UNIVERSITÀ DEGLI STUDI DI PALERMO
Dottorato di Ricerca in INGEGNERIA DELL'INNOVAZIONE TECNOLOGICA
Dipartimento dell'Innovazione Industriale e Digitale (DIID)
SSD: ING-INF/05

MULTISENSOR DATA FUSION IN PERVASIVE ARTIFICIAL INTELLIGENCE SYSTEMS

IL DOTTORE
Pierluca Ferraro

IL COORDINATORE
Ch.mo Prof. Salvatore Gaglio

IL TUTOR
Prof. Giuseppe Lo Re

CO TUTOR
Prof. Sajal K. Das
Missouri University of Science and Technology

CICLO XXIX
ANNO CONSEGUIMENTO TITOLO: 2017

To my family, for all that you do

Abstract

Intelligent systems designed to manage smart environments exploit numerous sensing and actuating devices, pervasively deployed so as to remain invisible to users and subtly learn their preferences and satisfy their needs. Nowadays, such systems are constantly evolving and becoming ever more complex, so it is increasingly difficult to develop them successfully. A possible solution to this problem might lie in delegating certain decisions to the machines themselves, making them more autonomous and able to self-configure and self-manage.

This work presents a multi-tier architecture for a complete pervasive system capable of understanding the state of the surrounding environment, as well as using this knowledge to decide what actions should be performed to provide the best possible environmental conditions for end-users, in line with the Ambient Intelligence (AmI) paradigm. To achieve such high-level goals, the system has to effectively merge and analyze heterogeneous data collected by multiple sensors, pervasively deployed in a smart environment. To this end, the proposed system includes a context-aware, self-optimizing, adaptive module for sensor data fusion. Contextual information is leveraged in the fusion process, so as to increase the accuracy of inference and hence decision making in a dynamically changing environment. Additionally, two self-optimization modules are responsible for dynamically determining the subset of sensors to use, finding an optimal trade-off to minimize energy consumption and maximize sensing accuracy. The effectiveness of the proposed approach is demonstrated with the application scenario of user activity recognition in an AmI system managing a smart home environment. In order to increase the resilience of the system to highly uncertain and unreliable information, the architecture is enriched by a filtering module to pre-process raw data coming from lower levels, before feeding them to the data fusion and reasoning modules in the higher levels.

Acknowledgments

I want to thank all the people who helped me along the way, both professionally and personally. First, I would like to sincerely thank my advisor, Prof. Giuseppe Lo Re, for all his help and guidance over the years. His encouragement and suggestions have been invaluable since the beginning of my Ph.D. research.

I wish to express my deepest gratitude to Prof. Salvatore Gaglio for always sharing his knowledge and experience with me.

I am especially thankful to Dr. Alessandra De Paola for constantly encouraging and inspiring me, and for all her comments and suggestions to improve my research projects. Without her dedication and hard work, none of this would have been possible.

I owe a huge debt of gratitude to Prof. Sajal K. Das and Prof. Simone Silvestri. Your brilliant insights have been a tremendous help, and I sincerely appreciate the time you took to guide me and my work.

I will be forever grateful to my dear friend Francesco Restuccia for his invaluable help over the past years and for being so incredibly supportive. You are the best.

Special thanks to all my amazing colleagues and friends at the NDS Research Group and CReWMaN Lab. It is always a great pleasure and honor to work with all of you.

I am truly lucky to be surrounded by friends who always support me. I cannot properly express what you all mean to me, but I will try nevertheless.

To Gloria Martorella, thank you from the bottom of my heart for your unfailing enthusiasm, laughs and encouragement. I don't know what I would do without you. To Vincenzo Agate, thank you for making me smile and laugh even on bad days. To Davide Vassallo and Antonino Piazza, I can't wait until the next time

we are all reunited, because it will be awesome. Thank you for everything you've done for me.

Last but not least, I can't begin to express how thankful I am to my family, and especially to my parents and my brother, who are always amazingly helpful and patient in dealing with me. Thank you for your unwavering and unconditional love and support. You make all the hard work worth it.

Contents

Abstract **ii**

Acknowledgments **iii**

Glossary **x**

1 Introduction **1**

1.1 Motivations and Goals 2

1.2 Contributions 6

1.3 Dissertation Outline 7

1.4 Publications 7

2 High-level System Architecture **9**

2.1 Related Work 10

2.2 Ambient Intelligence and BMS 11

2.3 Architecture Design 13

2.4 Ontology-based Self Modeling 17

2.4.1 General Concepts 18

2.4.2 Domain Concepts 20

2.5 Adaptive Behavior of the Monitoring System 22

2.5.1 Rule-based Reasoning 23

3 Context-Aware Multi-Sensor Data Fusion **30**

3.1 Related Work 31

3.2 Architecture of the data fusion system 33

3.2.1 Data Fusion Module 35

3.3 Context-awareness . . . 39
3.4 Experimental Analysis . . . 41
3.4.1 Simulation Setting . . . 41
3.4.2 Performance Metrics . . . 43
3.4.3 Experimental Results . . . 44

4 Context-Aware Self-Optimization 52
4.1 The Three-tier Architecture . . . 52
4.2 Context-aware Self-Optimization . . . 55
4.2.1 Feature Extraction . . . 56
4.2.2 Global Optimization . . . 59
4.2.3 Constrained Optimization . . . 63
4.3 Case Study: Activity Recognition in AmI Scenario . . . 65
4.4 Experimental Evaluation . . . 67
4.4.1 Experimental Results . . . 68

5 Filtering out unreliable data 75
5.1 Related Work . . . 76
5.2 Application Scenario . . . 78
5.3 Mobile Trusted Participants . . . 80
5.3.1 MTP Optimization Problem . . . 81
5.4 The FIRST Filtering Framework . . . 82
5.4.1 Computation of Validation Probability . . . 83
5.4.2 Likelihood Estimation Algorithm . . . 85
5.4.3 Solving the MTP Optimization Problem . . . 88
5.4.4 Practical Implementation . . . 90
5.5 Experimental Results . . . 91
5.5.1 Participatory Traffic Sensing . . . 91
5.5.2 Participatory PerCom . . . 98

Conclusions 102

Bibliography 104

List of Figures

2.1 High-level scheme of a WSAN installed in an office environment . . 12
2.2 Architecture of the autonomic AmI system 14
2.3 Details of the sub-modules of Understanding and Reasoning 16
2.4 Taxonomy of the SystemModule class 19
2.5 Data types used as inputs and outputs of the sub-modules 20
2.6 Templates used by the rule-based inference engine 24
2.7 Finite state machine of system conditions 27

3.1 Architecture of the context-aware data fusion system 34
3.2 Structure of the DBN used for the context-aware data fusion 35
3.3 Results of the system varying the granularity of the period of day . 45
3.4 Frequency of the activities during different periods of day 46
3.5 Frequency of the activities during different days of the week 46
3.6 Frequency of the activities during different months of the year . . . 47
3.7 Confusion matrix of the baseline data fusion system 48
3.8 Diversity index of each activity 49
3.9 Frequency of the activities performed in the kitchen 49
3.10 Improvement of context-aware inference accuracy using few sensors . 51

4.1 Three-tier architecture of the self-optimization module 54
4.2 Functions used in the Global Optimization module 60
4.3 The Global Optimization module influence diagram 62
4.4 The Constrained Optimization module influence diagram 64
4.5 Results of the three systems considered during a given week 68
4.6 True positives, precision and F-score of each activity 71

4.7 Results of the adaptive system in a single day 72
4.8 Sensor statistics usage with a fixed and variable cost function . . . 73

5.1 SCP system architecture 78
5.2 Example of an MTP moving over the sensing area 81
5.3 Components of FIRST 83
5.4 Computation of the probability of validating sensing reports 84
5.5 Heatmap of mobility traces vs. arterial roads 86
5.6 Example of the left-most insertion point algorithm 90
5.7 Comparison of mobility traces vs. Likelihood Estimation Algorithm 92
5.8 Error rate when varying the number of MTPs 93
5.9 Results of the MTP Optimization Algorithm 93
5.10 Results of the corruption attack 96
5.11 Results of the on-off attack 96
5.12 Results of the collusion attack 97
5.13 Acceptance probability in corruption, on-off, and collusion attacks . 98
5.14 Screenshots of the Android and iOS apps 99
5.15 Position of the Bluetooth beacons 100
5.16 Comparison of FIRST vs. other systems 101

List of Tables

3.1 Example Conditional Probability Table for state transition 38
3.2 Example Conditional Probability Table for sensor models 38
3.3 Results of the compared data fusion systems 47

4.1 Accuracy of the proposed and baseline systems in each test 69
4.2 Detailed results of the proposed and baseline systems 70

5.1 Experimental parameters used in the FIRST experiments 94

Glossary

ADL	Activities of Daily Living
AI	Artificial Intelligence
AmI	Ambient Intelligence
BMS	Building Management System
CPT	Conditional Probability Table
CVP	Computation of Validation Probability
DBN	Dynamic Bayesian Network
HVAC	Heating, Ventilation and Air Conditioning
LEA	Likelihood Estimation Algorithm
MOA	MTP Optimization Algorithm
MOP	MTP Optimization Problem
MTP	Mobile Trusted Participant
SC	Smartphone Crowdsensing
TPM	Trusted Platform Module
WSAN	Wireless Sensor and Actuator Network
WSN	Wireless Sensor Network

Chapter 1

Introduction

Smart environments and intelligent systems designed for real-world applications are often based on the *sensing-reasoning-acting* paradigm; generally, they exploit a sensory infrastructure to collect measurements, which is then used to obtain a high-level description of the state of the environment, and of the current context, reason about it, and select actions to be performed in order to achieve the desired system goals. In many cases the sensory infrastructure consists of a multitude of heterogeneous pervasive devices, which may produce a non-negligible uncertainty such as noisy data and measurements that impact the inference accuracy and energy consumption, and sensors on mobile devices, which are often energy hungry (De Paola et al., 2014; Rahmati et al., 2015). In such scenarios it is convenient to adopt a multi-sensor data fusion method, capable of dealing with uncertain situations.

This work proposes a multi-tier cognitive architecture for a complete smart environment management system which is able to self-configure and self-optimize its own submodules, so as to understand what is happening in the surrounding environment and plan the actions that should be performed to improve environmental conditions for end-users. To achieve this result, the system has to aggregate the raw data fed to it by its sensory infrastructure, integrating heterogeneous and distributed information to get a unified view of the current situation and efficiently plan the actions that must be performed to satisfy the needs of users.

Thus, one of the key components of the smart environment management system presented here is its data fusion subsystem. In particular, this work presents a context-aware, self-optimizing, adaptive system for sensor data fusion, which includes a self-optimization module that reconfigures the sensory infrastructure, by dynamically selecting the subset of sensors to use, in order to optimize the trade-off between inference accuracy of the system and energy consumption of the sensory devices.

To analyze the impact of feeding unreliable information to the system, the proposed architecture is enriched by a filtering module that aims to accurately classify reliable and unreliable raw data, before sending them to higher level modules.

1.1 Motivations and Goals

Ambient Intelligence (AmI) is an application paradigm of Artificial Intelligence, which focuses on users and their needs, with the aim of designing intelligent pervasive systems capable of providing the best possible environmental conditions for end-users. In order to achieve such goals, small intelligent devices are pervasively deployed in the environment and are programmed to act autonomously or in a collaborative approach to learn and satisfy user preferences (Cook et al., 2009). The goal is to create an intelligent environment which feels natural and with which people can interact easily and effortlessly (Ducatel et al., 2001; Remagnino and Foresti, 2005). However, traditional programming techniques may not be able to cope with the AmI paradigm (Sanz et al., 2007), which require autonomous systems that can manage and reconfigure themselves, thereby freeing designers from these demanding tasks.

Within such a scenario, this work proposes a multi-layer cognitive architecture for an autonomic AmI system, which is characterized by continuous self-modeling and self-monitoring, in line with the Autonomic Computing paradigm. According to this paradigm, a truly autonomous agent should rely on an explicit model of itself, of the environment in which it operates, and of its interactions with users. Exploiting such extensive knowledge, the system is thus able to understand the state of the environment and the preferences of its users, as well as using this knowledge to decide what actions should be performed next.

To this end, the proposed architecture exploits ontologies to model the domain knowledge in a unified and machine-computable format, in order to drive the process of knowledge abstraction from raw sensory data up to higher-level concepts. Using such approach, the system is able to understand the interactions between its cognitive subsystem, its physical subsystem (i.e., sensors and actuators) and the environment, thus acting in an appropriate manner to achieve its goals. By exploiting a rule-based inference engine to analyze the ontology and reason about its internal structure, the system is then able to self-configure its sub-modules and modify its own behavior at runtime, in order to minimize energy consumption.

One of the most challenging tasks the system has to perform is processing and merging large amounts of raw sensory data effectively. Multi-sensor data fusion is extensively used in literature to combine data collected by heterogeneous sensors (De Paola et al., 2013). However, since sensor data are often noisy and inaccurate, probabilistic techniques are widely adopted to explicitly model the noise and uncertainty of raw data (Cook, 2010). One of the most effective approaches for this purpose is the adoption of Bayesian belief networks (Koller and Friedman, 2009), which exploit the statistical correlation between sensory measurements and the peculiarities of the surrounding world, allowing the system to deal with the heterogeneity of information sources and with the uncertainty in sensory data and developed models. In particular, Dynamic Bayesian Networks (DBNs) (Murphy, 2002) take into consideration the past belief of the system, in addition to data coming from sensors, making it possible to handle the dynamicity of the observed phenomena.

This work proposes a context-aware multi-sensor data fusion system to infer high-level context information about the surrounding environment, leveraging low-level context information in order to refine the inference process. The output of such process is then exploited by higher level reasoning modules of the architecture to derive new knowledge.

The inference of context information, as a high-level description of the user's activities, is the main goal of the data fusion system; basic context attributes, such as time-related and location-related information, are used to refine the inference process. Such basic context attributes can be reliably and easily sensed, and thus do not increase the uncertainty of the system.

