different application protocols (CoAP, MQTT, HTTP, XMPP and others) offering an API that implements a custom message bus. Until few years ago Xively also provided a public instance of the cloud, making possible for the users to only create a client device without the need for a personal server, also generating an Open Data repository. Finally, another framework to mention, which has been standardized by the Open Mobile Alliance (OMA), is OMA-LwM2M[23], which defines a custom layer over CoAP focused on exchanging instances called "objects" and operating upon them via the custom interfaces.

2.3.2 What is Open Data and Why it is Important

In our research we make use of a concept that makes immediately possible, although not prone of difficulties, the integration of data from heterogeneous sources, i.e. the use of Open Data, a powerful source of information for devel-

[19] http://www.arrowhead.eu/

[20] https://www.cumulocity.com/

[21] https://openconnectivity.org/developer/reference-implementation/alljoyn

[22] https://www.xively.com/

[23] http://openmobilealliance.org/

oping novel IoT applications in domains such as smart cities, defense, healthcare and environmental monitoring, to name a few. Open Data is, as the name suggests, data that is freely accessible in machine-readable format from public repositories and might be either contributed by users or gathered in an open access form through an initiative. In fact, we group Open Data repositories as "reliable", that is, repositories maintained by organizations or governments, and "unreliable", that is, repositories created through crowdsourcing: users freely contributing in uploading datastreams through their personal devices [148]. Reliable Open Data repositories are preferred, since the data they provide follows some sort of annotation policy (i.e. we know exactly what it is), its updates are regular and its quality is guaranteed by the use of professional appliances. Examples of reliable Open Data repositories are the Environmental Protection Agency (EPA)[24], providing environmental monitoring data in the United States, the Regional Agency for the Protection of the Environment (ARPA, the equivalent of EPA in Italy)[25], various services related to weather and forecasts providing APIs such as WeatherUnderground[26] and DarkSky[27]. It is also woth mentioning several Open Data initiatives for Smart City projects such as the public repository in Singapore[28] or in the city of New York[29] or the Spanish Santander project [30] which include several datasets that can be useful to IoT applications. Unreliable Open Data repositories, on the other hand, provide IoT data for which there is no warranty about its veracity neither about what it actually measures; data is, in fact, typically unlabeled, poorly annotated and incomplete and need a data processing step to classify which datastreams are valuable and what do they actually measure. This opens up several issues, for instance we could possibly exclude valuable results due to their bad labeling, i.e. a temperature value could be named with a pointless name and thus not classified as meaningful, we could even include measurements that are not valuable for our system. Examples of crowdsourced (and unreliable) Open Data repositories are: ThingSpeak[31], a repository where users can upload data generated by their personal devices (mostly environmental) in "data channels", or OpenSignal[32], an Open Data repository that gathers readings about the signal strength of each base station for each cellular technology. Other Open Data repositories followed, until few years ago, the same approach of ThingSpeak. Few examples are SparkFun

[24] https://www3.epa.gov/
[25] https://www.arpae.it/
[26] https://www.wunderground.com/
[27] https://darksky.net/
[28] https://data.gov.sg/
[29] https://data.cityofnewyork.us/
[30] http://datos.santander.es/
[31] https://thingspeak.com/
[32] https://opensignal.com/

Electronics[33], an open hardware reseller (they shut down their public cloud in 2017), and Xively[34], the main product of which is a local data cloud for privates (they had a public instance of their cloud until 2017). There is also a third class of Open Data: the "Social Web", which creates a huge amount of content that, through various steps of processing, it can provide valuable information as well as enriching existing one; however, we do not deal with such data as is intended as a future work and, thus, outside of the scope of this dissertation.

Why then unreliable Open Data repositories should be of interest? First of all, it is worth noting how observations coming from reliable sources are given at a wide area granularity (often per-city), which, for some types of data, might be inaccurate. Examples include the noise level, which varies dramatically when the measurement is taken close to a highly crowded street or in a house backyard (the distance between the two can be small), or the temperature, which, for instance, drops in parks and rises in congested roads. Another reason lies upon the general trend in the usage of these platforms throughout a time window of few years. Let us consider the example of ThingSpeak, for which we analyzed all the public data channels (around 160.000 out of a total of 600.000 are public at the time of writing), all of them coming with a creation date and the time of the last update. We report the channels in the diagram in Figure 2.7. For each month in the diagram, the horizontal line inside the boxplot represents the number of active channels, the upper box is the number of newly created channels, and the lower box is the number of channels that have been updated for the last time on such month (we assume them to be inactive from that moment). Green boxes are those for which the number of created channels is higher than the number of channels that ceased their updates, red boxes are the opposite.

Starting from such analysis it results an exponential growth in created channels from 2011 until today. In particular, we can observe few steep increases, for instance the one starting at the end of 2014. A possible intuition behind this phenomenon is the parallel innovation in simple hardware modules, that, it, August 2014 corresponds to the launch of the first version of ESP8266[35] on the market and in October 2014 was possible to flash its firmware though an SDK [1]. Such analyses shed some light on how rapidly the world of Open Data is growing and people are gaining interest in using a platform that takes away the burden of creating a local ecosystem. Furthermore, taking a look at Figure 2.8, we also understand the importance of using heterogeneous data sources. The figure shows all the geolocated data channels that we extracted from the ThingSpeak and SparkFun Open

[33] `https://www.sparkfun.com/`
[34] `https://xively.com`
[35] `http://www.esp8266.com/`

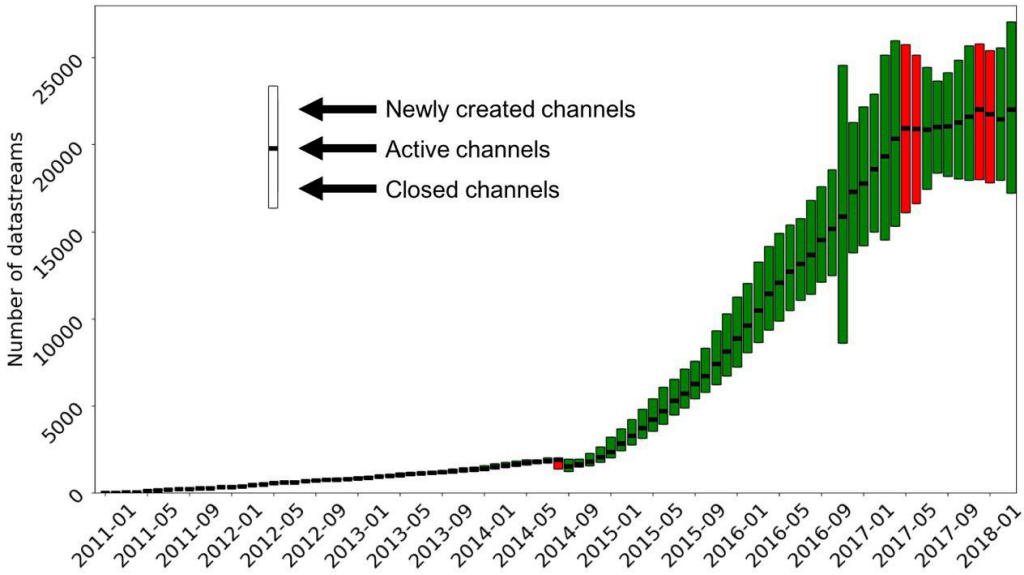


Figure 2.7: Trend in creation and update of ThingSpeak channels.

Data clouds in 2016 (even though the data channels in SparkFun were not really geolocated using GPS, but using the name of the city, which we converted in approximated GPS coordinates). Given such results, the importance of information fusion from different sources is clear, since merging such sources not only increments the sampling number of the sensing infrastructure, but also its coverage. In fact, ThingSpeak appears to have much more utilization in the European region, whereas SparkFun seems to be more popular in North America. Furthermore, this consideration might be extended to different macro topic areas, meaning that some Open Data sources are specialized on a specific field of measuring. For instance, governmental sources providing Open Data such as EPA are primarily focused on environmental data, whilst sources such as OpenSignal regard measurements on cellular network signal strength and coverage.

The data collection regarding the primary environmental and urban measurements, such as temperature, humidity, light intensity, noise, pressure, wind strength and many others, is currently considered an easy and inexpensive task. For this reason, location-aware community-based data collection (which is still considered as crowdsensing) through either embedded or general-purpose devices, has been found to be the basis for the development of novel IoT applications in domains such as smart cities, defense, healthcare and environmental monitoring, to name a few.

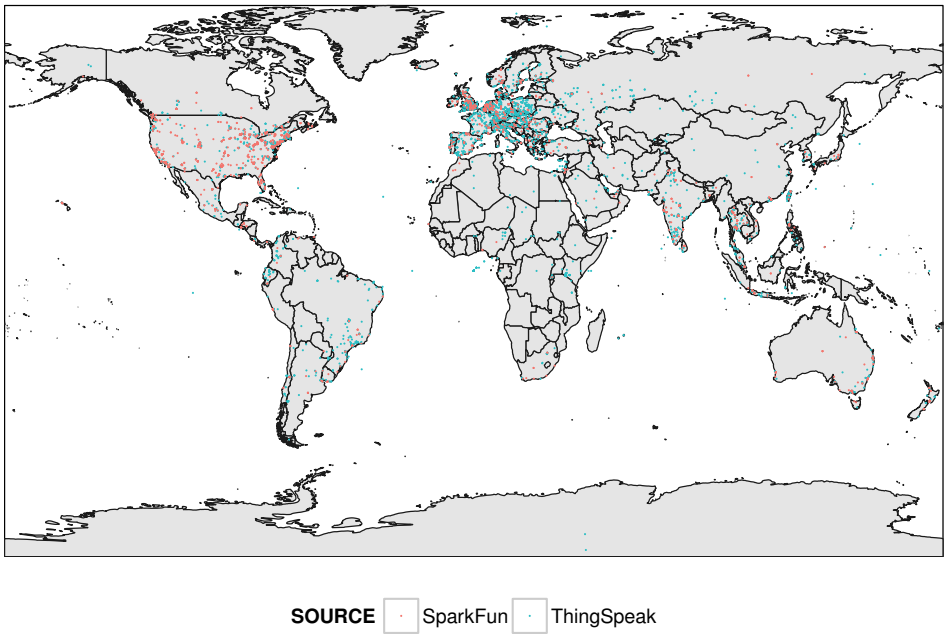


Figure 2.8: Location of all ThingSpeak and SparkFun sensing sources in 2016 [146].

2.3.3 The Issues of Crowdsourced Open IoT Datastreams

We use the term "datastream" to refer to an individual series of chronologically ordered numerical data points, each of them corresponding to an observation, together with its metadata, produced by a sensor on an IoT device. In contrast, we refer to "IoT data" as the collection of data produced by several IoT devices. We used such term to distinguish what we refer to as datastream and what is commonly referred to in literature as "data stream", since the latter is tied to the concept of streaming: a stream is "an appropriate model when a large volume of data is arriving continuously and it is either unnecessary or impractical to store the data in some form of memory" [160], whereas, we do not focus on this aspect and, sometimes, datastreams are stored and queried through batch analyses. Heterogeneous data collected from Open Data sources, as seen, often has the drawback of being unlabeled and sparse and, therefore, its meaning is hardly intelligible. Even metadata has a varying degree of availability and accuracy (partial to none). For such reasons, a data integration and classification layer is necessary in order to understand the semantics of the data collected. Open IoT datastream classification is a novel problem and has not been addressed well in the

literature. In order to support the vision of the IoT-driven Web 3.0 – i.e. moving from numerical data to smart data (data that is well described, allowing interoperability and re-purposing across several domains) – we focus on the challenge of annotating open IoT datastreams produced by *heterogeneous* IoT environments. Such heterogeneity, attributed to the nature of the Internet that allows everyone to contribute, is due to diversities in the way data is captured (e.g. location, the accuracy and range of the sensor producing the IoT datastream, non-alignment in time, etc.) and imposes additional challenges in solving the IoT datastream classification and annotation problem. In particular, our focus is on *classifying the observation type of an IoT datastream* under the following two cases 1) lack of metadata and 2) partial, incomplete or inaccurate textual metadata e.g., an IoT datastream that produces temperature may be described by the user using non machine interpretable names such as "temp", or "t1", or "T (°C)", or even something way less interpretable, such as "field_1".

Classification

The fact that unreliable datastreams tend to lack metadata rises the need for an automatic way to infer features such as the data class, which is crucial if such data is meant to be used by other application interoperably. This leads necessarily to the establishment of a datastream classification algorithm. In our environment, IoT datastreams are ordered sequences of sensor readings which can naturally be seen as attributes that form what in the literature is called a "time series". Time Series Classification (TSC) problems, indeed, differ from ordinary classification problems in that features are ordered (not necessarily in the dimension of time). Within the last years, several TSC approaches have been proposed [14] as an alternative to the one-nearest-neighbor (1NN) approach using the simple pointwise Euclidean Distance as a similarity measure between series as a standard benchmark distance measure. The common agreement accepted among researchers as a "hard to beat" standard distance measure between series has been Dynamic Time Warping (DTW) [17], for which several alternatives have been proposed in order to contrast its high time complexity [98]. The above mentioned methods consider the whole series, since they extract series similarities by pointwise comparison. Other recent TSC approaches aim to find a subsequence, called "shapelet", yielding the highest information gain that can discriminate among classes and using a tree-based classification algorithm [232]. Such *shapelet-based approaches* have been improved over time, especially due to their high time complexity [173]. Finally, a third type of well-performing methods, namely *dictionary-based approaches*, split the time series in time windows and extract patterns out of each window as new features. Such methods tend to be faster than the aforementioned ones due

to feature numerosity reduction. Examples of such methods are Bag-of-Patterns (BOP) [124], which uses piecewise aggregate approximation (PAA) through Symbolic Aggregation approXimation (SAX) words [123] and *Bag-of-SFA-Symbols* (BOSS) [188], which encodes subsequences through Discrete Fourier Transform (DFT), and Time Series Forest (TSF) [56], which implies a random forest approach on summary features extracted from the intervals (different from our random forest approach in that it still calculates localized features per interval). However, as we demonstrate in Section 4.2, TSC algorithms are not as suitable for a heterogeneous type of data such as IoT data is. In general, heterogeneous data classification is a strongly widespread research area, which has been studied over decades by many researchers and typically algorithms are modeled over specific data sets, while they perform badly over others (this is known as Wolpert's *no free lunch theorem* [222]). Nevertheless, it is worth mentioning some recent research efforts such as the one carried out in [76], in which a genetic algorithm that dynamically selects a combination of well-known classification algorithms is proposed. We also experienced a wide use of clustering algorithms for class inference in heterogeneous datasets, such as the one presented in [166] for land use tagging. Clustering algorithms are seldom used when a large volume of manually annotated data is available, since they rely on unsupervised or semi-supervised bases, and are less application-specific. Narrowing down, the classification of IoT data stemming from heterogeneous IoT devices has been in very few cases considered in literature. In [32], the authors imply a PAA-based approach which treats the sensor data classification as a dictionary-based TSC problem and uses interval slopes as features. A different approach has been taken in [26]. The authors extract user-annotated sensed data from a public platform, however, differently from our work, they infer the trustworthiness and reliability of such values in relation with a reference value, which is necessarily taken from a certified source (e.g. the well-known `Forecast.io`, now replaced by DarkSky for the weather data). On the one hand this is an efficient classification solution, taking into account the measurements instead of the annotations, on the other hand it limits the classified datastreams to only the ones for which a certified value is retrievable. In [148] the datastreams are classified only on top of the metadata provided, i.e. the user-assigned name.

In conclusion, no approach currently employs ensemble methodologies in order to consider varying degree of metadata quality and availability for classification; i.e. if no metadata is available it relies only on the numerical characteristics of data, whereas if limited and inaccurate metadata in the form of text is available, it uses a combination of numerical data and textual metadata. We address this gap in literature in Section 4.2, where we propose, TKSE, a novel ensemble classification algorithm which uses a combination of any available textual metadata

describing the datastream and numerical summaries.

Semantic Annotation

In the last decade, a significant number of ontologies have been proposed by the research community to address the semantic interoperability challenge [183] in IoT environments. These ontologies are mainly developed to enable automatic integration of IoT data exposed by smart things into applications. In the rest of this subsection a brief overview of the existing approaches approaches is given.

One of the most widely used semantic models in IoT is the Semantic Sensor Network (SSN) ontology [46]. The SSN ontology describes sensors in terms of their measurement capabilities, deployment environment, and observations. SSN is an extendable ontology and allows flexible descriptions of sensor-related information over a variety of different domains. However, this ontology suffers from two main drawbacks [112]. The first shortcoming of the SSN ontology is its inefficiency: SSN has many peripheral components that makes it quite ineffective for IoT environments. However, W3C proposed a new version of SSN ontology, which solves this shortcoming; the new version is built on top of a self-contained core ontology called SOSA (Sensor, Observation, Sample, and Actuator) that includes the SSN elementary classes and properties and can be independently used to create basic conceptual annotations. Another shortcoming of SSN is the lack of support for describing several important IoT-related concepts, such as units of measurement, time, locations and domain concepts. Therefore, a considerable number of projects extend the SSN ontology to support description of other IoT fields. One of these developed ontologies is IoT-Lite [22]. IoT-Lite is a lightweight variation of SSN ontology that extends it by introducing new concepts such as “iot-lite:Object”, and “iot-lite:Service”. IoT-Lite can also be combined with domain ontologies in order to represent IoT concepts with more details.

Many of the existing solutions that aim to fill the interoperability gap are oriented towards middleware-driven solutions for interoperability, however, automatic annotation of open sensor data still remains a challenge due to the heterogeneity of the Open Data sources providing sensor readings. Furthermore, there is no unique way to interpret such data, although there has been some efforts reported in the literature. In [32] the authors annotate data produced in Open Data clouds using the SSN ontology after a classification step. Similarly in [219], the authors propose semantic enrichment of IoT data with particular focus on developing a Semantic Sensor Web. They employ rule-based reasoning approaches to reason about additional metadata. In [195], the authors provide a recent survey on semantic enrichment of IoT data. However, all the aforementioned approaches require accurate metadata of the IoT sensor (e.g. temperature measured in Celsius) to perform the semantic enrichment. Lack of metadata will render these

approaches unusable.

In summary, the review of current studies show that automatic classification and semantic annotation of open IoT datastreams is essential to understand and re-purpose IoT data, but it still remains a challenge. With respect to this, in Section 4.3, we will use IoT-Lite as our core ontology in order to store and automatically annotate IoT datastreams.

2.4 Mobile Crowdsensing

In Section 2.2 we introduced CAPs and, in particular, we defined the concept of MCS, which is a central topic in the present dissertation, as many of our contributions are focused on such paradigm (Chapter 5). Due to such skyrocketing predictions, MCS has been a hot research area for computer scientists and engineers over the last years, as it is a quite articulate and debated paradigm. We foresee that future MCS applications for the IoT will have a significant influence in the data-driven decision making era, mainly due to their non-negligible advantages: ubiquity, extra low cost and extremely high potential in generating useful data. Nevertheless, many challenges lie ahead of these rapidly emerging technologies before mission-critical application on top of these technologies can be developed and enter the plateau of productivity [163]. MCS campaigns need to deal with the citizen participation, normally fostered through user motivation, low quality data and location awareness data mining, which are both challenging and currently studied problems [74]. The clear advantage of MCS is the huge amount of data samples that can be gathered due to the paramount spread of mobile general-purpose devices – although MCS includes whatever mobile device equipped with connectivity and location-awareness, in the vast majority of cases we imply the use of smartphones –, which grants a large spatial and temporal coverage and permits to observe a phenomenon through a significant number of different measurements. It has also been demonstrated to be efficient in several fields of application where the IoT is already considered as the key technology set that tackles the most challenging tasks [207].

Up to now, in Section 2.2 and 2.3, we covered the different ways in which citizens contribute to data collections and the importance of merging several data sources in order to build a new set of reusable knowledge. In this section we will give a closer look to the state-of-the-art in the area of MCS, as we will focus extensively on this paradigm, in particular, on a problem concerning the amount of data gathered, known as the *"Curse of Sensing"*, which will be extensively covered in Section 2.5. More in detail, in the rest of the section we will cite the most recent works in the areas of applied research in MCS – incentives, frameworks and applications – and will depict the future (or soon-to-be, where pioneering efforts are already in process) landscape in the MCS research.

2.4.1 Areas of applied research in MCS

Within this section we leave apart all the aspects related to the amount of data gathered by the campaign and how to control and balance it, since it is a problem that we deeply investigated and to which we dedicate the whole Section 2.5.

Incentive Mechanisms

MCS is efficient when the penetration of the application is wide, which, as said, is tough to obtain in case of a lack of incentives. Indeed, social and monetary incentives have demonstrated to be essential for pushing the users towards collaboration [95][241], in fact, any MCS application cannot be designed without a proper incentive scheme. According to [241], incentives for MCS can be grouped in three main categories: money-based, entertainment-based and service-based. Monetary rewards, such as refunds and actual payments, are the most immediate form of incentive and have been used in several forms. The first occurrences were static, such as the approach based on micro-payments presented in [179], or dynamic, such as the one presented in [118], based on the reverse auction dynamic price with virtual participation credit and recruitment algorithm. The latter relies on the concept for which participants are willing to sell their sensed data to an auctioneer, which, on the other hand, is willing to buy the least expensive measurements. After such cycle winners raise their prices and losers lower theirs in order to rotate over the data sellers. Reverse auctions have been established as an efficient standard for monetary incentive mechanisms, in fact, other researchers proposed alternative algorithms that depend on the objective function, e.g. in [228], the authors propose a polynomial-time reverse auction algorithm that aims to maximize the utility of both participants and the platform, whereas [118] focused on minimizing the platform costs. Other monetary approaches are based on a Stackelberg game, as in [61], in which a leader (the platform) proposes a least number of participants and a total available budget, on top of which users, assuming they are aware of their own cost, decide whether to participate or not. Other non-monetary approaches take into account personal interests of categories of users, especially focusing on entertainment. In particular, gamification through the proposal of online location-based games has shown to be efficient when connected to certain kinds of observation. Games can be devoted to study network coverage areas through games like Tycoon [20] in which users are pushed to explore as many areas as possible. Other games use GPS detection in order to either generate GPS traces for scientific experiment, as in [105], or classify Points-of-Interest in a city, as in CityExplorer [140]. Another type of incentive aims to give rewards in the form of services; this is common in situations where the observations are immediately useful for other users of the same type, such as in smart parking and transportation [83]. The challenge here is mainly to ensure real observations, which is tackled through a credit and refund system.

Frameworks

Currently, MCS-based architectures are characterized by two main components: the cloud backend and the client sensing application. Each cloud application is associated with the respective client application, often without any vertical interoperability among different applications. This is commonly referred to as an *application silos*, which results a wastage of resources and data redundancy, an issue that is specular to the problems presented for IoT data in Section 2.3. To address this problem from the architectural side, many research works focus in proposing efficient frameworks for MCS, in order to give a homogeneity and, ultimately, interoperability to MCS applications [130]. An example is McSense [37][207], a centralized MCS system that exploits monetary rewards to make the backend entity assign sensing tasks to the users. In such framework, the geolocalized task is automatically assigned to a minimal subset of users on top of their profile and relevance. Different solutions exist for another paradigm in which requesting users generate tasks and responding user can accept and execute them; one of the most famous implementation of such paradigm is given by Medusa [171], which provides an ad-hoc programming language for non-expert users for the task generation. The authors in [194] propose CAROMM, an MCS framework implementing energy efficiency through edge deduplication, which aggregates sensed data with social media information and uses online mining to reduce the amount of data redundancy. MOSDEN [97] proposes the collaborative reuse of sensor processing across several mobile apps and smartphones, alleviating the necessities for application-specific processing. Pick-a-Crowd [60] assigns MCS tasks to members of a crowd on top of their interests and skills. effSense [218] is a data uploading framework that, depending on the type of connection and battery conditions of each device, designs different uploading schemes. In [53] the authors design a oneM2M-based MCS framework for Smart Cities that is energy-aware, semantic-compliant and self-adaptive. in [99] the authors design a complete framework that has been deployed and tested in a campus. Frameworks are an important building block of the research in MCS, although MCS applications tend to be too diverse to be unified under a single architecture. Within the scope of this dissertation, we propose a framework for Smart Cities and environmental monitoring called SenSquare, focused on Collaborative IoT and MCS [149][148]. The MCS part is explained in detail in Section 5.1.

Applications

MCS is applied to a plethora of use cases, such as environmental monitoring [174], social trends detection [73] and traffic estimation [162]. In environmental monitoring the MCS paradigm finds its natural application, since sensing the

environment is a complex and challenging task to be performed by stand-alone devices and, often, requires a high number of sensing units. Moreover, since most devices natively support environmental sensors, leveraging the data coming from the crowd can provide a more complete view of the sensed environment. This is the example of SecondNose [119], which collects environmental data in order to infer a number of indices concerning air quality and pollution and sensitize the citizens. It is also integrated with specific portable multi-sensors in order to enlarge the number of detectable pollutants (such as benzene). Other applications make use of the microphone in order to keep track of the noise levels in different areas of a city, exploiting the concept of Mobile Learning (ML) [237], assigning the measuring task to a dedicated class of citizens (e.g. the bicycle couriers [102]), or focusing on a particular source of noise pollution (e.g. the traffic on highways [117]). Smart Cities are another area in which MCS is extensively applied. In fact, one of the most valuable application of MCS relies in Smart Cities, where phenomena are mostly caused by social contexts and have collective consequences. Citizens are more pushed to participate when their contribution leads to the development of a service that they can make use of. Applications span over Smart Parking, as exemplified in Section 2.2.3, where users publish information about free parking spots in cities. PCS applications have been proposed, where users intentionally post an information about parking spots [42], alternatively it can happen indirectly, for which a plethora of user activity detection methods have been proposed, such as measurements based on gyroscopes and accelerometers [186], sonars [122], magnetometers [215] and on-board cameras [70]. Crowdsensed user's transportation mode detection in smart cities is also well addressed in literature [18] [19], where the authors base their analysis solely on the smartphones' sensors. Public transportation is another area highly targeted by MCS application, such as in [131], where city-scale data is fused with MCS data to infer information about individuals. Moreover, MCS has been used in the fields of the emergency management [132] as well as the city mapping [35], for which many MCS applications make use of the GPS geo-fencing [187] in order to limit zones of interests [36]. Even the healthcare system has found advantages in MCS, such as in [44] or in [169], in which patients are collecting measurements about their daily activities in order to provide a significant dataset to be analyzed by doctors. Another application was established in Singapore for determining the usage patterns of air conditioning using data coming from a major initiative called the National Science Experiment[36] [75]. A plethora of applications making use of the smartphones' sensors have been proposed in literature. Given the current status of Collaborative IoT and MCS, we observed how most of the application relying upon such concepts are commonly driven by specific campaigns that focus on a

[36] `http://nse.sg/`

tiny portion of the aspects they can cover. To our knowledge, our work is the first attempt proposing a global platform able to cope with heterogeneous data coming from different available sources for environmental monitoring.

2.4.2 Future Research Landscape of MCS

MCS is a rising field of research and many solutions for a variety of issues concerning different application domains have been proposed at an exponentially growing rate over the last 5-6 years. Even though many of such issues have been tackled extensively, some of them can be considered as open, since no global consensus has been reached. In this section we enlist such open problems and analyze where major efforts should be focused in the current research trends.

Semantics and Interoperability

Ontologies and semantic models have been used for the past years to enable interoperability among different domains and applications. On the other hand, MCS applications are fundamentally correlated with each other, however, there is little effort in making their entities unified. A semantic categorization has been defined within the scope of WSN, namely Semantic Sensor Network (SSN) [46], which provides ontological homogeneity to entities such as sensors, actuators and data streams. In spite of this, SSN is currently not expressive enough to support mobility concepts in relation with MCS, making it challenging to be used. Furthermore, the heterogeneity of mobile devices makes imperative to refer to a global data structure for the device's capabilities and equipment. Such data structure is, at the moment, absent. Furthermore, the amount of data generated by IoT and MCS application is currently not exploited, as many heterogeneous systems behave as closed islands and control both data gathering and aggregation. Such problem has been recently addressed for common IoT application, however, very little effort has been undertaken within the scope of MCS. Works in [202] and [149] represent the first notable efforts in such sense. In particular, they rely on the availability of a large amount of environmental data coming from crowdsensing campaigns and open data repositories in order to avoid requesting data that is already in provided by another entity. Nevertheless, challenges about data homogenization and quality are yet to be undertaken.

Contextualization

Most of the applications that perform MCS need to process the data in real-time to be able to deliver the service on-time. On the other hand, one of the main

characteristics of the data collected from MCS applications is the ability to capture the contexts around the observations. These contexts can help to describe the sensing by adding more information about the data [230]. Furthermore contextualization of the data can improve processing of the data by considering only the data that are relevant to the particular situation [229]. As a result, contextualization can potentially improve data processing both in terms of efficiency and effectiveness. Contextualization of the data has a great potential to improve the processing time of the data in real-time applications. Contextualization has been tackled extensively in the field of SOA architectures for common IoT environments, an example is given by the approach for service coordination in [43]. The authors proposed a Situation Event Definition Language which is capable of defining situational event from basic events or other complex events via calculation or combination. Further, based on the proposed language, they presented an event detection algorithm and an architecture where event occurrence triggers a set of services according to a publish/subscribe mechanism. Although this and other similar studies consider contexts around data collection for data processing, there is a lack of platforms that take into account MCS and its characteristics while collecting the contexts around the observation.

Participation Rate

We describe the participation rate as a combination of *data quality* and *fairness*. These two parameters will directly impact the participation rate of participants in the MCS and that will underpin MCS generating sparse or dense data.

Data Quality: The evaluation of information quality in MCS has been covered extensively in literature, as it is a crucial point. Assessing the credibility of data is always subject to a certain degree of imprecision, due to the sparsity and the heterogeneity of the devices involved in the tasks and of the participants themselves. Without the goal of covering in detail such aspect, the interested reader is redirected to [182] for a deeper analysis). Within the scope of this dissertation, the vast majority of the recruitment frameworks, which will be covered in more detail in Section 2.5.4, are focusing solely on data quantity and often leave the concept of data quality apart, vice versa, solutions based on data quality do not consider if the amount of data meets the requirements of the application. These two concepts cannot be considered separately, which is why data quality needs to be taken into account at the time of design and not as a second overlaying step.

Fairness: Fairness is a key concept in MCS, since it preserves the users from dropping out of a campaign due to not being selected as contributors [94], i.e. an equal opportunity for all the participant. While this is in contrast with data trust-

worthiness, since the selection of only the most trusted participants is not compatible, it is an important metric addressed within the scope of incentive mechanisms. Despite this, it is seldom taken into account by solutions designed for dense MCS scenarios and the balance between the optimal data quality and an acceptable participant's fairness has yet to be undertaken.

Privacy

With the transition from WNSs to MCS ecosystems, privacy had become a crucial issue, since devices collect sensitive data of individuals and their disclosure can have serious implications [79]. The user's identity is typically recorded for logging reasons: to assess other parameters such as data credibility and user trustworthiness, as well as to enable participants-based push recruitment schemes, which would not be applicable otherwise. Many platforms also track GPS routes with a different degree of granularity, however, these information can lead to user profiling (an example could be the well-known framework CarTel [87]). Furthermore, some recruitment frameworks acquire logging data that can severely harm the user's privacy, like a record of the phone calls [239][224][223], which contributes to better design recruitment algorithms, however, in most cases, it would not be considered acceptable in terms of privacy. Many existing works in literature leveraged independently private information enclosure in order to cope with common de-anonymization oriented attacks (e.g. collusion and eavesdropping). Notable works are LOCATE [27], which distributes the users trajectories across all the participants to make them anonymous, PEPSI [54], which introduces an additional registration authority and obscures the sensitive data with an identity based encryption, and AnonySense [48], in which a task assignment language is designed using anonymization. More recent efforts take into account location privacy for MCS scenarios such as the framework in [210] and the technique outlined in [100], which shows that, with a great improvement in the entropy of the location of the user, a task-assigner application end up with negligible limitations.

2.5 The *"Curse of Sensing"*

A fundamental challenge faced by MCS application for IoT is the *Curse of Sensing*. Inspired by Big Data's *Curse of Data*, we define the *Curse of Sensing* as the inability of MCS applications to control sensing processes that may result in sparse or dense data. Sparse data may lead to insufficient information that will impact the ability to make data-driven decision and/or predictions; on the contrary, dense data could be influenced by Big Data's *Curse of Data* problem, i.e. too much data leading to insights that may not reflect the real-world state and/or can lead to inaccurate predictions. Most of the recent surveys in the area of MCS focus on various aspects of mobile crowdsensing including defining the concept of crowdsensing [64], inspecting new applications [134], privacy [214], cost and QoS [126], data quality and credibility [182], incentive techniques [241] etc., with little or no attention to the *Curse of Sensing*, that is impacted by issues such as participating rates and crowdsensing strategies [101]. However, there is a clear gap in the current literature in identifying techniques and challenges to address the *Curse of Sensing* problem. In summary, this section will provide the following contributions:

- A taxonomy of factors and objectives of MCS: in Sections 2.5.2 and 2.5.3 we identify the factors and objectives that will have a direct impact on producing sparse or dense data in MCS applications and, therefore, have an impact on the *Curse of Sensing* problem, defined in Section 2.5.1.
- Survey of techniques for coping with sparse and dense data in MCS: in Section 2.5.4 we provide a review of current approaches that have been carried out in recent years to address the *Curse of Sensing* problem. Here, we compare the techniques identified from the literature based on the previously established taxonomy.
- Challenges in coping with sparse and dense data: in Section 2.5.5 we conclude by discussing how the key challenges that are yet to be addressed in the literature, presented in Section 2.4.2, will aid or impact the *Curse of Sensing* problem in MCS applications.

2.5.1 Definition and Motivating Scenario

Depending on many factors, such as the type of incentive used and the interest of individual(s)/crowd in achieving a common goal, the number of participants of an MCS application may vary. This is largely attributed to the inherit nature of MCS i.e. based on crowd participation and contribution that cannot be controlled or planned. This also has direct impact on the amount of data generated by the

MCS application as they may also vary depending on the aforementioned factor. Hence, most MCS application from time to time will need to cope with sparse or dense data which we refer to as *"Curse of Sensing"* problem in MCS domain. Formally, we define the *"Curse of Sensing"* problem as the "inability of MCS applications to directly control sensing processes and their propensity to generate sparse or dense data, which can lead to significant gaps in the extracted knowledge".

We refer to data as *sparse* if the data collected via the MCS application is not sufficient to meet the MCS task requirements, due to lack of available data in the geographical areas covered by the MCS application and/or periods of time with no corresponding data samples. Conversely, we refer to data as *dense* when large amounts of data is contributed by the MCS participants, resulting in significantly more processing overheads due to increased volume of data and wastage of resources, such as the battery of the devices and the budget of the campaign issuer. In either of these cases (sparse and dense), the amount of data can lead to a fuzzy and inaccurate solution provided by the MCS application that provides a blurred view of the observed phenomenon.

An example on how the problem of *Curse of Sensing* can have a significant impact on the MCS application can be illustrated by the Smart Parking application presented in Section 2.2.3. Let us imagine that users report available parking spots within the parking areas through a Smart Parking application (either actively or opportunistically). If this crowdsensed data is sparse, the smart parking application may fail to make accurate and timely recommendation to users, due to lack of data that can cause areas with parking spots to be uncovered or obsolete data that does not reflect the real situation. Conversely, if too much data, due to no control in asking for participation, is produced via Crowdsensing, the application is then responsible for filtering the irrelevant data, aggregate multiple and potentially contrasting observations about the same event, process the data (which due to its volume and velocity can be characterized as Big Data and hence require significant resources for processing) and deliver timely and accurate recommendations to users. In either of these scenarios, the utility of the MCS application is thrown into serious doubt.

2.5.2 Taxonomy of Factors Influencing Sparse and Dense Data in MCS

In this section we enlist all the factors that can have an impact on MCS, resulting in either sparse or dense data.

Task

Building on the general definition of Task presented in 2.2.2, in this section we refer to the concept of task assignment and completion. A task can be *pull-based*, in which participants decide independently which tasks to participate in, or *push-based*, in which participants are assigned by the campaign owner one or more tasks to accomplish. Push-based and pull-based paradigms have a strong impact on the generation of sparse and dense data. In particular, pull-based tasks yield more decisions to the participants, thus are likely to generate more decentralized data, since the central entity has less control on them.

Location

In MCS paradigms location is always either *detected*, i.e. reported by the user's device with every observation (or group of observations), or *tracked*, i.e. reported continuously by the device, even without the occurrence of an observation. A tracked location helps more in designing a scheduling strategy for the individual, however, it can easily affect privacy, while a detected location may not be sufficient in order to avoid sparse and dense environments. Furthermore, the notion of sparse and dense depends strongly on the concept of location granularity, since information may be sparse in a fine grained location-based system, while it may be at the same time dense for a coarse grained location-based system.

Prediction

Prediction is a key aspect of MCS and has been widely used to solve many problems, including task allocation and tracking participant trajectories while applying prediction techniques on the actual captured data through the campaign. Prediction may be *macroscopic*, if it is performed over a crowd or a location and does not take into account the individual actions (e.g. the approximate number of users expected to be in a certain area at a given time frame), or *microscopic*, if it is performed over each participant (e.g. inferring the position of each user depending on his or her past and current trajectory).