However, it is not always convenient to blindly include all available context information in the data fusion process. On the contrary, as demonstrated by the experimental evaluation presented here, choosing the right combination of context information is fundamental to maximize inference accuracy. To this end, it is advisable to exploit only context attributes which are readily available and easy to measure in a reliable way, so as not to increase the uncertainty of the system. Extensive experiments prove that choosing the right combination of context information is fundamental to maximize inference accuracy, especially when only few sensors are available.

In addition, when performing data fusion, it might not always be efficient to continuously sample all available sensors. On the contrary, if the sensory infrastructure is composed of devices with limited energy resources, it may be useful to activate only a subset of sensors, in order to increase the lifetime of the whole network. This is the case of smart environments whose pervasive sensory infrastructure includes wireless sensor networks (WSNs) (Yick et al., 2008). WSNs are composed of devices, namely sensor nodes, characterized by programmability, wireless communications capability, and limited computational and energy resources, such that it is essential to extend the network lifetime by putting inactive nodes in stand-by state as long as possible. However, since different contexts may require different sensory capabilities, it is not desirable to determine a priori the subset of sensors to use. In a real-world scenario, the context conditions may change over time, implying the need for a system capable of dynamically selecting the subset of sensory devices.

Thus, the proposed data fusion system is able to dynamically reconfigure the sensory infrastructure, by selecting the subset of sensors to use in order to optimize its own performance. The system monitors the behavior of the data fusion process ascertaining if the inference can be considered reliable, and evaluates the behavior of the sensory infrastructure considering the contribution of different sensors. These observations are integrated with other context information to determine whether a different configuration of the sensory infrastructure may be more convenient. Such self-optimization process is performed through dynamic rules, so that the system is able to modify its own goals in response to context changes.

The main novelty of the proposed self-optimization is its adaptiveness to dynamic environments and its capability of extending the lifetime of the sensory infrastructure as the context changes, thus reducing human interventions. Moreover, the proposed solution has the capability of self-detecting those critical conditions in which a reconfiguration is needed in order to maintain a good quality of inference.

The validity of the proposed approach is tested in the application scenario of user activity recognition in an AmI system managing a smart home environment. The AmI scenario is particularly appropriate for the evaluation of the proposed data fusion system, since frequent user interactions make the context highly dynamic.

Furthermore, often sensory devices are only partially related to the observed phenomena, and thus non-negligible noises are typically introduced, e.g., poor sensor quality, lack of sensor calibration, hardware failures, noise from external sources, and imprecision in computing derived values from measurements (Wang et al., 2014b).

The system described so far does not filter out unreliable data coming from its sensory devices. However, this is a very important issue and a challenging task, especially if data are coming from users in addition to traditional sensors, and thus information quality is significantly uncertain.

Indeed, with the emergence of the Smartphone Crowdsensing (SC) paradigm, it is now possible to use humans *as sensors* (Wang et al., 2014a), and exploit their unique abilities to devise sensing applications that monitor the occurrence of complex events, which are difficult to detect with paradigms based only on traditional sensors.

However, the central role of people in the sensing process implies that the success of SC systems is strictly dependent on the reliability of the information sent by participants. As humans may exhibit selfish, opportunistic and unreliable behavior (Paxton and Benford, 2009), accurately classifying the reliability of user reports is paramount.

To this end, the proposed architecture is enriched by a filtering module to detect unreliable raw data before sending them to the data fusion and reasoning modules.

1.2 Contributions

The main contributions of the work presented in this dissertation are:

- The design of a multi-tier architecture for a complete autonomic AmI system capable of analyzing itself and its monitoring processes, and consequently of managing and reconfiguring its own sub-modules to better satisfy users' needs. To achieve such a degree of autonomy and self-awareness, the proposed AmI system exploits the knowledge contained in an ontology that formally describes the environment it operates in, as well as the structure of the system itself.

- The design and development of a context-aware Dynamic Bayesian Network which improves the accuracy of existing probabilistic systems by including basic context attributes to refine the inference process of high-level context information. The suitability of such an approach is demonstrated in the application scenario of user activity recognition in a smart home environment. Extensive experimental results prove that choosing the right combination of context attributes is fundamental to maximize the inference accuracy, especially when only few sensors are available.

- The design and development of a context-aware self-optimization system for sensor data fusion, based on a three-tier architecture. Heterogeneous data collected by sensors at the lowest tier are combined at the intermediate tier by the context-aware Dynamic Bayesian Network proposed in this work. At the highest tier, a self-optimization process dynamically reconfigures the sensory infrastructure, by sampling a subset of sensors in order to minimize energy consumption and maximize inference accuracy. Experimental results show that the proposed solution outperforms static approaches for multi-sensor data fusion, achieving substantial energy savings whilst maintaining a high degree of inference accuracy.

- The design and development of FIRST, a filtering module to classify and discard unreliable raw data coming from sensors and users alike, by exploiting mobile trusting participants (MTPs) to assess the truthfulness of sensing

reports. FIRST models and solves the challenging problem of determining, before deployment, the minimum number of trusted participants needed to achieve the desired classification accuracy, by exploiting a novel algorithm based on image processing.

1.3 Dissertation Outline

The remainder of the dissertation follows a top-down approach to describe the proposed architecture, starting with the reasoning module in the top level and the data fusion module to aggregate data coming from sensors, and ending with the filtering module in the lowest level to pre-process raw data.

Chapter 2 describes the high-level architecture for a complete autonomic AmI system, highlighting the aspects that ensure its adaptivity and context-awareness, and presents the ontology adopted by the system. Chapter 3 proposes a multi-sensor Data Fusion module exploiting a context-aware Dynamic Bayesian Network to infer users' activities in a smart home, and discusses various context attributes that can be exploited to increase the accuracy of the system, with experimental results to validate the analysis. Chapter 4 presents a context-aware self-optimization module which dynamically reconfigures the sensory infrastructure of the Data Fusion system, selecting the subset of sensors to be used to optimize the trade-off between inference accuracy and energy consumption of the sensory devices. Finally, Chapter 5 proposes a filtering module to classify between reliable and unreliable sensor reports before sending them to higher levels of the architecture, in the application scenario of Smartphone Crowdsensing.

1.4 Publications

Parts of the work in this thesis have been published in several referred conference proceedings and journals:

- Alessandra De Paola, Pierluca Ferraro, Salvatore Gaglio, and Giuseppe Lo Re. Autonomic Behaviors in an Ambient Intelligence System. *IEEE Sym-*

posium on Computational Intelligence for Human-like Intelligence (CIHLI 2014), Orlando, Florida, USA, 2014.

- Alessandra De Paola, Pierluca Ferraro, Salvatore Gaglio, and Giuseppe Lo Re. Context-Awareness for Multi-Sensor Data Fusion in Smart Environments. *Proceedings of the 15th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2016)*, Genoa, Italy, 2016.

- Alessandra De Paola, Pierluca Ferraro, Salvatore Gaglio, Giuseppe Lo Re, and Sajal K. Das. An Adaptive Bayesian System for Context-Aware Data Fusion in Smart Environments. *IEEE Transactions on Mobile Computing (TMC)*, 2016.

Chapter 2

High-level System Architecture

The proposed architecture is aimed at building an AmI system which is autonomous and able to self-configure and self-manage, in line with the paradigm of Autonomic Computing. The case study presented here comprises a Building Management System (BMS) capable of handling environmental conditions, such as temperature, humidity and lighting in an office environment, with the twofold goal of maximizing user comfort and minimizing the energy consumption of sensors and actuators (De Paola et al., 2014).

The remainder of this chapter is organized as follows. Section 2.1 analyzes some relevant approaches proposed in the literature in the fields of adaptive behavior and Autonomic Computing. Section 2.2 introduces the paradigm of Ambient Intelligence and presents Sensor9k, which is a testbed designed to facilitate reasoning about user comfort and energy saving in an office environment. Section 2.3 outlines the general architecture of the proposed system, highlighting the aspects that ensure its adaptivity and context-awareness. Section 2.4 describes the ontology adopted, which defines the concepts used by the system, that relate to the environment, the user and its interactions with it. Section 2.5 gives an outline of the rule-based inference engine, which allows the proposed system to configure and manage itself.

2.1 Related Work

In the field of Ambient Intelligence, as shown by (Hagras, 2007), it is very difficult, if not practically impossible, to build accurate mathematical models capable of grasping the complexity and dynamism of the real world. Therefore, it is necessary to adopt advanced techniques of Artificial Intelligence to manage the multitude of devices placed in the environment and build a layer of distributed intelligence so as to handle the uncertainty inherent in the data collected. The design of AmI systems thus employs techniques from different disciplines and research fields. Some of the most common AI techniques used in previous research to study the behavior of users and learn their preferences have been Bayesian networks (Kushwaha et al., 2004), fuzzy systems (Doctor et al., 2005) and neural networks (Mozer, 1998).

In such a scenario, the Autonomic Computing paradigm might be the key to developing truly adaptive agents (Jacyno et al., 2013). The paradigm was originally proposed by IBM, in analogy to the autonomic nervous system (Kephart and Chess, 2003), which allows the human body to maintain the balance between all of its complex subsystems, responding to unpredictable external stimuli and overcoming the dangers caused by external agents.

Similarly, computational architectures inspired by such a model are composed of collections of autonomic elements that are able to respond to changes autonomously, adapting to the environment in order to better achieve their goals. A self-configuring and self-managing AmI system should therefore include and analyze knowledge about itself and the environment in which it acts, so as to constantly update its model of the world (Kawamura et al., 2005).

By describing concepts from both domains in a machine-computable way, the agent is able to unify the mechanisms that it uses to reason about the world and about itself, exploiting the semantic enrichment of data processed by the system (Bermejo-Alonso et al., 2010).

One issue on which many researchers agree is that an autonomous agent should not be limited by fixed criteria, because real environments are characterized by uncertainty, and not all contingencies can be foreseen at design time. In this regard, a key aspect of every autonomous agent is its capacity of adaptive decision-making (Chadderdon, 2008), which allows it to make correct decisions based on

the current situation; this capability improves the context-awareness of the system, helping it to achieve its goals in the best possible way.

To implement these paradigms, various approaches have been proposed in the literature. For example, the authors of (Andry et al., 2004) propose an architecture based on neural networks that allows their robot to learn complex sequences of actions so as to coordinate vision and arms movements.

Others have used genetic algorithms (Ram et al., 1994), Bayesian networks (Ferreira et al., 2012) and reinforcement learning techniques such as Q-learning (Erbas et al., 2014) to build robust agents that are capable of operating in an uncertain world.

2.2 Ambient Intelligence and BMS

AmI envisages innovative scenarios in which intelligent systems assist human beings in their daily activities, without the inconvenience of intrusive technologies which, in this paradigm, remain confined to the background (Remagnino and Foresti, 2005).

The design of intelligent Building Management Systems (BMSs) is one of the most significant uses of AmI techniques. Other applications include care for the elderly, medical assistance, kindergartens, and car automation. The main goal of BMSs is the control of environmental conditions in buildings (e.g., temperature, humidity and lighting) to satisfy user requirements and minimize energy consumption. BMSs need a sensory and actuator infrastructure constituting their direct link to the real world. Sensors acquire environmental data (e.g., temperature, light intensity, noise, humidity) and context information (e.g., user presence and user activities), whereas the actuator infrastructure is composed of physical devices in the building that can influence the state of the environment (e.g., artificial lighting systems and heating, ventilation and air conditioning (HVAC) systems).

A key requirement of AmI systems is the low intrusiveness of the underlying technology. This means that the sensory and actuator infrastructures have to be characterized by a low degree of physical invasiveness.

This work is inspired by the BMS architecture proposed in Sensor9k (De Paola et al., 2012a), a pervasive testbed whose aim is to support the development

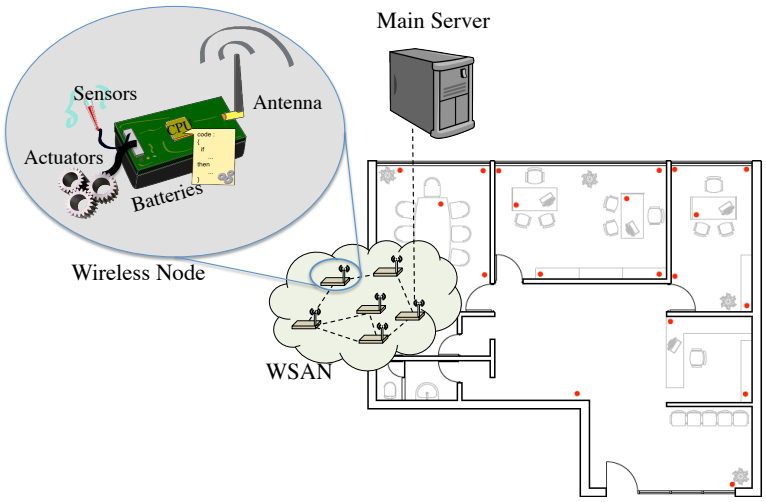


Figure 2.1: High-level scheme of a Wireless Sensor and Actuator Network (WSAN) installed in an office environment.

of energy-aware AmI systems. The name of the testbed recalls the fictional *HAL 9000* artificial intelligent system, whose sensory and actuator terminations permeated the spaceship in "2001: A Space Odyssey". The physical infrastructure of Sensor9k exploits the technology of Wireless Sensor and Actuator Networks (WSANs) (De Mil et al., 2008). These networks are composed of a set of small devices, called *nodes*, which are in most cases energetically autonomous, programmable, and able to perform small computations on boards and to wirelessly communicate with each other. Each node can be equipped with different sensors and can control certain connected actuators.

In the proposed setting, sensors installed on nodes gather information about the state of the environment and about the context, as well as information about the state of the node itself, such as residual battery charge. Furthermore, by exploiting a set of ad-hoc sensors, it is possible to monitor the energy consumption of all the electrical appliances installed in the monitored environment.

Appliances are controlled by the actuators installed on sensor nodes, by means of the transmission of opportune control signals. Moreover, the proposed architecture lets the user manually control the available appliances, and apposite sensors capture each of these interactions. The simpler actuators provided by the system are represented by remotely controllable power relays enriched with the capability

of providing information about their current state (e.g., the artificial lighting relay controller). More complex forms of actuators are those controlling domestic appliances (typically by means of IR remotes) frequently found in homes or offices, such as HVAC systems.

The sensory information acquired by WSANs, through the hierarchical communication infrastructure provided by Sensor9k, is sent to a centralized server, which hosts the reasoning components. The choice of centralizing reasoning activity makes it possible to preserve its consistency and uniqueness (Amigoni et al., 2004), even in the presence of a distributed and pervasive physical layer. A high-level scheme of a WSAN in an office environment is shown in Fig. 2.1.

Furthermore, the middleware provided by Sensor9k enables dynamic tuning of the sensors' behavior via control messages sent at runtime. By means of these control messages it is possible, for instance, to tune the sampling rate of sensor nodes, or to switch nodes to a low-power mode in order to minimize their energy consumption. In order to reduce the sensory infrastructure handling effort for the user, programmability is also exploited by the adoption of replicas of critical WSAN devices, thereby guaranteeing fault tolerance and robustness. The whole sensory infrastructure comprises a set of fully active sensor nodes and a complementary set of duplicate nodes. Replica nodes start in low-power mode with data gathering and transmission functionalities disabled. Listening to control messages is the only active functionality in low-power mode. A specific control message can switch a node from this condition to normal functioning mode in order to activate ambient monitoring. The proposed adaptive system exploits such features in order to resume its full operability when a given node switches into fault condition.

2.3 Architecture Design

The architecture proposed in this work is inspired by the paradigms of Ambient Intelligence and Autonomic Computing. The main goal is to design and develop an AmI system that is able to self-configure and self-manage, taking inspiration from certain intrinsic features of human beings, such as introspection and self-awareness, to make the architecture more robust and resilient and improve the adaptivity of the resulting system. Using the information represented by the ontologies and

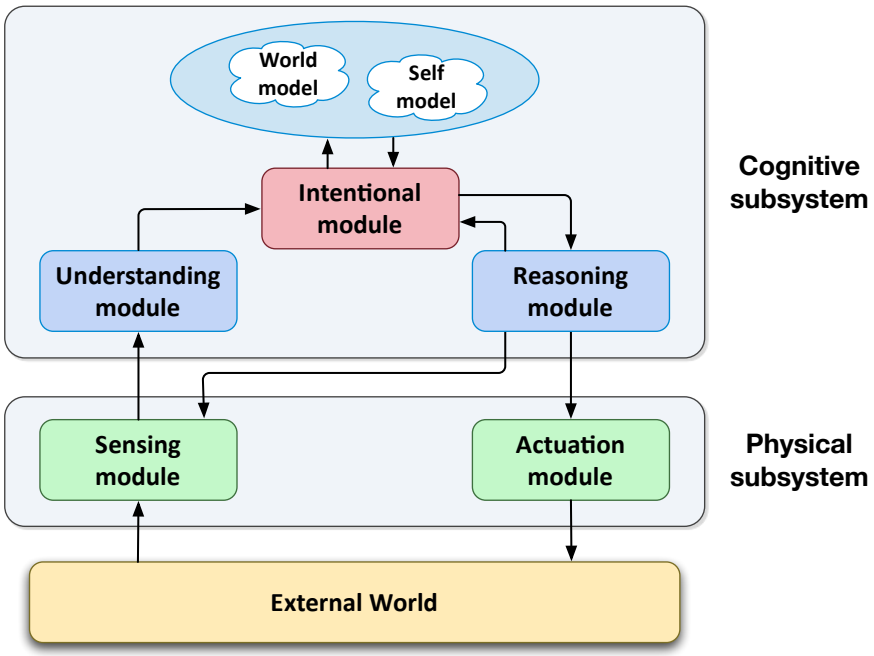


Figure 2.2: Architecture of the autonomic AmI system.

instantiated in the knowledge base of the system, the agent can then monitor the status of its own sub-modules, dynamically adapting its behavior to the context, so as to best suit the users' requirements, according to the high-level policies defined by administrators.

To achieve this result, the system must be able to aggregate the raw data fed to it by its own sensory infrastructure, integrating heterogeneous and distributed information to get a unified view of the current situation and efficiently plan the actions that must be performed to satisfy users' needs. Although Artificial Intelligence literature includes several techniques for representing the knowledge necessary for the functioning of such a system (Gu et al., 2004; Preuveneers et al., 2004), the amount of data collected by sensors in a realistic setting is huge, and would lead to serious problems in terms of computational efficiency.

For this reason, the proposed system filters information from sensors, by using modules that operate at different levels of abstraction and various techniques for the efficient fusion, modeling and interpretation of these data, as will be described

in the next chapters. As shown in Fig. 2.2, the architecture consists of five main modules, each of which has a highly flexible and configurable structure:

- a *Sensing module*, based on Wireless Sensor Networks (WSNs), which is responsible for collecting the raw sensory data about the relevant features of the surrounding environment and sending that information to higher levels;
- an *Understanding module* which processes sensory data, aggregating and representing them in gradually increasing levels of abstraction, so as to describe the environment through high-level concepts, which summarize in a succinct way the huge amount of data extracted by sensors;
- an *Intentional module* that directly accesses the ontological models of the environment and of itself, and updates them according to the data collected, deciding the short-term goals of the system in compliance with a high-level policy;
- a *Reasoning module*, which plans the most appropriate sequence of actions that the system must perform to achieve the goals set by the Intentional module, so as to better satisfy user needs;
- an *Actuation module* that can change the status of the environment, by operating heating, ventilation, air conditioning (HVAC) and lighting systems, as well as other elements that affect the environmental parameters.

The relationships between all the modules, the external world and the ontological model are also illustrated in Fig. 2.2. The key element of the architecture is the feedback loop, which requires that the system is aware of itself, of the surrounding environment and of their interactions with each other. This allows the system to keep its world model and self model constantly updated, based on the interactions between the physical subsystem, the cognitive subsystem and the external world.

In other words, by using an appropriate sensory infrastructure, continuous monitoring allows the system to identify relevant events concerning the users and the environment, so as to start the reasoning and planning processes, and then initiate the actions that are carried out by means of suitable actuators.

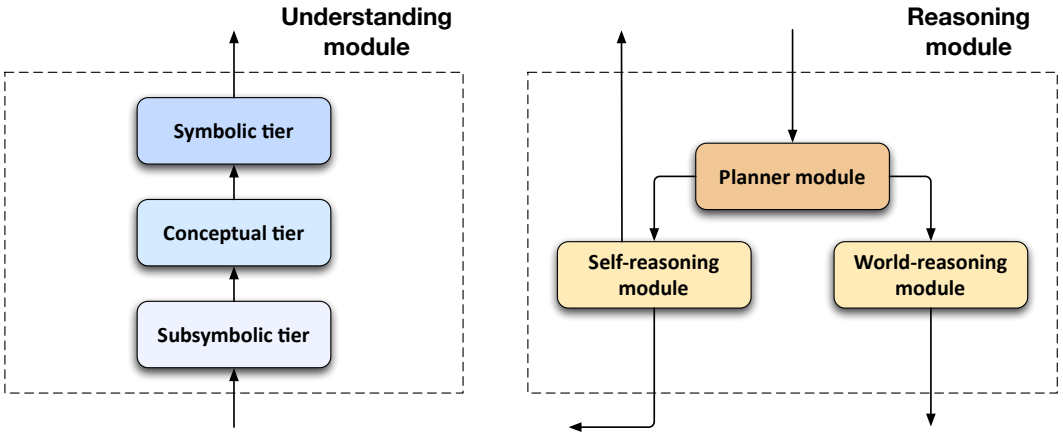


Figure 2.3: Details of the sub-modules of Understanding and Reasoning.

In addition, each of the main modules consists of sub-modules performing specific tasks within the feedback loop. The behavior of the system as a whole emerges from the interaction and collaboration of these subsystems.

As shown in Fig. 2.3, the Reasoning module is organized into three different sub-modules, named *Planner*, *Self-reasoning* and *World-reasoning*. The Understanding module is also divided into three tiers, namely the *Subsymbolic*, *Conceptual* and *Symbolic* tiers. Knowledge flows through these tiers, from raw sensory data up to qualitative descriptions of the environment at the highest level of abstraction.

The Subsymbolic tier is responsible for pre-processing the data coming from sensors, by uniforming the sampling rate and eliminating any outliers or unreliable information. The Conceptual module is an intermediate tier between the Subsymbolic and the Symbolic levels. This tier uses appropriate classification and data fusion techniques to identify high-level concepts which are then represented in the Symbolic level. The specific implementation of the sub-modules may vary depending on the specific application scenario. Examples of Conceptual and Subsymbolic modules in the application scenarios of activity recognition and smartphone crowdsensing are presented in the following chapters.

A more detailed description of the system structure and of the role played by each module is presented in the following section, along with an analysis of the ontology used by the system.

2.4 Ontology-based Self Modeling

In order to dynamically adapt its behavior to the context where it is running, an autonomous system should possess an explicit model of itself, of the surrounding environment, and of the ways it can interact with users (Hernández et al., 2009).

Therefore, the system has to manage a large amount of information that may change quickly and often. In order to represent such knowledge in an efficient and machine-computable way, then, it is advisable to use ontologies, which are documents that formally define the relationships among a set of terms belonging to a specific domain (Ribino et al., 2009). The proposed ontology, developed in the OWL 2 language (World Wide Web Consortium, 2012), provides a uniform terminology to describe the environment and its properties, the user and his interactions with it, as well as the system components and their interconnections. Furthermore, it describes how data flows within the system, highlighting the relationships between sensors and monitored environmental properties, and accomplishes several objectives:

- it represents in a formal and machine-computable way the relationships between concepts belonging to the domain of interest;
- it enables the system to dynamically and autonomously reconfigure itself, according to the goals defined by the Intentional module;
- it allows system administrators to define high-level policies affecting the overall behavior of the system;
- it makes it easy to represent the rules needed to infer appropriate control actions, taking into account the status of the environment and the current goals of the system.

The adoption of ontologies and semantic technologies is essential for the self-monitoring and self-management operations, and makes the system architecture sufficiently abstract and reusable in different application scenarios. In order to infer new knowledge from the concepts defined in the ontology, performing automated reasoning on the domain, rule-oriented languages are commonly used, such as the Semantic Web Rule Language (SWRL) (Horrocks et al., 2004).

This work proposes a semantic representation that exploits two different ontologies. The first one represents the structure of a generic building management system, and can be easily reused for several applications, since it is not tied to a specific Ambient Intelligence scenario. The second ontology extends the first one by defining subclasses and individuals, which describe a specific instance of the considered application scenario, i.e., the BMS for controlling ambient conditions in an office environment.

2.4.1 General Concepts

The top-level ontology contains the descriptions of basic elements constituting the sensory and actuator subsystems in the `Device` class and in its subclasses, such as `Sensor`, `Node` and `Actuator`.

The ontological description provides important information about sensors, such as energy consumption, sampling rate, continuity of monitoring and the node on which the sensor is installed. Each instance of `Device` perceives a specific attribute, or acts upon it. These properties, modeled by means of the `AmbientProperty` class and its subclasses, can be related to physical phenomena such as light and temperature, observable events like user activity, or the status of a particular environmental element, such as a window or an air cooler.

An important aspect modeled in the ontology concerns the basic topological elements of the environment in which the system operates. The `TopologicalElement` class and its subclasses (e.g., `Door` and `Room`) represent such knowledge. For instance, the `Room` subclass describes the properties defining the topological relationships between areas: `nextToRoom` indicates that two rooms are adjacent, whilst `isInRoom` represents each device in the room in which it is deployed.

In order to describe the status of a room, the `Room` subclass is enriched by the `SystemCondition` property. Considering such a property as input, the system dynamically adapts to the current situation by changing its configuration, its behavior and its goals so as to focus on the most important aspects of the external environment.

The `SystemCondition` affects the short-term goals identified by the Intentional module, and thus constitutes the basis of the adaptive and autonomic capabilities

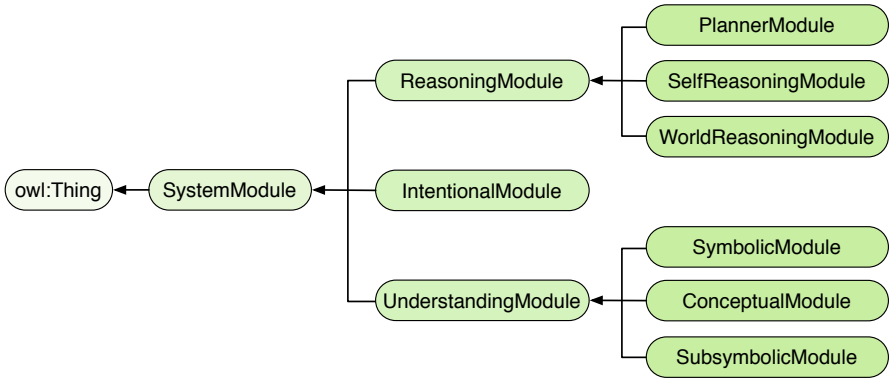


Figure 2.4: Taxonomy of the `SystemModule` class.

of the system. In other words, each condition is associated with a different set of rules, so the system responds differently to the same external stimuli depending on its current condition. `SystemCondition` is a generic concept, which can be exploited by several applications that use the top-level ontology.

The specific ways in which this property affects the functioning of the system depends on the application scenario. The possible values of this property are highlighted in Section 2.5.1 below, along with some of the special rules that are triggered in the various conditions and the rules for the transition between conditions.