Sampling

We refer to the definition of sampling and its division into sensing cycle and transmission cycle in 2.2.2. Sensing cycle can take place at a defined time frequency or scheduling without any control by any other component. In such case the sensing cycle is defined as *continuous*. Conversely, the measurement can be triggered by another entity and, in particular, by an event that occurs whenever a defined condition is met (e.g. the participant is in a certain location), in such case,

the sensing cycle is defined as *event-driven*. Moreover, the measurement can be directly triggered by the participant itself while performing an assigned task (e.g. by taking a picture), in such case the sensing cycle is *user-triggered*. Furthermore, transmission cycle can be categorized on top of the rate at which observations are reported to the central entity. Some applications require the measurement to be reported as soon as the data acquisition takes place and with no time window allowance permitted. We define them as *real-time*, whereas others, permitting the participant to upload data with some degree of intentional delay, are defined as *delay tolerant*. The way in which participants sample data has a deep impact on the generation of sparse and dense information. For instance, user-triggered sampling is likely to be less controllable and more prone to produce sparse information, whereas continuous sampling, if not tuned properly, can easily lead to dense information. The rate at which data is sent to the global server is another feature that can cause sparse or dense information. In fact, delay tolerant applications, according to what we defined in Section 2.2.2, may not reflect the current environment situation, thus leading more easily to decentralized information.

Figure 2.9 provides a taxonomy of the aforementioned factors that influence the generation of sparse or dense data in MCS.

2.5.3 Objectives for MCS Applications

In this section, we identify the main objectives that need to be addressed by MCS applications and have a direct impact on the generation of sparse and dense data. They normally result in a multi-objective optimization problem as one or more of these objectives may need to be considered for different applications. As stated, MCS, while introducing a fairly inexpensive way to obtain data, is limited by several aspects:

1. Participants are running applications on battery-powered devices.
2. The location of the sensing devices is mainly uncontrollable.
3. Each observation reported by a participant is meant to be worth some form of reward.
4. Participants excluded from the campaign are unlikely to participate again.
5. Participants are not meant to be experts and report data correctly and the data credibility may be not easy to assess due to the lack of ground truth.
6. Data collection from personal devices presents a high risk of sensitive data disclosure.

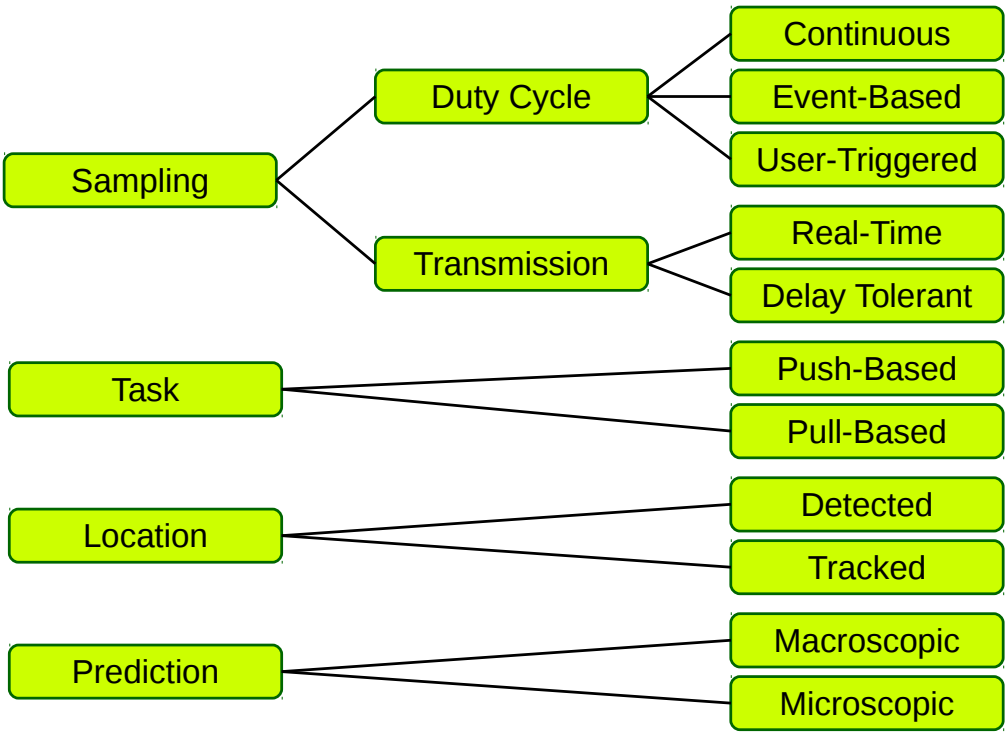


Figure 2.9: A taxonomy of different factors influencing sparse or dense data in MCS [154].

In this section we focus on a subset of these aspects that have been tackled in the literature, namely *energy*, *coverage*, *budget* and *privacy*. We define them as objectives and we break them into classes referring to the solutions present in literature. A graphical interpretation of the objectives is represented in Figure 2.10. Depending on the MCS application, each objective may have different priority. For instance, a campaign may have limited budget to carry out its task, thus it would try to achieve as much coverage as possible, while trading it off with the energy consumption of each device. Conversely, there might be the need of least coverage for which the campaign tries to allocate tasks in order to minimize the budget used. Designing a trade-off among such criteria is not an easy task, and current solutions tend to prioritize some of them while penalizing, or not considering at all, the others.

Energy

Battery depletion of end devices has been repeatedly addressed in MCS applications as a crucial issue. In general, end devices consume energy during the

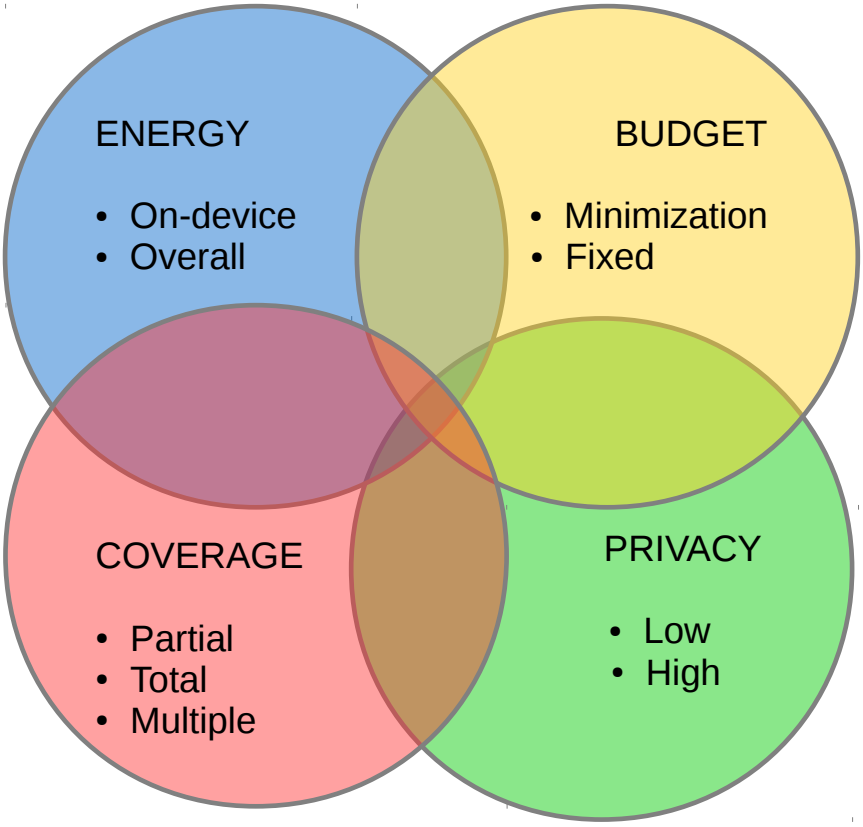


Figure 2.10: Objectives upon which MCS applications deal with sparse and dense information [154].

activities of sensing, computation and transmission, all of them part of the crowd-sensing life cycle. However, since data transfer consumes the highest amount energy by at least one order of magnitude [190], most approaches are focused on energy saving methods in transmission. Nonetheless, in contrast to what some solutions state, if mobile devices have a sufficient battery capacity (such as smart phones), then sensing and transmitting sensed data might be nearly negligible compared to all the other applications that run at the same time on such device. Hence, depending on the MCS application, the issue of battery depletion can have a fluctuating weight. On top of such premises, the approaches taken in literature to tackle this problem are diverse. We group them into the following categories: *on-device*, when the energy saving mechanism is operated by the end device (e.g. by piggybacking the transmission over cellular connections) and *overall*, when the recruitment strategy is oriented to the optimization of the overall battery lifetime of the crowd.

Coverage

The problem of capturing relevant and timely information in all the monitored areas depends strongly on the coverage of the MCS application. Solutions adopt different minimal requirement about coverage, which can either deal with sparse or dense scenarios. In particular, applications may need to limit the number of observations due to data redundancy or cost and rather infer the missing data points through interpolation-based methods. In any case, a least measure of coverage of the monitored areas is required. Some applications require a *total* coverage, which means that every monitored area must be observed by a participant at each time cycle of each task, while others might require (or sometimes force) a *partial* coverage of the monitored areas, enough to run an inference algorithm. Partial coverage methods are the only ones that can cope with sparse scenarios, whereas, in such cases, total coverage methods need to find other ways to recruit more participants through incentives. A third paradigm is given by the *multiple* coverage, which requires a minimum number k of observations for each area to be monitored.

Budget

Each MCS campaign is required to provide incentives to its participants. Whether they are monetary or not, we can assume the campaign issuer to rely on a certain budget, that can be expressed in units. Recruitment strategies, given they take into account such budget, focus either on the *minimization* of the expenses, or on the allocation of a certain *fixed* amount to be distributed among the participants, either evenly or as a reward for their performance.

Privacy

User privacy is a rather delicate topic in MCS. For example, in order to perform an accurate prediction, especially microscopic, many solutions track the user's features to different levels of extents. Nevertheless, privacy has to be taken into account at the time of designing MCS application as it can lead to sparse or dense data generated by corresponding campaigns. In fact, the number of participant can have a strong impact on privacy requirements and fulfillment (e.g. user's habits in sparse scenarios are likely to be tracked singularly, while, in dense scenarios, there might be the need of further tracking policies for participants selection) and it can be seen as an obstacle in granting, for instance, high coverage. For such reasons, it is not a surprise that many MCS frameworks do not take privacy into account.

Energy, coverage, budget and privacy play a fundamental role in MCS applications and has been considered in most papers in the literature and presented throughout this section at various degrees of importance. They are addressed either as goals or constraints of the MCS application, as a strong correlation among them has been observed: the more participants are recruited, the more coverage is assured; however, it requires more budget and can easily lead to dense data.

2.5.4 Current State-Of-The-Art in Addressing the *Curse of Sensing* Problem

In this section we focus on the diverse solutions and systems designed to cope with sparse and dense data in MCS, i.e. to address the *Curse of Sensing* problem. We describe each method in correlation with the factors as identified in the taxonomy in Section 2.5.2 and objectives as outlined in Section 2.5.3.

Compressive Sensing

Compressive sensing techniques for MCS scenarios leverage the fact that, due to spatial and temporal correlation of data over an area, there is less need to observe physically the whole ambient. Rather, in order to lower the amount of data to be acquired by mobile sensors and mitigate the problem of dense data, only a selected part of the environment has to be covered by observations, while the rest would be inferred. Such concept is based on a technique that has been introduced for problems in signal processing in which a signal is reconstructed using an amount of samples by far lower than the one required by the Shannon-Nyquist sampling theorem. This relies on the assumption that the signal is sparse (i.e. it has only a few non-zero entries) and reduces the problem to solving undetermined linear systems [33]. This technique can also be applied on sensed data samples, assuming a spatial and temporal correlation and redundancy as in environmental data such as temperature [133]. Such technique has been applied to MCS scenarios, as in [227], where the author presents Cost-Aware Compressive Sensing (CACS); a framework that leverages compressive sensing through randomized sampling. In CACS, devices exploit their sensing opportunities and independently calculate the probability of transmission using a cost function that takes into account the device energy and transmission costs combined with the need of real-time data. The web server component of CACS is committed to reply to each observation with a set of parameters that ensure a randomized data collection rate and fairness via the Distributed Weighted Sampling (DWS) or Pairwise Sampling (PW) heuristics. A slightly different approach is taken by Compressive CrowdSensing Task Allocation (CCS-TA) [216], in which direct task assignment is performed against the users. In such case, the whole environment is assumed

as a matrix – the case study is the city of Beijing – and aims to reduce the number of tasks in order to obtain a reasonably accurate sensing value for each sub-area, either through direct sensing or deduction. In order to do so, a sensing task is iteratively assigned to a participant in the matrix, then they assess the system satisfaction within the matrix through Bayesian inference and design a stopping criterion of top of it in order to decide if they should query someone else. As soon as the predicted data accuracy is higher than a predefined threshold, the task assignment stops. In such case the task assignment is performed sequentially, allowing only an applicability over sufficiently long cycles and assuming a wide user participation. It is also relevant the work in [217], in which the novel concept of "Sparse Mobile Crowdsensing" is defined. It is the paradigm used in compressive sensing-based solutions, that is, limiting the number of areas from which data is sampled in order to indirectly limit the amount of budget to be paid and the overall energy used by the end devices. Such solutions are typically divided in three phases: the reconstruction phase, in which missing data values are inferred through solving a matrix completion problem, optimal task allocation, in which the cells that experienced the highest variance are selected as to be sensed, and data accuracy estimation, which is carried out through the leave-one-out cross-validation in evaluation phase and inference subsequently.

Piggyback Crowdsensing

A number of works proposed over the last years have been leveraging the concept of "Piggyback Crowdsensing" (PCS), introduced officially in [110]. The authors leverage the possibility to piggyback sensing and participation tasks over the "Smartphone App Opportunities", i.e. those occasions in which the device performs primary actions like phone calls or accesses an application. In such cases, the energy required to sense – the authors use GPS, accelerometer, camera and microphone as examples – is significantly reduced, since the sensors no longer need to be woken up from an idle state. This applies also for transmitting chunks of data, in fact a data transfer consumes definitely more energy if performed with pauses. The work is supported by a prediction model that guides the device towards when to perform one of the scheduled tasks based on the user's habits. As a result of decreasing the energy consumption, PCS can improve the participation rates which leads to higher task coverage (i.e. avoid sparse data). The approach has been proposed in parallel with effSense [218], a framework that supports delay tolerant crowdsensing tasks and predicts on the edge when to upload the next chunk of data based on both user habits and the encounter of a non-cellular hotspot (being it a WiFi access point or a bluetooth-enabled user that could serve better the upload task). The approach is oriented to individual energy saving and cellular data cost lowering, however, both these approaches do not consider neither

incentives nor spatial coverage. Indeed, the coverage problem had been tackled in [239], where the authors design an algorithm that uses the PCS paradigm and predicts in a centralized way both the position and the likelihood of 3G voice calls for the users based on historical records in order to minimize the number of participants selected under coverage constraint. They chose to assign a spatial area to each cellular base station in order to be compliant with the dataset used for evaluation. A similar, more elaborated solution is EMC^3 [224], where the authors build complex prediction models on the 3G voice calls of the users and their mobility in order to design a sub-optimal task assignment policy that ensures coverage. This is subject to the concept of parallel transfer, used both for task assignment and data uploading. A different approach had been taken in [223], where two different kinds of problems are presented, both subject to a k-coverage parameter, meaning that every zone has to be covered k times. If the k-coverage is the constraint (every zone has to be covered *at least* k times), then the minimization object is the budget, whereas, if the budget is the constraint, then the maximization object is the coverage.

Edge Deduplication

The concept known as deduplication was one of the first introduced within the scope of mobile sensing. Most of the computation in this paradigm takes place at the client side, in order to reduce the energy consumption and the data redundancy as much as possible. Related works show that the size of processed data is considerably less than the raw data. Therefore, this approach not only reduces the energy consumption related to data transmission, but also avoids creation of dense data by reducing the amount of data transmitted to the MCS web server. Edge Deduplication has been revised as an MCS efficient solution in [126], where it is described as an edge computation technique in order to avoid duplication in acquired data. Sensing is performed continuously, however the data is not streamed to the upper layer unless it displays significant variations. This requires obviously online learning strategies, such as the one proposed in [194], one of the first works in this field. The idea is to perform an online mining algorithm based on clustering techniques in order to upload data only when there is a subtle change in the sensed observations, avoiding in this way to upload redundant data and to waste energy. Such method has been used previously in other mobile edge-based sensor data processing, such as in ACQUA [142]. Different is the approach adopted in [157], in which data deduplication is performed at the fog layer. The authors design an algorithm to eliminate similar data chunks while still keeping track of the users' contribution and focus more on the privacy and coverage issues; however, energy consumption is left apart.

Distributed Heuristics

When dealing with real scenarios, given that the control over the fleet of users is limited and many information (such as position and past records) may not be available, there is often a need for approximation solutions that tend to an optimum value by taking data density into account. The work in [192] designs a scheduling algorithm given the user's trajectories in advance and gives an heuristic to estimate such trajectories under realistic assumptions. In such work, the authors do not divide the area to be sensed in sub-areas, instead, they consider it to be limited to the roads, which are assumed to be narrow chains (i.e. having no thickness). The assumption is that mobile users are only moving along the roads and they have a sensing radius within which there is no need to sense twice. On top of these premises a scheduling algorithm is designed. Conversely, our work in [147], which is also presented in this dissertation in Section 5.2, designs a distributed algorithm for tuning the number of observations within a defined area to achieve reasonable data density level while having acceptable task coverage. The work is built on top of few assumptions, since the cloud entity is supposed to have no control over the users and no tracking whatsoever. Under the assumption of multiple coverage, users are only given a "satisfaction index" by the server, which states how much the coverage is satisfied, and tune their probability of upload upon each cycle using an inverse proportionality. A very similar algorithm is presented in [34], in which the needs of the platform are broadcasted periodically to the devices, whcih compute locally their utility in contributing based on several parameters, among which the battery consumption, the contextualization and their past contributions. A performance comparison between these two works has been recently performed in [211]. Other notable recent works base their heuristic on probability for selection over a crowd of sensor nodes [88] [238].

Optimization-based

Achieving optimality in distributing crowdsensing tasks often implies finding a balance among multiple metrics, as outlined in Section 2.5.3. Many algorithms presented in the literature consider one of such metrics to be the objective on which to design an optimization algorithm and, depending on the functional characteristics of the problem, select a number of constraint as input of the algorithm. One of the first recruitment algorithms in such fashion is the work in [180], which considers explicitly a participatory scenario in which users contribute in the form of pictures and, given the historical data about position, transportation mode and credibility, an optimization algorithm that maximizes the coverage in presence of a budget constraint is proposed. A more recent algorithm has been proposed in [240], in which a similar goal is given. The proposed scheme focuses on op-

timizing the coverage in a Point-of-Interest monitoring scenario given a budget constraint. The scenario is further complicated by the network deployment, since ad-hoc point-to-point communication technologies are implied (such as Bluetooth and WiFi direct).

Context and Logical Dependencies

Sometimes recruitment strategies are purely mathematic and blind to other contextual factors that can bring benefit to the selection. For such reason, some solutions are designed to bring the contextually best participant into the task leveraging the set of semantic information that could emerge from participants and tasks association at different levels. Works in this area try to achieve acceptable coverage (i.e. avoid sparse data) by selecting the minimum number of eligible users (i.e. avoid dense data) who are most likely able to participate in a given task. An example of this approach has been proposed in [86], in which an inter-data dependency is used as a key paradigm. The authors inspect as a case study a disaster recovery scenario, in which communication bandwidth (and, therefore, overall energy consumption) are considered as the parameters to minimize. Thus, sensing occurs continuously at the edges, however, data transfer is triggered by server queries that link concepts through logical clauses and data relationships in order to fetch only the required data. Another work tackling context as a key parameter for MCS campaigns is CATA [77]. According to such solution, in order to optimize the energy consumption of end devices, the authors design an algorithm that assigns opportunistic tasks only to a subset of users for which the context correspond to the one requested. The context is a set of meta information (personal data about the owner and technical data about the device as well as data about the activity) and can be compared to other contexts using a similarity function that returns a score. Such score is the basis upon which the participants are chosen. The work in [143] presents CloQue, a context-driver MCS query engine that forwards queries to the participants. The novelty here resides in the context evaluation and prediction: the context of each participant is assessed through a DNF-composed predicate that establishes which are the true condisitons of the context, then, all the predicates are evaluated in a specific order, on top of which one is expected to cover the most cases, resolves the minimum amount of uncertainty about all the other queries and incurs less energy to evaluate.

Table 2.5 summarizes and compares the papers we reviewed from literature and discussed in this section against the factors and objectives of MCS application that influence sparse and dense data.

	Sampling	Task	Location	Prediction	Energy	Coverage	Budget	Privacy	Type
[227]	continuous/both	pull	detected	-	on-device	partial	-	-	Opp.
[216]	event-based/real-time	push	tracked	-	overall	partial	minimization	-	Opp.
[217]	event-based/real-time	push	tracked	-	overall	partial	minimization	-	Opp.
[110]	event-based/delay tolerant	pull	detected	microscopic	on-device	-	-	low	Par.
[218]	event-based/delay tolerant	pull	tracked	microscopic	on-device	-	-	-	Par.
[239]	event-based/delay tolerant	push	tracked	macroscopic	on-device	total	minimization	-	Opp.
[224]	event-based/delay tolerant	push	tracked	microscopic	on-device	total	minimization	-	Opp.
[223]	event-based/delay tolerant	push	tracked	microscopic	on-device	multiple	min./fixed	-	Opp.
[194]	continuous/real-time	push	detected	-	on-device	-	-	-	Opp.
[192]	continuous/real-time	push	tracked	microscopic	overall	total	minimization	-	Opp.
[157]	event-based/delay tolerant	push	detected	-	-	total	-	high	Opp.
[147]	continuous/real-time	pull	detected	macroscopic	overall	multiple	-	high	Opp.
[34]	continuous/real-time	pull	detected	macroscopic	overall	multiple	-	high	Opp.
[180]	user-triggered/delay tolerant	push	detected	microscopic	-	partial	fixed	low	Par.
[240]	user-triggered/delay tolerant	push	detected	microscopic	-	partial	fixed	-	Par.
[86]	continuous/delay tolerant	push	detected	-	overall	partial	-	-	Par.
[77]	event-based/delay tolerant	push	detected	-	overall	total	-	high	Opp.
[143]	event-based/delay tolerant	push	detected	macroscopic	overall	partial	-	high	Opp.

Table 2.5: Table with all the State-of-the-Art solutions that deal with sparse and dense data, for which the relation with each factor and objective is stated.

2.5.5 Discussion and Challenges

Designing a solution to develop MCS applications is a challenging task especially when MCS applications need to cope with dense and/or sparse data over which it has no direct control. In Section 2.5.4 we discussed the approaches that underpin the current research landscape and their relation with the factors and objectives of MCS described in Sections 2.5.2 and 2.5.3, which are usually considered partially in each solution. In Section 2.4.2 we also presented the future research landscape of MCS, in particular the aspects that have not yet completely addressed. In this section we present a discussion on how such aspects would play an important role in addressing the *Curse of Sensing* problem.

As it is discussed earlier, privacy is one of the objectives and research challenges in MCS. Although privacy is part of the design criteria in Section 2.5.3, it is pointed out as a challenge in Section 2.4.2, and especially within the scope of sparse and dense crowdsensing scenarios. First, privacy has a different outcome depending on the type of scenario, second, we have observed that it is often treated separately from the other design criteria, at the same time we consider the inclusion of user privacy as of paramount importance in a solution design. Recently there have been several studies proposing privacy preserving approaches for MCS that aims to preserve the privacy of the users. However, they are not considering the fact that dense and sparse situations may have different privacy requirements. For example, when we are dealing with sparse situation the number of users are limited so it would be easier to reveal more information about the particular users who are providing the data. It is worth to mention that there are several studies in the literature that can potentially tackle privacy concerns in related areas dynamically and by considering sparse and dense situation/contexts as the input to adapt the privacy pervasive algorithm [231]. However, there is no particular study that considers such algorithms in MCS.

Fairness is another aspect that can be affected by the dense and sparse data. When the number of participants providing data is not enough or is limited, majority/all of the participants are required to contribute and provide data. However, in dense situations distribution of the tasks in a way to fairly utilize the resources of the participants providing the data can be challenging.

Data quality can be also impacted by dense and sparse situations equally and hence requires development of techniques that consider suitable trade-off depending on the MCS application. For instance, when we have more data through large contributions from participants, there is a potential for quality of data becoming fuzzy (as in the Curse of Data). Similarly, when only limited data is available due to limited participation, the quality of data can be low [23]. Moreover, one of the main challenges in MCS is the fact that we do not know how much data is sufficient data.

Contextualization is a way to improve processing of the data by taking into account the surrounding contexts. Contextualization is more effective when data volume is large as the filtering/aggregating of relevant data using relevant contexts results in better performance and accuracy. On the other hand, more dense data situations lead to collect more contexts that require more effective and high-performance techniques (including extrapolation and prediction). Semantics and interoperability mechanisms can help with integration and contextualization of the data. However, in more dense situations by increasing the volume, variety and, velocity of the data, interoperability of the data becomes more challenging.

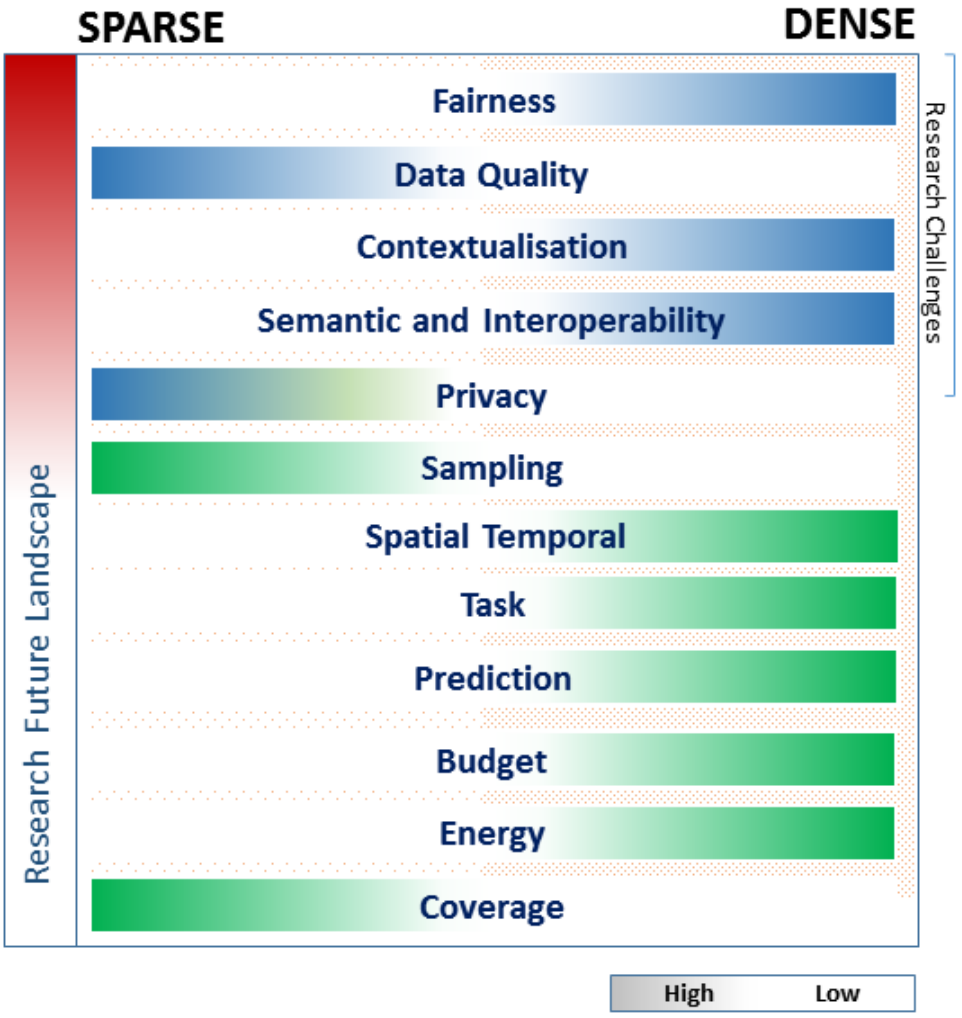


Figure 2.11: Challenges, factors and objectives with respect to sparse and dense data in MCS [154].

Objectives and factors discussed in Sections 2.5.2 and 2.5.3 also influence dense and sparse data situations in MCS. For example, sampling should be higher in sparse conditions to reduce the probability of missing data; on the other hand, proper prediction of sparse and dense situations can help the system to manage task distribution, budget and energy consumption in spatio-temporal environments in such a way to reduce the risk of not meeting the coverage requirements in sparse situations. Figure 2.11 provides a summary of this discussion where we illustrate 1) how each factor, objective influences the generation of dense or sparse data in MCS and 2) the future research landscape that indicates areas that have been well addressed in the literature (towards the bottom half of Figure 2.11) and areas that needs further investigation (towards the upper half of Figure 2.11).

Chapter 3

Motivations and Research Question

In this chapter we will outline the major contributions of the present dissertation, with respect to the current State-of-the-Art on Chapter 2 and as a consequence of the considerations that we introduced in Chapter 1. With this short chapter we aim to bind the two parts of the dissertation together – Part I is about the current research and industry landscape and Part II is about what is being presented as a contribution to the State-of-the-Art – and, in particular, to localize where our research work fits into the big picture, to extract the challenges on which we focused and, most importantly, why we think it is important and worth it. We will further channel our motivations in a research question we aim to answer through our contributions. Certainly, there are still aspects of this research that can be enhanced and others that still need to be addressed, nevertheless, we claim that the work presented in this thesis is a novel and important contribution to the field of the Collaborative IoT and its derivations.

The spread of the Internet of Things (IoT), being in its industrial, research or fan-made form, has been impressive over the last 15 years. The research in this direction has been largely fostered, also because the areas of research, as well as the use cases, are by now a lot and have different requirements. It is indeed impossible to give a unique definition of how an IoT ecosystem is ought to work and be structured. In fact, many of the use cases have orthogonal features and the areas of research pertaining the IoT are getting farther and farther from each other in dealing with such different sectors. This is evident, since nowadays many of the IoT-related works in literature highlight the growing number of connected devices and the amount of billions of dollars invested in the IoT market. Let us consider the amount of money and effort that has been put in M2M communication technologies for IoT in order to cope with the vast amount of use cases that the current demand has generated. Section 2.1 gives an idea of the plethora of technologies, standards and protocols that have been proposed and distributed in the market.

The outcome of this consideration, taking a look at the works that have been produced in different areas of IoT, is that this universe is currently moving at a pace that sometimes research and standardization efforts cannot keep up; as new needs come in, industrial ad-hoc efforts tend to come first. In fact, we are currently witnessing a set of "Intranets of Things" rather than a true Internet of Things [244], this is because current ecosystems tend to behave as closed islands with little or no interoperability between each other. This happens either because solutions need to be deployed "here and now", so there is more need for the least solution that solves the problem rather than something that can someday be useful to others, or else because widespread and different industrial solutions force the customer to stick to what has been envisioned by the manufacturer; therefore, interfacing with third parties entities is tough. Of course standardization efforts have been envisioned, especially from the architectural and semantic point of view. As a matter of fact, the literature is literally covered in proposals for new IoT frameworks, new IoT architectures, new IoT solutions that are expected to cover a plethora of possible use cases; however, many of them end up being yet another standard that a small amount of solutions adopt. Several of these standards are well written, however, the prosperity of a standard *"has not to be examined from the focus of whether the standard was written or even implemented (the usual metric), but rather from the viewpoint of whether the participants achieved their goals from their participation in the standardization process"* [38]. In our case, successfulness of standardization efforts in some IoT fields – areas like architectures, application layer protocols, services and semantics suffer the most from lack of standards – are thrown into doubt, because, if a political interest in a standard becomes too large, the various parties have too much at stake in their own vested interests to be flexible enough to accommodate a unified view. This, together with the pace at which the field is evolving, in many cases inhibits the institution of standard solutions or makes them adapt to the current practice, sometimes including technical mistakes or breaking the user base.

In Section 2.2 we introduced the concept of CAP for IoT scenario, a revolutionary way to think about IoT, to sense and to actuate. We specifically talked about the changes that cooperation and a wise use of a collective effort would bring to legacy IoT ecosystems. For some of the use cases (i.e. monitoring and the use of common resources in the Smart City) such paradigms have demonstrated to be a key enabler to a whole new level of pervasiveness of any application as well as, last but not least, a significant reduction in the costs. In fact what was formerly required (e.g. the deployment of a consistent WSN) is, for some applications, no longer necessary. Clearly, this paradigm is characterized by different issues, in particular, issues that formerly were purely technological now are shifted to social: people need to be instructed, encouraged to participate, incentivized and

satisfied of the results. The greatest outcome the CAPs bring to the current trends in some IoT use cases – we focus primarily in Smart Cities and environmental monitoring – is given by two major concepts:

1. The data about a phenomenon of common interest is probably already in place. The more the interest the more likely is to find data that can describe it, being it institutional or crowdsourced. Section 2.3 is about the research efforts carried out so far in this topic.

2. If the data is not available, the devices capable of reporting observations about such phenomenon are probably already in place. MCS, addressed in Sections 2.4 and 2.5, is the biggest trend that exploits this concept, however, any collaborative solution could make the difference in such sense.

The points above are the major pillars on top of which we structure our research work. In particular, with the final goal of bringing interoperability to IoT ecosystems and bridging the gaps of data and device redundancy, we do not aim to propose yet another standard framework/architecture/paradigm, rather we try to make the most out of what is already in place. Such solution does bring a significant added value to the current market, especially through saving economical and human resources. In summary, in this dissertation we aim to answer to the following research question:

For many application scenarios (i.e. environmental monitoring, Smart Cities) either data is already collected somewhere or devices capable of providing such data are already deployed. How can we make the most out of it and deliver services that fit the needs of the citizens?

Clearly, this question is generic, thus we break it into the following, more specific, ones:

1. We need to gather data from accessible and heterogeneous sources about the same phenomena. How do we integrate it? Moreover, many sources (i.e. crowdsourced ones) expose open public datastreams that are manually annotated by users and, sometimes, not even significant. How do we classify them into significant categories of data, i.e. how do we provide an automatic annotation?

2. We need to gather data from the personal devices of citizens, how do we structure the interplay of different actors in such an ecosystem and how do we ensure a control over the data gathering in order to get the correct amount of data (i.e. not incurring in the *"Curse of Sensing"* problem [154]) without

harming citizens' privacy, causing battery depletion and still getting enough data?

3. Once data is gathered from heterogeneous sources, how do we offer it to users who would aggregate it in a service-oriented fashion?

Such questions are addressed throughout Part II of this dissertation, in which our contributions are detailed. Although the chapter structure has been already introduced in Chapter 1, we recall here what is presented in Part II in order to better convey our work in light of the State-of-the-Art, the motivations and the research questions we proposed. Question 1 is tackled in Chapter 4, in which a common practice for the integration of heterogeneous data sources is proposed (Section 4.1, based on the findings of our work in [146]), a classification algorithm for annotating heterogeneous and crowdsourced IoT data streams is outlined (Section 4.2, based on the algorithm that we proposed in [155]) and an annotation framework is presented briefly (Section 4.3). Question 2 is addressed in Chapter 5, in which we propose, under the form of an exemplified framework, a rule-based paradigm for MCS in Smart Cities and environmental monitoring, meant to be used by participants and stakeholders (Section 5.1, based on our proposal published in [149]) and a novel distributed and probabilistic algorithm for data collection control in urban OCS scenarios (Section 5.2, based on our algorithm proposed in [147]). Finally, we answer to question 3 through the presentation of SenSquare (Chapter 6, based on our journal article in [148]), a Web platform that we developed as a prototype in order to prove the benefit that the exploitation of data coming from the sources cited bring and to show how the primary data points can be aggregated to form complex and customized services for each member of the society.

User Willingness to Participate in CAPs

Users are a fundamental building block of CAPs. In order to better support our motivations in using CAPs in the real world, in this section we present a user survey that we conducted as part of [148] in order to assess whether users are willing to participate in a data gathering campaign hosting one or more crowdsensing elements in their everyday life. More in detail, we proposed to the users two different ways of participating. First, we proposed to the participants to host a small multisensor device, acting as a weather station, in an outer part of their house (e.g. the rooftop, the windowsill, the balcony or the garden, if present). The device is embedded in a small box not bigger than a 5 cm-sided cube and hosts sensors for measuring temperature, humidity, pressure and environmental noise level. To report the data, the weather station has to be connected to the Internet, therefore we asked the participant to share their Wi-Fi connection with such devices. As an

alternative, the device should report the data either through cellular connection or through some other long-range technology, e.g. LoRa, rather than Wi-Fi, resulting in an increased cost for the device distributor. We still consider such approach as crowdsensing, since, even though users do not materially own the appliance, they have complete control on it. Second, we asked the participants about their willingness to install a mobile application in their personal smartphones, which runs in background and reports periodically sensed data to a central entity. For both installments we assure that the participant will get a personal consumer application able to monitor the data that their device, either the smartphone or the weather station, is sending to the remote platform.

Table 3.1: Table showing collected demographic data about the interviewed people.

Male	65.3%
Female	34.7%
Age 18-25	31.7%
Age 26-35	53.5%
Age 36-45	5.9%
Age 46-55	4.0%
Age 56-65	5.0%
Living in city (or town) center	57.4%
Living in the first outskirts	16.8%
Living in the periphery	13.9%
Living in the countryside	11.9%
Ownership of the roof	48.5%
Ownership of the garden	44.6%

We surveyed personally more than 100 individuals, all of them living within the Italian region Emilia-Romagna, which counts 9 different provinces and around 348 different municipalities. As a matter of fact, we do not intend to provide statistics over the general user acceptance of a crowdsensing paradigm, rather we wanted to prove that the population of a sample region tends to be positive towards an environmental crowdsensing campaign in exchange of a small reward. The user survey involved human beings of different ages, both females and males, and it is organized in three main sections: (A) The first section is about some general questions about the users participating in the survey, which made possible to report the participants demographics, outlined in Table 3.1. Such questions concern their age, their gender, in which context they live and the ownership of outer parts of their house. (B) We asked the participant whether he or she is willing to install the above mentioned weather station in the outer parts of his or her house and

report the data to our central database. The user can answer with a plain "Yes", a plain "No" or "Yes (without sharing the Wi-Fi)". Should the user select the plain "Yes", the survey skips to section C, otherwise the user is offered to answer the same question including a monetary reward of 5 € per month and, would he or she answer neither in this case with a plain "Yes", the reward is increased to 10 € per month. After such proposal, regardless of the answer, the survey continues with the subsequent section. (C) Here, the user is asked about his or her willingness to install the previously mentioned mobile application; in this case the only possible answers are "Yes" and "No". The flow is similar to the previous one, thus, if the user is not willing to install it, a monthly reward of 5 € is proposed and it is increased to 10 € in case of another negative answer. Both in sections B and C, if the user states to be willing to participate only in exchange of a monetary reward, he or she is asked whether is willing to accept such reward supplied in the form of a discount or an offer regarding a particular class of stakeholders (e.g. a discount for train tickets or for mobile phone costs).

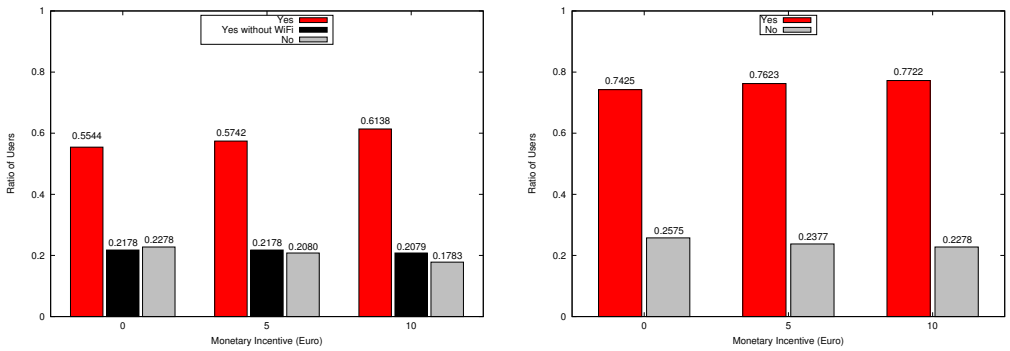


Figure 3.1: The results of the survey. Figure 3.1(a) shows the results concerning the weather station to be installed at home, while Figure 3.1(b) shows the results for the smartphone APP [148].

Focusing on Figure 3.1(a), we see that more than the half of the users is willing to install the reporting weather station and share the Wi-Fi connection. Interestingly, the number of users who either do not want to install it or simply to share the Wi-Fi is more or less constant regardless of the reward entity. Figure 3.1(a) reports the results of the survey concerning the installation of the mobile application on the participant's smartphone. The behavior is similar to the previous case, except for the fact that in this case we did not give the possibility to install the app without sharing a connection, which is needed in order to report the data. Nearly 80% of the participants replied that they would install the crowdsensing application on their smartphone. Again, similarly to the previous case, increasing the reward does not have the desired effect of increasing the number of positive

users as well. Instead, their percentage remains more or less constant regardless of the reward entity. An interesting aspect which emerges by comparing Figure 3.1(a) and Figure 3.1(b) is the fact that the users are more willing to install a mobile application reporting data rather than trusting the installation of a third-party device in their houses. Moreover, it is interesting to note that 73% of the participants who replied "No" or "Yes without WiFi" to the installation of the weather station for a 10 € per-month, answered positively to the installation of the mobile application question without reward.

Finally, the vast majority of users requiring a reward surprisingly accepted the alternative form of reward that we proposed. This suggests that a number of stakeholders, spanning from telecom companies to transportation companies to municipalities, are potentially motivated to take active part in such campaigns as distributors. Indeed, a vast amount of meaningful sensed data is a powerful source of knowledge that can help such stakeholders in planning and decision making. An example of such alternative form of reward has been adopted in the crowdsensing campaign issued by Doxa, an Italian institution for market researches. In particular, they proposed to the users to install a mobile application called *DoxaMeter*[1] which monitors the cellular connectivity and offers a monthly discount of 5 € for purchases on `amazon.it`.

[1] `http://doxameter.it/`

Part II

Contributions

Chapter 4

Integration of Heterogeneous Data Sources for Data Redundancy

This chapter outlines our contributions in the context of Open Data and how we used it to infer knowledge. Specifically, it discusses our general idea in using Open Data, featuring some examples of repositories that we took into account, an algorithm for classifying unannotated datastreams coming from Open Data repositories and a framework for automatically adding semantic connotation to homogenize the format of datastreams. The chapter is written on top of our works in [146], [150].

4.1 On the Integration of Heterogeneous Data Sources

With respect to Section 2.3, Open Data is a powerful source of information when data about an environment (typically the surroundings of the user) cannot be gathered by the end users themselves. In this section we report our early pioneering work [146] within the scope of Open Data for collaborative scenarios, with a focus on the integration of crowdsourced and heterogeneous Open Data. Specifically, in this work we describe in detail two crowdsourced Open Data repositories that we take as an example (Section 4.1.1), we inspect how they can be integrated using a common data structure (Section 4.1.2), we propose a sample architecture that leverages the usage of heterogeneous reliable and unreliable Open Data sources (Section 4.1.3) and, finally, we wrap up our findings (Section 4.1.4).

4.1.1 Open Data as a Source

In this section we outline two well-known sources that we considered in order to achieve an homogeneous data store. From both sources we extracted Open Data in the form of datastreams, in which the owner uploads measurements produced by its sensors. The information in this section has been extracted in 2016, in fact, the Open Data cloud of SparkFun has been dismissed in late 2017 and now it is not available anymore.

ThingSpeak

ThingSpeak[1], originally launched in 2010 by ioBridge, is an open source data platform and API for the IoT that allows the user to collect, store and analyze data. In more detail, it provides a personal cloud that users can deploy over their LAN and easily display the data produced by sensors using the straightforward API and front-end application of ThingSpeak. Data analysis and visualization have been possible due to the close relationship between ThingSpeak and Mathworks, Inc. since such functionalities are driven by the integrated MatLab support. Furthermore, what is more interesting to our research is that such a platform provides a global cloud[2] hosting millions of Open Data records, which are useful both to users who cannot deploy their own cloud and to consumers who need to infer information coming from the stored data. Data is organized in data channels, each of them belonging to a user, which are annotated with an absolute freedom of expression, meaning that they can be labeled with any name and do not need to stick to any constraint in terms of the amount and the quality of metadata. Data channels can be both private and public and provide also raw measurements encoded

[1] `https://thingspeak.com/`

[2] `https://thingspeak.com/channels/public/`

in XML, JSON or CSV and can be updated with new measurements every $15\ \mathrm{s}$, according to the ThingSpeak documentation page.

In the recent years, ThingSpeak had become very popular due to the rise of easily programmable IoT platforms such as Arduino, BeagleBone Black[3], ESP8266 and many others. As such devices become cheaper, getting started with them is easier. Nowadays, for instance, an ESP8266 is able to manage a sensor, get connected through WiFi, be programmed through the simple, C-like Arduino SDK and still cost less than 5 $ while its battery, if the duty cycle is light enough, is estimated to have a duration of around 7 years [59]. With a WiFi connection and an open platform such as ThingSpeak, a first home sensor network is very easy to bootstrap, since the device owner does not need to have the control on the cloud and, furthermore, the data produced by the sensor is easily displayable on the personal device of the end consumer, such as a Smartphone or similar.

Sparkfun

SparkFun Electronics, Inc.[4], founded in 2003 in Colorado, is a microcontroller seller and manufacturer, known for releasing all the circuits and products as open-source hardware. Along with the latter it also provides tutorials, examples and classes. For the purpose of the present work, SparkFun also hosted its own open source cloud of Open Data[5], on which the customers could test and upload the data collected by the embedded sensors. Users could push for free their data on such cloud in datastreams of $50\ \mathrm{MB}$ maximum size and with a maximum frequency of 100 pushes every 15 minutes. Unlike ThingSpeak, the location where the data comes from is specified at a coarse granularity since the name of the city is often obtainable, however the real GPS coordinates are never given. On the other hand, data coming from SparkFun could not be private and consumers could download datastream contents encoded in JSON, XML, CSV, MySQL, Atom and PostgreSQL.

4.1.2 Data Unification

In this section we point out the characteristics of the channels obtainable from our two sample Open Data clouds and how we aim to unify them onto a single data structure. We extracted from both the sources the whole repositories and parsed the JSON files in order to separate such data in components. Since the data structure does not force strong constraints, data is often incomplete in such a way that, in some cases, it is not usable, compliant with the issues introduced in

[3] `http://beagleboard.org/black`
[4] `https://www.sparkfun.com/`
[5] `https://data.sparkfun.com/`

Section 2.3.3. This happens when no location information is given, the channel name and the description is not understandable, the channel has not been recently updated and so on so forth. For both platforms, data channels are given together with metadata relative to the whole channel as well as one or more datastreams, each of which is represented by a chronologically ordered series of floating point values. A channel needs to be updated all at one, thus, every time the user updates a datastream adding a new value, it needs to do it for each datastream in the same channel through a single API call. This is typical when a IoT device is equipped with more than one sensor, or a sensor that performs more than one measurement at the same time (e.g. the popular DHT22 measures both temperature and relative humidity). Hereby the metadata that can be extracted from ThingSpeak or SparkFun data channels are enlisted:

- **Channel ID**: it is the unique ID of the channel. In ThingSpeak it is represented by an incremental number, which is assigned when the channel is created. In [146] we counted 28806 active and public channels with IDs spanning from 0 to 100172. In SparkFun the unique ID is given by a string of 20 random ASCII characters. In [146] we counted 3575 different SparkFun channels.

- **Channel name**: it present in both platforms and it is determined by the user with no constraint. It might carry or not useful information about the channel.

- **Geolocalization**: it is present in both platforms. In ThingSpeak not all the channels come with GPS data. Similarly, in SparkFun not all the channels are geolocalized, however, when they are, only the name of the city, or sometimes just the state or even just the country, is given. When extracting data in JSON from SparkFun, GPS coordinates are given, however we observed that such coordinates are probably obtained through some API converting the name of the city, since channels coming from the same city have the same GPS coordinates.

- **Tags**: are included in both platforms and represent the keywords that users assign to channels. They often help to infer useful information about the data.

- **Creation Timestamp**: it is included in all ThingSpeak and SparkFun channels as a metadata. It usually does not correspond to the timestamp relative to the first registered update, since each channel has a limited number of updates that can be permanently stored in the cloud, then the platform erases the oldest updates in excess. In SparkFun the limit is $50\ \mathrm{MB}$, while in ThingSpeak is 100 updates.

- **Last Update Timestamp**: it is included in all ThingSpeak channels as a metadata. In SparkFun is simply deducible from the timestamp of the last update in the channel, since the timestamp is implicitly included for each update.

- **Description**: it is a ThingSpeak metadata and its characterization is fully assigned to the user (who can also decide not to include it).

- **Elevation**: it is a ThingSpeak metadata and not always indicated, it represents the location of the source of the channel on the z axis, i.e. its vertical distance in meters from the sea level.

- **Metadata**: it is a non-mandatory ThingSpeak metadata which contains additional data for the channel in plain text. It is suitable for structured data such JSON and XML.

- **Url**: it is a non-mandatory ThingSpeak metadata indicating the address of the official web page of the channel.

- **Last Entry ID**: it is a ThingSpeak metadata, which points to the most recent update in all the datastreams of the channel, ordered using an incremental ID for each update.

Datastreams belonging to the same channel, both in ThingSpeak and SparkFun, also have their dedicated names, which represent the only way to discriminate which field registers which measurement. In both platforms each measurement comes together with an integrated timestamp. In [146], from each channel we extracted in particular the GPS position for a location analysis, finding that such position is indicated, with different degree of precision, in 6665 datastreams out of 32381 (nearly 21%). Thingspeak accounts for a total of nearly 14% of geolocalized datastreams, while Sparkfun takes the remaining 7%. However, as stated before, the GPS position provided by SparkFun indicates the center of the entity (the city, or the region) where the source is located. Without the location information, the channel is not useful, unless it can be inferred.

Therefore, a basic unification counting on an essential set of metadata is crucial, composing the minimum skeleton to which a datastream should be linked. For such purpose we aim to design an unique ID assignment policy, a geolocalization (in GPS coordinates with a precision error), the freshness of the information (given by the last update timestamp), when it was created (given by the channel creation date), a friendly name and an inferred measurement category for each field (such as temperature, humidity and so on) together with an unit of measure. The latter is essential, since most applications need to use services providing a

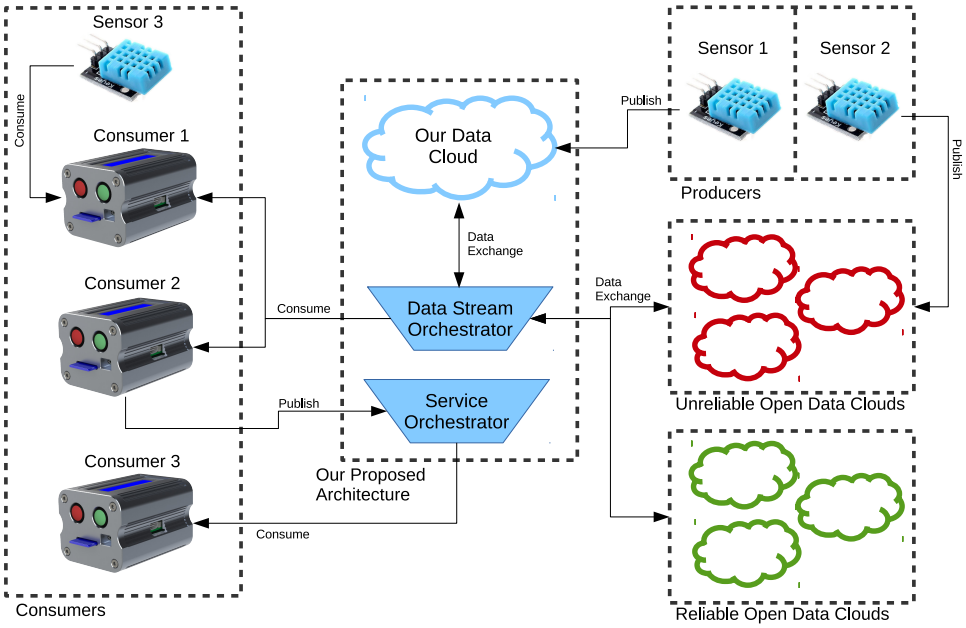


Figure 4.1: Our proposed Open Data architecture [146].

certain type of information, which will be given by the class assigned to the measurement field. Without such a semantic approach, each datastream will have no meaning. To this end, we will address more in detail the classification of unannotated datastreams in Section 4.2 and their semantic connotation in Section 4.3.

4.1.3 Our Architectural Proposal

In this section we discuss our proposal and how we plan to carry out the architectural design. We also give a glimpse on a set of possible case studies for this scenario in order to show why would someone use Open Data integration for measurement tasks. What is presented here as an "architecture" is meant to be integrated with our concrete architectural proposal SenSquare. As a matter of fact, the architecture proposed here is a sample view on how heterogeneous Open Data sources can be a significant improvement on the existing ecosystem. Figure 4.1 shows our proposed architecture depicting several use cases. Note that some entities are numbered in order to distinguish cases in which they are used and deployed differently, however they can be occurrences of the same entity (i.e. sensor stands for any type of sensor, consumer stands for any client device such as a smartphone, a Raspberry Pi or a PC).

In our proposed architecture we assume to have a middleware with three main components: a data cloud (which is dedicated to persistently store datastreams)

and two orchestrators, one for the raw datastreams and one for aggregated services. The Datastream Orchestrator is capable to return a datastream, given a set of parameters determined by the user's choice, from one among all the available sources, either reliable or unreliable (these concepts were defined in Section 2.3.2). User may, for instance, prefer only reliable sources, since they are provided by official and trustworthy organizations, however, their slow update rate introduces a trade-off whenever a user must choose between a high reliability or a fast update rate. Some applications, indeed, might require information at a finer granularity over time, for example when they want to detect instantaneous condition changes. In such cases the user, choosing the update frequency at the expense of reliability, will necessarily use the data or the services provided by neighbors. Furthermore, we assume to have our own data cloud for both datastreams and services that are not intended to be published onto one of the sources mentioned. As a case study, a user can build and run a custom application making use of different measurements, e.g. outdoor temperature and the amount of fine dust or pollen in the air, in order to infer an environmental condition or to trigger some action. For instance it would open automatically the window that is not facing the sun when it's too hot, but only if the pollen in the air is below a certain threshold, otherwise it turns on the air conditioning in order to avoid allergic reactions. This avoids pointless wastes of electrical energy while keeping the domestic environment safe. Such a case study can take place in different scenarios. A user, for example, might be the owner of the temperature sensor, since it is cheap and easily configurable, hence he or she will use it locally (in Figure 4.1 such case is represented by "Sensor 3"). However, there could be other sensors, such as pollen sensors or fine dust sensors, which might be too expensive or rare to get, or simply not owned by the end user and, therefore, measurements from other sources are needed, provided they are nearby enough (this is also why geolocalization is meant to be crucial). In this case, the outcome of the application is possible only through integration of local resources with other measurements, either reliable or unreliable, coming from other sources published in an Open Data platform (case represented by "Sensor 2" in Figure 4.1) or in our cloud (case represented by "Sensor 1" in Figure 4.1).

Our proposed architecture aims not only to unify raw datastreams and make them universally available, but also to make users able to share their service endpoint and to provide additional capabilities derived from both data aggregation and personal computational capabilities (represented by "Consumer 2" in the figure). As a simple example, a user receiving temperature and humidity data might calculate the heat index and expose it as a service interface. In such cases, the end consumer might not directly make use of the raw values of the datastreams, but it can query the orchestrator for a published and available service, providing processed and enhanced data, running on some private system. This reflects the

concept of SOA, a solution we actually carried out in SenSquare (Chapter 6). In conclusion, our proposal, given such a various set of use cases, provides the user with a wide variety of options regarding deployment and data retrieval. This is significantly straightforward, since the user is not forced to stick to a particular approach and gives a great advantage in an era where heterogeneity affects not only data and protocols, but also solutions.

4.1.4 Wrap Up and Future Perspectives

In this section we have introduced the challenging topic of data integration between heterogeneous data sources for the IoT. We have considered Open Data coming both from reliable sources like Governmental agencies as well as unreliable sources, made available through open clouds such as ThingSpeak and SparkFun. We analyzed the differences, and proposed a new architecture to integrate them together, along with the ability to deliver custom made services to the end users, using both reliable and unreliable data. This is an introductory study that opens up a new pathway for the research. Specific findings on this work are given in major detail in Section 4.2 and Section 4.3, whereas its practical use in our prototypical platform can be seen in Chapter 6.

Future works on this topic go through the integration of additional data sources, which will eventually provide a wider set of data. Furthermore, data sources might have a different update rate and a different "reliability", since they may belong to providers that are recognized as trustworthy or not. For non-official data we also plan to use a feedback policy based on the estimated precision and update rate of the datastreams as well as the opinions of users, helping consumers and orchestrators to perform choices based on a trustworthiness value.

4.2 Classification of Open IoT Datastreams

The IoT, underpinned by the principles of the Internet, has led to a phenomenal increase in the generation of IoT data that are contributed by users across the globe and can be publicly accessed via the Internet. We leverage the fact that a lot of data for such domains is available in open access forms from public platforms [146], as we pointed out in Section 2.3. Such "Open Data" is a powerful source of information for developing novel IoT applications, however, especially when such data is crowdsourced, it is currently hardly usable by third parties. A fundamental requirement in successfully re-purposing such open IoT data, in order to enable interoperability as envisioned by Semantic Web 3.0, is to be able to automatically characterize its metadata, i.e. information such as observation type (e.g. temperature, humidity), unit of observation (e.g. Celsius, Fahrenheit), location an many others. However, as validated by a recent study in the literature [199], most publicly available IoT data lack availability of such accurate metadata and, in most cases, even the observation type is unclear (i.e. what is actually being measured).

Recall that, in this chapter, as introduced in Section 4.1, we focus primarily on "unreliable" IoT data, that is, data that is uploaded by the end users, who also assign to each of their datastreams a set of metadata that is rarely complete and accurate. In particular, we use primarily ThingSpeak, which seems very popular. SparkFun and Xively used to be valid alternatives, used in literature for some works related to ours [26][32], however, they are not available anymore. This unreliable Open Data is crowdsourced, thus it is powerful and, looking at the datastream creation trend, potentially able to cover many needs in scenarios like environmental monitoring and smart cities. At the same time, it needs a processing stage in order to be usable.

In this section we describe the work we carried out in [155], and we focus primarily on assigning to unannotated (or poorly annotated) datastreams a class, i.e. an observation type. In fact, in order to semantically characterize a datastream, we first need to understand what is being measured (a temperature datastream could be named like ("temp", or "t1", or "T (°C)", or even something way less interpretable, such as "field_1") and if it is meaningful to the community (the temperature of the boiler of someone is not useful to anyone else). Therefore, the first step of assigning a class to the datastream is necessary. There are obviously other semantic properties that could be inferred and are extremely important, for instance the unit of measure or the location of non geolocated datastream. Deducing classes from names if available has been validated to be effective [148]. Therefore, the problem we are tackling here is annotating missing classes of datastreams produced from *heterogeneous* IoT environment which distinguishes itself from the conventional time series classification typically on *homogeneous* datasets [14], in fact, we will assess that conventional Time Series Classification (TSC) al-

gorithms are unsuitable for this type of problem. In order to achieve the goal, we imply Machine Learning (ML) techniques, in particular, in this section, we make the following contributions:

- A novel Class-wise Bag of Summaries (CBOS) approach, based solely on the numerical characteristics of the sensor readings. It is based on a TSC algorithm, however, it does not use time series properties and, even though its performance are not high enough, it is how we understood that problems with heterogeneous datastreams are not to be solved through TSC.

- A novel Top-k Sequential Ensemble (TKSE) approach, which uses a combination of any available textual metadata describing the datastream and numerical summaries. This is proven to outperform standalone ML algorithms.

- Creation and sharing of a dataset that comprehends several IoT datastreams from ThingSpeak in the form of time series along with their metadata (the numerical part of it could be integrated with the UCR dataset).

- Extensive experimental evaluations to validate the accuracy and the performances of the proposed approaches against the current state-of-the-art approaches in time series classification.

Our proposed sequential ensemble approach is a pioneering effort in the classification of IoT datastream that considers a combination of the numerical characteristics and partially available metadata of the IoT datastream. Our proposed methods are based on generic data mining and ML approaches and efficiently support classification and annotation of open IoT datastreams with diversity in accuracy and availability of metadata. The rest of the section is organized as follows: Section 4.2.1 introduces our IoT data classification problem and describes the proposed CBOS and TKSE approaches, Section 4.2.2 describes our experimental design and the IoT datasets used in our evaluations, Section 4.2.3 presents the results of our experimental evaluations and, finally, Section 4.2.4 summarizes the section with recommendations for future work.

4.2.1 CBOS and TKSE: Approaches for Classification and Annotation of IoT Datastreams

In this section we first formulate the problem of IoT datastream classification and annotation and then present our proposed algorithms: Classwise Bag of Summaries (CBOS) and Top-k Sequential Ensemble (TKSE). Our algorithms are

based on generic data mining and machine learning approaches that are extensible to other problem domains. The metadata can be further annotated through semantic approaches by means of ontologies, however, this aspect is discussed in Section 4.3.

Problem Formulation

Formally, we are given n IoT sensor datastreams $\mathbf{NS} = \{\mathbf{S_1}, \mathbf{S_2}, \ldots, \mathbf{S_n}\}$, each of which can be represented as an ordered tuple that resembles a time series with metadata. That is, $\forall i \in [n] : \mathbf{S_i} = \langle \mathbf{D}_i, r_{i,1}, r_{i,2}, \ldots, r_{i,m} \rangle$ where $[n]$ represents the interval from 1 to n, $\mathbf{D}_i$ denotes a dictionary of metadata on sensor stream i with or without annotations, and each $r_{i,j}$ is a numerical sensor reading (typically a floating point value) from i-th stream at time τ_j. For instance, consider a temperature stream $\mathbf{S_i}$ with annotated metadata 'name' and 'description' and the annotation for metadata $type$ missing. Then $\mathbf{D}_i = \{(name :$ "outdoorTemp"$), (description :$ "ESP8266 with DHT11"$), (type :$ ""$), \ldots\}$. Without loss of generality and for simplicity, we assume datastreams being of the same length m (thus, $\mathbf{NS}$ can be viewed as a column-ordered matrix of size $n \times m$) and time intervals $\{\tau_{j+1} - \tau_j\}$ between two consecutive readings in each datastream being near-uniform. As, in our scenario, the values for the textual metadata $type$, that indicate the classes of datastreams, are missing, our *goal* is to recover the datastream class (the $type$ value in $\{\mathbf{D}_i\}$) from the information of sensor readings and/or other aiding metadata. To achieve this, types or classes are mapped to numerical labels $\{y_i\}$, i.e. datastreams become of the form $\{\mathbf{S_i}, y_i\}$. Specifically, streams with known classes in $\mathbf{NS}$ form the training set of size t, otherwise the testing set of size $n - t$. From the training set, existing classes are mapped to c distinct numerical labels $\mathbf{L} = \{l_1, l_2, \ldots, l_c\}$[6] and, normally, $c \ll n$. In training phase classifiers are built for the later testing phase to inference, from $\mathbf{L}$, which class each missing y_i in the test set belongs to. Throughout the paper, we use bold symbols to denote multi-dimensional data structures such as vectors, matrices and dictionaries.

For the above formulated classification and annotation problem, we first propose CBOS, a preliminary data mining solution based on statistical summaries of numerical readings $\{r_{i,j}\}$, what we call as "bag of summaries" (BOS). Then, we introduce the best performer TKSE that sequentially and incrementally ensembles multiple classifiers trained from the textual metadata and numerical summaries respectively.

[6]For convenience, we can treat "class" and "label" equivalently.

Classwise bag of summaries (CBOS)

Our first approach took place in investigating TSC algorithms for inferencing the type of data measured by each datastream. Given the noisy fashion of each datastream, we attempted to tackle the problem using Bag-of-Patterns-Features (BOPF) [121], a recent phase-invariant dictionary-based approach based on SAX words to encode local patterns. In essence, such approach splits the z-normalized time series through a sliding window in time-ordered chunks and extracts a SAX word by aggregating symbolic mappings of the mean values of the chunks. Features are then represented by the distribution of SAX words across sliding windows. In the training set, features are first ranked by their global ANOVA-F value (the mean variance of a feature value across the whole dataset divided by the sum of within-class variances of that feature) and then by means of a leave-one-out cross validation together with the computed ANOVA-F ranking, a subset of features yielding the highest cross-validation accuracy is selected.

Although TSC approaches are widely known to work well on homogeneous datasets, such algorithm, as well as other TSC approaches we attempted, did not perform the way we expected on IoT datasets (as shown later in Section 4.2.3) due to heterogeneity in IoT data. To cope with this phenomenon, after some experimentation, we observed that a set of global statistical summaries of each non-normalized datastream can be highly discriminative for certain classes. For example, pressure values tend to have an average of the order of magnitude of 10^4 hPa, thus, the unnormalized mean has a higher information gain since is far from any other; likewise, values like RSSI (the received signal strength by a wireless appliance) are often negative, thus the minimum value of the datastream is likely to be indicative. Hence, we adopt a set of meaningful statistical summary features $\{F_1, F_2, \ldots F_f\}$ – defined as *bag of summaries* (BOS) – i.e. mean, median, minimum value, maximum value, root mean squared error (RMS) and standard deviation. Moreover, we propose the algorithmic approach CBOS, built on BOS features, that differentiates classwise features, i.e. each class of instances is trained to have its own/different discriminative subset of BOS features. This is in contrast with the aforementioned BOPF approach, where identical global features are shared by all classes. The proposed CBOS algorithm is presented in Algorithm 1.

In Algorithm 1, instead of ranking global features by ANOVA-F values, we compute *classwise* ANOVA-F (CANOVA-F) distributions $\mathbf{AF}$ on the training set (line 4), where $AF_{i,j}$ is the global variance of feature F_j divided by the local variance of F_j among instances of class i ($i \in [c]$ and $j \in [f]$). Moreover, values in $\mathbf{AF_i}$ are normalized into *weights* (for our later weighted cross validation and testing) by forcing them to sum up to 1. Similar to BOPF, both our cross validation and testing phases rely on a simple 1NN feature distance classification against the

Algorithm 1 CBOS Algorithm

Require: Training set $\mathbf{TRAIN} = \{(\mathbf{S_1}, y_1), \ldots, (\mathbf{S_t}, y_t)\}$, test example $(\mathbf{T}, y)$
Ensure: $y \in \{1, 2, \ldots, c\}$

1: {TRAINING PHASE}
2: $\{(\mathbf{F_1}, y_1), \ldots, (\mathbf{F_t}, y_t)\} := ExtractSummaryFeatures(\mathbf{TRAIN})$
3: **for** $i := 1$ **to** c **do**
4: $\quad \mathbf{AF_i} := NormalizedCANOVA(i, \{(\mathbf{F_1}, y_1), \ldots, (\mathbf{F_t}, y_t)\})$
5: $\quad \mathbf{Cen_i} := CalculateCentroids(\{(\mathbf{F_1}, y_1), \ldots, (\mathbf{F_t}, y_t)\}_i)$
6: **end for**
7: h^* := $LeaveOneOutCV(\{(\mathbf{F_1}, y_1), \ldots, (\mathbf{F_t}, y_t)\}, \mathbf{AF}, \mathbf{Cen})$
8:
9: {TESTING PHASE - 1NN}
10: **for** $i := 1$ **to** c **do**
11: $\quad d_i := 0$
12: $\quad$ **for** $j := 1$ **to** h^* **do**
13: $\quad\quad d_i := d_i + \|(\mathbf{F}[j] - \mathbf{Cen_i}[j]) \cdot \mathbf{AF_i}[j]\|$ # $\mathbf{F}$ is the feature value vector of $\mathbf{T}$
14: $\quad$ **end for**
15: **end for**
16: **return** $y := \arg\min_i d_i$

training class centroids $\mathbf{Cen_i}$ (line 5), obtained by averaging out the BOS feature values within the same class. The main difference is our leave-one-out cross validation (line 7), which performs classwise discriminative feature selection on all classes, i.e. it incrementally tries the $h \in \{1, 2, \ldots, f\}$ highest CANOVA-F-ranked features of each class and eventually finds the best h^* that yields the maximum cross-validation accuracy. For each class with its respective centroids and CANOVA-F values, the distance between a new or cross-validated example and the centroids of such class is calculated as their sum of feature distances *weighted by* the respective feature CANOVA-F values (lines 10 to 16).

Top-k sequential ensemble (TKSE)

Within the scope of classification on data streams and time series, ensemble algorithms have been shown to be effective and able to capture different facets of the type of data [184]. However, most of the existing ensemble algorithms are designed in a parallel fashion, in that a number of classifiers are built and trained on the original data and the class of an unseen example is typically guessed via majority vote. In our case, as stated in Section 4.2.1, IoT datastreams may come with partial and inaccurate metadata (e.g. the ThingSpeak dataset in Section 4.2.2),

which can provide a powerful source of information from a different dimension, for instance the dimension of the natural language. For such reason, we propose a novel *sequential ensemble* of classifiers that sequentially combines the text-based Natural Language Processing (NLP) and numerical value-based classification techniques, on the metadata and sensor readings respectively, so that they both contribute in classifying a datastream enriched with annotated metadata. In particular, our proposed *Top-k Sequential Ensemble* (TKSE) algorithm aims to independently train two or more classifiers of different nature and then classify a new example in a pipeline, rather than doing it in parallel. Our choice of a sequential ensemble relies on the fact that data is *noisy* and presents features in *several dimensions*. Hence, we think that a parallel ensemble of classifiers, each of them trying to guess one class, would be hardly sufficient to get rid of the noise and does not have a way to assign weights to classifiers operating on different dimensions. Conversely, sequential classifiers iteratively get rid of sets of classes that are highly unlikely to be the correct one.

Suppose that two classifiers Γ_1 and Γ_2 can be independently trained on the same training set $\mathbf{TRAIN} = \{(\mathbf{S_1}, y_1), \ldots, (\mathbf{S_t}, y_t)\}$ with training/ground truth classes $|\mathbf{L}| = c$ and $\forall i \in [t] : \mathbf{S_i} = \langle \mathbf{D}_i, r_{i,1}, r_{i,2}, \ldots, r_{i,m} \rangle$. Then, for an unseen example $(\mathbf{T}, y)$, the predicted classes for these two classifiers are $y_1 = \Gamma_1(\mathbf{T}, 1, \mathbf{L})$ and $y_2 = \Gamma_2(\mathbf{T}, 1, \mathbf{L})$ respectively. We will call the predicted class by each classifier as its TOP-1 predicted class; in fact, in many cases, classifiers produce a probabilistic rank of classes based on classification accuracy and the best ranked class (the class that is most likely to be the correct one according to the classifier) is selected as the final prediction. During the testing, TKSE instead first applies the classifier $\Gamma_1(\mathbf{T}, k, \mathbf{L})$ that outputs TOP-k ranked classes $\subseteq \mathbf{L}$ (this can also be viewed as a filtering classifier) based on learning from annotated textual metadata. Then, these output classes from Γ_1 are fed as the input class labels of TOP-1 Γ_2 trained from sensor readings. Therefore, the final ensemble prediction result becomes $y = \Gamma_2(\mathbf{T}, 1, \Gamma_1(\mathbf{T}, k, \mathbf{L}))$. Note that if $k = 1$ then TKSE reduces to $\Gamma_1(\mathbf{T}, 1, \mathbf{L})$, and if $k = c$ it reduces to $\Gamma_2(\mathbf{T}, 1, \mathbf{L})$.

In all our experiments we use as Γ_1 a simple supervised dictionary-based NLP classifier, which we will refer to as a Dictionary Damerau-Levenshtein NLP (DDL-NLP) classifier, first introduced in an earlier work [148]. We chose to use it as Γ_1, because of its TOP-k accuracy (i.e. the probability that the correct class falls into its TOP-k guessed classes) is significantly higher than one of the other classifiers we have considered. In Figure 4.2 the TOP-k accuracy of DDL-NLP is shown; it is clearly increasing as k increases, because the TOP-k accuracy includes all the guesses in which the true positive is among the k classes chosen.

The algorithm focuses on the similarity of the metadata 'name' attributed to data streams as a classifying parameter. Algorithm 2 outlines the proposed TKSE algorithm, detailing also the implementation of the DDL-NLP algorithm:

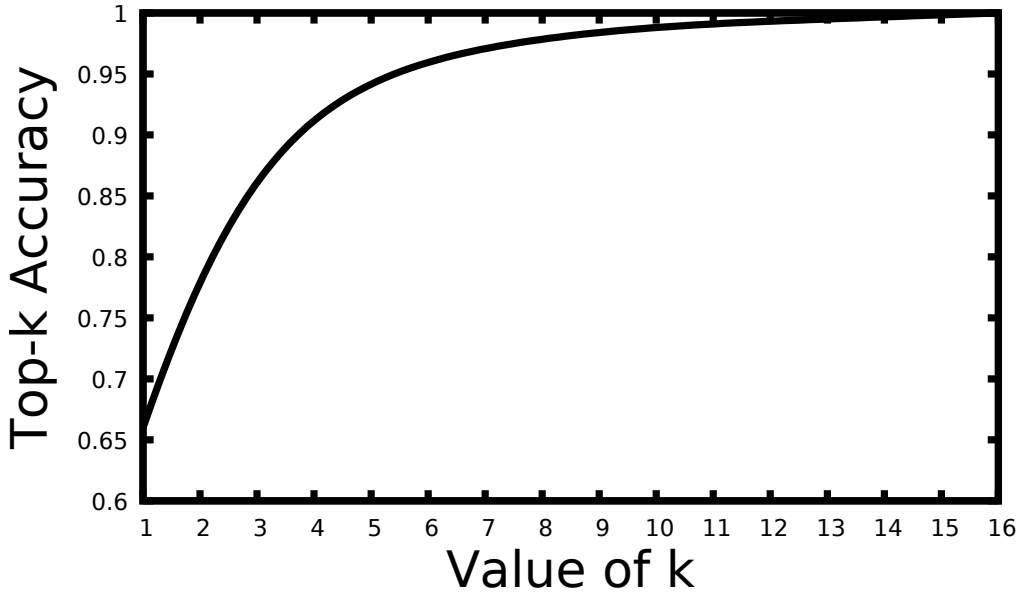


Figure 4.2: TOP-k Accuracy of standalone DDL-NLP [150].