All the architectural components described in Section 2.3 are represented by the `SystemModule` class, whose taxonomic organization is shown in Fig. 2.4. The main categorization is functional, and differentiates between the Understanding, Intentional and Reasoning modules, while further subdivisions identify the various specialties.

All the data processed by the system are modeled by the `DataType` class, while the relationships existing between each module and its input and output data is determined by two ad hoc properties (`hasInputData` and `hasOutputData`). By exploiting such properties, it is possible to analyze how the data flows between modules, abstracting the knowledge processing that takes place within them through the integration of information from raw sensory data up to high-level symbolic concepts. This integration of data, which gives the system a unified view of the current situation, constitutes the basis of its self-awareness and self-reasoning capabilities.

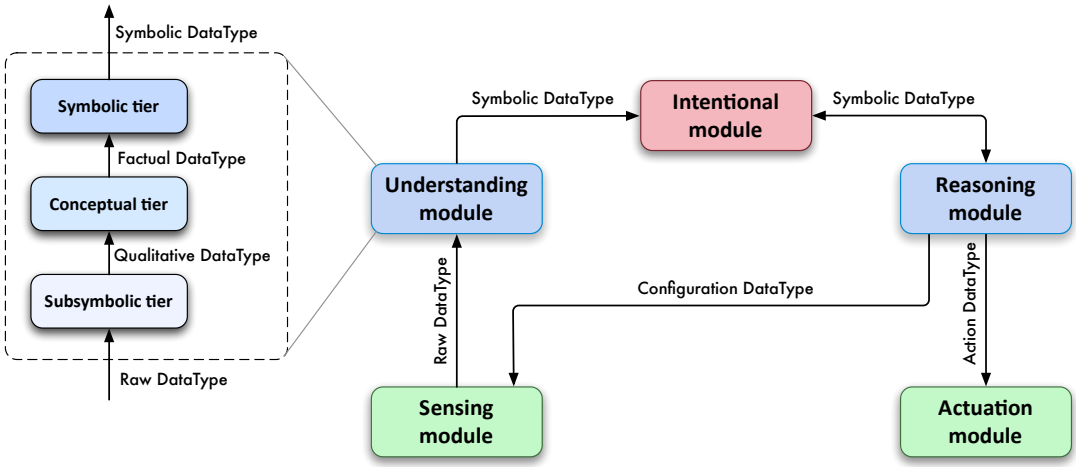


Figure 2.5: Data types used as inputs and outputs of the different sub-modules.

Fig. 2.5 shows the data flow among the modules that comprise the AmI system, highlighting the process of knowledge abstraction from raw sensory data up to higher-level symbolic data, which occurs in the `UnderstandingModule`. The basic sensors deployed in the monitored premises represent the input devices of the system, and their collected data are sent as instances of `RawDataType` to the `UnderstandingModule`. In addition, the nodes of the wireless sensor network send information to the `UnderstandingModule` in order to communicate their status, thus allowing the AmI system to reason about its own sub-modules.

Symbolic data constitute the input of the `IntentionalModule`, which updates the world and self models according to the data collected, planning the short-term goals of the system and triggering the `ReasoningModule` and its sub-modules, which carry out the reasoning and self-reasoning activities, as described in Section 2.3. Finally, in order to close the loop, the `WorldReasoningModule` sends action commands to the actuators, while nodes and sensors receive configuration commands from the `SelfReasoningModule`.

2.4.2 Domain Concepts

The specific domain ontology is based on the general one, enriching it with specific knowledge related to a specific AmI system instance. As previously described in Section 2.2, the application scenario considered here is that of Sensor9k (De Paola

et al., 2012a), a testbed designed for reasoning about user comfort and energy saving in an office environment. Each of the nodes installed in the office environment is included in the ontology as an instance of the `Node` class, and is related to the specific sensors that it hosts through the property `isEquippedWith`. Several features of the domain ontology are represented by means of properties that can be physical or abstract, simple or complex. The `PhysicalProperty` class is instantiated by each of the environmental properties monitored in the application scenario, such as light, temperature, humidity, barometric pressure, loudness and so on.

As an example of the self-reasoning capabilities of the system, it is worth noticing that when defining properties, it is not necessary to specify which of them are directly observable by the sensory infrastructure, since the system can automatically infer that any properties perceived by at least one of its sensors are directly observable. This reasoning capability is easily implementable by exploiting one of the SWRL rules exploited by the inference engine:

$$\texttt{Sensor(?x)} \wedge \texttt{AmbientProperty(?y)} \wedge \texttt{senses(?x, ?y)} \rightarrow \texttt{ObservableProperty(?y)}$$

In this way, by exploiting the functionalities of self-diagnosis and self-monitoring, the system recognizes dynamically, at runtime, which of the environmental properties can be perceived by the sensors actually available. This feature also affects the short-term goals decided on by the `IntentionalModule`, since the system might easily infer that it is not adequately able to measure certain complex properties, which are defined as a composition of other basic properties.

In such situations, the system recognizes its inadequacy for the assigned task and, in the absence of a viable alternative, may notify the administrators, as explained in Section 2.5.1 below.

Complex properties allow the system to perform high-level reasoning using abstract concepts, and also form the basis of many of the autonomic and self-reasoning capabilities of the system. Examples of complex properties defined in the domain ontology are `UserInBuilding`, `UserInOffice` and `RoomOccupancy`, which represent, respectively, the presence of the user in the building or in his own office, and the number of people in a monitored room. `UserInOffice`, for

instance, is affected by several basic properties, such as `RFIDStriping`, `WorkstationActivity`, `DoorStatus` and `Loudness`, which in turn are instances of a subclass of `AmbientProperty`.

To represent the system architecture within the domain ontology, the taxonomic organization defined in the top-level one was extended with new subclasses, and the actual modules were instantiated as individuals of these subclasses, so as to reflect the topological and semantic organization of the monitored environment.

As an example of such a process, the domain ontology particularizes a class of sub-symbolic modules capable of processing the temperature information, and this class is instantiated in a set of specific individuals. Each of these individuals is devoted to the monitoring of a given managed room. The same process is followed for all other monitored environmental properties, such as humidity and lighting.

2.5 Adaptive Behavior of the Monitoring System

On the basis of the ontological representation of the system, the surrounding environment, and their interactions, this section illustrates how the Reasoning module manages the activities of self-monitoring and self-configuration, emphasizing its integration into the multi-tier cognitive architecture of the system.

The implementation of the Autonomic Computing paradigm proposed here is based on the classic monitor-analyze-plan-execute cycle, which underlies an intelligent control loop (Kephart and Chess, 2003), as explained in Section 2.4 in the description of the data flow within the sub-modules constituting the AmI system.

Although this paradigm requires the system to maintain a constantly updated model of itself and of the surrounding environment, it enables dynamic and runtime reconfiguration of the monitoring subsystem.

In other words, the planning module implements rule-based reasoning behavior capable of reconfiguring the sensors on the basis of certain parameters, such as a user's presence, the degree of accuracy of the monitored information, the state of the sensors, their energy consumption, and the residual lifetime of the battery-powered sensor nodes. However, if the system detects serious problems requiring the intervention of administrators, opportune alerts are generated.

Exploiting its reasoning modules, the system is able to understand which submodules are needed to infer certain concepts, and identifies the relationships between the environmental properties of interest and the specific sensors that perceive them.

The information required for the self-reasoning and self-configuration of the monitoring system come from the subsystems of the Understanding module, and are used by the Intentional module to update the facts defined in the ontology. This, in turn, affects the short-term goals planned by the Intentional module to comply with the high-level policy set by administrators, such as the minimization of energy consumption when the user is not present within the building.

To achieve these objectives, the Self-reasoning module exploits a set of rules that use the knowledge contained in the ontology in order to infer new evidence and plan the sequence of actions that the system must perform. At the end of the planning process, the Self-reasoning module sends the resulting configuration commands to the sensors.

The following section introduces the rule-based inference engine at the core of the system, describing some of the most important rules for the functioning of the automatic reasoner.

2.5.1 Rule-based Reasoning

To implement the operations of reasoning and self-reasoning, this work adopts Jess (Java Expert System Shell) (Friedman, 2003), a rule-based inference engine developed in Java.

Jess expresses logical rules with a syntax similar to LISP, and uses a pattern-matching algorithm to query the knowledge base and extract the information required by the system. Each fact contained within the knowledge base is a true proposition about the world or the system itself, and belongs to a template, just as every object is a member of a class in the object-oriented programming style.

Jess templates define the name of a fact, the properties that a fact possesses and, optionally, the corresponding range of values that properties may assume. Four types of templates are used in the system:

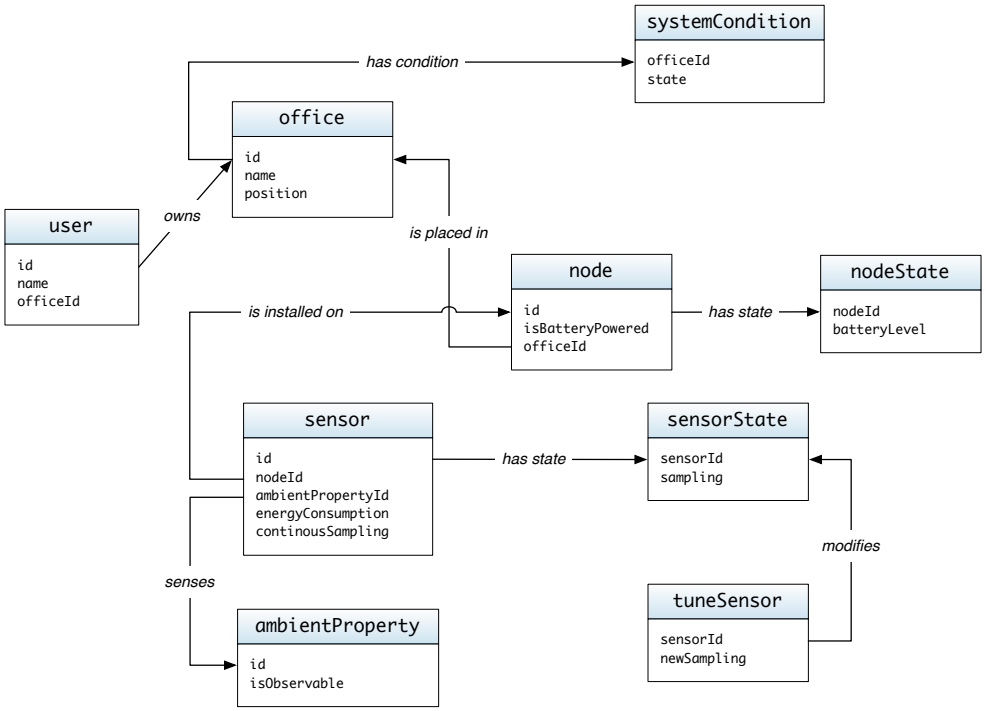


Figure 2.6: Examples of the templates used by the rule-based inference engine.

- *static knowledge* templates, representing the initial facts known by the reasoner during the setup of the system, such as the locations of nodes and sensors, or static information about the users and their offices;
- *dynamic knowledge* templates, relating to information that is continuously updated at runtime, such as the status of sensors and actuators, the presence of users within their offices, or the accuracy of the monitoring activity that is estimated by the self-diagnosis modules;
- templates for facts which represent the output of the planning operations of the `WorldReasoningModule` and the `SelfReasoningModule`, that is *action commands* for the actuators and configuration commands for the sensors;
- templates relating to the *alerts* that are sent to administrators when the self-monitoring modules report any inadequacy of the system with respect to the operations that should be performed.

Fig. 2.6 illustrates some examples of the templates used by the system, showing the relationships between nodes, sensors and environmental properties.

The `tuneSensor` template, for instance, allows the planner to modify the sensor sampling rate, thus allowing a reduction in the energy consumption of a node, or a more accurate acquisition of a specific environmental variable. Possible values are {`on, off, up, down, min, max`}. This template allows the system to dynamically adapt to the context, using the residual energy of the battery-powered nodes in an intelligent way.

Besides generating configuration commands for sensors, the Self-reasoning module also yields alerts for system administrators, such as insufficient sensing capability or short sensing time. This latter event occurs if all the sensors capable of measuring a particular environmental phenomenon are battery-powered, and currently register a low level of residual energy.

Jess allows an easy implementation of rules that produce new knowledge, starting from the information already acquired. Each rule takes the form of an "*if* <*conditions*> *then*" construct, and it is activated when all its conditions are satisfied, that is, when the knowledge base contains facts that match all the antecedents of the rule. Rules are executed only once for any given set of facts, and are re-evaluated only after the insertion of new facts within the knowledge base.

In order to minimize energy consumption, the system dynamically decides whether and how to change the sampling rate of sensors, depending on the current value of the `SystemCondition` property, which was introduced in Section 2.4 along with the top-level ontology.

In this way, limited resources such as residual energy in battery-powered nodes can be managed efficiently, minimizing energy consumption and increasing the network lifetime. The possible values of the state slot in the `SystemCondition` property are {`normal, stress, attentive`}.

A `stress` condition occurs when the amount of energy currently used is no longer sustainable, i.e., when all the sensors devoted to monitoring a given environmental variable are installed on sensor nodes that are battery-powered, and such devices have a low level of residual energy.

When the system falls into a `stress` condition, the Reasoning module activates a set of special rules to minimize energy consumption and put the system back into

a condition of `normal` functioning, indicating balanced behavior, and improving the chances of meeting both the short-term and long-term goals of the system.

Finally, the `attentive` condition is triggered when users are occupying the monitored premises. In this state, the system gives priority to user comfort, activating a set of special rules in order to maximize the accuracy of monitoring. Some rules are described below which govern state transitions.