- In training phase, a "dictionary" for each class L_j is constructed in the form of Bag-of-Words (BOW_j) including all stream names in metadata attributed to data streams within the same class (line 2). In practice, the name of every datastream in the training set belonging to class L_j will be added to BOW_j.

- In testing phase, for each class L_j, the classwise minimum edit distance $d_j = min\{ed(w, s) \mid s \in BOW_j\}$ of a testing example w is computed (lines 4-5). For the distance function ed, we leverage the Damerau-Levenshtein edit distance [51] normalized by the maximum length between the two words.

- The algorithm then picks the closest classes through the TOP-k smallest distances among $\{d_1, \ldots, d_c\}$ (line 7).

- Subsequently inputs these to a second vanilla machine learning classifier Γ_2 (line 8) such as decision tree, random forest and SVM (as experimented in Section 4.2.3) trained on all the BOS features.

In order to properly combine two classifiers and achieve the best accuracy, the optimal value of k has to be determined, since, an inappropriate k would impact negatively on the performance by either missing many classes or introducing much noise. The straightforward approach would try all *discrete* values

Algorithm 2 TKSE Algorithm with Γ_1 as DDL-NLP

Require: Training set $\mathbf{TRAIN} = \{(\mathbf{S_1}, y_1), \ldots, (\mathbf{S_t}, y_t)\}$, test example $(\mathbf{T}, y)$, k

Ensure: $y \in \{1, 2, \ldots, c\}$

1: **for all** $(\mathbf{S}_i, y_i) \in \mathbf{TRAIN}$ **do**
2: $\quad BOW_{y_i} \leftarrow \mathbf{D_i}('name')$ # $\mathbf{S_i} = \langle \mathbf{D}_i, r_{i,1}, r_{i,2}, \ldots, r_{i,m} \rangle$
3: **end for**
4: **for** $j := 1$ **to** c **do**
5: $\quad d_j = min\{ed(\mathbf{D}('name'), s) \mid s \in BOW_j\}$ # $\mathbf{T} = \langle \mathbf{D}, r_{i,1}, r_{i,2}, \ldots, r_{i,m} \rangle$
6: **end for**
7: $\mathbf{C} = \{L_j \mid d_j \in kmins\{d_1, \ldots, d_c\}\}$ # $kmins$ picks the k lowest values
8: $y = \Gamma_2(\mathbf{T}, 1, \mathbf{C})$

of $k \in [c]$ and choose the value k^* which yields the highest accuracy in k cross-validation rounds of Γ_2 on the training set, however, such method could be slow when c is large. This is also unlike applying stochastic gradient descent on approximating continuous non-convex functions. Instead, we perform a faster logarithmic search heuristic as a simple approximation: first let ACC_k denote the cross-validation accuracy for the chosen k, then from 1 to k we incrementally try values $\{2^0, 2^1, 2^2, ..., k\}$ and pick $k' = 2^p$ with the highest $ACC_{k'}$. As intervals between $[2^{p-1}, 2^p]$ and $[2^p, 2^{p+1}]$ might get larger, we can then recursively perform the above logarithmic guesses in these intervals until they are small enough and find the overall best guess $k^* = \arg\max_{k'} ACC_{k'}$. Note that if we have more useful annotated metadata, TKSE can then be extended to a chain of more than two classifiers, although this brings additional issues that are not discussed here.

4.2.2 Experimental Design

In this section we describe in detail the datasets we used to validate the proposed approaches, our experimental objectives and design consideration.

Public Open IoT Datasets

We chose a combination of unannotated and partially annotated open source IoT datasets, namely: *The Swiss Experiment* dataset[7] and a dataset we extracted from the public channels on the *ThingSpeak* cloud platform, which has been used in our introductory work in Section 4.1, in order to validate the performance and accuracy of the proposed approaches. The latter is entirely extracted, cleaned,

[7] `http://www.swiss-experiment.ch/`

annotated and formatted by us and it is made available online, as we believe it brings a substantial contribution in this research area.

The Swiss Experiment It is a platform that allows publishing environmental sensor data located within the Swiss Alps mountain range on the web in real-time. Data is highly noisy, comes from different microscopic locations and it is taken within different time spans. The sampling rate is also different among sensors making the phase shift of data series very significant. Neither semantic annotation nor timestamps were originally provided, thus data is unusable as it is, since the only information provided is the numerical datastream. To the best of our knowledge, the Swiss Experiment dataset (for which a manually annotated version is available at `http://lsirpeople.epfl.ch/qvhnguye/benchmark/`) is one of the few heterogeneous datasets used in research for our type of problem [32]. The authors in [] have used the dataset for their classification experiments using an encoding with slopes, however, they still relied on TSC-based algorithms. They also used data extracted from AEMET[8], the Spanish meteorological office, however, such datasets are constantly changing and the extracted data is usually not made available. The Swiss Experiment dataset consists in datastreams measuring 11 different environmental parameters: CO_2, humidity, lysimeter, moisture, pressure, radiation, snow height, temperature, voltage, wind speed and wind direction. Time series are of different length, however, some of the algorithms we have tested require the time series to have the same length, therefore we cut each time series to the length of the shortest stream in the dataset. After such step, the dataset is composed by 346 datastreams without metadata and 445 data points each.

ThingSpeak To recap what has been introduced in Section 4.1.1, it is an online cloud platform to which users can subscribe and push sensor data produced by their personal device onto their personal "channel" through dedicated APIs. Channels are optionally public and are composed by a set of time series (one for each sensor) and partially and mostly inaccurate user-annotated metadata: each channel has a name, a description, a name for each datastream and, optionally, a geolocation in GPS coordinates. Each name (as well as the other metadata) is user-assigned, thus it can be informative as well as useless. Each channel can be updated at any time rate above $10\,\mathrm{s}$ and the cloud keeps permanently in memory the last 8000 measurements. Channels are numbered progressively and the data streams belonging to the public ones are available as JSON objects. We built our dataset by scraping the first 500,000 channels through parsing a JSON object returned by the dedicated HTTP call `https://thingspeak.com/`

[8] `http://www.aemet.es/en/portada`

channels/{ch}/feed.json?results=8000&start=2018-01-01 00:00:00, where {ch} is the number of the channel. With such call, the metadata and the last 8000 readings in year 2018 from all the datastreams belonging to the queried channel are retrieved and, subsequently, all the datastreams were made independent from their channel, thus the initial dataset is composed by 11,742 datastreams, each including a time series of sensor readings, with associated timestamps, and a set of metadata. In order to provide a least consistency among environmental value readings in data streams, we operated both a spatial and a temporal clustering: we first clustered the data streams by location using DBSCAN. We tested the algorithm with different radii in order to reach a consistent number of instances within the same cluster and chose a radius of about $500\ \mathrm{km}$ over the most populated cluster: an area in central Europe, which includes parts of Germany, Poland, Czech Republic, Slovakia, Hungary and Austria. The amount of streams is then reduced to 1,803. We further filtered out private streams, non geolocated streams and streams with less than 5,000 readings in 2018; and clustered the rest both by time: in order to perform a temporal consistency, we first homogenized each time series clustering the data points in 15 minutes time chunks. Then, we filtered our dataset calculating a time window of 2 days within which we have the maximum number of series having measurements in such window and including only those that have measurements in at least 45 chunks belonging to the most "populous" interval of 24 hours within such window. The final number of data streams is 1,275, with measurements starting on February 22nd 2018 at 6PM and ending on February 23rd 2018 at 6PM. The data, at this point, displays few missing points, which we interpolated my means of cubic splines. Subsequently, we manually annotated all the data streams, assigning them a class according to the environmental parameter measures by means of the human interpretation of the metadata. The task has been challenging due to different languages with which the metadata has been annotated (English, German, Czech, Hungarian etc.) and some of the streams had been excluded due to a too high uncertainty, therefore the final number of streams is 1,091, belonging to 16 different classes: non-air temperature, humidity, pressure, wind speed, wind direction, UV, light, sound, air quality, electrical parameter, RSSI, indoor air temperature, outdoor air temperature, heath index, rain index, dew point. In summary, the dataset is composed by 1,091 datastreams with metadata and 96 data points each. We made the dataset available for download at https://github.com/stradivarius/TSopendatastreams.

Experimental Objectives

In order to evaluate the effectiveness and efficiency of our proposed IoT datastream classification algorithms, we performed extensive experiments against other

state-of-the-art approaches on the above open IoT datasets. Experimental results reveal that these datasets display similar behaviors over the evaluated approaches.

For the purpose of experimental evaluation, we use the following three metrics: *Accuracy, Macro averaged F1-Score* – defined in [201] as the harmonic mean of macro averaged precision and recall over all classes – and the *average runtime performance* over the number of folds in seconds. For the sake of clarity, we report here the formulas for calculating accuracy (A), macro averaged precision (P), macro averaged recall (R) and macro averaged F1-Score ($F1$) on a dataset of size m given, for a given class c, the number of true positives (tp_c), the number of false positives (fp_c) and the number of false negatives (fn_c):

$$A = \frac{\sum_{i=1}^{C} tp_i}{m}; \quad P = \frac{\sum_{i=1}^{C} \frac{tp_i}{tp_i+fp_i}}{C}; \quad R = \frac{\sum_{i=1}^{C} \frac{tp_i}{tp_i+fn_i}}{C}; \quad F1 = \frac{2 \cdot P \cdot R}{P+R}$$

We chose to report the macro averaged F1-Score as the number of instances per class in each of the datasets is unbalanced and, while a high accuracy indicates the overall success, a high F1-Score implies that all classes have been equally considered. Through the usage of such metrics, we performed the following experiments:

Evaluation of Accuracy and F1-Score The objective of this experiment is to validate the proposed algorithms on both our IoT datasets against time-series and BOS-based algorithms from the literature. Each algorithm has been validated through a k-fold cross validation. On Swiss Experiment we used 5-fold cross validation (same as [32]), whereas on ThingSpeak, we used 10-fold, as the number of instances is much higher. The algorithms are tested both on the original and z-normalized time series, as the conventional homogeneous time series analysis often requires normalized data.

Impact of K on TKSE This experiment has the goal of validating the behavior of TKSE for different values of k. It is designed such that our BOS-based algorithms are tested as a second step in TKSE together with the DDL-NLP method (as described in Section 4.2.1) in a 10-fold cross validation on the ThingSpeak dataset (with annotated names) on all values of k. Although the TKSE method imposes a logarithmic search of the optimum value of k, for the sake of completeness, we report the performances of such algorithms on the ThingSpeak datasets using all the 16 values of k. Again, for each value of k, we use a 10-fold cross validation and calculate the features over the non normalized data.

Evaluation of Runtime Performance This experiment has the purpose of validating the efficiency of the considered algorithm in time. They are tested on both

datasets and the average runtime over the number of folds in seconds is reported.

Experimental Design

Our approaches CBOS and TKSE together with our adopted vanilla classifiers: decision trees (C4.5) [170], random forest (RF)[9] [28] (which have been shown to perform well on remote sensing scenarios [161]) and Support Vector Machines (SVM) [85] are compared with the following TSC algorithms and the results obtained with the slope-based algorithm in [32] for sensor data:

The golden standard As sensor reading streams can be easily interpreted as time series, we took into account the most widely used TSC algorithm: *One Nearest Neighbor with Dynamic Time Warping* (1NN-DTW) [175] – considered as the golden standard for TSC [17]. It is a whole series method that has been proposed to cope with phase shifting when calculating series similarity. In particular, suppose we want to compute the similarity between two series $a(a_0 \dots a_n)$ and $b(b_0 \dots b_n)$ and we calculate the pointwise $n \times n$ distance matrix D. The DTW distance between the series is the minimum warping path $P = ((e_1, f_1), \dots, (e_n, f_n))$ that defines the transversal of the matrix N, i.e. intuitively a path that leads from the point (1,1) to the point (n,n) increasing, at each step, the column, the row, or both. It is superior to the typical whole series baseline, the 1NN with Euclidean Distance (1NN-ED), which computes the similarity between two series by summing up the euclidean distances among their data points in the same position.

Dictionary approaches Within the scope of TSC, we also included two recent dictionary-based approaches: *Bag-of-Pattern-Features* (BOPF) [121], which our CBOS is based on as outlined in Section 4.2.1 and *Bag-of-SFA-Symbols* (BOSS) [188]. BOPF is a recent linear time implementation of the well known BOP approach relying on the series transformation in approximated SAX words. In particular, the algorithm slides a window along the time series and breaks the interval into chunks that are transformed into a SAX word according to some property (usually their mean). A histogram counting how many times the same word occurs within the series constitutes the bag of features for such series, which is then classified through 1NN approach. This method is of particular interest since, given the noise and the phase shift affecting the data, it is expected to perform better as it is focused more on micro patterns. The recent linear time implementation uses, as training features, the per-class centroid or the tf-idf of the series in the dictionary rather than comparing each unseen example with every training instance.

[9] Although [56] has considered a more sophisticated RF, this is not the focus in dealing with heterogeneous IoT data.

BOSS is a recent dictionary-based time series classification algorithm that, as well as BOPF, slides a window along the series sampling each time the words using, instead of a PAA, encodes subsequences through Discrete Fourier Transform and establishes the breakpoints a priori via Multiple Coefficient Binning, instead of using fixed intervals. As well as BOPF, the method is an ensemble that validates several parametrized classifiers over the training set keeping only the best ones and guessing the unseen class via majority vote.

All experiments are performed on a computer with an Intel Core i7-7700HQ CPU @ $2.80\ \mathrm{GHz} \times 4$ and $8\ \mathrm{GB}$ RAM while running Linux Mint 18.2 64-bit. All tested algorithms are implemented in Python 3.5.2 except the implementations of BOPF and BOSS, for which we used the original C++ code provided by the authors.

4.2.3 Experimental Results and Discussions

In this section we provide results and insights about the experiments outlined in Section 4.2.2.

Evaluation of Accuracy and F1-Score

We evaluated our proposed algorithms with the ones outlined in Section 4.2.2 using the Swiss Experiment and ThingSpeak datasets. Both accuracy and F1-Score are reported for the datasets (Swiss Experiment in Figure 4.3 and ThingSpeak in Figure 4.4) in their original and z-normalized form. Observing the outcomes of such analysis, it is immediately clear how data normalization causes the loss of important features and, hence, impacts the accuracy of classification making this a much harder problem. In fact, only BOSS and BOPF achieve similar results for both normalized and unnormalized datasets, in that such algorithms were designed specifically to operate on z-normalized series and some parameters are hard coded to cope with the underlying data. Nevertheless, they still achieve a similar accuracy on the original series. Our first summary-based approach CBOS performed poorly on z-normalized data. This is expected, as z-normalization loses most of the information based on statistical summaries. But, on the non z-normalized data CBOS improves on BOPF, which it is based on, and is later shown to be faster than BOSS. In summary, the above mentioned TSC approaches still do not achieve the desirable IoT data classification accuracy. On the other hand, the golden standard DTW tends to perform better on non z-normalized data, sometimes achieving good results on ThingSpeak. However, this further validates our findings i.e. the trend of the time series is not as indicative as the absolute value ranges in heterogeneous IoT data. These absolute value ranges are better captured when the data is not z-normalized.

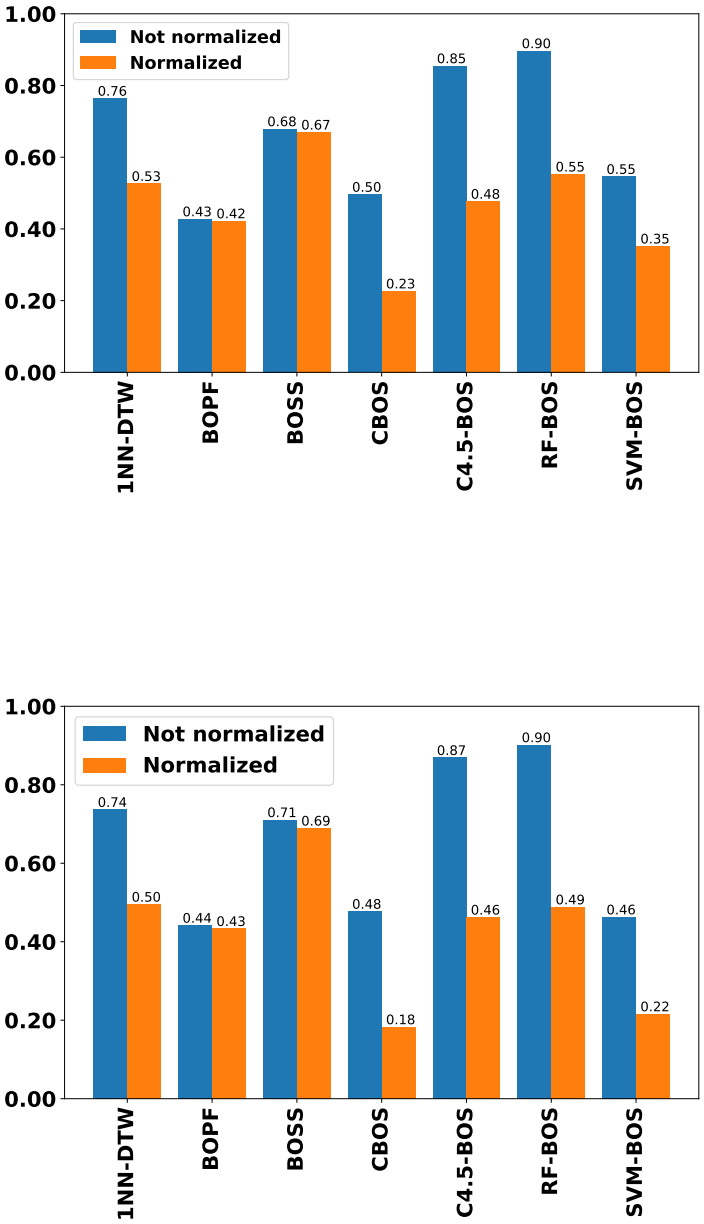


Figure 4.3: Accuracy (above) and F1-Score (below) for Swiss Experiment Dataset [150].

Based on this inference, we further found through our experiments that the vanilla machine learning methods performed using our BOS features achieve significant accuracy and F1-Score gains. Our justification for such phenomenon lies behind the fact that, due to heterogeneity in IoT data, different classes exhibit different behavior (and, therefore, bias) in terms of such features. For such reason, tree-based classifiers (C4.5 and RF), built in a way in which the attribution of an example to a class is driven by a sequence of decisions based on the threshold of the feature itself, seem intuitively suitable for open sensor data. This is not the way in which SVM are designed, in fact, they do not seem to work as well. It is also interesting to notice that SVM tend to assign more likely the most populated classes, resulting in a fairly poor F1-Score. Furthermore, on the Swiss Experiment dataset, the authors in [32] have reported on normalized data in the form of a pattern-based time series approach, achieving an accuracy of 67.7%, whereas our RF approach on BOS easily achieves above 80%.

By looking at Figure 4.4 it is possible to see the performances of the DDL-NLP algorithm and the proposed TKSE (which includes DDL-NLP as a first step). DDL-NLP alone has been applied on open datasets previously (data extracted from ThingSpeak and SparkFun) [148] with an accuracy close to 88%, however, such datasets are purely composed by data streams annotated in English, whereas the ThingSpeak dataset presented in Section 4.2.2 is annotated in different languages and the performances of the DDL-NLP approach alone drop to 65% in both Accuracy and F1-Score. Looking at the bar charts, it is clear that TKSE with tree-based methods (RF and C4.5) bring significant improvements (at $k^* = 4$), whereas all others coupling with DDL-NLP diminish the performances as detailed in the next subsection, demonstrating the important influence of annotated metadata.

Impact of k on TKSE

The performance of the proposed TKSE approach has been evaluated via coupling the DDL-NLP classifier with other feature-based classifiers included in our previous test. For the purpose of this experiment we only used original data due to their preserved distinguishing power. For completeness in the illustration we tried each value of k rather than using a recursive logarithmic search. As stated before and shown in Figure 4.5(a), only tree-based approaches display a performance increase with TKSE, whereas the others tend to be pejorative. It is also interesting to notice how the curves are similar in their trend, in particular they all display a local maximum for $k \simeq 4$ while performance starts dropping for greater values (as more noises are introduced). On the other hand, F1-Score, as shown in Figure 4.5(b), is not positively affected by TKSE on ThingSpeak, since DDL-NLP is already the highest among the algorithms – this is mostly due to the reason that

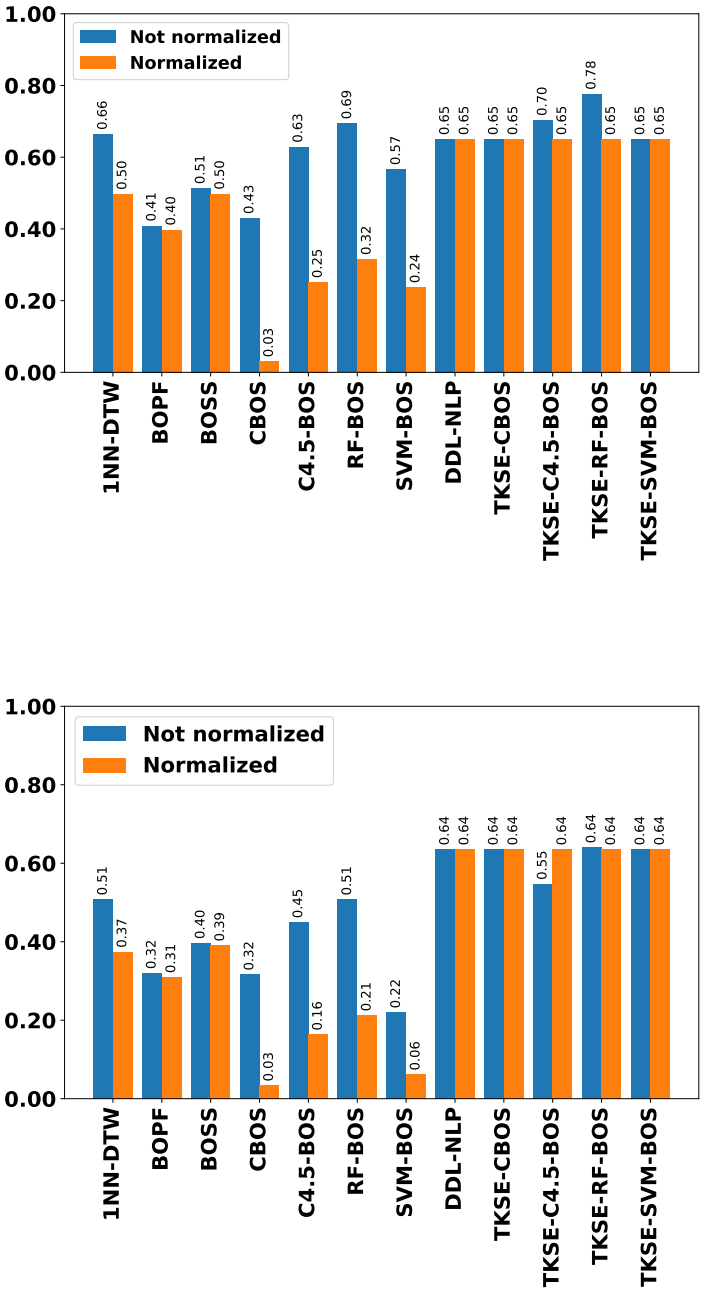


Figure 4.4: Accuracy (above) and F1-Score (below) for ThingSpeak Dataset [150].

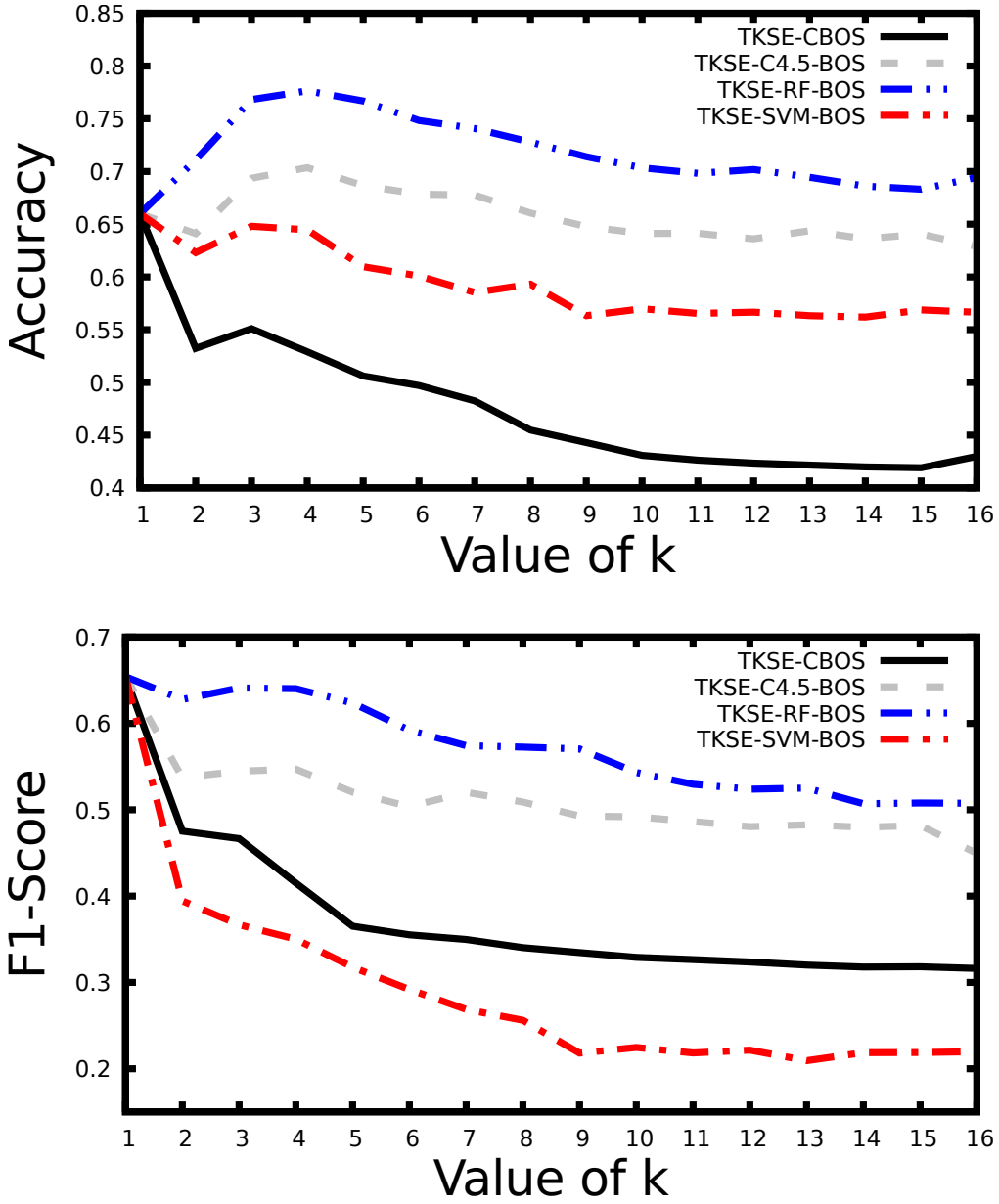


Figure 4.5: Accuracy (above) and F1-Score (below) of TKSE approaches evaluated on different algorithms. Tree-based algorithms display an evident increase in accuracy [150].

the proposed TKSE searching/tuning process is designed with the objective of optimizing the classification accuracy instead of the F1-Score. This phenomenon requires our future investigation, but nevertheless TKSE-RF keeps the F1-Score

to approximately the same as DDL-NLP alone. Therefore, along with its superior accuracy, TKSE-RF on BOS is by far the best choice among all the algorithms.

Evaluation of Runtime Performance

We have tested extensively the runtime performances of all the evaluated methods. The results are reported in Table 4.1 and time is measured in seconds over the cross-validation process divided by the number of the folds (which gives the actual training and testing time over a training and a test set). In particular, the Swiss Experiments has a training set size of 273 and a test set size of 73, while the ThingSpeak dataset has a training set size of 974 and a test set size of 117. It is interesting to notice how algorithms perform differently on such datasets due to the relatively larger number of instances in ThingSpeak and longer series in the Swiss Experiment (e.g. BOPF performs better on the Swiss Experiment, whereas BOSS on ThingSpeak). The slow performance of DTW is expected due to its high worst-case time complexity $\mathcal{O}(n^2m^2)$, where n is the number of stream instances and m is the series length. Other time series based methods mostly have a linear complexity (w.r.t the input size of nm data points): BOPF and CBOS both have a complexity of $\mathcal{O}(nm)$, although, in practice, CBOS is significantly faster, and BOSS has a complexity of $\mathcal{O}(nm^{\frac{3}{2}})$. TKSE-based experiments were performed with logarithmic search that is relatively slower but more accurate. Overall, except for SVM and DTW all the other methods perform well in runtime, and, in particular standalone RF and TKSE-RF flexibly trade off some runtime for classification quality and exhibit better performances than others on both datasets.

4.2.4 Wrap Up and Future Perspectives

In this section we proposed novel algorithms to tackle the challenge of annotation and classification of open IoT datastreams produced from *heterogeneous* IoT environments. In particular, we first proposed CBOS, a bag of summary-based approach to classify IoT datastreams based on numerical characteristic of the underlying IoT datastream. Through experimental evaluations and validation, CBOS showed that, although IoT datastreams are reminiscent of time series datasets, due to the heterogeneity in the sensor readings produced by IoT devices, classic TSC approaches perform poorly while vanilla classifiers such as decision trees and random forest based on bag-of-summaries (BOS) perform significantly better when only considering the numerical characteristics of the IoT datastream. Our second proposed algorithm namely TKSE uses a novel sequential ensemble approach to take advantage of both 1) partially available textual metadata that

[10]This algorithm is written in C++ since the original code provided by the authors has been used, implying stronger runtime baselines to compare with.

Table 4.1: Runtime performances in training and testing of all the algorithms on both Swiss Experiment and ThingSpeak datasets

	Time in seconds (Swiss Experiment)	Time in seconds (ThingSpeak)
1NN-DTW [175]	5989.694	1528.244
BOPF[10] [121]	5.87	36.589
BOSS[10] [188]	79.028	15.628
CBOS	0.068	0.265
C4.5-BOS	0.045	0.138
RF-BOS	0.589	1.178
SVM-BOS	24.194	845.437
DDL-NLP	-	1.286
TKSE-CBOS	-	11.670
TKSE-C4.5-BOS	-	7.195
TKSE-RF-BOS	-	21.510
TKSE-SVM-BOS	-	5082.260

describes the IoT datastream and 2) the numerical characteristics of the IoT datastream. Through extensive experimental evaluations and comparisons with state-of-the-art approaches in the literature, we validated the significant gain in accuracy of the proposed TKSE algorithm while imposing minimal impact on runtime performance. Future work can be devoted into further improving annotation quality with more sophisticated features and ensembles that leverage all useful metadata to some extent, as well as studying the behavior of sequential ensembles in the presence of more than two classifiers. In fact, if we have more useful annotated metadata, TKSE can then extend to a chain of Θ classifiers as below:

$$y = \Gamma_{\Theta}(\mathbf{T}, 1, \Gamma_{\Theta-1}(\mathbf{T}, k_{\Theta-1}, \ \ldots \ (\Gamma_1(\mathbf{T}, k_1, \mathbf{L})) \ \ldots \ ))$$

where $\forall\theta \in \{2, 3, \ldots, \Theta\} : 0 < k_\theta < k_{\theta-1}$ and $k_1 < c$. However, we still lack an efficient algorithm that finds all the optimal k_i. Furthermore, as shown in experiment Section 4.2.3 our simple DDL-NLP approach alone works reasonably well for ThingSpeak, although a larger (external) dictionary corpus and a more sophisticated NLP technique (might be much slower) could be later considered. However, the focus of our approach until now has been a novel and efficient way of ensembling classifiers for better classification.

4.3 INFORM: Framework for Open IoT Data Annotation

The process of classification itself is not enough to guarantee the reuse and the adaptability of Open IoT datastreams. In fact, many of these datastreams lack semantic annotations, an essential component required to effectively use the IoT data in contexts where interaction between several components is required. Furthermore, in order to monetize the IoT data[11] [89], it is imperative to understand, contextualize and categorize the IoT datastream, e.g. whether a particular IoT datastream is indoor or outdoor temperature, what is its unit of measure, etc. Hence, a tool that is able to automatically categorize IoT data and semantically enrich them with metadata extracted from domain ontologies and semantic sensor ontologies is imperative. To address this issue, we the platform INFORM that: 1) automatically infers types of IoT datastreams through the novel classification algorithm TKSE presented in the previous section (Section 4.2.1); and 2) annotates datastreams with metadata extracted from relevant domain-specific ontologies and semantic sensor ontologies such as SOSA. Although still a future work, we developed the prototype of inform for a submitted work [78].The reference ontology used for this work is IoT-Lite [22], proposed within the FIWARE project. The Proof of Concept (PoC) implementation takes non-annotated IoT datastreams and produces a semantically annotated IoT datastream using IoT-Lite for annotation and TKSE for classification. At the moment we only classify the observation type, as pointed out in the previous section (Section 4.2.1), however, we envision to be able to infer much more metadata in the future. For such reasons, no individual evaluation is available for INFORM, however, we necessarily introduce this piece of software as a fundamental component that will put sensor classification and annotation into the big picture of our framework, outlined in Chapter 6. Figure 4.6 illustrates the overall architecture of the INFORM tool. INFORM tool consists of two main components: 1) Top-k Sequential Ensemble (TKSE) algorithm that underpins the IoT datastream classification (presented in Section 4.2.1) and 2) IoT Datastream Annotator (DSA) that provides the mechanism to semantically annotate IoT datastreams.

DSA is responsible for annotating the datastreams (using IoT-Lite) classified by the TKSE component thus making them discoverable for other IoT applications and providing a subscription model for IoT applications to subscribe their interest and specify a corresponding domain specific concepts (in the form of an ontology) for semantic enrichment purposes. The annotation of datastreams associates the IoT-Lite class with the output class from the TKSE module and we expect it to semantically enrich the annotated IoT datastreams with domain specific concepts

[11] `http://www.terbine.com/slidedeck.html`

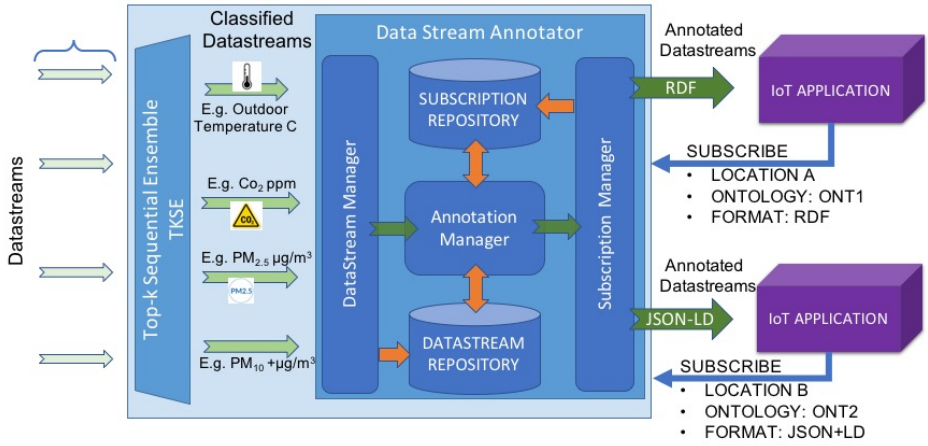


Figure 4.6: INFORM Architecture.

provided by the IoT application. The DSA component has three modules, namely Datastream Manager (DSM), responsible for annotating and storing datastreams, the Annotation Manager, which produces a semantically annotated datastream based on IoT application requirement, and Subscription Manager, which handles IoT application subscriptions.

Based on the proposed architecture, we implemented a prototype of the INFORM tool as a JavaEE web application and deployed it on Payara Server 4.2.

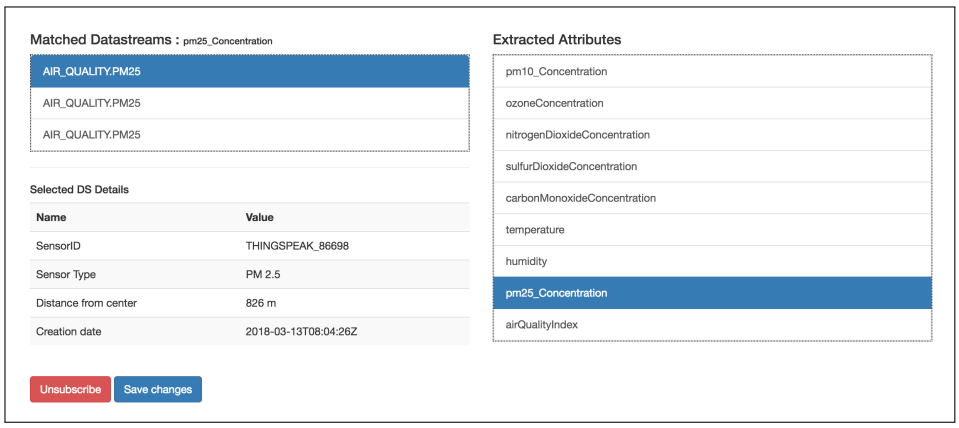


Figure 4.7: Inform subscription manager UI.

This application has a number of Restful APIs that allow IoT applications to easily interact with INFORM. We developed a desktop application which allows the IoT application users to subscribe for their datastreams of interest and manage their subscriptions through a user-friendly UI.

Figure 4.7 depicts the user interface of the subscription manager developed

to test the functionalities of the INFORM tool. The provided screenshot shows an example of subscription in INFORM where the domain ontology of interest is `http://opensensingcity.emse.fr/ontologies/airQuality/`, which is a quite simple ontology for monitoring air quality. This screen consists of two columns, Matched datastreams and Extracted attributes. The extracted attributes column lists all the attributes INFORM extracts from the domain ontology provided by the IoT application in the subscription stage. When the user clicks on one of these attributes, based on the selected attribute, a list of automatically matched datastreams will be loaded in the matched datastreams column.

Although a missing piece of the work, since more classified metadata would be needed in order to perform an evaluation, we presented this software as a practical support to the classification module in our big picture.

Chapter 5

Mobile Crowdsensing for Device Redundancy

This chapter outlines our contributions in the context of MCS and how we use it to get the data we need. Specifically, it discusses a framework we proposed for MCS applications in Smart Cities where both participants and stakeholders take part for the common benefit as well as a distributed probabilistic algorithm that we designed for urban contexts that could be affected by the *"Curse of Sensing"*.This Chapter is written on top of our works in [149] and [147].

5.1 MoCroSS: A Mobile Crowdsensing framework for Smart Cities and Environmental Monitoring

In Sections 2.4 and 2.5 we talked extensively about MCS and its main challenges in the current and future research landscape. What is also evident from the current trend in dealing with MCS is that, typically, research is oriented to the user perspective of MCS. As a matter of fact, some types of data could be costly for the user to get, such as the throughput of the cellular connection, therefore, their transmission should be limited to what is strictly needed, in order to overcome the *Curse of Sensing* problem. Moreover, crowdsensing campaigns are typically designed on a single goal, thus, an individual participating in multiple campaigns results in running different applications and potentially uploading the same measurement to different servers, resulting in a wastage of resources, particularly at the user's side. A lot of data is already collected on the field by privates and institutions, such as the air quality, environmental noise, temperature and many others. This is another facet of data redundancy, which has been discussed in Chapter 4. MCS represents also an opportunity for stakeholders, that is to say, those who have interest in gathering and analyzing data. An example comes from the mobile network operators, which need to monitor the performance of the cellular connectivity for their users by testing it on the field. Indeed, thanks to MCS, such operators could potentially ask directly to the end users to perform connectivity measurements and report the data. In such case, gathering a sufficient number of users producing the measurements is clearly not straightforward, but it is less expensive than sending a dedicated specialized team in the location of interest to measure the connectivity, despite the additional cost that a data analysis step at the server side in order to filter out poor quality MCS data would involve.

One of the major problems in this domain is the fact that a general purpose paradigm in which stakeholders can declare their needs and users can report their data to be analyzed by stakeholders is missing. In this section we propose a novel paradigm, through a platform, that fills this gap, designed to be a module of our prototypical framework, SenSquare, described in its big picture in Chapter 6. In this document we will refer to this module as MoCroSS (Mobile Crowdsensing module for SenSquare) (although it has been published as SenSquare [149], as it was published before our more general architecture was designed). In MoCroSS stakeholders can declare their campaign and ask for certain data, users can then subscribe to the needs of stakeholders and report the data needed by them. Requests for the same data by different stakeholders are automatically handled so that users only report data once for all the requesting stakeholders when is needed.

We compare our smart system with its non-smart counterpart, in which all the sensing clients send their data with respect to a local timer and are totally unaware

of the community and the issues related to the *Curse of Sensing* that may occur. We claim that our system brings several advantages and an overall benefit more efficiently that the non-smart one for the following reasons:

- Depending on the targeted scenario, our proposal limits significantly the amount of redundant data to be sent, received, stored and processed by the remote data aggregator, especially during "data rush hours". This is done without affecting the results, hence avoiding power and resources waste. The design of an algorithm that optimizes the amount of data to be sent is beyond the scope of this section, which is mainly dedicated to outline the architecture. In fact, here we introduce a simple rule-based upload policy, while the actual algorithm is discussed in Section 5.2. The upload policy presented here significantly outperforms the non-smart policy and it is a first step towards the design of the distributed algorithm, which uses the framework and the rules presented in this section.

- Crowdsensing applications are known to meet scarcely the needs of the final user when the personal income is not immediately visible due to a lack of incentives, thus a rewarding mechanism has to be adopted. MoCroSS is fed by both the users' willingness to get rewards from the stakeholders and the stakeholders' needs to get data from the users.

The rest of this section is organized as follows: Section 5.1.1 details our proposed architecture; Section 5.1.2 focuses on the server side and its reasoning capabilities; Section 5.1.3 follows with the details of a possible client implementation compliant with our architecture (we shown this as a prototype); Section 5.1.4 summarizes the results.

5.1.1 System Architecture

In this section we outline the architecture and the main components characterizing MoCroSS. The system is organized in a star client-server topology, meaning that client entities have no real or virtual communication link between each other and the whole set of reasoning capabilities is assigned to a centralized component, which we refer to as the Central Coordination Unit (CCU). The overall architecture is shown in Figure 5.1. In our scenario there are two sets of client entities. The first one, which we refer to as the sensing clients or the participants – a term recalling our original definition in 2.2 –, include any device equipped with sensing hardware and network connection, belonging to a final user and capable of reporting sensed data through the Internet, such as smartphones and embedded devices. We also envisage a number of stakeholders to take part in our scenario as the second set of client entities. In particular, they are able to push rules to

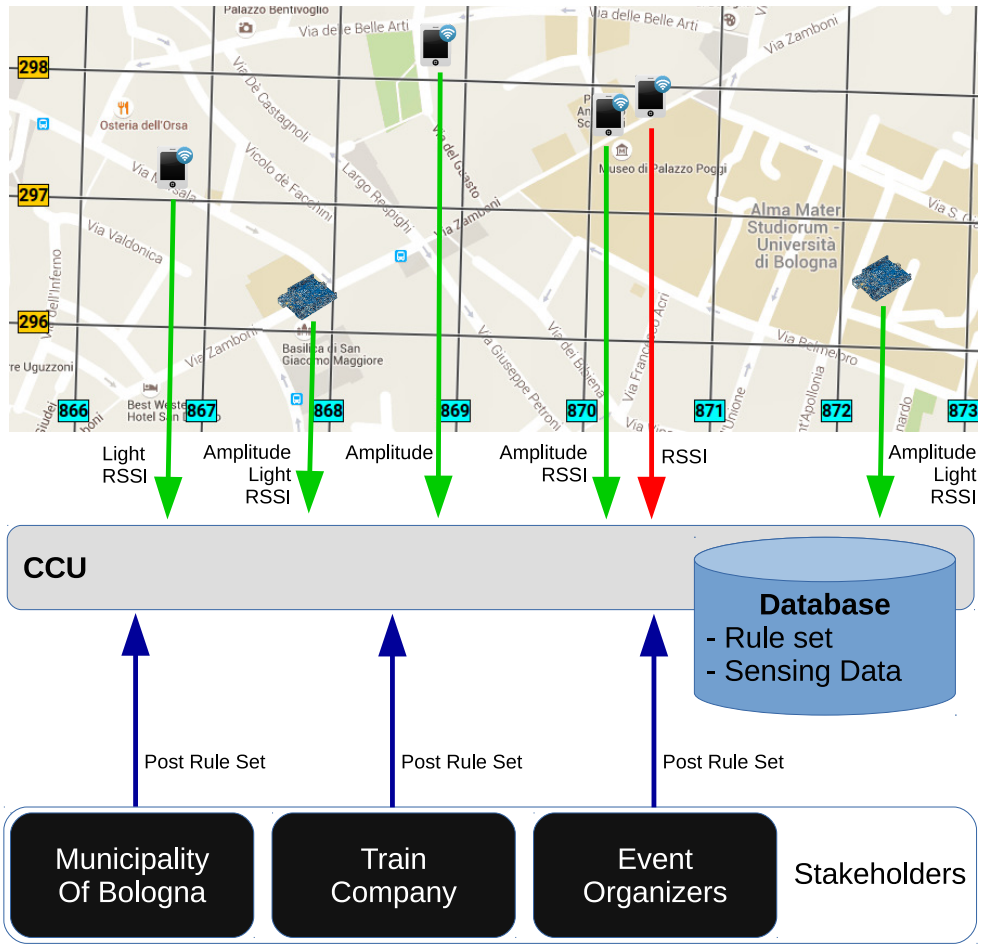


Figure 5.1: The overall system architecture. It is highlighted how a measurement coming from two smartphones in the same zone is not required to be updated by both (the red cross outlines a denied transmission) [149].

the CCU in order to orchestrate the update rate of the sensing clients that are subscribed to the campaigns they issued. On the other hand, stakeholders can obtain the subscription of participants to their campaigns by offering some kind of revenue, depending on both the needs of both parts. The task of the CCU is to gather and store the measurements of the participants and to provide each of them, in a smart way, with a command stating the update rate needed for each measurement with respect to the rules of each campaign stored by stakeholders. More in detail, whenever a sensing client starts its activity, by sending all the measurements available from its sensors, the CCU replies, for each sensor type, with a temporal constraint (correspondent to a timer after which the measurement from that sensor is not valid anymore and has to be resent) and a spatial constraint (correspondent to a zone outside which the measurement from that sensor in not valid anymore has to be resent). Clearly, the spatial constraint is significant only when the sensing client is a mobile system, such as a smartphone or a wearable module. More in detail, temporal and spatial constraints are based on a set of rules stored in the CCU, which are formally defined in Section 5.1.2. In summary, each rule is dedicated to one and only one type of measurement and specifies time and space properties:

- The rule specifies a duration, which is the absolute time interval within which the rule is valid, and a sampling time span, which indicates how frequently the measurement should be updated (within the same zone).
- The rule specifies a zone of validity, which is the absolute zone within which the rule is valid, and a sampling area granularity, which indicates the spatial granularity of each measurement, that is, the area size outside which each measurement is not considered to hold anymore and, thus, it needs to be updated because it belongs to another sampling area.

Through our system, users can decide to subscribe to multiple campaigns issued by stakeholders instead of sticking to the default rules. In fact, each type of measurement has its own default rule, however, stakeholders can add to the database their own rules, overriding the default ones, so that the participants subscribed to a campaign of a stakeholder switches to a different update rate and can potentially face a heavier resource consumption. On the other hand, stakeholders offer a dedicated revenue to users in order to balance out the (often) higher participation rate that they ask. As an example, a mobile network operator might be interested in knowing the average cellular throughput (a costly resource in terms of cellular traffic for the users, because they would need to perform data bursts) with a spatial constraint of 10 meters and a temporal constraint of 5 minutes in the zone around a point of interest (such rate is likely to be significantly lower in the default rules). After a configurable number of uploads from a single client,

it can offer a discount on the monthly phone plan for a subsequent period, which represents a possible incentive pushing users to collaborate to the data collection. It is worth mentioning that similar spatio-temporal control rules mechanisms have been used for sensors and actuators such as in [15] and [11], however, the interaction with the stakeholders and the way in which spatio-temporal rules can be included into each other has not, to the best of our knowledge, been presented so far.

5.1.2 The Central Coordination Unit

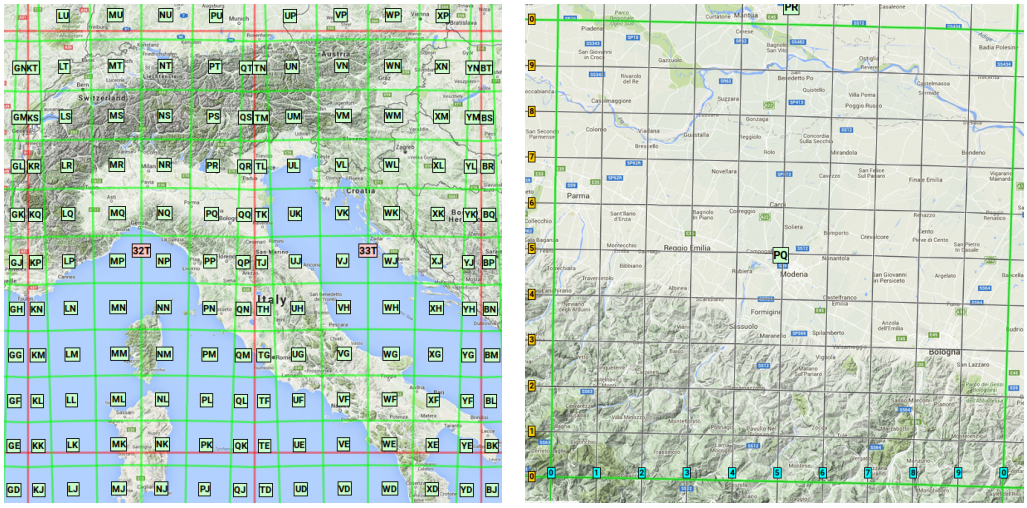
As we already stated before, all the computational burden generated by our ecosystem is concentrated on the CCU, which has always the control over the monitored scenario. A section of its data store is dedicated to the set of rules that determine the CCU's responses, which, as said, include time and space constraints for the user's next updates. More in detail, the CCU receives a set of measurements from the sensing infrastructure and, for each of them, sends back to the respective user a timer and an area that characterizes where and when such measurement is valid, thus, implies in which cases the next update is required.

Timer for Measurement Update

A time constraint is represented by a timer after which the relative resource is no longer valid and it is required to be updated. Clearly, some measurements are supposed to change more often, hence different timers are associated to different sensors. For instance, the microphone, used as noise sensor, is expected to be queried often, since the ambient noise can change rapidly and instantaneous measurements are likely to give an accurate average value when their number is not small. On the other hand, we expect air pressure values to change over a long period of time, thus only few measurements per day are needed from a single user. Time constraints can be fixed or change over time according to a set of other options, for instance, the amount of measurements received recently from a certain zone. In this section we do not deal with such case, which is shown in detail in Section 5.2, thus, for simplicity, we consider the time constraints to be fixed.

Zone for Measurement Update

Similarly to the time constraint, we use a spatial constraint in order to cope with mobility within our scenario. For the sake of zone labeling, we use Military Grid Reference System (MGRS) [156], which hierarchically encodes the world map in square areas. As it can be seen in Figure 5.2(a), the world is divided in 6° by 8° rectangular geographic areas (except for a few cases close to the northern



(a) (b)

Figure 5.2: Italy and the city of Bologna represented through the MGRS scheme. In Figure 5.2(a) the system is at a coarse granularity, representing the 6° by 8° GZIs and their irregular division into 100 km sided squares. In Figure 5.2(b) the 100 km sided square around the city of Bologna is represented, with its regular 10 by 10 square division [149].

and southern poles), uniquely identified by a Grid Zone Identifier (GZI), encoded in two digits and one letter. Each of these grid zones is subdivided in 100 km sided squares, identified by two more letters after the GZI. Such division is imprecise, in fact some squares are cut when crossing a grid zone edge. This happens unavoidably due to the sphericity of the planet, for which, on a large scale, it is impossible to set up a perfect squared grid. Each of the above mentioned squares, as can be seen in Figure 5.2(b), is then further subdivided in 100 squares, distributed on a 10×10 grid, each with a 10 km side. All of them are identified using two more digits (one for the x and one for the y coordinate) and further subdivided repeatedly and hierarchically with the same system with a maximum precision of 1 meter sided squares. Such reference system allows both to uniquely identify squared zones of different granularity and to hierarchically obtain one of the surrounding squares of bigger magnitude setting up a mask on the identifier itself. Finally, a maximum precision of 1 meter is enough for our purposes, also given the possible inaccuracies of the GPS. As an example, the north of Italy is assigned to GZI *32T* and the city of Bologna, which represents the area where we deployed our test, as we will describe in Section 5.1.3, lies into a 100 km sided square identified by *32TPQ*. If, for instance, a smartphone geo-locates itself to be into the square of 10 meters side identified by *32TPQ 5731 2957* (four digits for

the x and four for the y), then we can infer that it also lies into the 100 m sided square identified by *32TPQ 573 295*, as well as the 1 km sided square identified by *32TPQ 57 29*, and so on so forth.

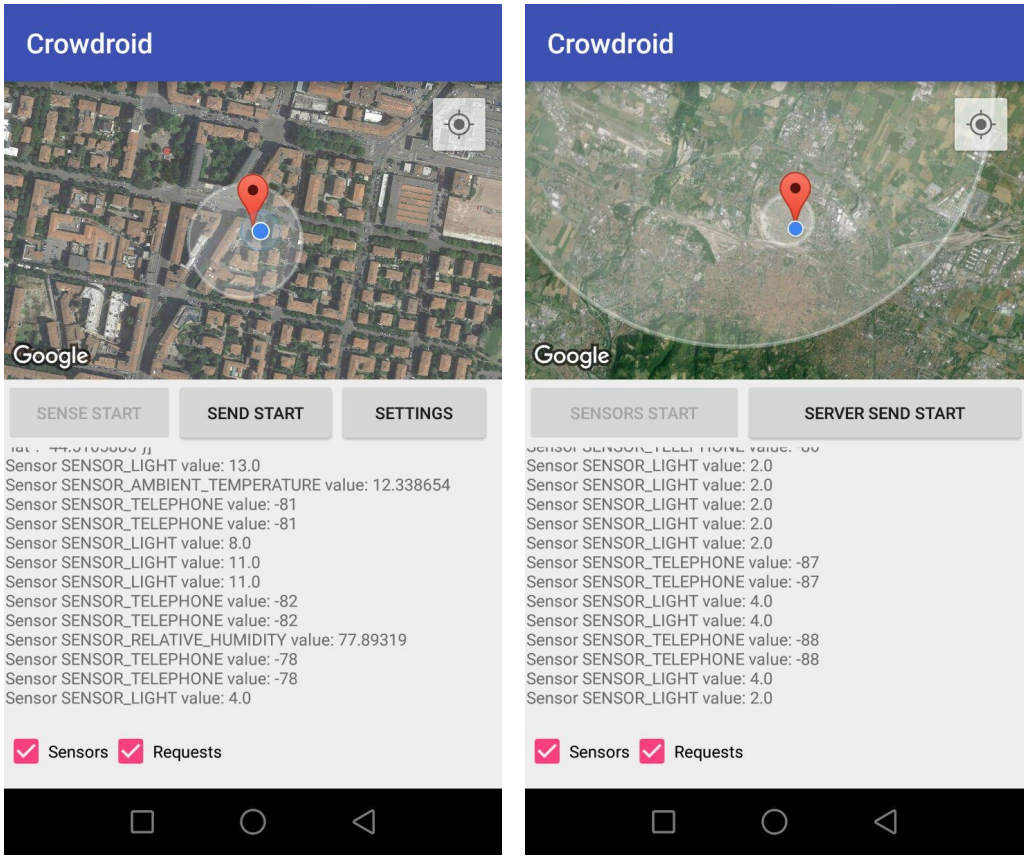
The Rules

Time and space conditions are based on a set of rules stored in the CCU's database, which can be either the default or specified by a stakeholder. In particular, a rule R is defined by a tuple $\langle$*stakeholderID, sensorType, validityArea, sampleGranularity, validityTime, sampleTimer*$\rangle$, where R.*stakeholderID* is the identifier for the stakeholder who owns the rule (or *"default"* if the rule has no owner), R.*validityArea* identifies the MGRS identifier of the square within which the rule is valid, R.*sampleGranularity* defines the area size within which each measurement is valid and spans from 1 (100 km side) to 6 (1 meter side), R.*validityTime* defines the timestamp until when the rule is valid and R.*sampleTimer* defines the time span in seconds within which each measurement is considered valid. We exploit squares of different sizes, depending on the nature of the sensor and the rules stored in the database. In case of multiple rules involving the same sensor for the same area (for example when participating in multiple campaigns requiring similar observations) we always choose the smallest one for the sake of identifying the spatial constraint for the resource. Similarly, the shortest timeout is picked for the time constraint. In other words, we always pick the strictest spatial and temporal constraints as, by satisfying the most demanding rules, we are also able to fulfill the requirements of the coarser ones. We also define two operators. The unary operator $\diamond$ extracts the granularity from an MGRS coordinate. For instance, let C be *32TPQ 5731 2957*; then, $\diamond C = 5$. The operator $\oplus$ is the mask operation: $A \oplus b$ extracts the MGRS area of granularity b containing the MGRS area A. This is easily obtained by removing the last $\diamond A - b$ digits from both the x and the y coordinates of A, which has granularity $\diamond A$. Clearly, $\diamond A \geq b$, otherwise the operation is not possible. For instance, using C defined above, $C \oplus 3 =$ *32TPQ 57 29*.

More formally, let R_1 and R_2 be two different rules characterized by the respective tuples in the database. They are both applicable to the same measurement if and only if they are both valid, i.e. when the following conditions are verified:

$$\begin{cases} \{R_1, R_2\}.\textit{validityTimer} \geq \textit{currentTime} \\ R_1.\textit{sensorType} = R_2.\textit{sensorType} \\ R_1.\textit{validityArea} \subseteq R_2.\textit{validityArea} \end{cases},$$

where $\subseteq$ represents the geographical inclusion between square areas. If $R_1, \ldots, R_n$ are all applicable, then we instruct the participants by sending a configuration with $max\{R_1.\textit{sampleGranularity}, \ldots, R_n.\textit{sampleGranularity}\}$ as the spatial con-



(a) (b)

Figure 5.3: Two screenshots of the prototype client application implemented in Android. They show the geo-fencing active on two instances of the smartphone application. Figure 5.3(a) it resembles the surrounding circle for a 100 m sided square, while in Figure 5.3(b), where the application is subscribed to more than one stakeholder, a second circle for a 10 km sided square is shown [149].

straint and $min\{R_1.sampleTimer, \ldots, R_n.sampleTimer\}$ as the temporal constraint. By performing this operation we assure that all the rules are satisfied.

5.1.3 The Crowdroid mobile application: a prototype implementation

In this section we present the implementation of our proposal and an Android client mobile application developed as a demonstrator. Smartphones belonging to users are potentially a gratuitous set of environmental sensors and the most representative example of MCS. In our scenario they are the core entities collabora-

tively forming the sensing infrastructure, configured by the CCU. As a deployment for our proposed architecture, we built a Python CoAP/REST server together with a MySQL database as the CCU and we also developed an Android application for smartphones, named Crowdroid, as the mobile sensing clients for the scenario. The application, at its bootstrap, checks the presence of the sensors we considered for crowdsensing purposes, performs the respective measurements and sends the whole set of sensed data to the CCU, which returns a spatial and a temporal constraint for each sensor involving the subsequent updates as explained in Section 5.1.2. The sensors and measurements we take into account are: barometer, light sensor, noise sensor, temperature, relative humidity, signal strength (RSSI) and throughput for both WiFi and the cellular technolgy used (LTE, UMTS). In particular WiFi RSSI is useful when assessing connectivity relative to WiFi infrastructural networks (such as the ones used by universities or companies) and public networks. In our case, we detect BSSID, RSSI and SSID of the connected access point. Observation of the RSSI over the cellular infrastructure gives us feedbacks on the connectivity in different city zones and times. Furthermore, we perform throughput tests as measurements using a default of 50 kB burst in uplink and downlink. The latter represents a case for which measurements are very costly if performed frequently. In addition, we note that stakeholders can specify different packet sizes to be downloaded or uploaded for the throughput test, depending on their monitoring purposes. For some of the measurements, due to the Android listeners architecture, are not triggered by the application, thus the actual measurement is not driven by the CCU's command. This is the case of temperature, air pressure, light and humidity, which are updated onto the client whenever a change is detected, while the CCU only instructs the client on when to upload them. The sound level is measured using an internal software timeout and the RSSI measurement is updated internally upon its variation, while the throughput test is performed upon a request by the CCU in order to avoid a resource waste. Each measurement is encoded in a JSON record and forwarded to the CCU using the CoAP application protocol, which is significantly lighter than HTTP and more suitable for large-scale sensing applications [191]. The client receives the update timer and the center as well as the length of half the diagonal of the designed MGRS square delimiting the validity of the measurement. The diagonal of the square is used in order to exploit the geo-fencing facility on the smartphones [187], which allows the application to declare circular areas and send a notification when the client detects itself to get in or drop out of the selected region. As geo-fencing uses circular areas, we established to use the circumscribed circle for each of the square areas, using half the diagonal as the circle's radius. With this method we ensure that no observation inside the MGRS area is lost, even though the spatial precision is lower [31]. A more precise localization would be possible by continuously polling the GPS, which, however, would rapidly deplete the bat-

tery level of the device, hence limiting its practical use and potentially decrease the willingness of users to install the application. A screenshot of the application showing the geo-fencing is shown in Figures 5.3(a) and 5.3(b).

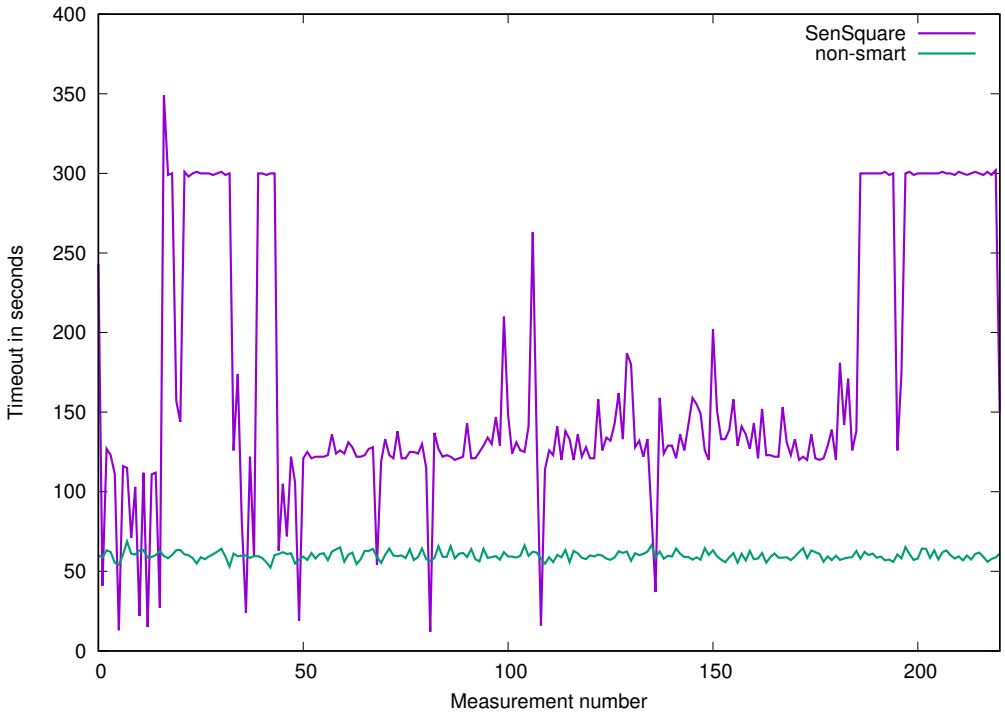


Figure 5.4: Diagram showing the variability of the update time over 230 measurements compared with the ideal non-smart approach, highlighting the different granularities that can be configured with MoCroSS [149].

We select as a test case the city of Bologna, which happens to be split in two 10 km sided squares in the MGRS system, one identified by *32TPQ 8 2* (the southern one) and the other by *32TPQ 8 3* (the northern one). We set up the default rule RL for the light sensor using $RL.\textit{sensorType} = Light$, $RL.\textit{sampleTimer} = 300$ and $RL.\textit{sampleGranularity} = 2$, that is, a 100 km sided square. Finally, we introduce a stakeholder interested only in the northern half of the city, which declares a new rule RS, setting up $RS.\textit{sensorType} = Light$, $RS.\textit{sampleGranularity} = 4$, that is, a 100 m sided square, and the area of validity $RS.\textit{validityArea} = \textit{32TPQ 8 3}$. The diagram in Figure 5.4 shows how the timer for the update varies over a time span of about 10 hours for a smartphone owned by one of the test users. More in detail, on the x-axis the updates in chronological order are labeled with an increasing index, while on the y-axis we show the number of seconds elapsed from the previous update. It is clear how, while the user is within the southern square, the updates

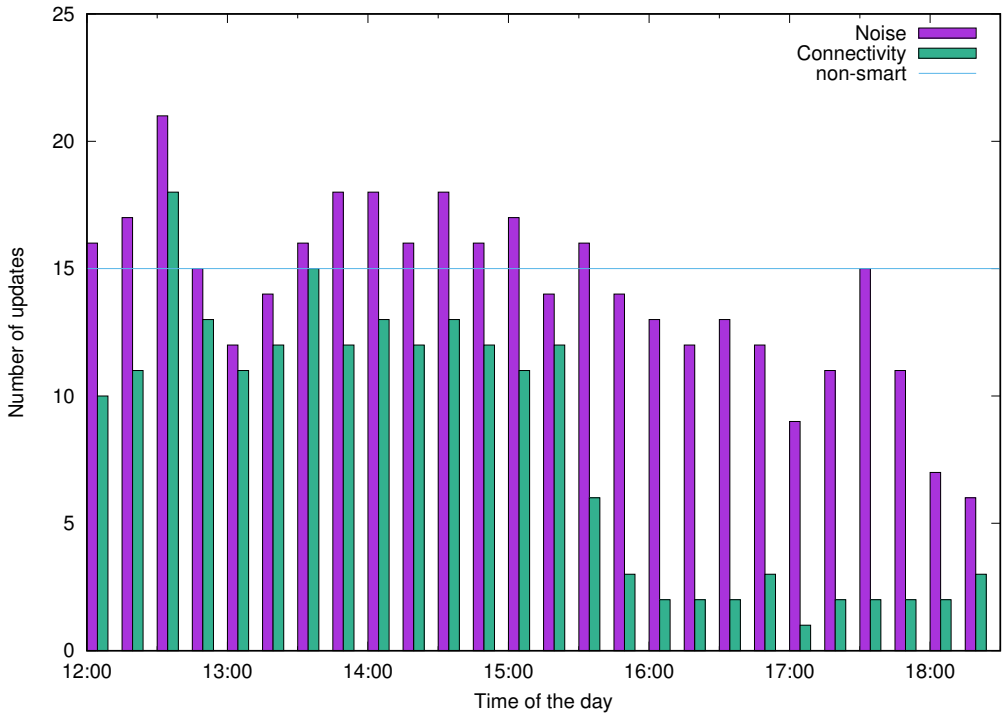


Figure 5.5: Histogram showing the number of updates required by MoCroSS to two different sensors, grouped in 15 minutes time slots [149].

are triggered at a regular interval and mostly regulated by the time constraint. In such case, few irregularities are found, probably due to connection failures (high spikes) or sporadic geo-fencing border crossing (low spikes). On the other hand, while the user is within the northern square (where the train station is located), the spatial constraint appears to be more effective, making the device send updates more frequently. During this period, which approximately relies within the interval $[40, 180]$, high spikes probably denote a user moving slowly or standing still, while low spikes denote a user walking fast and/or crossing a small lateral portion of a 100 meter sided square. If compared to a non-smart application, which sends data at a constant rate, the difference is evident. We assumed a rate of 60 seconds for the non-smart approach, which guarantees a constant update, however it is clear from the figure that the same number of updates covered a period of about 3.5 hours, thus, a lot more updates are required, resulting in a possible waste of mobile device resources.

In Figure 5.5 we show a time-based analysis concerning the amount of updates for MoCroSS and its non-smart counterpart, considering a sample user reporting data to our infrastructure. We performed a test on two different measurements in the southern square of Bologna (i.e. identified by the *32TPQ 8 2* MGRS square).

We set the rule RN noise sensor (i.e. $RN.$*sensorType* $=$ *Noise*) with a default time constraint of $RN.$*sampleTimer* $= 80$ seconds, while the default rule RC for the cellular RSSI (i.e. $RN.$*sensorType* $=$ *CellRSSI*) has a time constraint of $RC.$*sampleTimer* $= 300$. The considered stakeholder requires for both the measurement a spatial constraint of 10 meters. During the first part of the time span (before 3:00 PM) it is evident that the update rate is more spatial-driven, because of the user was walking around, while when the user stands still, in the second part of the time span (after 3:00 PM), the update rate is more time-driven. This behavior can be clearly seen by the noise bars reporting the data around 15 times per quarter-hour, while the connectivity report is given around 3 times per quarter-hour. On the other hand, the non-smart approach requires the same update rate regardless of the spatial conditions, asking for useless measurements, as the horizontal line at 15 updates per quarter-hour shows.

5.1.4 Wrap Up and Future Perspectives

In this section we presented MoCroSS, a MCS architecture for collaborative IoT and its application to a case study performed with mobile devices. We outlined the advantages that MoCroSS brings both for users and stakeholders in terms of saving resources, limiting the amount of data to be published while still keeping its precision and pushing the users to collaborate through monetary incentives. We also developed an Android mobile application, namely Crowdroid, as a demonstrator in order to prove its effectiveness, comparing it to its non-smart counterpart. The advantages of a Smart MCS architecture over the non-smart proposal is evident, both in terms of measurement precision as well as resource waste.

We envisage as a future work the inclusion of machine learning techniques addressed to establish the needed number of measurements in a specified zone, determined by the data variance and the number of reporting users. In this way the rules will shift from static to dynamic, outlining a self-adapting elastic IoT ecosystem. The next section (Section 5.2) outlines a distributed algorithm to control thedynamically the data collection in an area. We also want sensing clients to produce data either by sensing or enter the observations through a participatory approach. Although the latter requires a more active user participation, it is still supported by our architecture, in fact, clients can submit their data manually and gain consequently a reward, provided that the information is relevant and trustworthy. Data reliability could be based on other user's feedbacks, but we leave this mechanism as a future work as it is out of the scope of this section. Furthermore, in a future development we foresee to use rules to identify malicious users, attempting to corrupt the data pushing more frequently than requested and with measurements far from the mean values.

5.2 Distributed Data Collection Control

Recalling what has been discussed in the previous section (Section 5.1), there is currently a huge interest in monitoring relevant environmental and urban phenomena. Most of the times, the costs for such activities may be unpredictable and, often, investing dedicated resources results in a consistent loss of money. Nevertheless, tasks such environmental and infrastructural monitoring are considered of paramount importance for predicting events, avoiding dangers and providing dedicated and improved services to citizens. In this section, as in the previous one, we focus on the usage of MCS in such use cases and address the problem of regulating the amount of data to be transmitted by the participants. In fact, according to the definition of the *Curse of Sensing*, controlling properly the data flow results in an overall energy saving, since devices are discouraged to transmit when not necessary, and limits data redundancy while still satisfying the required coverage.

In light of this, in this section we present the work published in [147], a distributed probabilistic algorithm that aims to control the amount of data generated and transmitted by participants, so that it tends to a defined value. The scenario we take into account is the same as MoCroSS, previously introduced in Section 5.1. It belongs to a poorly investigated category of MCS architectures; in fact, the coordinator is unable to infer the number of participants in the zone, as their position is not continuously tracked. Therefore, we designed a distributed algorithm capable to regulate the number of observations over time towards a desired value without the need of extended common knowledge about the scenario. In particular, the algorithm falls in the category of "Distributed Heuristics" discussed in Section 2.5.4. Another key parameter we take into account is the fairness: as probabilistic solutions cannot assure that participants contribute equally, we are interested in granting as much fairness as possible in order not to rely on few devices and leave others underutilized. This is important when considering energy depletion, since, without fairness, we would only care about the overall energy without considering how much individuals consume. We assess the performances of our proposal through extensive simulations in order to estimate its effectiveness and its adaptability. We also take into account macro mobility of users using a publicly available mobility dataset for pedestrians through which we will show the algorithm's reactiveness to changes.

The work presented in this section differs substantially from the related works enlisted in Section 2.5.4 and brings a novel and lightweight approach to a barely investigated problem. The scenario that we are taking into account is characterized by the following novel features:

1. Our ecosystem is user-oriented and push-based. Participants and their devices are not instructed directly by any entity, instead, they decide where and when to contribute with their data, upon the suggestion of stakeholders and actors who can provide incentives, as indicated in Section 5.1 as well as in [149] and [148]. As a consequence, we cannot identify the scenario as task-oriented, since no task is effectively assigned to users as in other works. Instead, tasks are normally continuous and, despite they can be attributed with an expiry date, they do not have the notion of being completed, since they refer to periodic environmental measurements.

2. The platform does not have any direct control over the users, meaning that any information pushed by a participant is atomic and the participant is not tracked. Although the central entity knows who uploads each observation, it cannot infer who is in the interested zone at a defined moment, thus it has a limited knowledge about the number of participants in such area. This distinguishes the present work from the ones presented in the literature, where the position of each participant is typically known. We decided to pursue this choice because it is way more privacy-friendly compared to approaches that explicitly task individual nodes.

3. The system is centralized and assumes an infrastructure-based communication technology (e.g. LTE), thus we do not focus on communication issues that may occur.

The section is divided in the following structure: Section 5.2.1 defines the problem and all its parameters, Section 5.2.2 illustrates the baseline algorithms we use and the one we proposed, Section 5.2.3 describes how we carried out our evaluation tests and their results and, finally, Section 5.2.4 gives a summary of what has been discussed here.