For example, when the sensory infrastructure signals that the user is not present in his office, and the knowledge base contains at least one statement of `short-sensing-life` type instantiated in the same room, the following rule puts the system into the `stress` condition:

defrule setStressCondition:

if (no user is in room R) **and** ($\exists$ "low" battery node in R) **then**
system-condition $\leftarrow$ "stress"

For each environmental property, the following rule checks that there exists at least one sensor node with a sufficient residual charge. If this is not the case, the rule sends a `short-sensing-life` alert to administrators:

defrule computeSensingLife:

if (all sensors that monitor ambient property AP are installed on "low" battery nodes) **then**
send *short-sensing-life* alert to administrators

More specifically, the system finds all the sensors that monitor a given property within a room. If all of these sensors are installed on battery-powered nodes, and if all these nodes have a `low` level of residual energy, the system sends an alert to administrators, warning them in a proactive way.

The `minimizeRoomSensing` rule makes it possible to infer that, if the system is in a `stress` condition, it is necessary to minimize the monitoring activities for all the measured quantities in the room.

defrule minimizeRoomSensing:

if (*system-condition* is "stress") **then**
for each sensor S in room R **do**
set minimum sampling rate for S

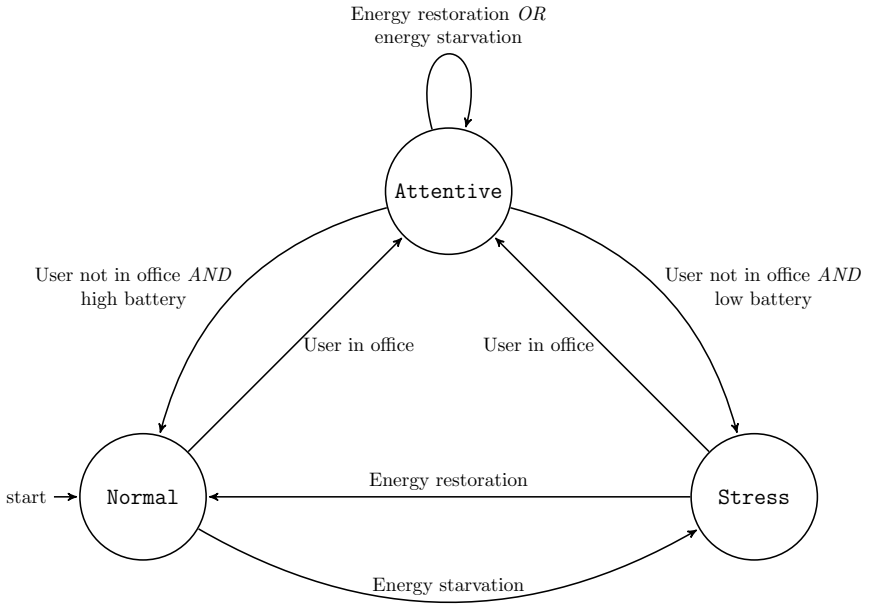


Figure 2.7: Finite state machine showing the transitions between system conditions.

The previous rule sets the sampling rate of all the sensors installed in the room at the minimum value, thus minimizing energy consumption. This is an example of a rule that is only triggered when the system is in a particular condition, thus improving adaptability and context awareness.

The `attentive` condition is activated when the user enters his office and the following rule succeeds. As shown by the finite state machine in Fig. 2.7, this condition has the highest priority. That is, the system activates the `attentive` condition when the user enters his office, and remains in this state until the user leaves, regardless of other factors, thus prioritizing the accuracy of monitoring.

defrule setAttentiveCondition:

if (user U is in room R) **then**

system-condition ← "attentive"

When in such a condition, the system triggers a set of rules that increase the sampling rate of sensors in order to better monitor the environment and satisfy users' needs. For instance, the `increaseRoomSensing` rule detailed below increases the monitoring rate of any ambient property characterized by a `low` level of accu-

racy. The rule is triggered only if the system is in the `attentive` condition, and hence, indirectly, if a user is in his office.

defrule increaseRoomSensing:

if (*system-condition* is "attentive") **and**
(∃ "low" accuracy ambient property AP in room R) **then**
increase sampling rate for AP in R

The rule works to increase the sampling rate of the sensors, so as to maximize the accuracy of monitoring. If the sampling rates of all available sensors are already at their maximum value, the rule sends an `insufficient-sensing` alert to administrators, informing them that the sensory infrastructure of the system can not adequately achieve the goals set by the reasoner.

The `setNormalCondition` rule puts the system into the `normal` condition, which is triggered when the user is not present in his office and the knowledge base does not contain `short-sensing-life` statements related to the current room:

defrule setNormalCondition:

if (no user is in room R) **and** (∄ "low" battery node in R) **then**
system-condition ← "normal"

Finally, in cases where any active node consumes all the residual energy, the `activateBackupNode` rule leverages the redundant sensory infrastructure of the WSAN, as described in Section 2.2, to enable the backup sensor nodes to operate.

In particular, the rule searches all sensor nodes with a `very low` battery level and tries to activate the corresponding backup nodes. If the operation is successful, the system can delegate the gathering and transmission of sensory measurements to the backup nodes, which then become fully active and replace the original nodes in all respects.

defrule activateBackupNode:

if (∃ "very low" battery node N in room R) **then**
activate backup node of N

The purpose of this special rule is to restore the system from a `stress` condition and put it back to a `normal` one, thus implementing the self-healing capabilities required by an autonomic system. If the activation of the backup nodes

is not successful, the system signals a `discharged-battery` alert to administrators, warning them about the energy shortage and urging the replacement of the uncharged batteries.

Considering that the activation of the backup nodes increases the survivability of the system and does not involve a decrease in monitoring accuracy, the `activateBackupNode` rule is independent of the active `SystemCondition`, and can therefore be activated even if the user is present in his office and the system is currently `attentive`.

It is important to note that the rules described are static and there is no mechanism for learning them automatically. The adaptivity of the system lies in the ability to dynamically modify the state of its sensory infrastructure. Nevertheless, the rules developed are generic and do not depend on a particular instance of the system. Moreover, they implicitly manage the conflicting goals of maximizing sensing accuracy and minimizing energy consumption.

Chapter 3

Context-Aware Multi-Sensor Data Fusion

Multi-sensor data fusion is extensively used to merge data collected by heterogeneous sensors deployed in smart environments, so as to infer high level concepts exploited by the reasoning module. However, data coming from sensors are often noisy and inaccurate, and thus probabilistic techniques, such as Dynamic Bayesian Networks (DBNs), are adopted to explicitly model the noise and uncertainty of data.

This work proposes to improve the accuracy of probabilistic inference systems by including context information, and proves the suitability of such an approach in the application scenario of user activity recognition in a smart home environment. The selection of the most convenient set of context information to be considered is not a trivial task. To this end, this chapter presents an extensive experimental evaluation which shows that choosing the right combination of context information is fundamental to maximize inference accuracy.

The remainder of this chapter is organized as follows. Section 3.1 briefly discusses related work on multi-sensor data fusion and context-aware systems. Section 3.2 describes the multi-layered architecture of the data fusion system, focusing on the context-aware DBN that performs the inference. Section 3.3 discusses context information that can be exploited to increase the accuracy of the system. Section 3.4 presents the experimental setting and the evaluation results.

3.1 Related Work

Multi-sensor data fusion techniques are widely used to meld data acquired by sensors deployed in the environment, in order to drive the process of knowledge abstraction from raw data and generating high-level concepts. Although plenty of research work has been done on data fusion, many challenging problems still remain unsolved, especially those inherent to the fusion of multi-source information, such as scalability, data inaccuracy and heterogeneity, outliers, or conflicting information. Further problems arise from the modeling of dynamic phenomena varying over time, or when addressing privacy and security concerns. A comprehensive survey on the state of the art of multi-sensor data fusion solutions can be found in (Khaleghi et al., 2013).

Existing literature addresses the problem of data imperfection by proposing different approaches, based on various theoretical foundations. Probabilistic techniques (Koller and Friedman, 2009), such as Naive Bayes classifiers, Hidden Markov Models, and Conditional Random Fields (CRFs), representing data uncertainty by using probability distributions, have been described in (Cook, 2010), which analyzes and compares the performance of these approaches in a smart home scenario. Alternative approaches deal with other aspects of data inaccuracy, such as vagueness, for example those based on the fuzzy set theory (Zadeh, 1965). The authors in (Stover et al., 1996) proposed a general-purpose fuzzy logic architecture that combines multi-sensor data for automatic object recognition, control of system resources, and automated situation assessment. In (Zheng and Zheng, 2010), a multi-sensor image fusion for surveillance systems is proposed, which exploits fuzzy logic in order to enhance the fused image.

The problem of detecting sensory measurements that mismatch with the expected pattern of observed data has been thoroughly studied in the literature (Zhang et al., 2010), with the aim of eliminating outliers from the fusion process. An adaptive, distributed Bayesian approach for detecting outliers in data collected by a wireless sensor network is proposed in (De Paola et al., 2015) to guarantee reliability and fault tolerance, as well as to reduce energy consumption for unnecessary transmissions.

Heterogeneity of information sources is another significant challenge for data fusion systems. The raw data to analyze might be generated by a large number of heterogeneous sensors and extensive research effort has been devoted to coherently and accurately combine the resulting data (Hall and McMullen, 2004; De Paola et al., 2013). In recent years, a lot of attention has been paid to include context information in the data fusion process in order to reduce ambiguities and errors (Huebscher and McCann, 2004). HiCon (Cho et al., 2008) is a hierarchical context aggregation framework that deals with a broad spectrum of contexts, from personal (e.g., the activities of individuals) to city-wide (e.g., locations of groups of people and vehicles) and world-wide (e.g., global weather and financial data). The authors in (Padovitz et al., 2005) defined a formal model capable of representing context and situations of interest, and developed a technique that exploits multi-attribute utility theory for fusing the modeled information and thereby attain situation awareness. An extensive framework to mediate ambiguous contexts in pervasive environments is presented in (Roy et al., 2011, 2012).

When dealing with phenomena that evolve over time, *adaptiveness* is a fundamental feature. In such cases, where the external environment may constantly change, the system needs to dynamically adapt to the situation, and modify its behavior accordingly. Dynamic Bayesian Networks (DBNs) (Murphy, 2002), in addition to current sensory readings, consider the past belief of the system, thus capturing the dynamicity of the phenomena under observation. Several works adopt DBNs to perform adaptive data fusion for different applications, such as target tracking and identification (Zhang and Ji, 2006), user presence detection (De Paola et al., 2012b, 2011), user activity recognition (Tapia et al., 2004; van Kasteren and Krose, 2007), and healthcare (Roy et al., 2007). Other works integrate machine-learning algorithms in the data fusion process, to develop adaptive systems. Interesting examples of this trend are reported in (Hossain et al., 2009), which exploits reinforcement learning, and in (Fabeck and Mathar, 2008), which proposes kernel-based learning methods to improve the effectiveness of data fusion.

In the field of pervasive systems, and particularly in Ambient Intelligence, the adoption of an adaptive information fusion scheme, capable of exploiting context information, is fundamental to develop a truly smart environment (Cook and Das, 2004) that is able to dynamically adapt to the external events. Context infor-

mation, such as time, location, and user presence or activity, allows for refining the inference process, thus significantly improving the accuracy of reasoning (Dey et al., 2001). The authors in (Roy et al., 2012) propose a resource-optimized framework for sensor networks based on DBNs and information-theoretic reasoning to minimize ambiguity in the context estimation process and context quality determination.

When the pervasive sensory infrastructure is enriched with mobile devices, as proposed in (Yurur et al., 2014), the set of sensors capable of perceiving context information becomes larger. Nevertheless, those sensors are often energy hungry, thus many solutions in literature focus on the reduction of their consumption. In (Rahmati and Zhong, 2011) a static scheme of exclusion of most costly sensors is proposed. SeeMon (Kang et al., 2010) is an energy-efficient context monitoring framework for mobile devices, which adopts event-based monitoring policies to save energy by reducing unnecessary wireless communications. ACE (Acquisitional Context Engine) (Nath, 2012) is a middleware for context-aware applications that dynamically learns associative rules among context attributes, so as to infer the value of expensive attributes by sensing cheaper ones. On a similar note, CAR-LOG (Jiang et al., 2014) adopts a rule-based approach to minimize bandwidth usage, energy and latency, and supports multiple concurrent queries of context attributes, further minimizing bandwidth usage.

The authors in (Rahmati et al., 2015) propose a solution that reduces energy consumption by selecting the set of sensors that achieves the minimum value of accuracy of estimation. The authors of (Yurur et al., 2014) suggest instead to reduce the sensors sampling rate and propose a data fusion approach capable of dealing with such inhomogeneity of data sources.

3.2 Architecture of the data fusion system

This work proposes a novel approach to multi-sensor data fusion for intelligent systems based on the use of pervasive sensors. The system presented here is highly scalable and its inference tier can be used to implement the *Understanding* module of the autonomic AmI architecture described in Chapter 2, so as to infer high-level concepts from raw sensory data.

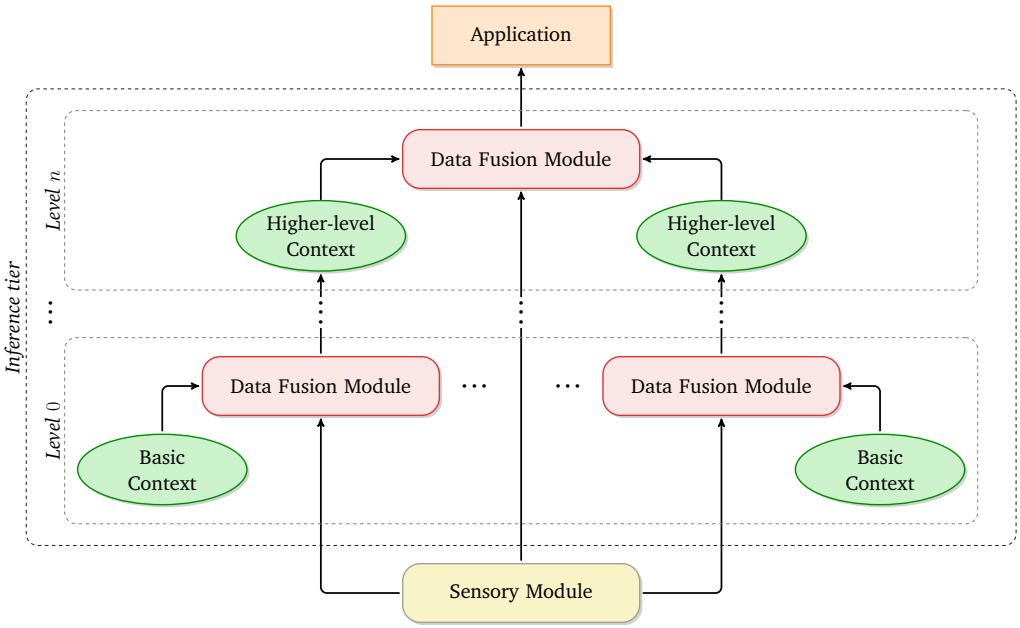


Figure 3.1: Multi-layered architecture of the context-aware data fusion system.