5.2.1 Problem Statement

Given the assumptions, the background and the ideal goal that we outlined in Section 2.5, the essence of this work is concentrated on minimizing the number of data transfers in excess performed by participants. The priority of the central platform remains, however, the achievement of a defined number M_0 of observations within a given time span, a scenario strongly subject to the *Curse of Sensing*. Furthermore, the systems aims to homogenize the contribution given by each actor, that is, maximize the level of fairness of the ecosystem. Through the rest of this section, we assume to be part of an MCS platform such as the one presented in Section 5.1, thus, we can make the following assumptions:

- With respect to the MGRS encoding, on top of which our platform in [149] and [148] is built, we can assume that our algorithm operates within a square area in which every observation is assumed to refer to the same phenomenon (e.g. if the participants are measuring the temperature within a $100\ \mathrm{m} \times 100\ \mathrm{m}$ square, the temperature values are assumed to be homogeneous).
- The participants communicate with the central entity via IP-based technologies, therefore connection issues and delays are not considered, since technologies such as WiFi and LTE are assumed to provide enough coverage to the whole area.
- The central entity does not track the identity of the participants posting observations, it only receives such observations seamlessly. As a consequence, it has no clue about the number of active participants within the monitored area, nor it can estimate it from the number of observations.

Given such assumptions, we can model the problem as N different stations that adhere to the MCS campaign and perform observations against a given phenomenon. Such number N can vary over time due to mobility, in particular, participants may leave the interested area, whereas new ones may join it. We assume to split our timeline in time slices Δt_i, that represent the atomic units during which a station cannot transmit more than once due to internal clocks. We also assume that the stations will send periodically observations relative to a certain resource Ψ_0. The central entity's goal is to obtain exactly M_0 observations about Ψ_0 within every time window T_i, the length of which is given by $|T| = w$. We follow the approach of the sliding window, thus $T_i = \{\Delta t_{i-w}, \ldots, \Delta t_i\}$, this means that T_i and T_{i-1} are overlapping by $w - 1$ time slots. The central entity displays the performances of the data collection through a Satisfaction Index (SI), which is calculated upon each time window T_i and it is defined as $SI_i = \frac{m_i}{M_0}$, where m_i is the actual number of observations received within T_i. The aim of the central entity is to obtain a SI equal to 1 (or as close as possible).

5.2.2 Proposed Algorithm

It is easy to observe that, from the point of view of the central entity, estimating the number of participants within the interested area is an ill-posed problem. For such reason, we propose a probabilistic distributed algorithm that reaches asymptotically the problem's fixed point. In order to achieve it, we make use of two different variations of the well known Asymptotically Optimal Backoff (AOB) algorithm, which was introduced and validated firstly in [25], although originally studied for a different problem. Such algorithm was developed for estimating the optimal backoff for a number of stations aiming to transmit to an access point

(AP) in CSMA/CA-based environments, such as IEEE 802.11. In brief, the basic algorithm works as follows: stations are provided with the SI_i, relative to t_i, of the central entity (the AP), which spans from 0 to 1; subsequently, the stations calculate the probability of transmitting at the step t_{i+1} as $P_{i+1} = 1 - SI_i$. This way, the more the central entity is occupied, the more stations are discouraged to transmit and vice versa. However, such algorithm was designed and tuned for environments in which only one station could transmit at a time due to collisions, whereas, in our scenario, all participants are allowed to transmit in the same time slot. We avoid further details for space constraints, and we refer the interested reader to [25]. Hence, here we present two variations of the AOB algorithm that we designed in order to achieve the goals outlined in Section 5.2.1.

Asymptotic Opportunistic algorithm for Satisfaction Index (AO-S)

The balance provided by legacy AOB method is not enough for our scenario to reach a SI close to 1, thus we design a booster mechanism that pushes the number of transmissions when the SI tends to stabilize at a low level. Conversely, it hinders further the transmissions whenever the SI is too high. Given such premises, we define the transmission probability as:

$$P_{i+1} = 1 - SI_i^b \tag{5.1}$$

where b is defined as the booster factor, which is modified iteratively until SI reaches a satisfactory point of stability. In particular, we aim to force $SI \in [\sqcap_{SI}; \sqcup_{SI}]$, where $\sqcap_{SI}$ and $\sqcup_{SI}$ are, respectively, the lower and upper bound for considering the SI acceptable. In order to estimate the point of stabilization of the SI, we use the average SI over its last θ values:

$$ASI_i = \frac{\sum_{j=i-\theta}^{i} SI_j}{\theta} \tag{5.2}$$

Ideally, we aim to increase or decrease b with respect to the current value of ASI iteratively, that is, AO-S algorithm periodically runs Algorithm 3, which increments or decrements b by one unit per stage, however, b does not follow the integer numbers' succession. In particular, we define the function $inc(b)$ and $dec(b)$ as:

$$inc(b) = \begin{cases} \frac{1}{(1/b)-1} & \text{if } b < 1 \\ b + 1 & \text{otherwise} \end{cases} \tag{5.3}$$

$$dec(b) = \begin{cases} b - 1 & \text{if } b > 1 \\ \frac{1}{(1/b)+1} & \text{otherwise} \end{cases} \tag{5.4}$$

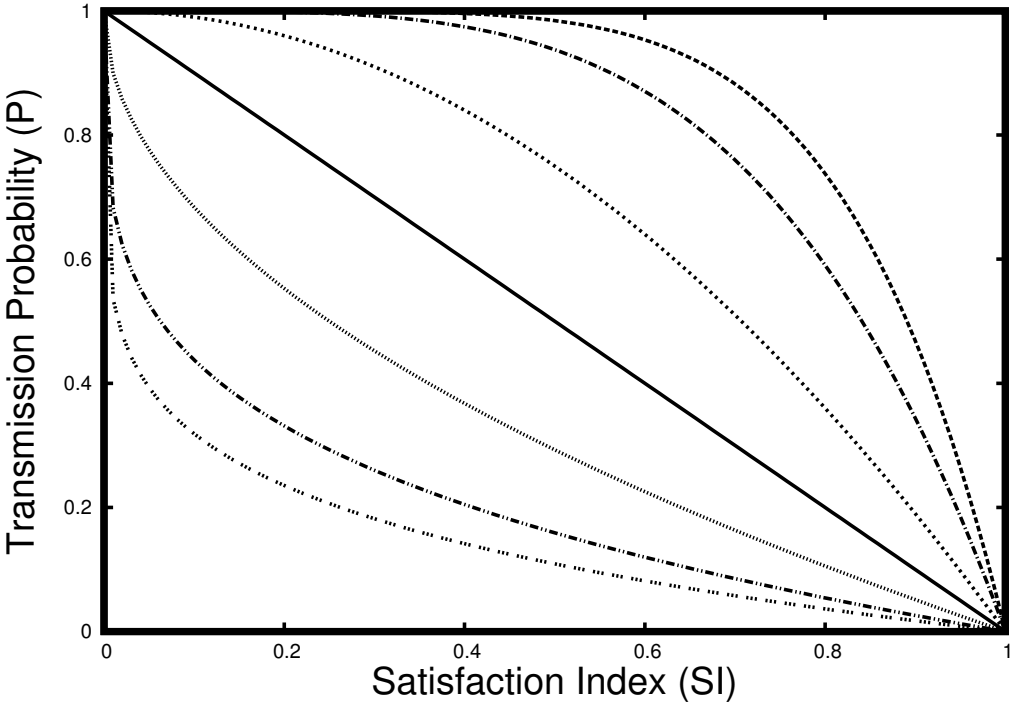


Figure 5.6: Diagram showing the probability curve with respect to the last received value of the SI. The solid line represents the probability calculated upon $P_{i+1} = 1 - SI_i$, the other curves show $P_{i+1} = 1 - SI_i^b$ for different values of b [147].

In other words, the scale of values for b is designed to enable different curves of increased (or decreased) steepness, forcing higher or lower values of the probability, as shown in Figure 5.6. In conclusion, we set the goal of AO-S as minimizing the deviation index DI, i.e. the difference over time (up to a time slot t_τ) between ASI and 1:

$$DI = |1 - \frac{\sum_{i=0}^{\tau} ASI_i}{\tau}| \tag{5.5}$$

Asymptotic Opportunistic algorithm for Fairness (AO-F)

While AO-S algorithm is committed to grant the achievement of a SI close to the optimum, it still lacks fairness. In other words, there is no evidence that the load of transmissions is balanced equally among all the participants. For such reason we adopt a fair variant of the AOB algorithm, leveraging a concept introduced in [25], defined as AO-F. In particular, we define the transmission probability as

$$P_{i+1} = 1 - SI_i^{1+k}$$

where k is the attempting factor, tuned upon the number of time slots elapsed since the last transmission, i.e. the number of consecutive time slots for which the participant decided to back off. More in detail, k is calculated as $k = \lfloor \log_2(j) \rfloor$, where $j > 0$ is the number of t_i slots elapsed since the last transmission. It is clear that such mechanism pushes stations that experienced a significant number of consecutive backoffs to transmit with a higher probability than the others. More specifically, with the AO-F algorithm, we aim to maximize the transmission fairness among the participants, for which we adopt Jain's Fairness Index (JFI) [96]:

$$JFI = \frac{(\sum_{h=1}^{N} x_h)^2}{N \cdot \sum_{h=1}^{N} x_h^2} \tag{5.6}$$

where x_h indicates the total number of transmissions performed by the participant h.

Combining the approaches

In this section we provide a combination of the AO-S and the AO-F algorithms, which we define as Asymptotic Opportunistic for Joined Fairness and Satisfaction index (AO-JFS). We aim to:

1. Minimize the DI, which has an optimum value of 0 .
2. Maximize the JFI, which ranges from $1/n$ to 1.

Hence, we define the transmission probability as

$$P_{i+1} = 1 - SI_i^C \tag{5.7}$$

where C is given by the combination of b and k as $C = inc^k(b)$, with $f^n(x)$ indicating the iterative composition as $f^n(x) = f \circ f^{n-1}(x)$. After extensive experiments we found that, for different values of b, k can sometimes be disruptive for the DI. For such reason we redefined k on top of b as follows:

$$k = \begin{cases} \lfloor \log_2(j) \rfloor & \text{if } b < 1 \\ j \cdot b & \text{if } b \geq 1 \end{cases} \tag{5.8}$$

The resulting AO-JFS algorithm, formalized in Algorithm 3 calculates the b value at each time slot and it is performed by the central entity, which is committed to broadcast such value to the participants. Each participant, at each time slot, calculates its own probability of transmission on top of the received value of b and its own j. Such procedure is outlined formally in Algorithm 4.

Participants are allowed to operate in two different modes:

Algorithm 3 Update the booster value b

calculate new SI
if $SI > \sqcap_{SI}$ **then**
  $b := dec(b)$
else if $SI < \sqcup_{SI}$ **then**
  $b := inc(b)$
end if
broadcast(SI, b)
sleep($|t|$)

Algorithm 4 Calculate locally the probability of transmission

receive(SI, b) {Only in active mode}
if $b < 1$ **then**
  $k := \lfloor \log_2(j) \rfloor$
else
  $k := j * b$
end if
$C := b$
for $i := 1$ **to** k **do**
  $C := inc(C)$
end for
$P := 1 - SI_i^C$
transmit with probability P
if transmitted **then**
  $j := 0$
  receive(SI, b) {Only in power save mode}
else
  $j := j + 1$
end if

1. Participants operating in **active mode**, at each time slot t_i, as indicated in Algorithm 4, perform a listening phase in which they capture the broadcast message from the central entity. Thus, the SI and the b values are updated constantly.

2. Participants operating in **power save mode**, as indicated in Algorithm 4, obtain the updated values of SI and b only as a response to their transmission. This way, they are not updated in real time at each t_i. It is clearly a disadvantage for the sake of performances; however, it is a viable choice for participants willing to consume as less as possible while still being part of

the sensing community.

5.2.3 Simulation

In this section we demonstrate the effectiveness of our algorithm through extensive simulations. In this work we perform our evaluation study over both a static scenario, i.e. where the number of participants in the interested zone does not change over time; and a mobile scenario, i.e. where the number of participants changes according to mobility traces. In particular, we assessed the performance of the AO-JFS algorithm presented in Section 5.2.2 in comparison with AO-S and AO-F algorithms as baselines.

Static Scenario

Within the scope of the static scenario, we took into account both sparse and dense environments; in particular, we assigned to N the values 5, 10, 20, 50, 100, 200. The time slot t is set to 10 s and every participant performs a single transmission decision located randomly within such time span. T is set to 30 and M_0 is set to 100. We set the upper bound $\sqcap_{SI}$ to 1.15 and the lower bound $\sqcup_{SI}$ to 0.95 and we also limited b to a minimum of $1/8$ and a maximum of 50. In order to avoid extreme behaviors we needed to force P to never assume the values 0 or 1. For such reason, we introduced 0.001 and 0.999 as respectively lower and upper bound for P. Given such parameters, we performed a consistent number of simulations for each chosen value of N assuming participants operating both in active mode and in power save mode.

Mobile Scenario

We represented participants' motion through the use of macro mobility – we only consider the events that cause participants to exit and/or enter the interested zone – considering all of them as pedestrians. In particular, we used the "KTH Walkers Dataset" of pedestrian traces [81], that have been generated using different levels of density using the urban area of Östermalm – a district in Stockholm – as a location. We manipulated the traces in order to extract the instants where participants either join or leave the area. For the purpose of the present work, we selected one trace such that, after a transient in which participants are spawned, it reaches a steady state which counts around $N_0 = 200$ participants on the average at the same time. In order to reach comparable conditions with the static scenario, we introduced, for each join event, a probability $P_s = 1 - (N/N_0)$ of blocking the creation of the relative participant for each value of N used in the static scenario. Each other parameter is set as in the static scenario.

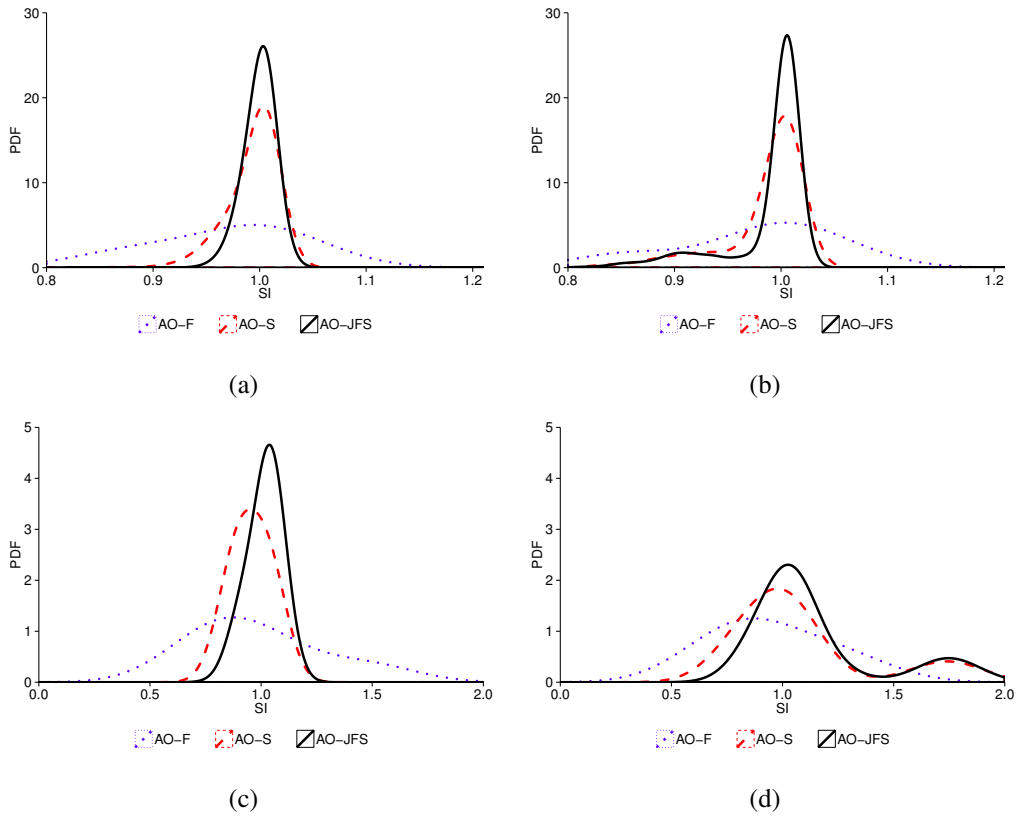


Figure 5.7: Diagram showing the aggregated PDF with respect to all the collected values of the SI over different values of N in a static scenario (a) and a mobile scenario (b), with all nodes operating in active mode. The case of nodes operating in power save mode is given in (c) for the static case and (d) for the mobile analysis.

Minimizing the deviation

Here, we focus on the goal of obtaining the number of observations M_0 that the central entity requires, i.e. minimizing the DI. Figures 5.7(a)-5.7(b) show the performance of the three algorithms with all the participants operating in active mode; in particular, the aggregated Probability Density Function over all the used values of N is shown with respect to the SI. In particular, we collected all the SI values for a 3-hours simulation leaving out the first λ values, which are subject to the transient bias, as we are interested only in the steady state behavior of the SI. In all our simulations we set $\lambda = 100$. In general, it is clear how AO-F fails in achieving a good value of DI as it is too dependent on the number of participants and the SI assumes many more different values. As a matter of fact, in sparse scenarios, i.e. when N is small, the SI gets stuck to a value lower than required;

Table 5.1: DI and JFI with participants in active mode [147].

	$N = 5$	$N = 20$	$N = 100$	$N = 200$
Static				
DI (AO-S)	0.0389	0.0114	0.0029	0.0066
DI (AO-F)	0.3101	0.0404	0.0327	0.4796
DI (AO-JFS)	0.0314	0.0011	0.0038	0.0133
JFI (AO-S)	0.9761	0.8711	0.6430	0.5529
JFI (AO-F)	0.9999	0.9870	0.8458	0.9843
JFI (AO-JFS)	0.9796	0.8803	0.8395	0.7513
Mobile				
DI (AO-S)	0.1196	0.0198	0.0048	0.0038
DI (AO-F)	0.3877	0.0660	0.0056	0.0555
DI (AO-JFS)	0.1166	0.0057	0.0058	0.0037

conversely, in highly dense scenarios the SI is too high. With respect to such metric, we observe that AO-S and AO-JFS reach successfully a high number of SI values close to 1 for mostly all the values of N. Comparing Figures 5.7(a) and 5.7(b), respectively the static and the mobile scenario, we observe that AO-S is more affected by mobility rather than AO-JFS. This suggests that AO-JFS is more robust against changes and quicker in reverting to a steady state when a perturbation occurs. We can observe from Table 5.1 that DI is minimized in AO-S in dense scenarios, whereas in sparse scenarios it is minimized in AO-JFS. In any case, AO-JFS and AO-S perform similarly, while AO-F performs poorly.

Figures 5.7(c)-5.7(d) show the performance of the three algorithms with all the participants operating in power save mode, using a PDF as in the previous case. Since the power save mode yields by assumption worse performances, here we used a different scale of values on the axes in order to better highlight the differences. In fact, we can observe that, as expected, the lines tend to cover many more values due to the inaccuracy of the updates received by each participant. Furthermore, as in the previous case, AO-F fails in reaching an acceptable value of SI, especially in sparse and dense scenarios. We also notice that AO-S, similarly to the active mode, does not achieve a precision as good as AO-JFS does. In addition, we observe in Figure 5.7(d) that many values of the SI tend to cluster in another spot. This is mainly due to new participants joining the area, who do not receive the value of the SI before any transmission, thus they transmit for the first time upon a default value of P even when the SI is already too high. We can notice in Table 5.2 that, similarly to the active mode case, AO-S and AO-JFS share almost equally the minimum DI, although the overall performances of the power save mode are poorer than the ones for the active mode.

Table 5.2: DI and JFI with participants in power save mode

	$N = 5$	$N = 20$	$N = 100$	$N = 200$
Static				
DI (AO-S)	0.1749	0.0906	0.0199	0.0843
DI (AO-F)	0.3124	0.1813	0.1346	0.5629
DI (AO-JFS)	0.0750	0.0020	0.0728	0.1351
JFI (AO-S)	0.9582	0.7828	0.4290	0.3781
JFI (AO-F)	0.9993	0.9611	0.9670	0.9854
JFI (AO-JFS)	0.9986	0.6715	0.6039	0.5917
Mobile				
DI (AO-S)	0.2402	0.0190	0.0805	0.7587
DI (AO-F)	0.3661	0.1284	0.3155	1.5702
DI (AO-JFS)	0.1597	0.0200	0.1589	0.7866

Maximizing the fairness

Here, we focus on maximizing the fairness of the system, i.e. maximizing the calculated JFI for each configuration. Since AO-F is specifically designed for fairness, it performs always better than the others in terms of balance. We can observe it from Tables 5.1 and 5.2, in which the JFI value for AO-F is higher than the others. However, such algorithm is itself not sufficient to achieve a good value of DI, thus, we aim to consider solely the other two algorithms as potential candidates. We can notice that, apart for only one case, our AO-JFS performs significantly better than AO-S in terms of fairness. We did not collect the JFI values in the mobile case, since it would need more microscopic information as participants do not linger in the interested zone for the same period of time. Fairness degrades naturally with the increase of participants because, assuming that they transmit with the same probability, fewer participants would more naturally balance, as each contribution tend to matter more when calculating the SI.

5.2.4 Wrap Up and Future Perspectives

In this paper we have presented a distributed probabilistic solution for achieving a satisfactory amount of observations in opportunistic MCS scenarios that addresses the *Curse of Sensing* problem. In particular, we defined our goals as the closeness of the number of observations to a certain value as well as the fairness among the participants, in order to grant a minimum overhead of messages, a maximum balance of the messages among participants, and the achievement of a desired number of observations per time unit. We introduced the MCS problem relative to a scenario different from the vast majority of the ones present in

literature and we provided two distributed algorithms as a baseline for solving the problems. We also defined a combined approach using such algorithms and we showed that it has good performances over all the problem requirements. Future works are oriented to extending such work to a multi-sensor case and across multiple areas. Furthermore, we plan to integrate the participatory case, which is particularly useful when, despite the distributed algorithm, some areas remain uncovered.

Chapter 6

SenSquare: a Collaborative IoT architecture for Smart Cities and Environmental Monitoring

This chapter outlines our contributions in prototyping frameworks and platforms that take advantage of the data collected through CAPs. Specifically, it discusses SenSquare, our demonstrator in which we developed a service-oriented layer devoted to compose customized services using the data that we collected collaboratively. This Chapter is written on top of our work in [148].

6.1 The Framework

As we first introduced in the preface to Chapter 2, a key feature of a complete IoT ecosystem its the ability to get knowledge out of the data. Canonically, the main trend in tackling such challenge relies in service composition and discovery [13], typical features of a SOA. In fact, pretty much all the IoT high-level architectures and middlewares proposed so far have a SOA connotation. In the same way, following up our research question in Chapter 3 – i.e. how do we offer the information that could be inferred from the multitude of raw datastreams collected through CAPs –, we designed a SOA middleware that is responsible for aggregating raw datastreams in composite services and deliver them to the final user. In order to do so, in accordance with the philosophy that we conveyed throughout the whole dissertation, we enable the service and knowledge sharing among all the users of our ecosystem (being them privates or companies). Actors in our ecosystem can be participants and provide their sensed data through MCS and Open Data paradigms (as in Chapters 4 and 5). They can also be stakeholders and provide a revenue soliciting more contribution regarding certain areas or certain types of data that are more of interest than others (as in Section 5.1). Whichever of these role they cover (if any), they can also be final users of our platform, who create, share and use customized services that actually transform datastreams in valuable knowledge for their needs and the common benefit.

The effectiveness and the exploitability of this paradigm in a plethora of IoT application, both in the rural and the urban context, is demonstrated through the real implementation of our prototype platform: SenSquare. It has been under development and improvement for some years and the architecture changed slightly as we introduced new components. The first version was introduced in [148], which was our pioneering effort in the field of Collaborative IoT and, since then, we have explored all its components (many of them presented as contributions in previous sections), until we reached the current version. We recall that this is not a proposal for a new IoT architecture to which we expect public and private entities to adhere, rather, it is a prototype that demonstrates the potential of a CAP-based solution for IoT applications. In fact, we still consider all the previous or different client interfaces that we developed throughout our research work to be valid contribution and possible options, even though we made significant changes over time. Nevertheless, when we refer to SenSquare, we specifically focus on the current version of SenSquare.

The overall architecture is presented in Figure 6.1, which we can conceptually separate in three different areas:

1. The **data gathering** part is devoted to collect data through CAPs. It occu-

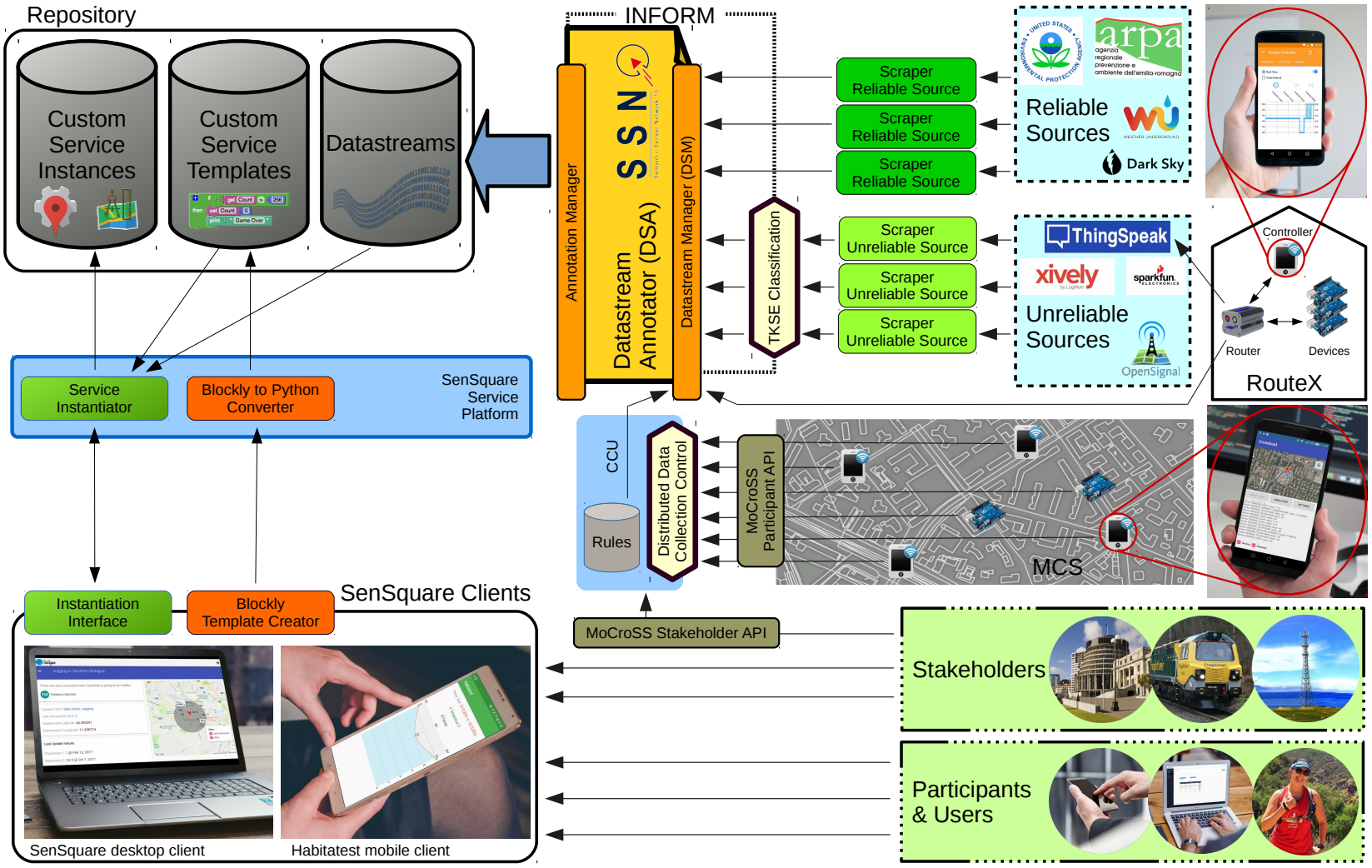


Figure 6.1: Architecture of the whole ecosystem together with all of its external and internal components and actors.

pies the whole upper right corner of the figure and it includes all the components that are actually committed to produce data. Specifically, it includes reliable and unreliable Open Data repositories, together with the classification infrastructure, as well as the whole MCS area, together with the CCU and the architectural components that manage the data collection in MCS. The gateway to other architectural sections is the INFORM architecture, in particular the Annotation Manager, which outputs annotated and uniform datastreams.

2. The **data aggregation** part is committed to unify data coming from different sources and forms a layer of abstraction that transforms raw data in complex services. It is the core of the SenSquare architecture, where the service composition takes place. It is located in the upper left corner of the figure and includes the persistent storage system (i.e. the three databases shown in the figure) and the SenSquare service platform. This macro component takes in input the unified raw datastreams and outputs aggregated and ad-hoc services, shared with the community and based on the needs of users.

3. The **user experience** part provides the users with an access to the system and allows them to create, share and make use of customized and dedicated services, depending on their needs as an individual or a company.

We will discuss these components in detail throughout this section.

Data Gathering

The lowest layer of the architecture of SenSquare is in charge of retrieving useful datastreams from publicly available resources and potential contributors. This task is clearly non-trivial due to the heterogeneity of the data sources as well as their potentially variable data quality, an issue that we discussed extensively in Chapter 4 and 5.

Within the scope of public Open Data (Chapter 4), the only way to retrieve all the possible datastreams is to construct a dedicated data scraper for each source that periodically performs HTTP requests in order to extract the updated data points from the web. In the figure we represented both reliable and unreliable Open Data Sources, with some examples for each category, as well as their dedicated scrapers in dark and light green boxes. Data scrapers are scripts that run periodically and update the knowledge base with fresh data points. Both reliable and unreliable resources have been tested with success, in particular, we extracted air quality datastreams from the Regional Agency for the Protection of the Environment in the Italian region Emilia-Romagna (ARPAE)[1]. For the unreliable

[1] `https://www.arpae.it/`

data sources, we extracted the whole knowledge base of the Open Data cloud of ThingSpeak. Every extracted datastream for unreliable data sources goes through an additional classification step through TKSE before getting into the DSM.

MCS is the other important powerful source of information for our framework (Chapter 5). In particular, at the current stage, we focused on the opportunistic collection of data. In particular, in the figure we represented symbolically a urban environment with a fleet of mobile devices belonging to participants to our campaign. Each of them runs the Crowdroid application (or a hypothetic equivalent for other devices) and the environment is supervised by the MoCroSS framework . The only difference with the original implementation of MoCroSS is that, in this deployment, we use the distributed data collection control algorithm outlined in Section 5.2. This means that the original MoCroSS protocol is slightly modified as the time constraint in the rules no longer specifies when to upload a new measurement, rather it instructs when the participant needs to check whether the update has to be uploaded or not using AO-JFS. In fact, the SI calculated by the CCU has to be integrated as an additional rule field. This assumes that every MGRS area would have a dedicated optimal number of observations on top of which the SI is calculated.

In order to step into the stage of service composition, we use the INFORM architecture, introduced in Section 4.3, as a hub for the datastreams coming from all the aforementioned sources.

As a potential client platform for SenSquare we reported as an example the RouteX platform, a Home Automation System introduced in [151] and developed by us, however, it can be any private IoT ecosystem.

Service Delivery

Service Oriented Architectures (SOA) are the added value to pure IoT applications, since they leverage the service composition of raw data streams and add reasoning capabilities, making pure observation much more meaningful to humans. In our case, raw data is commonly not made for being accessible to everyone as it is. In particular, whenever a user is willing to consume one or a set of particular datastreams, her or his request is conveyed through the instantiation of a service: the data aggregation is established upon the creation of service templates, which specify the type of data to be consumed, and instantiated in a certain zone as service instances. We implemented a mechanism by means of which users can aggregate raw data streams and compose services, that can be exploited by other users too. As an example, we can simply think about all the well-known information that are obtainable by combining raw sensing measurement such as temperature and humidity. This is the case of the heat index, or humidex, which is a derived measurement calculated upon the values of temperature and humid-

ity and it is commonly referred to as the "human-perceived temperature". Another example is the Dew Point, which corresponds to the maximum temperature at which water vapor in the air condenses and forms liquid dew. This is again dependent on the values of humidity and air temperature and can be calculated upon such measurements. Moreover, the definition of derived quantities can be extended to custom ones. For instance, within the scope of house automation, a participant may be interested in opening automatically the windows whenever the environmental temperature reaches a value over a certain threshold. At the same time, such participant may want to combine the value of temperature with some other due to certain requirements, e.g. he or she might be allergic to pollen, thus, if there is a high concentration of airborne pollen, the participant would rather use air conditioning. This approach is a simple example of data aggregation as a custom service that a user can create which, as a consequence, results in a combination of energy saving and safe health. In our proposed architecture, the service "aeration for pollen intolerant" is intended to be created only once as a template that can be instantiated by several participants in different locations, provided that the right sensors are available in such places.

In order to give a formal and more detailed shape to such definition, services are composed through two main primitive entities: the raw datastreams and the Custom Service Templates (CST), defined as combinations of primary data classes through a mathematical expression and shared in a common repository to encourage reuse. They are abstract compositions and users design them in the same way a programmer writes a function: using a simplified language that we first defined in Backus-Naur Form (BNF) in [148]. The current version of such language includes basic arithmetic and relational operations between datastream classes, the `if-then-else` clause and logical connectives. Formally, a CST is defined by a mathematical expression E. The BNF expression of its current version is as follows:

$$E := c \mid DC \mid (E + E) \mid (E - E) \mid (E * E) \mid (E/E) \mid IFTE(C, E, E)$$

$$C := b \mid C \wedge C \mid C \vee C \mid \neg C \mid E > E \mid E \geq E \mid E < E \mid E \leq E \mid E = E \mid E \neq E$$

where c is a constant floating point value, b is a boolean value and DC is a datastream class. $IFTE(C, E_1, E_2)$ is the `if-then-else` clause, which executes E_1 if C is true, E_2 otherwise. When defining each DC, the CST specifies whether it should correspond to a single datastream; in alternative, aggregated measures for all the datastreams of the same type can be used (i.e. the maximum, the minimum and the average). A CST is stored in the database as a Python script with the used datastream classes as parameters. For instance, the heat index would be a CST that takes in input a temperature and a humidity value and returns a numeric value.

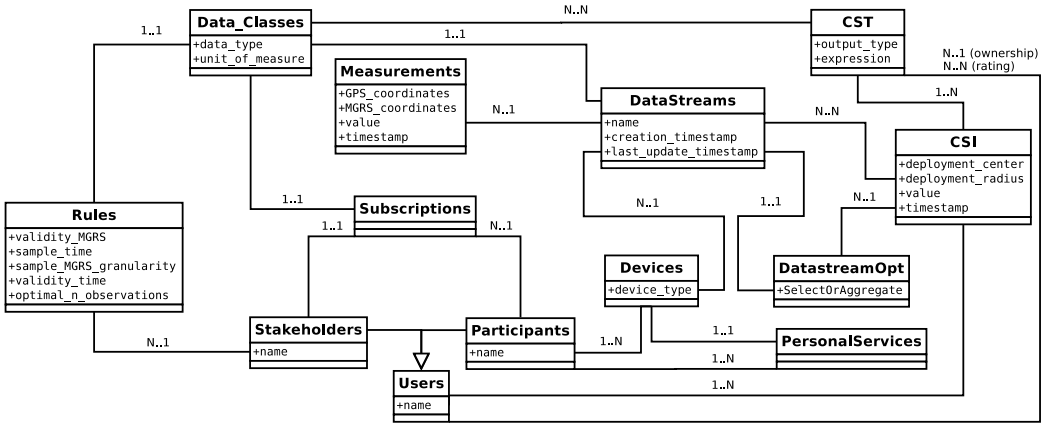


Figure 6.2: Relational diagram representing the repository of SenSquare [148].

Given such primitives, the actual services are defined as Custom Service Instances (CSI), again generated by the users through the instantiation of a CST in a specific geographical area. Here we recall that actual observations belonging to datastreams are used only with the instantiation of a CST to a CSI; a CST alone is just a template and specifies only data classes. When a CSI is instantiated, the user must choose the specific datastream of each type located in the area of interest and required by the respective CST to be used for the calculation. If the CST requires an aggregate measure instead, this step is ignored. We note that, as MCS sources are not static, they can only take part in aggregates and cannot be selected singularly. After this stage, the CSI behaves as a new geolocated datastream itself, which takes in input raw measurements and returns periodically a numeric output, using the expression contained in its CST as a calculation function. Therefore, it could be potentially aggregated further, although the current implementation does not allow it yet. CSTs and CSIs repositories are public, thus, once they are created, they are accessible to all the users of the platform. For the sake of clarity, in the next section we give an example of usage for CSTs and CSIs through the desktop user interface.

Datastreams, CSTs and CSIs are stored onto a persistent database (the current version of the architecture uses a MySQL database, therefore, the semantics added by INFORM are not currently used). The relational diagram of the database is shown in Figure 6.2. In particular, the system is characterized by two special types of actors: the stakeholders, which, as said, propose data collection campaigns, and the participants, who are able to produce data. A user is willing to consume data in the form of services and can be a stakeholder or a participant as well (but not necessarily). Each participant owns one or more devices, which are the physical entities committed to sense the environment. Each device can

produce one or more datastream, which are characterized by a single data type, marked with a pre-defined data class. Finally, each datastream refers to a set of measurements, to which, for each update, a new one is added. A special case is given by the data channels retrieved from the open data platforms, which are considered as single devices although they are not necessarily physical devices and can refer to multiple ones. Each stakeholder can submit a number of rules referring to a particular zone, and, whenever a participant decides to attend to the campaign proposed by a particular stakeholder, its subscription is registered onto a specific table. Users can create one or more CSTs and instantiate one or more CSIs. Multiple CSIs can be instantiated from the same CST. A special type of CSI, the "personal service" has the same structure as the CSI, but it is belonging only to the user who owns the device and simply wants to receive updates from his or her own datastreams.