One of the main features of the system is its capability of dealing with inaccurate and noisy data coming from sensory devices. In particular, the use of probabilistic techniques allows the system to merge information coming from multiple sensors by explicitly modeling the noise and uncertainty of data (Khaleghi et al., 2013).

Fig. 3.1 shows the multi-layered architecture of the system. At the lowest tier, the *Sensory* module perceives the world through the pervasive sensory infrastructure. The inference tier is composed of multiple levels: at each level, one or more *Data Fusion* modules exploit context attributes coming from lower levels to perform probabilistic inference on the pre-processed sensory data, fusing them to infer new context information which provides a higher level description of the environment. The process of knowledge abstraction continues until the context information requested by the top-level application is inferred.

This work focuses on a single *Data Fusion* module, and on the impact that context information has on its inference accuracy. A more accurate description of the *Data Fusion* module is presented in the following section. Chapter 4 describes a self-optimizing module that enriches the system presented here by selecting a

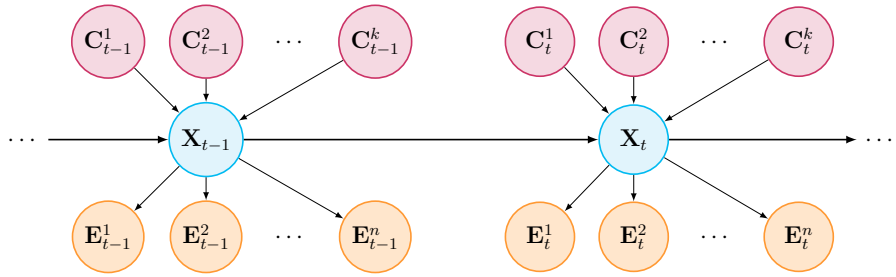


Figure 3.2: Structure of the DBN used for the context-aware data fusion.

subset of sensors to use in the data fusion process, so as to find the best trade-off between inference accuracy and energy consumption of sensory devices.

3.2.1 Data Fusion Module

The proposed data fusion system is based on a Dynamic Bayesian Network (DBN), which models the observed phenomena by taking into account the past state of the world besides current sensory readings. DBNs are a specialization of Bayesian Networks that guarantee a great flexibility in model expressiveness (De Paola and Gagliano, 2014). They pose no restrictions on conditional probability distributions, differently from Kalman filters (Meinhold and Singpurwalla, 1983), and allow for more general topologies than Hidden Markov Models (Sanchez et al., 2007). A DBN is partitioned in temporal slices, where each slice represents the state of the world in a given moment, besides the evidences representing the observable manifestation of the hidden state of the world. Each slice of a DBN can have any number of state variables and evidence variables. Since DBNs represent a good trade-off between expressiveness and tractability (Murphy, 2002), they provide a useful tool for performing data fusion.

Fig. 3.2 shows the structure of the designed DBN. Its main goal is to infer the state of the world, in the form of a given feature of interest, which is represented by the hidden variable X_t. The belief is updated on the basis of a set of sensory readings, represented by the evidence nodes $\boldsymbol{E}_t = (E_t^1, \ldots, E_t^n)$ at any time slice t. Differently from prior work, the system also exploits a set of context information, represented by the evidence nodes $\boldsymbol{C}_t = (C_t^1, \ldots, C_t^k)$ in the time slice t. The set of context information heavily depends on the application scenario; however, it is

crucial to limit the number of context variables in order to control the size of the conditional probability tables (CPTs) and, consequently, the number of parameters to learn in the training phase. Section 3.3 analyzes in detail the choice of which context information to use.

To fully characterize the DBN, it is necessary to define the *sensor model* and the *state transition model* (Murphy, 2002). The probability distribution $P(\boldsymbol{E}_t|X_t)$ expresses how sensory readings are affected by the state variable, and is named *sensor model*. The *state transition model*, defined as $P(X_t|X_{t-1}, \boldsymbol{C}_t)$, represents the probability that the state variable takes a certain value, given its previous value and the current context information.

The *belief* of the system about a specific value of the state variable at time t is defined as:

$$Bel(x_t) = P(x_t|\boldsymbol{E}_{1:t}, \boldsymbol{C}_{1:t}). \tag{3.1}$$

By following a procedure analogous to that adopted in (Thrun et al., 2005) for deriving the equation of Bayes filters, it is possible to express Eq. (3.1) as follows:

$$\begin{aligned} Bel(x_t) = P(x_t|\boldsymbol{E}_{1:t}, \boldsymbol{C}_{1:t}) = P(x_t|\boldsymbol{E}_{1:t-1}, \boldsymbol{E}_t, \boldsymbol{C}_{1:t}) = \\ = \eta \cdot P(\boldsymbol{E}_t|x_t, \boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t}) \cdot P(x_t|\boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t}), \end{aligned} \tag{3.2}$$

where η is a normalizing constant. By using the Markov assumption, and considering that the sensor nodes in $\boldsymbol{E}_t$ do not depend on the context variables in $\boldsymbol{C}_t$, given the state variable X_t, the following holds:

$$P(\boldsymbol{E}_t|x_t, \boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t}) = P(\boldsymbol{E}_t|x_t, \boldsymbol{C}_{1:t}) = P(\boldsymbol{E}_t|x_t). \tag{3.3}$$

Assuming that sensor measurements are mutually conditionally independent, given the value of the parent node X_t, Eq. (3.3) can be further decomposed into:

$$P(\boldsymbol{E}_t|x_t) = \prod_{e_t^i} P(e_t^i|x_t), \tag{3.4}$$

where e_t^i is the specific value of the sensory reading gathered by sensor i in time slice t.

Moreover, the last term in Eq. (3.2) can be expressed as follows:

$$\begin{aligned} P(x_t|\boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t}) &= \sum_{x_{t-1}} P(x_t, x_{t-1}|\boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t}) = \\ &= \alpha \cdot \sum_{x_{t-1}} P(x_t|x_{t-1}, \boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t}) \cdot P(x_{t-1}|\boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t}), \end{aligned} \tag{3.5}$$

where α is another normalizing constant. $\boldsymbol{C}_t$ can be safely omitted from the last term, since X_{t-1} does not depend on the next context $\boldsymbol{C}_t$ if the next state X_t is not considered. Thus, using the Markov assumption, Eq. (3.5) can be expressed as:

$$\begin{aligned} &P(x_t|\boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t}) = \\ &\quad = \alpha \cdot \sum_{x_{t-1}} P(x_t|x_{t-1}, \boldsymbol{C}_t) \cdot P(x_{t-1}|\boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t-1}) = \\ &\quad = \alpha \cdot \sum_{x_{t-1}} P(x_t|x_{t-1}, \boldsymbol{C}_t) \cdot Bel(x_{t-1}). \end{aligned} \tag{3.6}$$

Finally, by substituting Eq. (3.4) and Eq. (3.6) into Eq. (3.2), the belief can be defined with the following recursive formula:

$$Bel(x_t) = \eta \cdot \prod_{e_t^i} P(e_t^i|x_t) \cdot \sum_{x_{t-1}} P(x_t|x_{t-1}, \boldsymbol{C}_t) \cdot Bel(x_{t-1}), \tag{3.7}$$

where α is integrated in the normalization constant η. Using Eq. (3.7), the inference can be performed by storing the last two slices of the DBN, and thus the time and space required for updating the belief do not increase over time.

The computational complexity of calculating the belief for a single x_t is $O(n + m)$, where n is the number of sensor nodes and m is the number of possible values of X_t. Thus, the overall complexity of computing $Bel(x_t)$ for all values of X_t is $O(m^2 + m \cdot n)$.

In order to populate the conditional probability tables of the DBN, several different methods can be adopted, depending on the available training set of historical data, Ξ. In a fully labeled dataset, it is possible to compute sample statistics for each node. For example, let $\boldsymbol{P}_i$ denote the parents of a node V_i. The sample statistic $P(V_i = v_i|\boldsymbol{P}_i = \boldsymbol{p}_i)$ is given by the number of samples in Ξ having $V_i = v_i$ and $\boldsymbol{P}_i = \boldsymbol{p}_i$ divided by the number of samples having $\boldsymbol{P}_i = \boldsymbol{p}_i$. To learn the CPTs, it suffices to calculate these sample statistics for all the nodes in the network. If

Table 3.1: CPT for state transition: $P(\mathbf{X}_t|\mathbf{X}_{t-1}, \mathbf{C}_t^1)$.

		$\mathbf{X}_t$	
$\mathbf{C}_t^1$	$\mathbf{X}_{t-1}$	**0**	**1**
0	**0**	0.75	0.25
0	**1**	0.11	0.89
1	**0**	0.85	0.15
1	**1**	0.07	0.93
2	**0**	0.95	0.05
2	**1**	0.10	0.90

Table 3.2: CPTs for sensor models: $P(\mathbf{E}_t^1|\mathbf{X}_t)$, $P(\mathbf{E}_t^2|\mathbf{X}_t)$ and $P(\mathbf{E}_t^3|\mathbf{X}_t)$.

		$\mathbf{E}_t^1$			$\mathbf{E}_t^2$		$\mathbf{E}_t^3$	
		0	**1**	**2**	**0**	**1**	**0**	**1**
$\mathbf{X}_t$	**0**	0.50	0.20	0.30	0.87	0.13	0.68	0.32
	1	0.15	0.05	0.80	0.36	0.64	0.10	0.90

the values of one or more of the variables are missing for some of the training records, different techniques can be used, such as the Expectation Maximization (EM) algorithm or gradient ascent (Koller and Friedman, 2009).

Running Example

The following running example illustrates how the DBN of the *Data Fusion* module operates. The example network has a single state variable $\mathbf{X}$, with two possible different values, three evidence nodes $\mathbf{E}^1$, $\mathbf{E}^2$, $\mathbf{E}^3$, taking 3, 2 and 2 values respectively, and a single context node $\mathbf{C}^1$, taking three possible values; the network is defined through the CPTs reported in Tables 3.1 and 3.2.

Let $P(\mathbf{X}_0) = \langle 0.85, 0.15 \rangle$ represent the probability distribution for the state variable at time $t = 0$, i.e., $P(\mathbf{X}_0 = 0) = 0.85$ and $P(\mathbf{X}_0 = 1) = 0.15$. If the sensory readings at time $t = 1$ are $[E_1^1, E_1^2, E_1^3] = [1, 0, 1]$, and the value of the context node C_1^1 is 0, the corresponding belief for the state variable can be

computed according to Eq. (3.7):

$$
\begin{aligned}
&Bel(X_1 = 0) = \\
&\quad = \eta \cdot (0.2 \cdot 0.87 \cdot 0.32) \cdot (0.75 \cdot 0.85 + 0.11 \cdot 0.15) = \\
&\quad = \eta \cdot 0.037, \\
&Bel(X_1 = 1) = \\
&\quad = \eta \cdot (0.05 \cdot 0.36 \cdot 0.9) \cdot (0.25 \cdot 0.85 + 0.89 \cdot 0.15) = \\
&\quad = \eta \cdot 0.006,
\end{aligned}
$$

where the normalization constant can be computed as $\eta = \frac{1}{0.037+0.006} = 23.256$. Thus, the belief at time $t = 1$ is $Bel(X_1) = \langle 0.867, 0.133 \rangle$. At time $t = 2$, for the sake of the example, let the sensors produce the readings: $[E_2^1, E_2^2, E_2^3] = [2, 1, 0]$, and let the value of the context node C_2^1 be 1. Applying again Eq. (3.1), it is possible to calculate $Bel(X_2) = \langle 0.606, 0.394 \rangle$. As mentioned, context information greatly influences inference results. Indeed, if the value of the context node C_2^1 had been 0 at time $t = 2$, the corresponding belief would have been $Bel(X_2) = \langle 0.507, 0.493 \rangle$.

3.3 Context-awareness

The role of context in the system is twofold. The main goal is the inference of context information, intended as a high-level description of the surrounding world. In particular, this chapter focuses on recognizing activities performed by users in a smart home environment, which in turn will enable higher-level *Reasoning* modules to provide the most appropriate services to users, as described in Chapter 2.

Low-level context information, such as time and location, can be exploited by such data fusion system to improve the accuracy of reasoning by refining the inference process, as demonstrated by many context-aware data fusion systems proposed by researchers over the years (Nimier, 1995; Padovitz et al., 2005).

However, using too many context attributes can actually be detrimental to the inference accuracy, as will be demonstrated in Section 3.4.3, and increases the computational burden of the system, especially in the training phase. Thus, it is

important to analyze the possible context information and select only the most informative attributes, which may vary depending on the application scenario.

It is possible to identify some principles that can drive the selection of context attributes. First of all, context information should be readily available in all situations, regardless of the sensors used. Therefore, the proposed system does not include information provided by users manually, as well as context attributes which are difficult to sense or that cannot be sensed directly and reliably, thus introducing new elements of uncertainty in the system.

The authors of (Dey et al., 2001) provide a widely accepted definition of context, which identifies the *primary* categories of contextual information, i.e., identity, activity, location, and time. Identity and activity are high-level attributes, while location and time are low-level attributes. Thus, according to the principles stated above, this chapter focuses on location-related and time-related context information, analyzing the possible benefits they can provide to the system, and validating the proposed approach in the experimental section.

Time-related context information is used by most context-aware systems in literature, since it is very easy to obtain (i.e., it is sufficient to check the current date and time). For activity recognition systems, in particular, time-related context information provides remarkable improvements to the accuracy (Perera et al., 2014). First of all, intuitively, activities performed by users may vary a lot in different periods of day: for example, sleeping is the most probable activity during the night, and many users have lunch and dinner at regular time each day. Thus, exploiting this context attribute should improve the accuracy of the system, with almost no drawbacks. However, the number of periods in which a day is divided can influence the performance of the system, as demonstrated in Section 3.4.3. Both too coarse-grained periods (e.g., intervals of 12 hours) and too fine-grained ones (e.g., intervals of 1 minute) do not convey much information; hence, finding the best granularity is very important.

Similarly, activities performed by users might be influenced by the current day of the week and, to a lesser extent, by the month of the year. However, users' activities should be less correlated to these context attributes, with respect to the period of day. For example, it is possible that users will behave differently during weekends, but it is unlikely that activities will change much among the other days.

Further considerations regarding the day of the week and month of the year are deferred to the experimental section. Other time-related context information, such as the timezone, might be interesting for different scenarios, but are irrelevant for the case study of activity recognition in a smart house.

As regards location-related context information, this work focuses primarily on the position of users, leaving to future work an analysis on how to exploit the position of objects to improve the awareness of the system about users' surroundings. In the case of a smart home, with no strong assumption on the kind of sensors used, this work proposes to exploit user location information with a room-level granularity. Regardless of the sensors used, estimating the position of users with this level of detail is required to correctly inferring their activities.

However, a system that relies primarily on location-related context information will encounter difficulties in recognizing certain activities. Intuitively, this can be explained by considering that some activities are performed in well-defined locations (e.g., sleeping in the bedroom), and therefore are well recognized using this kind on information, while other activities are more irregular (e.g., housekeeping, which may be carried out in all rooms of the smart home), and more heterogeneous context information should be exploited to recognize them with higher accuracy.

3.4 Experimental Analysis

In order to evaluate the possible contribute of different context information to the data fusion process, the performance of the proposed DBN will be tested while varying the type and granularity of context information.

3.4.1 Simulation Setting

This section evaluates the performance of the system in a smart home where several programmable wireless sensor nodes are deployed. The traces, collected by a set of real sensors, have been exploited in order to simulate the interaction of sensory devices with the environment, according to principles defined in (Lalomia et al., 2009). Sensory traces and corresponding user activities were obtained from the Aruba dataset of the CASAS Smart Home Project (Cook, 2010), developed at

Washington State University. This dataset contains annotated data collected in a smart apartment with a single resident, over a period of seven months. Events are generated by 31 motion sensors, 3 door sensors, and 5 temperature sensors, deployed in 8 rooms (5 sensors per room on average).

A pre-processing of the original data was required to test the system. The sequence of sensor events was partitioned into time windows of 30 seconds, counting how many times each sensor was activated during each slice. Considering the low correlation between temperature readings and the activity performed by the user, this information has been discarded.

Moreover, a heuristic was developed to label each interval with the predominant activity performed by the user during that time slice. The Aruba dataset considers eleven activities of daily living (ADLs), i.e., *Bed to Toilet*, *Eating*, *Enter Home*, *Housekeeping*, *Leave Home*, *Meal Preparation*, *Relax*, *Resperate*[1], *Sleeping*, *Wash Dishes*, and *Work*. To better evaluate the system, a new activity, named *Outside*, has been added to the dataset. This activity takes into consideration the periods of time when the user is not at home, i.e., the intervals between *Leave Home* and *Enter Home*. This information can be used by the system to further optimize the energy consumption of the sensory infrastructure, since all the sensors installed in the smart home can be deactivated when no one is present, with the exception of the door sensors, thus minimizing energy consumption without sacrificing inference accuracy.

In a real scenario, it is very difficult, if not impossible, to predict all the activities that will be performed by users and, furthermore, a fixed list of activity classes cannot take into consideration the unavoidable transitions between activities. To address both of these problems, a further activity class was added to the ones annotated within the original dataset, as suggested in (Krishnan and Cook, 2012). This special activity, named *Other*, groups all the sensor events that do not match any of the known activities. It is essential to detect this activity class accurately in a real world scenario, since nearly 20% of the sensor events in the dataset considered here belong to the *Other* class. However, considering the heterogeneity of the activities grouped by this class, it is very challenging to recognize it with good

[1]Resperate is a device used for the treatment of high blood pressure.

accuracy, and many approaches in the literature ignore it altogether, relying on a static list of known activities, as noted in (Krishnan and Cook, 2012).

The cross validation method was used to evaluate the system, dividing the dataset into ten parts. For each test, nine parts were used for learning the CPTs (Conditional Probability Tables) of the DBN, and the tenth was used for the test. This process was then repeated changing the test set ten times and averaging the results.

After the pre-processing phase, the dataset consisted of 633 468 sensor events. Each test of the cross validation used 570 121 sensor events as training cases, and 63 347 sensor events as test cases. All experiments have been performed on a workstation equipped with an Intel® Core™ i5-3470 CPU (4 cores, 3.20 GHz, 4 GB RAM). The training phase required 4 914 ms on average.

3.4.2 Performance Metrics

The most common metric to evaluate activity recognition systems is the average accuracy, defined as:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}, \tag{3.8}$$

where *TP*, *TN*, *FP*, and *FN* are, respectively, the true positives, true negatives, false positives and false negatives. However, accuracy alone is not sufficient to evaluate different approaches, since data are skewed towards the most probable activities. In fact, activities such as *Sleeping* and *Relax* account for a large number of time slices, while others like *Resperate* and *Leave Home* or *Enter Home* are much rarer and shorter. For this reason, the evaluation proposed here adopts additional metrics to provide a more detailed analysis of the performance of the systems.

An index based on the classic definition of Shannon entropy (Shannon, 2001) has been exploited to measure the uncertainty of the probabilistic reasoning performed by the systems. Another metric that will be used in the following is the average cross-entropy error function, which is defined as follows:

$$CE = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} \log p_{ij}, \tag{3.9}$$

where N is the number of timesteps, M is the number of activity classes, and y_{ij} and p_{ij} are, respectively, the ground truth and the predicted probability for the j^{th} activity class at time i. The cross-entropy error is an information-theoretic measure of accuracy that incorporates the idea of probabilistic confidence, measuring the cross-entropy between the distribution of true labels and the prediction of the system. This kind of error becomes extremely large (i.e., $+\infty$ in the extreme case) if the system is over-confident about a wrong prediction, and it is thus useful to evaluate the accuracy of the belief with a fine granularity. Finally, the *precision* (positive predictive value) has been used as fidelity measure, and the *recall* (sensitivity) has been selected for measuring completeness. These metrics are defined as follows:

$$\begin{aligned} precision &= \frac{TP}{TP + FP}, \\ recall &= \frac{TP}{TP + FN}. \end{aligned} \tag{3.10}$$

Precision and recall, in turn, are used to calculate the *F-score*, which is defined as the harmonic mean of precision and recall, as follows:

$$F\text{-}score = 2 \cdot \frac{precision \cdot recall}{precision + recall}. \tag{3.11}$$

3.4.3 Experimental Results

Time-related context information

The first set of experiments presented here is a detailed analysis on the importance of some time-related context attribute, i.e., *period of day*, *day of week*, and *month*.

The analysis starts by studying the performance of the system when changing the granularity of the *period of day* node. Fig. 3.3 shows the accuracy, uncertainty, F-score and cross-entropy error of a system exploiting the *period of day* node, as a function of the number of periods in which a day is divided, starting from a single period (i.e., a single interval of 24 hours) up to a maximum of 48 periods (i.e., 48 intervals of 30 minutes). The figure shows an increment of the accuracy and F-score when increasing the granularity up to 6 periods (i.e., intervals of 4 hours). Likewise, uncertainty and cross-entropy error are very low using this granularity.

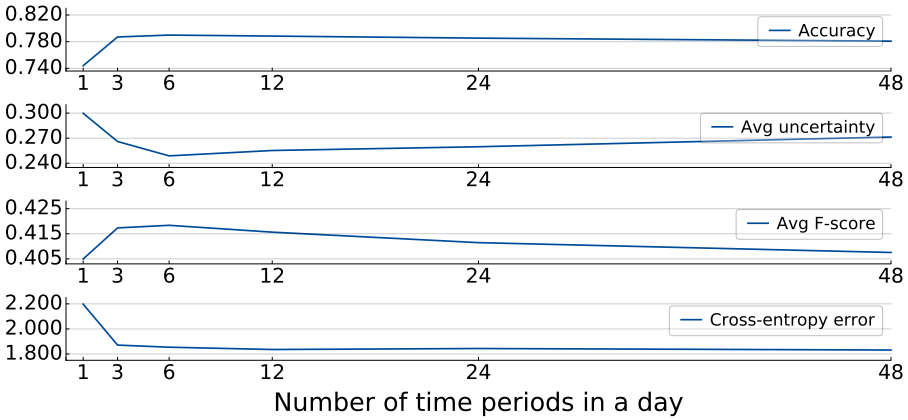


Figure 3.3: Accuracy, uncertainty, F-score, and cross-entropy of the data fusion system when varying the granularity of the *period of day* node.

However, dividing the day in more than 6 periods yields a steady decrease of the F-score, as well as an increase of uncertainty, whilst accuracy and cross-entropy remain unchanged. Thus, increasing time granularity is beneficial only up to a point; going further only adds to the noise, resulting in a system that performs worse with no added benefits.

The experimental results show that it is possible to improve accuracy and F-score of the system even more by dividing the day manually in four periods, namely *morning* (8AM - 12PM), *afternoon* (12PM - 8PM), *evening* (8PM - 11PM) and *night* (11PM - 8AM). This way, periods closely follow the phases of the day when the type of activities performed by typical users changes, as shown in Fig. 3.4. As the figure points out, the user's behavior changes remarkably during the day. For example, the *Housekeeping* and *Wash Dishes* activities are much more probable during morning or afternoon, and it seems the user works prevalently on afternoons. As expected, activities such as *Sleeping* and *Bed to Toilet* take place mainly at night. However, some activities, such as *Other*, show less variance during the day, and are thus more difficult to identify. Results show that this granularity yields the best accuracy, F-score and uncertainty (0.793, 0.416, and 0.231, respectively), and one of the lowest cross-entropy errors, i.e., 1.858.

The frequency of the user's activities, during the week (Fig. 3.5) and among different months (Fig. 3.6), was analyzed to evaluate the effect of context information concerning the day of week and the month. It is worth noting that the

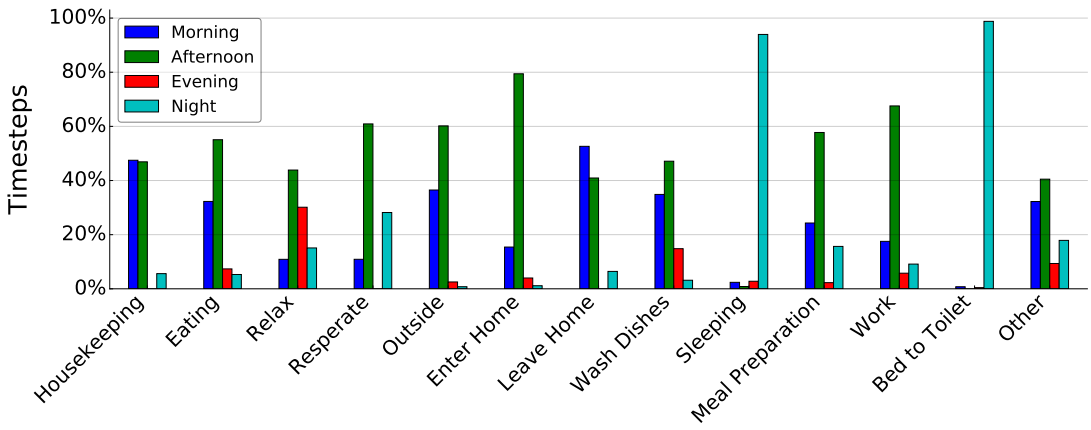


Figure 3.4: Frequency of the activities during morning, afternoon, evening, and night.

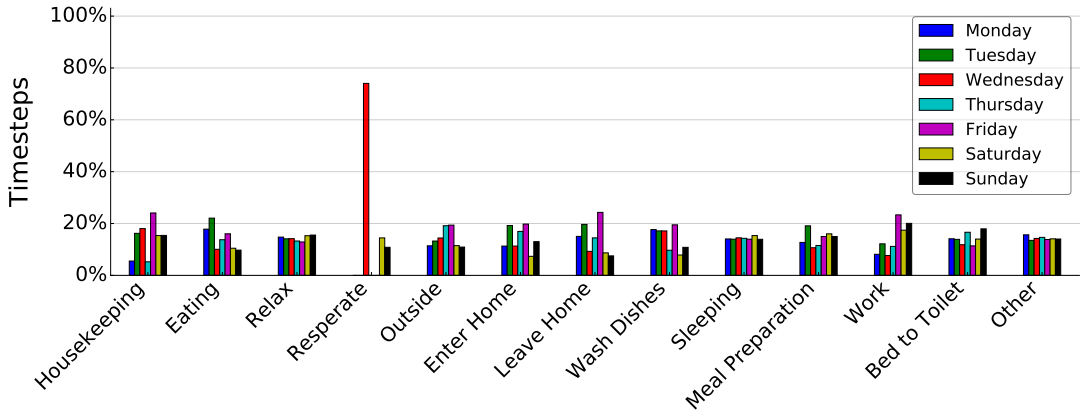


Figure 3.5: Frequency of the activities during different days of the week.

user's behavior is pretty regular during the week, including weekends. The only exception appears to be the *Resperate* activity; however, this is a fairly rare activity, thus its weight when determining the accuracy of the system is limited. Even among different months, the activities are quite regular (only *Housekeeping* and *Resperate* show a remarkable variance). By analyzing such results, it is easy to predict the different impact of the *period of day (PoD)* node with respect to the *day of week (DoW)* and *month* ones.