6.2 Prototyping

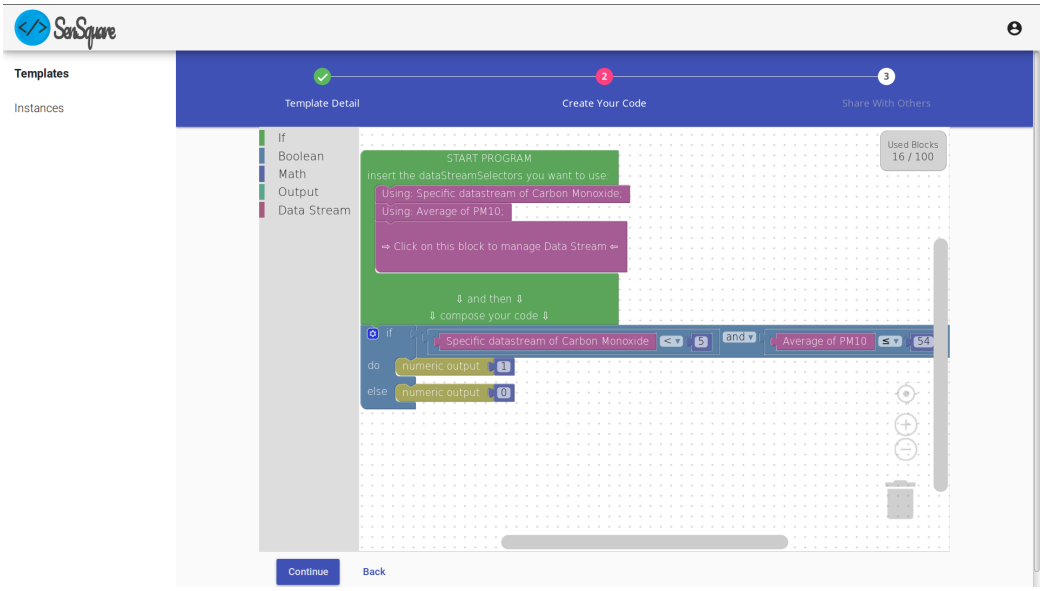
In this section we outline our implementations of SenSquare. Section 6.2.1 shows the web application for the users of SenSquare; Section 6.2.2 describes briefly a mobile application for SenSquare that has been developed previously; Section 6.2.3 wraps up the contributions of this section and outlines possible future directions.

6.2.1 The SenSquare Web Application

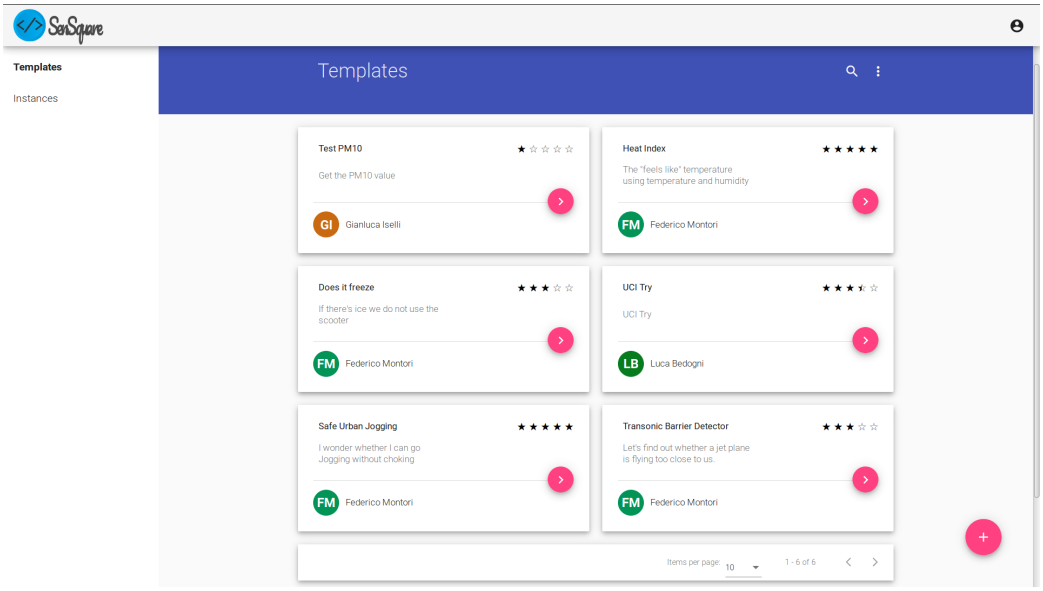
Throughout the development of our ecosystem we implemented several user interfaces that reflect a possible usage of the architecture potentially interested users. The first client applications were presented in [148] (both the desktop and the mobile interfaces), however, several other features have been added over time, therefore, here we present the current SenSquare desktop client application[2], where users make use of the datastreams gathered through collective awareness by creating CSTs and instantiating them into CSIs. In order to better explain the usage of the platform we will walk the interested reader through an example that better clarifies each step. Let us say that a user is particularly sensitive to urban pollution, however, he or she is also interested in jogging in zones included in the urban area. In such case, the user would start with the creation of a CST that informs whether the outdoor air quality is good enough to preserve her health. Looking at the EPA air quality indices (AQI), we can establish the maximum level of PM10 (suspended particulate matter below $10\ \mu\mathrm{m}$) bearable for a good AQI is $54\ \mu\mathrm{g/m^3}$, whereas the maximum level of CO (Carbon Monoxide) is $5\ \mathrm{ppm}$. Hence, we would write a CST that, if both the levels are below the respective thresholds, would output a positive value, negative otherwise. We do not assume that inexperienced users have programming capabilities, therefore, we leverage the paradigm of visual programming, widespread in the field of education, for the composition of a new CST. In particular, we used the well-known plugin Blockly by Google[3] with customized functionalities in order to cover the only the cases outlined in Section 6.1 (i.e. avoiding cycles) and provide as variables only parametric values coming from datastreams. Whenever selecting a possible datastream that can be part of the CST, we ask to the user whether it has to be a specific value or an aggregate. In our example, the composition of the CST through Blockly is depicted in Figure 6.3(a), in which new blocks can be dragged and dropped from the left end side into the main dashboard. We set the value of PM10 to be an aggregate (the average value), whereas the value of CO has to be

[2] `http://sensquare.disi.unibo.it/`

[3] `https://developers.google.com/blockly/`



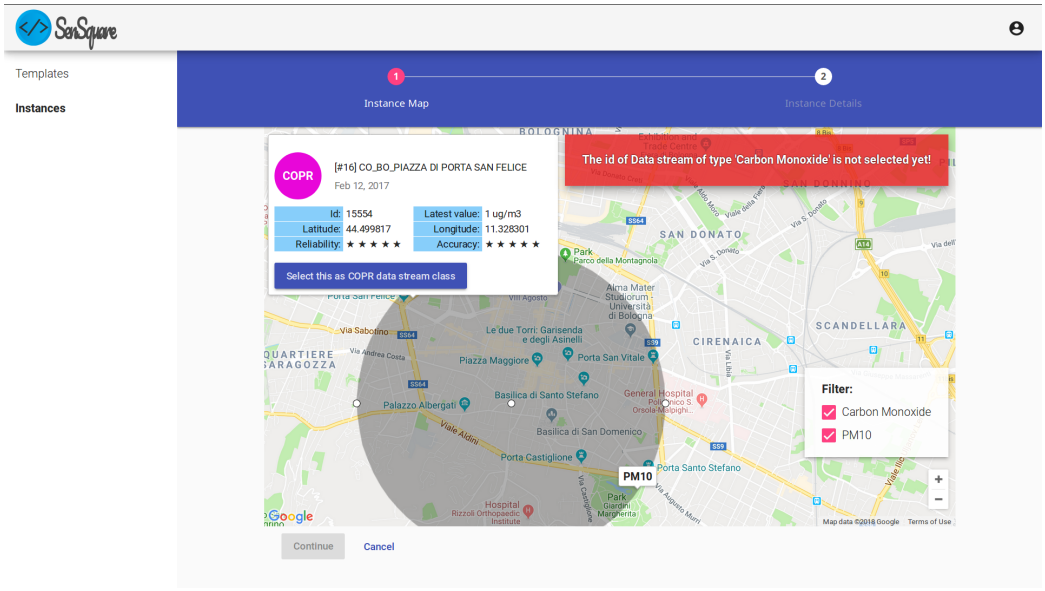
(a)



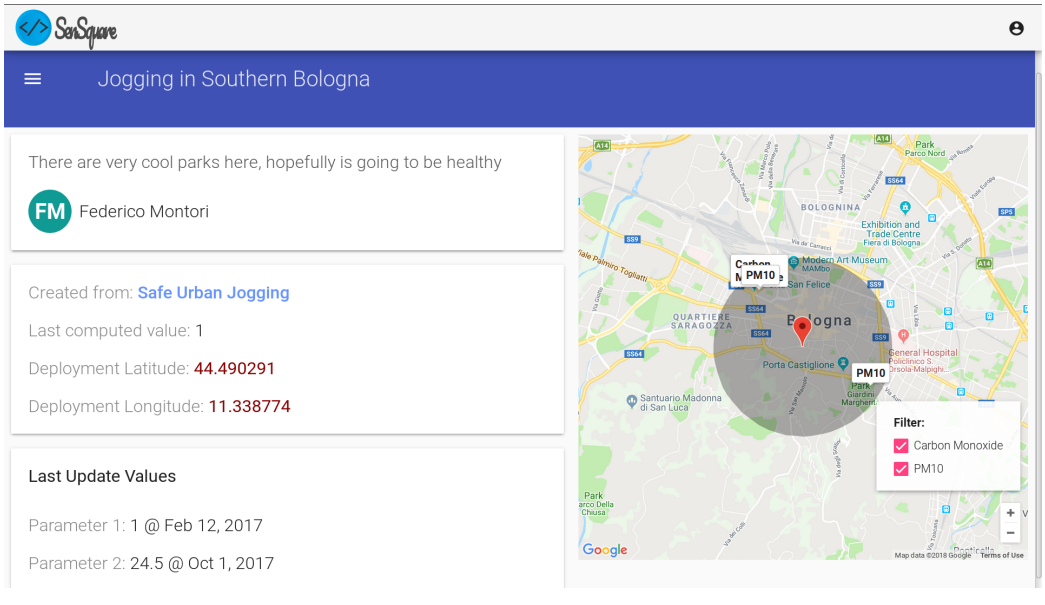
(b)

Figure 6.3: Screenshots of the SenSquare Web application in creating a CST using Blockly.

selected from a specific datastream at the time of instantiation. Our modification to Blockly only allows to output a numeric value, therefore, we will interpret 1 as a "positive answer" and 0 as a "negative answer". Once the CST is generated, it is stored together with all the other CSTs created by other users. The list of



(a)



(b)

Figure 6.4: Screenshots of the SenSquare Web application in creating a CSI through instantiation.

CSTs is shown in the screen in Figure 6.3(b), where users can explore all the created CSTs and select to instantiate one of them. Given that CSTs are created through crowdsourcing, we introduced a rating mechanism in order to quantify

the trustworthiness. Once the user selects one of the CSTs from the list, she will be displayed with the instantiation wizard screen, which consists in a map where the user should indicate a circular zone of interest (both the center and the radius are customizable). As the user moves and edits the circle, all the static and active datastreams within such area are displayed with a marker on the map (the datastreams coming from MCS sources are not displayed, since they are moving constantly and, therefore, will only take part in the aggregates). Only the datastreams of the same classes as the ones required by the respective CST are shown, in our example only datastreams measuring PM10 and CO. Furthermore, in order to complete the instantiation, the user must necessarily select a single datastream for each class not used as an aggregate; in Figure 6.4(a), following our example, we need to select a single CO datastream, whereas for PM10 it is not necessary, as we are using the average over all the PM10 datastreams in the area. Once the CSI is created it will be available to the whole community to be visualized. In Figure 6.4(b) we show the CSI visualization screen for our instantiated example. On the right side the map with the circular area highlighted in dark grey is displayed, together with the markers representing all the static sources taken into account. It is also possible to filter them by type. The part on the left is dedicated to all the metadata about the CSI, including its name, its location and the user who created it, as well as the observation values, both by category and the final value computed through the function implemented in the respective CST. In our example, we can see that the values measured for PM10 and carbon monoxide are respectively 1 and 24.5, thus, the final value computed is 1 as expected, which stands for a good AQI. We can conclude that jogging in such area is safe even for susceptible individuals. The whole platform has been developed using Angular 2.0 and Django and its front-end interface has been designed following the guidelines of Material Design to promote intuitiveness.

In [148] we presented the first version of the Web application, which used Angular 1.0.

6.2.2 The Habitatest Mobile Application

In this section we describe our Android mobile application, called Habitatest, which is composed by an Android activity to merge services together, and widgets to monitor the services of the user. It has been proposed in [148] and it is a simplified access to the repositories without using CSTs and CSIs.

In Figure 6.5(a) we show the main screen of the Habitatest app, where the user is able to select the datastreams to monitor. The selection can be made either by inserting the ID string or by scanning the QR code which can be retrieved from the web service described in Section 6.2.1. The user can declare any number of

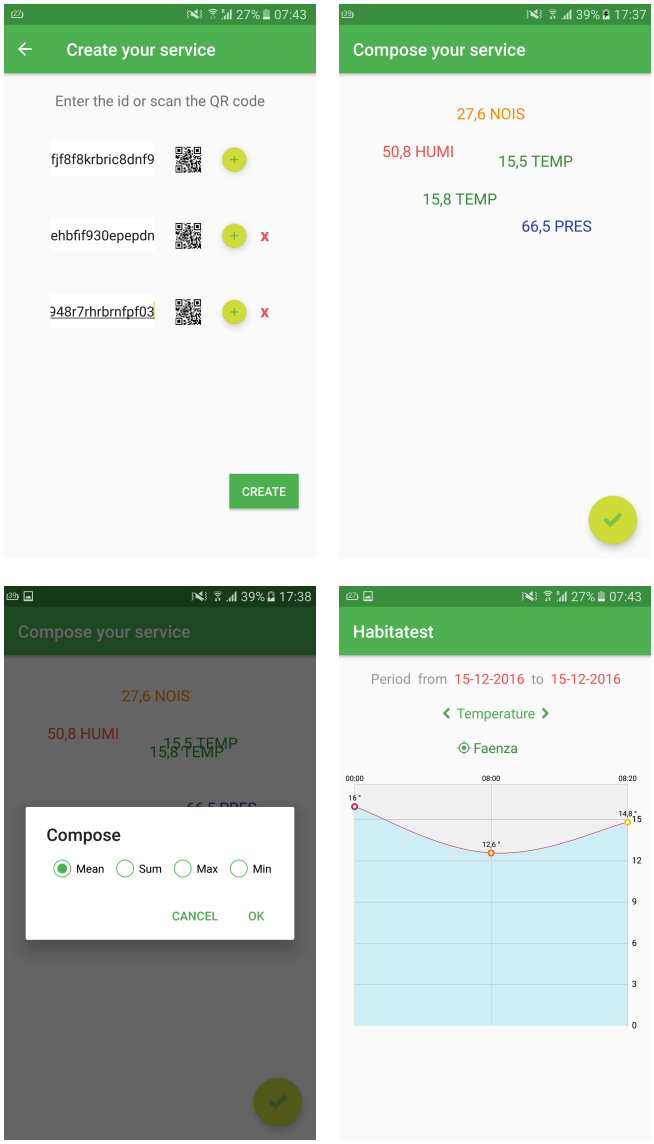


Figure 6.5: The Habitatest APP. Figure 6.5(a) shows the service composition screen; Figure 6.5(b) shows the aggregation of datastreams, Figure 6.5(c) shows the selector for the merging method; finally, Figure 6.5(d) presents the charts about the desired service.

datastreams of interest to be monitored through the Habitatest app and an update frequency. After this step, the user is redirected to the activity shown in Figure 6.5(b), from which the user can select all the data is interested in. He or she can also merge together data of the same class through 4 different methods, which we show in Figure 6.5(c). The user can merge the data either by extracting the

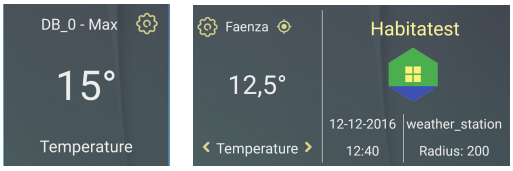


Figure 6.6: The widgets provided by the Habitatest APP. In Figure 6.6(a) we show the simple one, while Figure 6.6(b) offers more information to the user, such as the service range validity, the date, and geolocation information.

mean of the instances of the requested data class, their sum, the maximum or the minimum value. In the example in the figure the user drags and drops all the temperature sensors together each other as shown in Figure 6.5(b), and chooses to get the maximum out of them as in Figure 6.5(c). In case the user is not interested in some of the data classes, he or she can just drag them out of the screen to remove them from the monitoring area. After the selection of the data classes and their aggregation method, a local and personal service within the application is created. The service is not saved onto the SenSquare repository, as it is saved and aggregated locally. The user can then choose to monitor the service directly through the Habitatest app, or by using one of the widgets provided, as in Figure 6.6(a) or Figure 6.6(b). After selecting the smaller (Figure 6.6(a)) or the bigger version (Figure 6.6(b)), the user can then get updates directly on his or her home screen. If the user selected more than one data class to be monitored, the widget will give the possibility to the user to switch from one type of data to the other through a right and left arrow.

The Habitatest app runs then in the background, gathering all the information selected by the user and aggregating them together locally following the directions given, assuming that the user selected to do so. The user also has the opportunity to click on the data class to get the historian of the measurements, as shown in Figure 6.5(d).

6.2.3 Wrap Up and Future Perspectives

In this section we presented SenSquare, a prototype middleware platform that manages the multitude of datastreams that we collect through CAPs and aggregates them in customized services that users can create, share and use as a community. We believe that a platform as SenSquare brings a significant contribution in filling the gap of many IoT applications in the real world: transforming raw data in valuable knowledge for the personal and common benefit. With SenSquare, we recall that we do not propose a framework that is expected to be used by individuals and companies, rather, we bring this platform as a demonstrator to

prove the potential of CAPs when paired with a SOA able to provide users with services. Clearly, there is much to do in improving such a platform, first of all, we aim to redesign the repository in order to comply with the paradigm of the Web of Things (WoT) and the Semantic Web, fully unleashing the potential of our INFORM architecture. Another future work is the management of CSTs and CSIs as microservices, so that they can be further combined together with primary datastream classes.

Chapter 7

Conclusions

With this dissertation we had the goal to prove the effectiveness of CAPs in several IoT applications and how they can exponentially increase the knowledge base. We took into account a currently major issue in IoT ecosystems: the inability to cooperate and to share data that can potentially and exponentially enlarge that knowledge base due to incompatibilities and interests of the various parties. We tackled this problem in a different way compared to the vast majority of the research literature, which is typically focused in proposing new IoT frameworks, new IoT platforms, new IoT architectures, new IoT standards that hardly meet the interests of citizens and companies, mainly because the field is growing at an impressive pace. Instead, we aim to make use of what is already in place, since right now we are just lacking edges to connect the enormous amount of nodes in the IoT world; that is to say, we can get a lot of information from what is already in place. In this dissertation, in order to take advantage of such data sources, we make use of CAPs (introduced and classified in Section 2.2), paradigms that offload data collection tasks to a crowd of participants, either directly (i.e. specifically recruiting the participants) or indirectly (i.e. making use of the Open Data published by the participants on third parties repositories). In particular, we analyzed two main systems: Open Data and MCS.

Open Data has been introduced in Section 2.3. We provided definitions, examples in the real world and the (scarce) research landscape around it. We defined the concepts of reliable and unreliable data sources: the former host basically data provided by governmental or trustworthy sources which annotate their data and use good appliances, the latter are formed by crowdsourced data coming from privates who typically annotate poorly the datastreams. Unreliable data sources are growing and can potentially be an extremely useful added value to IoT applications, however, they need a processing step in order to be usable. To this end, we studied the feasibility of their integration in Section 4.1, we proposed an efficient

ensemble algorithm for the classification of unannotated, unreliable datastreams in Section 4.2 and a framework that semantically annotates datastreams according to a given set of ontologies in Section 4.3. Future works in this field are envisioned, as the proposed classification algorithm, namely TKSE, is designed currently for an ensemble of two different classifiers, we need to further explore the possibility of adding an arbitrary number.

MCS has been introduced in Section 2.4, in particular, we focused on the problem of the *"Curse of Sensing"*, introduced separately in Section 2.5, defined as the inability of current MCS applications in dealing with sparse or dense data. We specifically took into account applications in the scope of Smart Cities and environmental monitoring and proposed a paradigm, namely MoCroSS (Section 5.1, that elaborates the requests for participants on top of a set of rules established by the stakeholders, who are active actors in our architectures and represent the demand for data in particular zones at a defined time. Furthermore, we tackled the *"Curse of Sensing"* problem through a distributed algorithm that solicits the participants in contributing opportunistically with a rate proportional to the demand (Section 5.2). Future works here are mainly oriented to enhance the distributed algorithm in order to consider an arbitrary number of sensors and zones, as well as the possibility to include a participatory fashion. Better simulations and comparison with the State-of-the-Art are as well subject to studies.

Chapter 6 discussed our prototype platform SenSquare, that we developed in order to demonstrate the effectiveness and the usability of the CAPs that we proposed, as well as the plethora of possibilities derived from their usage. SenSquare represents our vision for a global system that is fed with data coming from CAPs, dictated by the needs of the users themselves, and displays end users with personalized, usable and flexible services. In particular, in Section 6.1, we defined a language for composing service templates, used through the tool Blockly in order to be usable by people with a non-programming background, and discussed how users can instantiate services according to their needs in the area of interest. Future works will be focused on defining a microservice-based architecture in which service instances can be composed further in a layered fashion.

With this dissertation we brought the attention of the reader over the CAPs and, in general, over a paradigm that, in this world where data is among the most important goods, instead of building a new data architecture, makes the most of what is already available. We know for a fact that the union of the efforts is way more powerful that the sum of the parts, here we demonstrate how the power of such union brings an exponentially growing benefit compared to self-acting ecosystems, even though using data provided by crowds carries along a number

of challenges. We firmly believe that this work opens up a plethora of novel possibilities in research as well as in any entity interested in building IoT applications for the common benefit; in fact, much of the data needed for such applications is already available, we just need to be aware of it.

Acknowledgements

My PhD path has been a long, difficult journey that resulted in a lot of personal gratifications and many experiences that shaped the man that I am, personally, professionally and culturally. I owe all this to my supervisor, Luciano, who followed me throughout my whole path within the university. I also thank a lot my mentors, Tullio, Marco and especially Luca, who are responsible for my knowledge baggage. The members of my research group and my colleagues, in particular Angelo, Luca, Francesco, Vincenzo, Tong, Stefano and Fabio helped me a lot in carrying out the outcome of this journey. I also owe a lot to the people who hosted me in Australia in an amazing professional and cultural experience: Dimitrios, Prem, Ali and Kewen. Thank you all. I also thank Archan Misra and Cristian Borcea, who spent their time in reviewing this work giving me valuable advice.

I also need to thank the incredible amount of friends that supported me throughout this journey, I cannot name them all due to spatial constraints otherwise the acknowledgements section would be longer than the thesis. But I need to spend few words to nominate Mandiza, Leonardo, Danilo, Nicola and Mehmet, who became an imperishable part of me. And Valeria, she has always been with me even though we have been separated more than we have been together. And I haven't felt anybody as close to me as she has been. In the same way I also need to thank my fantastic family, in particular my parents, Lucia and Guido. I don't think I could have asked for more, they are the best part of me. My final thank goes to my brother, Francesco, I wouldn't be even the half of what I am if I haven't had him reminding me what is really important.

Bibliography

[1] Espressif SDK Releases. http://bbs.espressif.com/viewforum.php?f=46.

[2] LTE-M and NB-IoT Commercial Launches. https://www.gsma.com/iot/mobile-iot-commercial-launches/.

[3] SigFox Coverage. https://www.sigfox.com/en/coverage.

[4] 3GPP. Standardization of Machine-Type Communications (MTC), v0.2.4. Technical report, 2014.

[5] 3GPP. Extended Coverage GSM (EC-GSM) for support of Cellular Internet of Things. Tsg geran wg1, 2015.

[6] Karl Aberer, Manfred Hauswirth, and Ali Salehi. The global sensor networks middleware for efficient and flexible deployment and interconnection of sensor networks. Technical report, 2006.

[7] Ejaz Ahmed, Ibrar Yaqoob, Abdullah Gani, Muhammad Imran, and Mohsen Guizani. Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges. *IEEE Wireless Communications*, 23(5):10–16, 2016.

[8] Godfrey Anuga Akpakwu, Bruno J. Silva, Gerhard P. Hancke, and Adnan M. Abu-Mahfouz. A survey on 5g networks for the internet of things: Communication technologies and challenges. *IEEE Access*, 6:3619–3647, 2018.

[9] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.

[10] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys and Tutorials*, 17(4):2347–2376, 2015.

[11] Soko Aoki, Yukihiko Kirihara, Jin Nakazawa, Kazunori Takashio, and Hideyuki Tokuda. A sensor actuator network architecture with control rules. In *Networked Sensing Systems (INSS), 2009 Sixth International Conference on*, pages 1–4. IEEE, 2009.

[12] Kevin Ashton. That 'Internet of Things' Thing. *RFID Journal*, 22(7):97–114, 2009.

[13] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[14] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

[15] Athanasios Bamis and Andreas Savvides. Stfl: a spatio temporal filtering language with applications in assisted living. In *Proceedings of the 2nd International Conference on PErvasive Technologies Related to Assistive Environments*, page 5. ACM, 2009.

[16] Andrea Bartoli, Mischa Dohler, Juan Hernández-Serrano, Apostolos Kountouris, and Dominique Barthel. Low-power low-rate goes long-range: the case for secure and cooperative machine-to-machine communications. In *NETWORKING 2011 Workshops*, pages 219–230. Springer, 2011.

[17] Gustavo E. Batista, Xiaoyue Wang, and Eamonn J Keogh. A complexity-invariant distance measure for time series. In *Proceedings of the 2011 SIAM international conference on data mining*, pages 699–710. SIAM, 2011.

[18] Luca Bedogni, Marco Di Felice, and Luciano Bononi. By train or by car? Detecting the user's motion type through smartphone sensors data. In *2012 IFIP Wireless Days*, pages 1–6. IEEE, nov 2012.

[19] Luca Bedogni, Marco Di Felice, and Luciano Bononi. Context-aware Android applications through transportation mode detection techniques. *Wireless Communications and Mobile Computing*, 16(16):2523–2541, nov 2016.

[20] Marek Bell, Matthew Chalmers, Louise Barkhuus, Malcolm Hall, Scott Sherwood, Paul Tennent, Barry Brown, Duncan Rowland, Steve Benford, Mauricio Capra, et al. Interweaving mobile games with everyday life. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 417–426. ACM, 2006.

[21] Oded Berger-Tal and José J Lahoz-Monfort. Conservation Technology: The next generation. *Conservation Letters*, page e12458, 2018.

[22] Maria Bermudez-Edo, Tarek Elsaleh, Payam Barnaghi, and Kerry Taylor. Iot-lite: a lightweight semantic model for the internet of things. In *Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 Intl IEEE Conferences*, pages 90–97. IEEE, 2016.

[23] Jean Boivin and Serena Ng. Are more data always better for factor analysis? *Journal of Econometrics*, 132(1):169–194, 2006.

[24] Rick Bonney, Caren B. Cooper, Janis Dickinson, Steve Kelling, Tina Phillips, Kenneth V Rosenberg, and Jennifer Shirk. Citizen Science: a developing tool for expanding science knowledge and scientific literacy. *BioScience*, 59(11):977–984, 2009.

[25] Luciano Bononi, Marco Conti, and Enrico Gregori. Runtime Optimization of IEEE 802.11 Wireless LANs Performance. *IEEE Transactions on Parallel and Distributed Systems*, 15(1):66–80, 2004.

[26] João B. Borges Neto, Thiago H. Silva, Renato Martins Assunção, Raquel A. F. Mini, and Antonio A. F. Loureiro. Sensing in the collaborative Internet of things. *Sensors*, 15(3):6607–6632, 2015.

[27] Ioannis Boutsis and Vana Kalogeraki. Privacy preservation for participatory sensing data. In *Pervasive Computing and Communications (PerCom), 2013 IEEE International Conference on*, pages 103–113. IEEE, 2013.

[28] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[29] Bob Buckiewicz. Technical Overview of DECT ULE. *LSR White Paper*, 2016.

[30] Nicola Bui and Michele Zorzi. Health care applications: a solution based on the internet of things. In *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*, pages 131:1—-131:5, 2011.

[31] Jeffrey A. Burke, Deborah Estrin, Mark Hansen, Andrew Parker, Nithya Ramanathan, Sasank Reddy, and Mani B. Srivastava. Participatory sensing. *In: Workshop on World-Sensor-Web (WSW'06): Mobile Device Centric Sensor Networks and Applications*, pages 117–134, 2006.

[32] Jean-Paul Calbimonte, Oscar Corcho, Zhixian Yan, Hoyoung Jeung, and Karl Aberer. Deriving semantic sensor metadata from raw measurements. 2012.

[33] Emmanuel J. Candès. Compressive sampling. *Proceedings of the International Congress of Mathematicians*, pages 1433–1452, 2006.

[34] Andrea Capponi, Claudio Fiandrino, Dzmitry Kliazovich, Pascal Bouvry, and Stefano Giordano. A cost-effective distributed framework for data collection in cloud-based mobile crowd sensing architectures. *IEEE Transactions on Sustainable Computing*, 2(1):3–16, 2017.

[35] Giuseppe Cardone, Andrea Cirri, Antonio Corradi, and Luca Foschini. The participact mobile crowd sensing living lab: The testbed for smart cities. *IEEE Communications Magazine*, 52(10):78–85, 2014.

[36] Giuseppe Cardone, Andrea Cirri, Antonio Corradi, Luca Foschini, Raffaele Ianniello, and Rebecca Montanari. Crowdsensing in Urban areas for city-scale mass gathering management: Geofencing and activity recognition. *IEEE Sensors Journal*, 14(12):4185–4195, 2014.

[37] Giuseppe Cardone, Luca Foschini, Paolo Bellavista, Antonio Corradi, Cristian Borcea, Manoop Talasila, and Reza Curtmola. Fostering participaction in smart cities: A geo-social crowdsensing platform. *IEEE Communications Magazine*, 51(6):112–119, 2013.

[38] Carl F. Cargill. Why standardization efforts fail. *Journal of Electronic Publishing*, 14(1), 2011.

[39] Ricardo C. Carrano, Diego Passos, Luiz C. S. Magalhaes, and Celio V. N. Albuquerque. Survey and taxonomy of duty cycling mechanisms in wireless sensor networks. *IEEE Communications Surveys and Tutorials*, 16(1):181–194, 2014.

[40] Patrick Cerwall. Ericsson Mobility Report. Ericsson White Paper, 2018.

[41] Georgios Chatzimilioudis, Andreas Konstantinidis, Christos Laoudias, and Demetrios Zeinalipour-Yazti. Crowdsourcing with smartphones. *IEEE Internet Computing*, 16(5):36–44, 2012.

[42] Xiao Chen, Elizeu Santos-Neto, and Matei Ripeanu. Crowdsourcing for on-street smart parking. In *Proceedings of the second ACM international symposium on Design and analysis of intelligent vehicular networks and applications*, pages 1–8. ACM, 2012.

[43] Bo Cheng, Ming Wang, Shuai Zhao, Zhongyi Zhai, Da Zhu, and Junliang Chen. Situation-aware dynamic service coordination in an iot environment. *IEEE/ACM Transactions on Networking*, 25(4):2082–2095, 2017.

[44] Atanu Roy Chowdhury, Ben Falchuk, and Archan Misra. Medially: A provenance-aware remote health monitoring middleware. In *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*, pages 125–134. IEEE, 2010.

[45] Cisco. Cisco Visual Networking Index: Forecast and Methodology, 2016–2021. Cisco White Paper, 2017.

[46] Michael Compton, Payam Barnaghi, Luis Bermudez, RaúL GarcíA-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, et al. The ssn ontology of the w3c semantic sensor network incubator group. *Web semantics: science, services and agents on the World Wide Web*, 17:25–32, 2012.

[47] Cathy C. Conrad and Krista G. Hilchey. A review of citizen science and community-based environmental monitoring: Issues and opportunities. *Environmental Monitoring and Assessment*, 176(1-4):273–291, 2011.

[48] Cory Cornelius, Apu Kapadia, David Kotz, Dan Peebles, Minho Shin, and Nikos Triandopoulos. Anonysense: privacy-aware people-centric sensing. In *Proceedings of the 6th international conference on Mobile systems, applications, and services*, pages 211–224. ACM, 2008.

[49] Coronis Systems. Wavenis Technology Plateform. *Product Summary*, 2013.

[50] Jeremy Cowan. On-Ramp Wireless rebrands as Ingenu and launches US-wide M2M wireless public network, 2015.

[51] Fred J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.

[52] DASH7 Alliance. DASH7 Alliance Wireless Sensor and Actuator Network Protocol VERSION 1.0. *DASH7 Alliance Specification*, pages 1–69, 2015.

[53] Soumya Kanti Datta, Rui Pedro Ferreira Da Costa, Christian Bonnet, and Jérôme Härri. onem2m architecture based iot framework for mobile crowd sensing in smart cities. In *Networks and Communications (EuCNC), 2016 European Conference on*, pages 168–173. IEEE, 2016.

[54] Emiliano De Cristofaro and Claudio Soriente. Participatory privacy: Enabling privacy in participatory sensing. *IEEE Network*, 27(1):32–36, 2013.

[55] Jerker Delsing. *Iot automation: Arrowhead framework*. CRC Press, 2017.

[56] Houtao Deng, George Runger, Eugene Tuv, and Martyanov Vladimir. A time series forest for classification and feature extraction. *Information Sciences*, 239:142–153, 2013.

[57] Hasan Derhamy, Jens Eliasson, Jerker Delsing, and Peter Priller. A survey of commercial frameworks for the Internet of Things. In *Emerging Technologies & Factory Automation (ETFA), 2015 IEEE 20th Conference on*, pages 1–8. IEEE, 2015.

[58] Harpreet S. Dhillon, Howard Huang, and Harish Viswanathan. Wide-area wireless communication challenges for the internet of things. *IEEE Communications Magazine*, 55(2):168–174, 2017.

[59] Attilio Di Nisio, Tommaso Di Noia, Carlo Guarnieri Calò Carducci, and Maurizio Spadavecchia. Design of a low cost multipurpose wireless sensor network. In *Measurements & Networking (M&N), 2015 IEEE International Workshop on*, pages 1–6. IEEE, 2015.

[60] Djellel Eddine Difallah, Gianluca Demartini, and Philippe Cudré-Mauroux. Pick-a-crowd: tell me what you like, and i'll tell you what to do. In *Proceedings of the 22nd international conference on World Wide Web*, pages 367–374. ACM, 2013.

[61] Lingjie Duan, Takeshi Kubo, Kohei Sugiyama, Jianwei Huang, Teruyuki Hasegawa, and Jean Walrand. Incentive mechanisms for smartphone collaboration in data acquisition and distributed computing. In *INFOCOM, 2012 Proceedings IEEE*, pages 1701–1709. IEEE, 2012.

[62] Enrique Estellés-Arolas and Fernando González-Ladrón-de Guevara. Towards an integrated crowdsourcing definition. *Journal of Information Science*, 38(2):189–200, 2012.

[63] Dave Evans. The internet of things: How the next evolution of the internet is changing everything. *CISCO white paper*, 1(2011):1–11, 2011.

[64] Raghu Ganti, Fan Ye, and Hui Lei. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine*, 49(11):32–39, 2011.

[65] Vangelis Gazis. A Survey of Standards for Machine to Machine (M2M) and the Internet of Things (IoT). *IEEE Communications Surveys & Tutorials*, (c):1–1, 2016.

[66] Orestis Georgiou and Usman Raza. Low power wide area network analysis: Can lora scale? *IEEE Wireless Communications Letters*, 6(2):162–165, 2017.

[67] Carles Gomez, Joaquim Oller, and Josep Paradells. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *Sensors (Switzerland)*, 12(9):11734–11753, 2012.

[68] Claire Goursaud and Jean-Marie Gorce. Dedicated networks for IoT: PHY / MAC state of the art and challenges. *EAI Endorsed Transactions on Internet of Things*, 1(1):1–11, 2015.

[69] Jorge Granjal, Edmundo Monteiro, and Jorge Sá Silva. Security for the internet of things: a survey of existing protocols and open research issues. *IEEE Communications Surveys & Tutorials*, 17(3):1294–1312, 2015.

[70] Giulio Grassi, Matteo Sammarco, Paramvir Bahl, Kyle Jamieson, and Giovanni Pau. Poster: ParkMaster: Leveraging Edge Computing in Visual Analytics. *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pages 257–259, 2015.

[71] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645–1660, 2013.

[72] Wael Guibene, Keith E. Nolan, and Mark Y. Kelly. Survey on Clean Slate Cellular-IoT Standard Proposals. In *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable,*

Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on, pages 1596–1599. IEEE, 2015.

[73] Bin Guo, Huihui Chen, Zhiwen Yu, Xing Xie, Shenlong Huangfu, and Daqing Zhang. FlierMeet: A Mobile Crowdsensing System for Cross-Space Public Information Reposting, Tagging, and Sharing. *IEEE Transactions on Mobile Computing*, 14(10):2020–2033, 2015.

[74] Bin Guo, Zhiwen Yu, Xingshe Zhou, and Daqing Zhang. From participatory sensing to Mobile Crowd Sensing. In *2014 IEEE International Conference on Pervasive Computing and Communication Workshops, PERCOM WORKSHOPS 2014*, pages 593–598, 2014.

[75] Gabriel Happle, Erik Wilhelm, Jimeno A Fonseca, and Arno Schlueter. Determining air-conditioning usage patterns in Singapore from distributed, portable sensors. *Energy Procedia*, 122:313–318, 2017.

[76] Mohammad Nazmul Haque, Nasimul Noman, Regina Berretta, and Pablo Moscato. Heterogeneous ensemble combination search using genetic algorithm for class imbalanced data classification. *PLoS ONE*, 11(1), 2016.

[77] Alireza Hassani, Pari Delir Haghighi, and Prem Prakash Jayaraman. Context-aware recruitment scheme for opportunistic mobile crowdsensing. In *Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS*, volume 2016-Janua, pages 266–273, 2016.

[78] Alireza Hassani, Federico Montori, Kewen Liao, Pari Delir Haghighi, Prem Prakash Jayaraman, Luciano Bononi, Dimitrios Georgakopoulos, and Arkady Zaslavsky. {Submitted for publication}.

[79] Daojing He, Sammy Chan, and Mohsen Guizani. User privacy and data trustworthiness in mobile crowd sensing. *IEEE Wireless Communications*, 22(1):28–34, 2015.

[80] Tobias Heer, Oscar Garcia-Morchon, René Hummen, Sye Loong Keoh, Sandeep S. Kumar, and Klaus Wehrle. Security challenges in the IP-based Internet of Things. In *Wireless Personal Communications*, volume 61, pages 527–542, 2011.

[81] Olafur Helgason, Sylvia T. Kouyoumdjieva, and Gunnar Karlsson. Opportunistic Communication and Human Mobility. *IEEE Transactions on Mobile Computing*, 13(7):1597–1610, 2014.

[82] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for network sensors. In *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, volume 35, pages 93–104, 2000.

[83] Baik Hoh, Tingxin Yan, Deepak Ganesan, Kenneth Tracton, Toch Iwuchukwu, and Juong-Sik Lee. Trucentive: A game-theoretic incentive platform for trustworthy mobile crowdsourcing parking services. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 160–166. IEEE, 2012.

[84] Jeff Howe. The rise of crowdsourcing. *Wired magazine*, 14(6):1–4, 2006.

[85] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks*, 13(2):415–425, 2002.

[86] Shaohan Hu, Shen Li, Shuochao Yao, Lu Su, Ramesh Govindan, Reginald Hobbs, and Tarek F. Abdelzaher. On exploiting logical dependencies for minimizing additive cost metrics in resource-limited crowdsensing. In *Proceedings - IEEE International Conference on Distributed Computing in Sensor Systems, DCOSS 2015*, pages 189–198, 2015.

[87] Bret Hull, Vladimir Bychkovsky, Yang Zhang, Kevin Chen, Michel Goraczko, Allen Miu, Eugene Shih, Hari Balakrishnan, and Samuel Madden. CarTel : A Distributed Mobile Sensor Computing System. *The 4th ACM Conference on Embedded Networked Sensor Systems*, pages 125–138, 2006.

[88] Inseok Hwang, Qi Han, and Archan Misra. Mastaq: A middleware architecture for sensor applications with statistical quality constraints. In *Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops*, PERCOMW '05, pages 390–395, Washington, DC, USA, 2005. IEEE Computer Society.

[89] IBM. Monetizing IoT Data.

[90] IEEE Computer Society. IEEE Std 802.15.4™-2003. 2003.

[91] Insteon Alliance. Insteon Whitepaper: The Details. *White Paper*, 2013.

[92] Alan Irwin. *Citizen science: A study of people, expertise and sustainable development*. Routledge, 1995.

[93] ISA Standard. Wireless systems for industrial automation: process control and related applications. *ISA-100.11 a-2009*, 2009.

[94] Luis G. Jaimes, Idalides Vergara-Laurens, and Miguel A. Labrador. A location-based incentive mechanism for participatory sensing systems with budget constraints. *2012 IEEE International Conference on Pervasive Computing and Communications*, (March):103–108, 2012.

[95] Luis G. Jaimes, Idalides J. Vergara-Laurens, and Andrew Raij. A Survey of Incentive Techniques for Mobile Crowd Sensing. *IEEE Internet of Things Journal*, 2(5):370–380, 2015.

[96] Raj Jain, Dah-Ming Chiu, and William R Hawe. *A quantitative measure of fairness and discrimination for resource allocation in shared computer system*, volume 38. Eastern Research Laboratory, Digital Equipment Corporation Hudson, MA, 1984.

[97] Prem Prakash Jayaraman, Charith Perera, Dimitrios Georgakopoulos, and Arkady Zaslavsky. MOSDEN: a scalable mobile collaborative platform for opportunistic sensing applications. *ICST Transactions on Collaborative Computing*, 2014.

[98] Young-Seon Jeong, Myong K. Jeong, and Olufemi A. Omitaomu. Weighted dynamic time warping for time series classification. *Pattern Recognition*, 44(9):2231–2240, 2011.

[99] Thivya Kandappu, Nikita Jaiman, Randy Tandriansyah, Archan Misra, Shih-Fen Cheng, Cen Chen, Hoong Chuin Lau, Deepthi Chander, and Koustuv Dasgupta. Tasker: behavioral insights via campus-based experimental mobile crowd-sourcing. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 392–402. ACM, 2016.

[100] Thivya Kandappu, Archan Misra, Shih-Fen Cheng, Randy Tandriansyah, and Hoong Chuin Lau. Obfuscation at-source: Privacy in context-aware mobile crowd-sourcing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2(1):16:1–16:24, March 2018.

[101] Salil S. Kanhere. Participatory sensing: Crowdsourcing data from mobile smartphones in urban spaces. In *2011 IEEE 12th International Conference on Mobile Data Management*, volume 2, pages 3–6, June 2011.

[102] Eiman Kanjo. NoiseSPY: A real-time mobile phone platform for urban noise monitoring and mapping. *Mobile Networks and Applications*, 15(4):562–574, 2010.

[103] Elli Kartsakli, Aris S. Lalos, Angelos Antonopoulos, Stefano Tennina, Marco Di Renzo, Luis Alonso, and Christos Verikoukis. A survey on M2M systems for mHealth: a wireless communications perspective. *Sensors*, 14(10):18009–18052, 2014.

[104] Keysight Technologies. Internet of Things (IoT), 2016.

[105] Peter Kiefer, Sebastian Matyas, and Christoph Schlieder. *Playing location-based games on geographically distributed game boards*. Citeseer, 2007.

[106] Anna N. Kim, Fredrik Hekland, Stig Petersen, and Paula Doyle. When HART goes wireless: Understanding and implementing the WirelessHART standard. In *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, pages 899–907, 2008.

[107] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. Ddos in the iot: Mirai and other botnets. *Computer*, 50(7):80–84, 2017.

[108] Margaret Kosmala, Andrea Wiggins, Alexandra Swanson, and Brooke Simmons. Assessing data quality in citizen science. *Frontiers in Ecology and the Environment*, 14(10):551–560, 2016.

[109] Srdan Krco, Boris Pokric, and Francois Carrez. Designing IoT architecture (s): A European perspective. In *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, pages 79–84. IEEE, 2014.

[110] Nicholas D Lane, Yohan Chon, Lin Zhou, Yongzhe Zhang, Fan Li, Dongwon Kim, Guanzhong Ding, Feng Zhao, and Hojung Cha. Piggyback CrowdSensing (PCS): Energy Efficient Crowdsourcing of Mobile Sensor Data by Exploiting Smartphone App Opportunities. *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, pages 7:1–7:14, 2013.

[111] Nicholas D. Lane, Shane B. Eisenman, Mirco Musolesi, Emiliano Miluzzo, and Andrew T. Campbell. Urban Sensing: Opportunistic or Participatory? *HotMobile '08, Proceedings of the 9th workshop on Mobile computing systems and applications, February 26-26, 2008, Napa Valley, CA, USA*, pages 11–16, 2008.

[112] Jorge Lanza, Luis Sanchez, David Gomez, Tarek Elsaleh, Ronald Steinke, and Flavio Cirillo. A proof-of-concept for semantically interoperable federation of iot experimentation facilities. *Sensors*, 16(7):1006, 2016.

[113] Mads Lauridsen, Huan Nguyen, Benny Vejlgaard, István Z Kovács, Preben Mogensen, and Mads Sørensen. Coverage Comparison of GPRS, NB-IoT, LoRa, and SigFox in a 7800 km^2 Area. In *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, 2017.

[114] Anna Lawrence. 'No Personal Motive?' Volunteers, Biodiversity, and the False Dichotomies of Participation. *Ethics, Place & Environment*, 9(3):279–298, 2006.

[115] Andres Laya, Luis Alonso, and Jesus Alonso-Zarate. Is the random access channel of LTE and LTE-A suitable for M2M communications? A survey of alternatives. *IEEE Communications Surveys and Tutorials*, 16(1):4–16, 2014.

[116] Mihai T. Lazarescu. Design and field test of a WSN platform prototype for long-term environmental monitoring. *Sensors (Switzerland)*, 15(4):9481–9518, 2015.

[117] Simone Leao, Kok Leong Ong, and Adam Krezel. 2Loud?: Community mapping of exposure to traffic noise with mobile phones. *Environmental Monitoring and Assessment*, 186(10):6193–6206, 2014.

[118] Juong Sik Lee and Baik Hoh. Dynamic pricing incentive for participatory sensing. In *Pervasive and Mobile Computing*, volume 6, pages 693–708, 2010.

[119] Chiara Leonardi, Andrea Cappellotto, Michele Caraviello, Bruno Lepri, and Fabrizio Antonelli. SecondNose: an air quality mobile crowdsensing system. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction Fun, Fast, Foundational - NordiCHI '14*, pages 1051–1054, 2014.

[120] Jian Li, Qiang Sun, and Guangyu Fan. Resource allocation for multiclass service in IoT uplink communications. In *Systems and Informatics (ICSAI), 2016 3rd International Conference on*, pages 777–781. IEEE, 2016.

[121] Xiaosheng Li and Jessica Lin. Linear Time Complexity Time Series Classification with Bag-of-Pattern-Features. In *Data Mining (ICDM), 2017 IEEE International Conference on*, pages 277–286. IEEE, 2017.

[122] Ruizhi Liao, Cristian Roman, Peter Ball, Shumao Ou, and Liping Chen. Crowdsourcing On-street Parking Space Detection. *arXiv preprint arXiv:1603.00441*, 2016.

[123] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing SAX: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2):107–144, 2007.

[124] Jessica Lin, Rohan Khade, and Yuan Li. Rotation-invariant similarity in time series using bag-of-patterns representation. *Journal of Intelligent Information Systems*, 39(2):287–315, 2012.

[125] Jie Lin, Wei Yu, Nan Zhang, Xinyu Yang, Hanlin Zhang, and Wei Zhao. A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications. *IEEE Internet of Things Journal*, 4(5):1125–1142, 2017.

[126] Jinwei Liu, Haiying Shen, and Xiang Zhang. A survey of mobile crowdsensing techniques: A critical component for the internet of things. In *2016 25th International Conference on Computer Communications and Networks, ICCCN 2016*, 2016.

[127] LoRa Alliance. LoRa Specification v1.0. Technical report, 2015.

[128] LoRa Alliance. Where does LoRa Fit in the Big Picture? LoRa Alliance White Paper, 2015.

[129] LoRa Alliance. 2017 End of the year report. Technical report, 2017.

[130] Malamati Louta, Konstantina Mpanti, George Karetsos, and Thomas Lagkas. Mobile crowd sensing architectural frameworks: a comprehensive survey. In *Information, Intelligence, Systems & Applications (IISA), 2016 7th International Conference on*, pages 1–7. IEEE, 2016.

[131] Yu Lu, Archan Misra, and Huayu Wu. Smartphone sensing meets transport data: A collaborative framework for transportation service analytics. *IEEE Transactions on Mobile Computing*, 17(4):945–960, 2018.

[132] Thomas Ludwig, Tim Siebigteroth, and Volkmar Pipek. Crowdmonitor: Monitoring physical and digital activities of citizens during emergencies. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8852, pages 421–428, 2015.

[133] Chong Luo, Feng Wu, Jun Sun, and Chen Wen Chen. Compressive Data Gathering for Large-Scale Wireless Sensor Networks. *Mobicom*, (800):145–156, 2009.

[134] Huadong Ma, Dong Zhao, and Peiyan Yuan. Opportunities in mobile crowd sensing. *Infocommunications Journal*, 7(2):32–38, 2015.

[135] Machina Research. LPWA Technologies: unlock new IoT market potential. LoRa Alliance White Paper, 2015.

[136] Anzar Mahmood, Nadeem Javaid, and Sohail Razzaq. A review of wireless communications for smart grid, 2015.

[137] Thomas W. Malone, Robert Laubacher, and Chrysanthos Dellarocas. Harnessing crowds : Mapping the genome of collective intelligence. *MIT Sloan School of Management*, 1:1–20, 2009.

[138] Graham Martin. Wireless sensor solutions for home & building automation. *EnOcean White Paper*, pages 1–7, 2007.

[139] Suhas Mathur, Tong Jin, Nikhil Kasturirangan, Janani Chandrasekaran, Wenzhi Xue, Marco Gruteser, and Wade Trappe. Parknet: Drive-by sensing of road-side parking statistics. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, MobiSys '10, pages 123–136, New York, NY, USA, 2010. ACM.

[140] Sebastian Matyas, Christian Matyas, Christoph Schlieder, Peter Kiefer, Hiroko Mitarai, and Maiko Kamata. Designing location-based mobile games with a purpose: collecting geospatial data with CityExplorer. In *Proceedings of the 2008 international conference on advances in computer entertainment technology*, pages 244–247. ACM, 2008.

[141] Kais Mekki, Eddy Bajic, Frederic Chaxel, and Fernand Meyer. A comparative study of lpwan technologies for large-scale iot deployment. *ICT Express*, 2018.

[142] Archan Misra and Lipyeow Lim. Optimizing sensor data acquisition for energy-efficient smartphone-based continuous event processing. In *Mobile Data Management (MDM), 2011 12th IEEE International Conference on*, volume 1, pages 88–97. IEEE, 2011.

[143] Tianli Mo, Lipyeow Lim, Sougata Sen, Archan Misra, Rajesh Krishna Balan, and Youngki Lee. Cloud-based query evaluation for energy-efficient mobile sensing. *Pervasive and Mobile Computing*, 38:257–274, 2017.

[144] Yuqi Mo, Claire Goursaud, and Jean-Marie Gorce. Theoretical analysis of unb-based iot networks with path loss and random spectrum access. In *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2016 IEEE 27th Annual International Symposium on*, pages 1–6. IEEE, 2016.

[145] Prashanth Mohan, Venkat Padmanabhan, and Ramachandran Ramjee. Nericell: Rich monitoring of road and traffic conditions using mobile smartphones. In *ACM Sensys*. Association for Computing Machinery, Inc., November 2008.

[146] Federico Montori, Luca Bedogni, and Luciano Bononi. On the integration of heterogeneous data sources for the collaborative Internet of Things. In *2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a Better Tomorrow, RTSI 2016*, pages 1–6, 2016.

[147] Federico Montori, Luca Bedogni, and Luciano Bononi. Distributed data collection control in opportunistic mobile crowdsensing. In *SMARTOBJECTS 2017 - Proceedings of the 3rd Workshop on Experiences with the Design and Implementation of Smart Objects, co-located with MobiCom 2017*, pages 19–24, 2017.

[148] Federico Montori, Luca Bedogni, and Luciano Bononi. A Collaborative Internet of Things Architecture for Smart Cities and Environmental Monitoring. *IEEE Internet of Things Journal*, 5(2):592–605, 2018.

[149] Federico Montori, Luca Bedogni, Alain Di Chiappari, and Luciano Bononi. SenSquare: A mobile crowdsensing architecture for smart cities. In *2016 IEEE 3rd World Forum on Internet of Things, WF-IoT 2016*, pages 536–541, 2016.

[150] Federico Montori, Luca Bedogni, Marco Di Felice, and Luciano Bononi. Machine-to-Machine Wireless Communication Technologies for the Internet of Things: Taxonomy, Comparison and Open Issues. *Pervasive and Mobile Computing*, 2018.

[151] Federico Montori, Luca Bedogni, Filippo Morselli, and Luciano Bononi. Achieving IoT interoperability through a service oriented in-home appliance. In *2017 IEEE Global Communications Conference, GLOBECOM 2017 - Proceedings*, 2017.

[152] Federico Montori, Riccardo Contigiani, and Luca Bedogni. Is WiFi suitable for energy efficient IoT deployments? A performance study. In *RTSI 2017*

- IEEE 3rd International Forum on Research and Technologies for Society and Industry, Conference Proceedings, 2017.

[153] Federico Montori, Marco Gramaglia, Luca Bedogni, Marco Fiore, Farid Sheikh, Luciano Bononi, and Andrea Vesco. Automotive communications in LTE: A simulation-based performance study. In *IEEE 86th Vehicular Technology Conference (VTC-Fall)*, 2017.

[154] Federico Montori, Prem Prakash Jayaraman, Ali Yavari, Alireza Hassani, and Dimitrios Georgakopoulos. The Curse of Sensing: Survey of techniques and challenges to cope with sparse and dense data in mobile crowd sensing for Internet of Things. *Pervasive and Mobile Computing*, 2018.

[155] Federico Montori, Kewen Liao, Prem Prakash Jayaraman, Luciano Bononi, Timos Sellis, and Dimitrios Georgakopoulos. Classification and Annotation of Open Internet of Things Datastreams. In *19th International Conference on Web Information Systems Engineering*, 2018.

[156] National Geospatial-Intelligence Agency. Military Map Reading 201. Technical report.

[157] Jianbing Ni, Xiaodong Lin, Kuan Zhang, and Yong Yu. Secure and deduplicated spatial crowdsourcing: A fog-based approach. In *Global Communications Conference (GLOBECOM), 2016 IEEE*, pages 1–6. IEEE, 2016.

[158] Prusayon Nintanavongsa, Rahman Doost-Mohammady, Marco Di Felice, and Kaushik R. Chowdhury. Device characterization and cross-layer protocol design for RF energy harvesting sensors. *Pervasive and Mobile Computing*, 9(1):120–131, 2013.

[159] Nokia. LTE-M - Optimizing LTE for the Internet of Things. Nokia Networks White Paper, 2014.

[160] Liadan O'Callaghan, Nina Mishra, Adam Meyerson, Sudipto Guha, and Rajeev Motwani. Streaming-data algorithms for high-quality clustering. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 685–694. IEEE, 2002.

[161] Mahesh Pal. Random forest classifier for remote sensing classification. *International Journal of Remote Sensing*, 26(1):217–222, 2005.

[162] Bei Pan, Yu Zheng, David Wilkie, and Cyrus Shahabi. Crowd Sensing of Traffic Anomalies Based on Human Mobility and Social Media. *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 344–353, 2013.

[163] Kasey Panetta. Top Trends in the Gartner Hype Cycle for Emerging Technologies. 2017.

[164] Gaetano Patti, Luca Leonardi, and Lucia Lo Bello. A Bluetooth low energy real-time protocol for industrial wireless mesh networks. In *Industrial Electronics Society, IECON 2016-42nd Annual Conference of the IEEE*, pages 4627–4632. IEEE, 2016.

[165] James W. Pearce-Higgins, Stephen R. Baillie, Katherine Boughey, Nigel A. D. Bourn, Ruud P. B. Foppen, Simon Gillings, Richard D. Gregory, Tom Hunt, Frederic Jiguet, Aleksi Lehikoinen, Andy J. Musgrove, Rob A. Robinson, David B. Roy, Gavin M. Siriwardena, Kevin J. Walker, and Jeremy D. Wilson. Overcoming the challenges of public data archiving for citizen science biodiversity recording and monitoring schemes. *Journal of Applied Ecology*, 2018.

[166] Tao Pei, Stanislav Sobolevsky, Carlo Ratti, Shih-Lung Shaw, Ting Li, and Chenghu Zhou. A new insight into land use classification based on aggregated mobile phone data. *International Journal of Geographical Information Science*, 28(9):1988–2007, 2014.

[167] Juha Petäjäjärvi, Konstantin Mikhaylov, Marko Pettissalo, Janne Janhunen, and Jari Iinatti. Performance of a low-power wide-area network based on lora technology: Doppler robustness, scalability, and coverage. *International Journal of Distributed Sensor Networks*, 13(3):1550147717699412, 2017.

[168] Juha Petäjäjärvi, Konstantin Mikhaylov, Rumana Yasmin, Matti Hämäläinen, and Jari Iinatti. Evaluation of LoRa LPWAN technology for indoor remote health and wellbeing monitoring. *International Journal of Wireless Information Networks*, 24(2):153–165, 2017.

[169] Rudiger Pryss, Manfred Reichert, Jochen Herrmann, Berthold Langguth, and Winfried Schlee. Mobile Crowd Sensing in Clinical and Psychological Trials – A Case Study. In *2015 IEEE 28th International Symposium on Computer-Based Medical Systems*, pages 23–24, 2015.

[170] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[171] Moo-Ryong Ra, Bin Liu, Tom F. La Porta, and Ramesh Govindan. Medusa: a programming framework for crowd-sensing applications. *Proceedings of the 10th international conference on Mobile systems, applications, and services - MobiSys '12*, (Section 2):337, 2012.

[172] Ajinkya Rajandekar and Biplab Sikdar. A survey of MAC layer issues and protocols for machine-to-machine communications. *IEEE Internet of Things Journal*, 2(2):175–186, 2015.

[173] Thanawin Rakthanmanon and Eamonn Keogh. Fast shapelets: A scalable algorithm for discovering time series shapelets. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 668–676. SIAM, 2013.

[174] Rajib Kumar Rana, Chun Tung Chou, Salil S. Kanhere, Nirupama Bulusu, and Wen Hu. Ear-Phone : An End-to-End Participatory Urban Noise Mapping System. *Proceedings of the International Conference on Information Processing in Sensor Networks IPSN*, pages 105–116, 2010.

[175] Chotirat Ann Ratanamahatana and Eamonn Keogh. Three myths about dynamic time warping data mining. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, pages 506–510. SIAM, 2005.

[176] Tifenn Rault, Abdelmadjid Bouabdallah, and Yacine Challal. Energy efficiency in wireless sensor networks: A top-down survey. *Computer Networks*, 67:104–122, 2014.

[177] Marvin Rausand and Arnljot Hsyland. System Reliability Theory: Models, Statistical Methods, and Applications. *Wescon/96*, 2004.

[178] Usman Raza, Parag Kulkarni, and Mahesh Sooriyabandara. Low Power Wide Area Networks: An Overview. *IEEE Communications Surveys & Tutorials*, 2017.

[179] Sasank Reddy, Deborah Estrin, Mark Hansen, and Mani Srivastava. Examining micro-payments for participatory sensing data collections. *Proceedings of the 12th ACM international conference on Ubiquitous computing - Ubicomp '10*, page 33, 2010.

[180] Sasank Reddy, Deborah Estrin, and Mani Srivastava. Recruitment framework for participatory sensing data collections. In *International Conference on Pervasive Computing*, pages 138–155. Springer US, 2010.

[181] Machina Research. Global M2M market to grow to 27 billion devices, generating USD1.6 trillion revenue in 2024.

[182] Francesco Restuccia, Nirnay Ghosh, Shameek Bhattacharjee, Sajal K. Das, and Tommaso Melodia. Quality of information in mobile crowdsensing: Survey and research challenges. *ACM Transactions on Sensor Networks (TOSN)*, 13(4):34, 2017.

[183] Jérémy Robert, Sylvain Kubler, Niklas Kolbe, Alessandro Cerioni, Emmanuel Gastaud, and Kary Främling. Open iot ecosystem for enhanced interoperability in smart cities—example of métropole de lyon. *Sensors*, 17(12):2849, 2017.

[184] Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39, 2010.

[185] Ahmad-Reza Sadeghi, Christian Wachsmann, and Michael Waidner. Security and privacy challenges in industrial internet of things. In *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*, pages 1–6. IEEE, 2015.

[186] Rosario Salpietro, Luca Bedogni, Marco Di Felice, and Luciano Bononi. Park Here! a smart parking system based on smartphones' embedded sensors and short range Communication Technologies. In *IEEE World Forum on Internet of Things, WF-IoT 2015 - Proceedings*, pages 18–23, 2015.

[187] Douglas R. Sanqunetti. Implementing geo-fencing on mobile devices, 2006.

[188] Patrick Schäfer. The BOSS is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29(6):1505–1530, 2015.

[189] Curt Schurgers, Gautam Kulkarni, and Mani B. Srivastava. Distributed on-demand address assignment in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 13(10):1056–1065, 2002.

[190] Curt Schurgers, Gautam Kulkarni, and Mani B. Srivastava. Distributed on-demand address assignment in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 13(10):1056–1065, 2002.

[191] Zach Shelby, Klaus Hartke, and Carsten Bormann. The Constrained Application Protocol (CoAP). *Rfc 7252*, page 112, 2014.

[192] Xiang Sheng, Jian Tang, and Weiyi Zhang. Energy-efficient collaborative sensing with mobile phones. In *Proceedings - IEEE INFOCOM*, pages 1916–1924, 2012.

[193] Zhengguo Sheng, Shusen Yang, Yifan Yu, Athanasios Vasilakos, Julie McCann, and Kin Leung. A survey on the IETF protocol suite for the internet of things: Standards, challenges, and opportunities. *IEEE Wireless Communications*, 20(6):91–98, 2013.

[194] Wanita Sherchan, Prem Prakash Jayaraman, Shonali Krishnaswamy, Arkady Zaslavsky, Seng Loke, and Abhijat Sinha. Using on-the-move mining for mobile crowdsensing. In *Proceedings - 2012 IEEE 13th International Conference on Mobile Data Management, MDM 2012*, pages 115–124, 2012.

[195] Feifei Shi, Qingjuan Li, Tao Zhu, and Huansheng Ning. A survey of data semantization in internet of things. *Sensors*, 18(1), 2018.

[196] Akhilesh Shrestha and Liudong Xing. A Performance Comparison of Different Topologies for Wireless Sensor Networks. *2007 IEEE Conference on Technologies for Homeland Security*, pages 280–285, 2007.

[197] Akhilesh Shrestha and Liudong Xing. Quantifying application communication reliability of wireless sensor networks. *International Journal of Performability Engineering*, 4(1):43–56, 2008.

[198] Dhananjay Singh, Gaurav Tripathi, and Antonio J Jara. A survey of Internet-of-Things: Future vision, architecture, challenges and services. In *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, pages 287–292. IEEE, 2014.

[199] Eugene Siow, Thanassis Tiropanis, Xin Wang, and Wendy Hall. TritanDB: Time-series Rapid Internet of Things Analytics. *arXiv preprint arXiv:1801.07947*, 2018.

[200] Emiliano Sisinni, Abusayeed Saifullah, Song Han, Ulf Jennehag, and Mikael Gidlund. Industrial Internet of Things: Challenges, Opportunities, and Directions. *IEEE Transactions on Industrial Informatics*, 2018.

[201] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437, 2009.

[202] John Soldatos, Nikos Kefalakis, Manfred Hauswirth, Martin Serrano, Jean-Paul Calbimonte, Mehdi Riahi, Karl Aberer, Prem Prakash Jayaraman, Arkady Zaslavsky, Ivana Podnar Žarko, Lea Skorin-Kapov, and Reinhard Herzog. Openiot: Open source internet-of-things in the cloud. In *Interoperability and open-source solutions for the internet of things*, pages 13–25. Springer, 2015.

[203] Erfan Soltanmohammadi, Kamran Ghavami, and Mort Naraghi-Pour. A survey of traffic issues in Machine-to-Machine communications over LTE. *IEEE Internet of Things journal (in print)*, pages 1–21, 2015.

[204] John A. Stankovic. Research directions for the Internet of Things. *IEEE Internet of Things Journal*, 1(1):3–9, 2014.

[205] Dipak Surie, Olivier Laguionie, and Thomas Pederson. Wireless sensor networking of everyday objects in a smart home environment. In *ISSNIP 2008 - Proceedings of the 2008 International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pages 189–194, 2008.

[206] James Surowiecki. The Wisdom of Crowds. *American Journal of Physics*, 75(0908):336, 2005.

[207] Manoop Talasila, Reza Curtmola, and Cristian Borcea. Mobile Crowd Sensing. In *Handbook of Sensor Networking: Advanced Technologies and Applications*, number JANUARY 2014. 2015.

[208] Jasper Tan and Simon G. M. Koo. A survey of technologies in Internet of Things. In *Proc. of the IEEE International Conference on Distributed Computing in Sensor Systems (ICDCSS)*, 2014.

[209] Thread Group. Thread Usage of 6LoWPAN. *White Paper*, 2015.

[210] Hien To, Gabriel Ghinita, and Cyrus Shahabi. A framework for protecting worker location privacy in spatial crowdsourcing. *Proc. VLDB Endow.*, 7(10):919–930, June 2014.

[211] Mattia Tomasoni, Andrea Capponi, Claudio Fiandrino, Dzmitry Kliazovich, Fabrizio Granelli, and Pascal Bouvry. Profiling energy efficiency of mobile crowdsensing data collection frameworks for smart city applications. In *Mobile Cloud Computing, Services, and Engineering (MobileCloud), 2018 6th IEEE International Conference on*, pages 1–8, 2018.

[212] Lorenzo Vangelista, Andrea Zanella, and Michele Zorzi. Long-range iot technologies: The dawn of lora™. In *Future Access Enablers of Ubiquitous and Intelligent Infrastructures*, pages 51–58. Springer, 2015.

[213] Anitha Varghese and Deepaknath Tandur. Wireless requirements and challenges in Industry 4.0. In *Proceedings of 2014 International Conference on Contemporary Computing and Informatics, IC3I 2014*, pages 634–638, 2014.

[214] Idalides J. Vergara-Laurens, Luis G. Jaimes, and Miguel A. Labrador. Privacy-preserving mechanisms for crowdsensing: Survey and research challenges. *IEEE Internet of Things Journal*, 2016.

[215] Félix Jesùs Villanueva, David Villa, Maria José Santofimia, Jesùs Barba, and Juan Carlos Lòpez. Crowdsensing Smart City Parking Monitoring. In *IEEE World Forum on Internet of Things, WF-IoT 2015 - Proceedings*, 2015.

[216] Leye Wang, Daqing Zhang, Animesh Pathak, Chao Chen, Haoyi Xiong, Dingqi Yang, and Yasha Wang. CCS-TA: Quality-Guaranteed Online Task Allocation in Compressive Crowdsensing. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 683—-694, 2015.

[217] Leye Wang, Daqing Zhang, Yasha Wang, Chao Chen, Xiao Han, and Abdallah M'Hamed. Sparse mobile crowdsensing: Challenges and opportunities. *IEEE Communications Magazine*, 54(7):161–167, 2016.

[218] Leye Wang, Daqing Zhang, and Haoyi Xiong. effSense: Energy-Efficient and Cost-Effective Data Uploading in Mobile Crowdsensing. *UbiComp Adjunct*, pages 1075–1086, 2013.

[219] Wang Wei and Payam Barnaghi. Semantic annotation and reasoning for sensor data". In Payam Barnaghi, Klaus Moessner, Mirko Presser, and Stefan Meissner, editors, *Smart Sensing and Context*, pages 66–76, Berlin, Heidelberg", 2009. Springer Berlin Heidelberg.

[220] Matthew Weiner, Milos Jorgovanovic, Anant Sahai, and Borivoje Nikolie. Design of a low-latency, high-reliability wireless communication system for control applications. In *2014 IEEE International Conference on Communications, ICC 2014*, pages 3829–3835, 2014.

[221] Graham Whitelaw, Hague Vaughan, Brian Craig, and David Atkinson. Establishing the Canadian community monitoring network. *Environmental Monitoring and Assessment*, 88(1-3):409–418, 2003.

[222] David H. Wolpert. The Lack of A Priori Distinctions Between Learning Algorithms. *Neural Computation*, 8(7):1341–1390, 1996.

[223] Haoyi Xiong, Daqing Zhang, Guanling Chen, Leye Wang, Vincent Gauthier, and Laura E. Barnes. ICrowd: Near-Optimal Task Allocation for Piggyback Crowdsensing. *IEEE Transactions on Mobile Computing*, 15(8):2010–2022, 2016.

[224] Haoyi Xiong, Daqing Zhang, Leye Wang, and Hakima Chaouchi. EMC3: Energy-Efficient Data Transfer in Mobile Crowdsensing under Full Coverage Constraint. *IEEE Transactions on Mobile Computing*, 14(7):1355–1368, 2015.

[225] Xiong Xiong, Kan Zheng, Rongtao Xu, Wei Xiang, and Periklis Chatzimisios. Low power wide area machine-to-machine networks: Key techniques and prototype. *IEEE Communications Magazine*, 53(9):64–71, 2015.

[226] Li Da Xu, Wu He, and Shancang Li. Internet of things in industries: A survey, 2014.

[227] Liwen Xu, Xiaohong Hao, Nicholas D. Lane, Xin Liu, and Thomas Moscibroda. Cost-aware compressive sensing for networked sensing systems. In *Proceedings of the 14th International Conference on Information Processing in Sensor Networks - IPSN '15*, pages 130–141, 2015.

[228] Dejun Yang, Guoliang Xue, Xi Fang, and Jian Tang. Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing. In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 173–184. ACM, 2012.

[229] Ali Yavari, Prem Prakash Jayaraman, and Dimitrios Georgakopoulos. Contextualised service delivery in the internet of things: Parking recommender for smart cities. In *Internet of Things (WF-IoT), 2016 IEEE 3rd World Forum on*, pages 454–459. IEEE, 2016.

[230] Ali Yavari, Prem Prakash Jayaraman, Dimitrios Georgakopoulos, and Surya Nepal. Contaas: An approach to internet-scale contextualisation for developing efficient internet of things applications. In *Proceedings of the 50th Hawaii International Conference on System Sciences*, 2017.

[231] Ali Yavari, Arezou Soltani Panah, Dimitrios Georgakopoulos, Prem Prakash Jayaraman, and Ron van Schyndel. Scalable role-based data disclosure control for the internet of things. In *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*, pages 2226–2233. IEEE, 2017.

[232] Lexiang Ye and Eamonn Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 947–956. ACM, 2009.

[233] Robert Young and David Zhang. Introduction to "Clean Slate" cellular IoT radio access solution. Presentation, 2014.

[234] Z-Wave Alliance. Z-Wave Protocol Overview. 4th May, 2006.

[235] AA Zaidan, BB Zaidan, MY Qahtan, OS Albahri, AS Albahri, Mussab Alaa, FM Jumaah, Mohammed Talal, KL Tan, WL Shir, et al. A survey on communication components for IoT-based technologies in smart homes. *Telecommunication Systems*, pages 1–25, 2018.

[236] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. Internet of things for smart cities. *IEEE Internet of Things Journal*, 1(1):22–32, 2014.

[237] Marco Zappatore, Antonella Longo, Mario A Bochicchio, Daniele Zappatore, Alessandro A Morrone, and Gianluca De Mitri. A crowdsensing approach for mobile learning in acoustics and noise monitoring. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 219–224. ACM, 2016.

[238] Bo Zhang, Zheng Song, Chi Harold Liu, Jian Ma, and Wendong Wang. An event-driven qoi-aware participatory sensing framework with energy and budget constraints. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3):42, 2015.

[239] Daqing Zhang, Haoyi Xiong, Leye Wang, and Guanling Chen. CrowdRecruiter: selecting participants for piggyback crowdsensing under probabilistic coverage constraint. *UbiComp*, pages 703–714, 2014.

[240] Maotian Zhang, Panlong Yang, Chang Tian, Shaojie Tang, and Baowei Wang. Toward Optimum Crowdsensing Coverage with Guaranteed Performance. *IEEE Sensors Journal*, 16(5):1471–1480, 2016.

[241] Xinglin Zhang, Zheng Yang, Wei Sun, Yunhao Liu, Shaohua Tang, Kai Xing, and Xufei Mao. Incentives for mobile crowd sensing: A survey. *IEEE Communications Surveys & Tutorials*, 18(1):54–67, 2016.

[242] Kan Zheng, Suling Ou, Jesus Alonso-Zarate, Mischa Dohler, Fei Liu, and Hua Zhu. Challenges of massive access in highly dense lte-advanced networks with machine-to-machine communications. *IEEE Wireless Communications*, 21(3):12–18, 2014.

[243] Zigbee Alliance. Zigbee Specification. *Zigbee Alliance website*, pages 1–604, 2008.

[244] Michele Zorzi, Alexander Gluhak, Sebastian Lange, and Alessandro Bassi. From today's INTRAnet of things to a future INTERnet of things: A wireless- and mobility-related view. *IEEE Wireless Communications*, 17(6):44–51, 2010.