In order to verify such analysis, Table 3.3 reports the comparison of eight systems that exploit different combinations of context information. The difference in accuracy between the best and worst combination of context nodes is more than

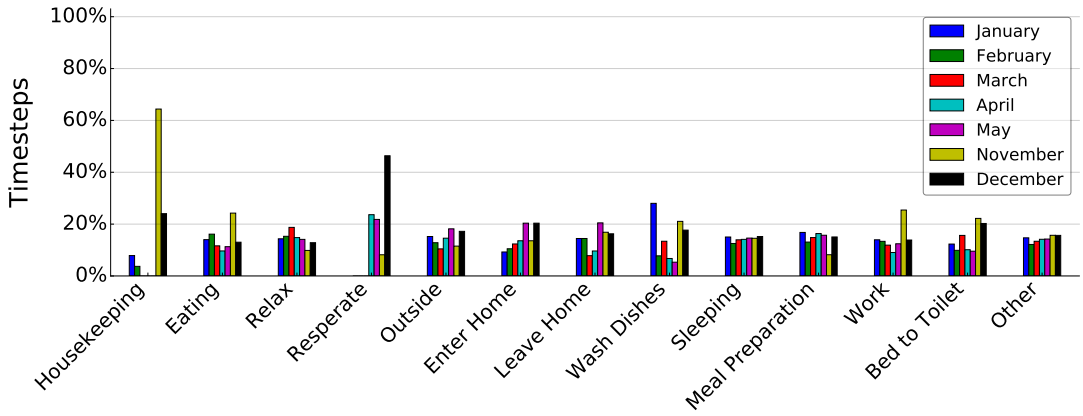


Figure 3.6: Frequency of the activities during different months of the year.

Table 3.3: Average accuracy (Acc), uncertainty (Unc), cross-entropy error (CE), and F-score of the analyzed systems, sorted by accuracy in descending order.

PoD	DoW	Month	Acc	Acc w/o *Other*	Unc	CE	F-score
✓	–	–	0.793	0.889	0.231	1.858	0.416
✓	✓	–	0.779	0.874	0.245	1.950	0.400
✓	–	✓	0.778	0.876	0.280	1.815	0.385
–	–	–	0.760	0.853	0.282	2.146	0.403
✓	✓	✓	0.739	0.833	0.373	1.911	0.366
–	✓	–	0.734	0.826	0.347	2.232	0.390
–	–	✓	0.714	0.800	0.429	2.222	0.363
–	✓	✓	0.690	0.772	0.562	2.347	0.349

10%. It is worth noting that four systems out of the five with highest accuracy exploit the *period of day* node. Moreover, these systems show high F-scores and low uncertainty and cross-entropy errors. As expected, the accuracy of all systems improves significantly if the *Other* activity class is ignored, increasing by about 10% on the average. Surprisingly, the system which includes all three context nodes performs worse than the one which excludes them. This can be explained by the interference of the *month* and *day of week* nodes. In fact, the system that exploits only these two context nodes is the worst according to all metrics. Conversely, the system that performs better is the one which uses only the *period of day* node. Activities are too regular during the week and among months, and therefore the usefulness of the *day of week* and *month* nodes is limited. Thus, at a first glance,

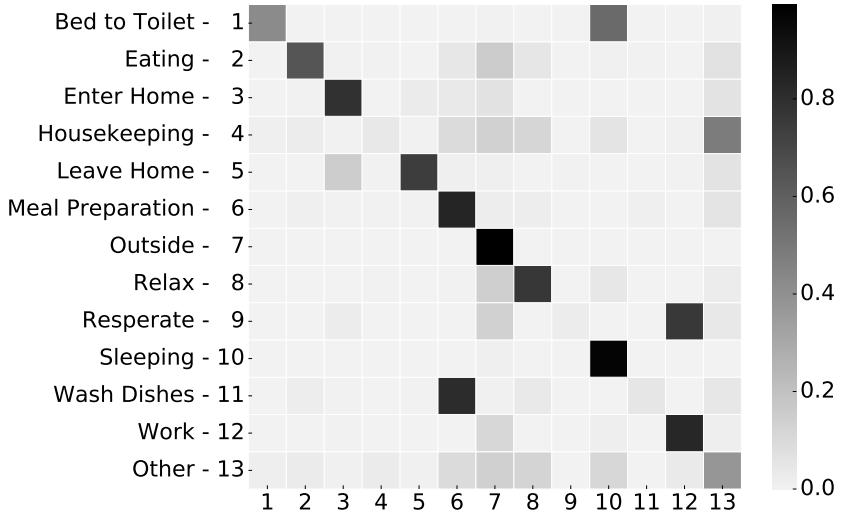


Figure 3.7: Confusion matrix of the baseline data fusion system.

it appears that the *day of week* and *month* nodes are not needed to improve the performance of the data fusion system, and can, in fact, be detrimental.

The system which uses only the *period of day* node will be considered as baseline for comparison with other systems in the following experiments. To provide a more detailed analysis of its performances, its confusion matrix, row-wise normalized, is presented in Fig. 3.7. Each cell C_{ij} represents the number of instances of class i predicted to be in class j by the system. Therefore, diagonal entries correspond to true positives, and non diagonal entries correspond to classification errors. Darker blocks indicate higher values, as suggested by the color bar. The large number of dark diagonal entries and the light entries in the rest of the matrix imply a high true positive rate and low confusion among activities. However, the matrix highlights that some activities are difficult to handle, i.e., *Housekeeping* (often misclassified as *Other*), *Resperate* (misclassified as *Work*), and *Wash Dishes* (misclassified as *Meal Preparation*).

In the following, the location in which each activity is carried out will be analyzed in detail, so as to explain why some activities are more difficult to recognize than others.

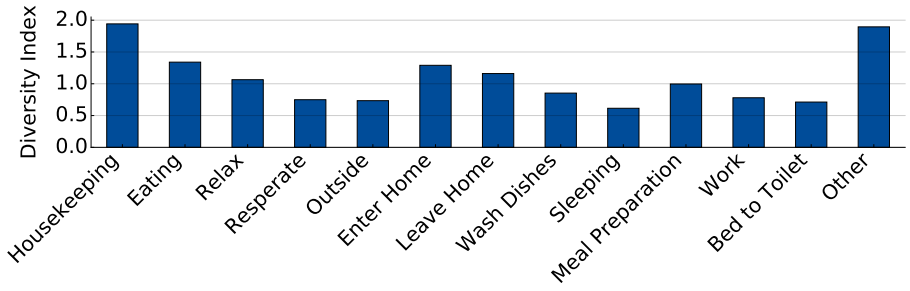


Figure 3.8: *Diversity index* of each activity.

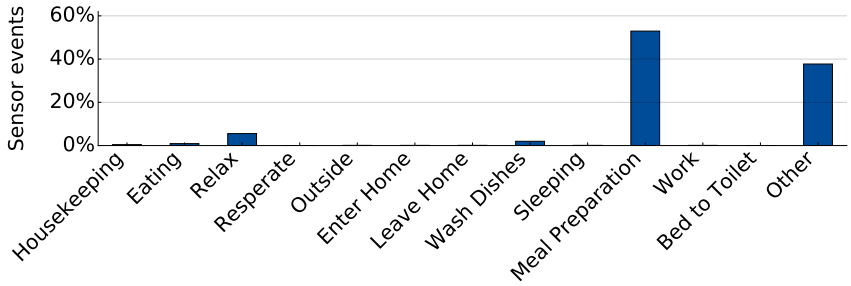


Figure 3.9: Frequency of the activities performed in the kitchen.

Location-related context information

Intuitively, some activities are performed in well-defined locations, and therefore are well recognized using only motion sensors, while other activities are more irregular. Furthermore, it is reasonable to assume that some activities are performed mainly in the same rooms (and roughly in the same time periods), such as *Wash Dishes* and *Meal Preparation*. To verify these hypotheses, the smart house was divided in rooms, so as to measure the variability of the association between activities and rooms, exploiting the *diversity index*, defined as the classical Shannon entropy (Shannon, 2001).

Fig. 3.8 summarizes the *diversity index* of the activities, which indicates how they are carried out in different rooms. Activities performed in a well-defined location have a low diversity index, while activities carried out in different rooms exhibit a high diversity index. As expected, activities which are difficult to recognize correctly, such as *Housekeeping* and *Other*, exhibit the highest diversity indices. On the other hand, activities that are easier to classify accurately, such as *Sleeping*, have low *diversity indices*. The *Wash Dishes* activity seems to contra-

dict this statement, since it sports a low *diversity index*, but is often misclassified. However, this activity only takes place in the kitchen, since almost 80% of sensor events associated with it comes from sensors deployed there. This accounts for its low *diversity index*, but, as Fig. 3.9 shows, there are much more probable activities taking place in the same room, such as *Meal Preparation* (52.9% of sensor events) and *Other* (37.7% of sensor events); even *Relax* is a more probable activity than *Wash Dishes*, in the kitchen. Therefore, it is understandable that a system which relies mostly on motion sensors will have a hard time identifying this kind of activity. To overcome this problem, it is possible to exploit the information associated to the duration of each activity.

Duration of activities

The experiments reveal that some activities exhibit a much longer average duration than others. For example, *Sleeping* has an average duration of about 4 hours, while *Eating* generally takes about 10 minutes. Thus, it is intuitive that making use of this kind of context information should be beneficial to the system.

In order to verify the usefulness of such information, a system with an additional context node exploiting the duration of activities was tested and compared to the baseline system. Surprisingly, the resulting accuracy was lower than the baseline system by about 2%, and the other metrics were unchanged or slightly worse. A closer look at the data reveals that only a couple of activities (i.e., *Sleeping* and *Outside*) have average durations longer than one hour. Most of the other activities have durations similar to each other, generally between 10 and 30 minutes. It seems that this type of context information fails to help the system if it is possible to exploit enough data coming from the sensory devices.

However, when performing data fusion, it might not always be efficient to sample all available sensors. On the contrary, it may be useful to activate only a subset of sensors, depending on the application scenario. For instance, if the sensory infrastructure is composed of devices with limited energy resources, the use of a subset of devices might increase the lifetime of the whole network.

For this reason, the comparison experiments were repeated using only a subset of sensors, discarding the rest of the data. As expected, in these conditions

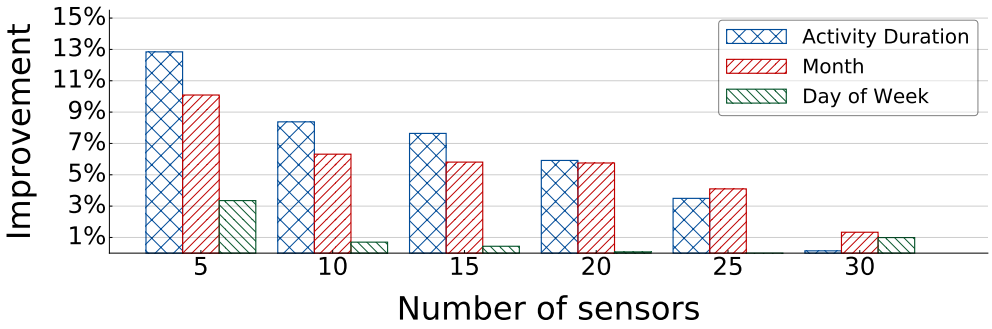


Figure 3.10: Improvement of inference accuracy when exploiting context information with different number of sensors.

context information proved to be much more valuable. Using only 10 sensors (out of 34), the accuracy of the baseline system is 65.87%, while exploiting duration information results in an accuracy of 74.25%, with a significant improvement of 8.38%. As it turns out, the same is true for other context information as well.

Fig. 3.10 shows the improvement in accuracy of systems exploiting *activity duration*, *month* and *day of week*, with respect to the baseline system (i.e., the one exploiting only the *period of day*) as a function of the number of sensors used. It can be noted that, in the extreme case of using only 5 sensors, exploiting the *activity duration* improves the accuracy of the system by almost 13%. Conversely, the benefits of using context information decrease when there is enough data coming from the sensory devices. The same holds true for the *month* node, whilst the improvement when using the *day of week* is negligible even with few sensors.

Chapter 4 analyzes how to further use context information to dynamically reconfigure the sensory infrastructure, by exploiting a self-optimization module which samples a subset of sensors in order to minimize energy consumption, whilst maintaining a high degree of inference accuracy.

Chapter 4

Context-Aware Self-Optimization

This chapter proposes a context-aware self-optimizing module for multi-sensor data fusion in pervasive computing scenarios, enriching the Data Fusion system described in Chapter 3. Differently from previous works on the subject, the proposed self-optimization module focuses on dynamic management of sensors, finding the best trade-off between inference accuracy and energy consumption of sensory devices. Furthermore, this work exploits contextual information in a novel way, both to increase the accuracy of reasoning and to improve the adaptiveness of the system, thereby reducing energy consumption.

The remainder of this chapter is organized as follows. Section 4.1 outlines the proposed self-optimization system and provides a high level description of its modules. Section 4.2 highlights the dynamic reconfiguration capabilities, explaining the behavior of the self-optimization modules. Section 4.3 presents the case study used to demonstrate the effectiveness of the proposed approach with the help of activity recognition in an AmI scenario. Section 4.4 describes the experimental setting and analyzes the experimental results.

4.1 The Three-tier Architecture

One of the main features of the approach proposed in Chapter 3 is its capability of dealing with the inherent inaccuracy of sensors. Probabilistic techniques, such as Dynamic Bayesian Networks, enable the fusion of information coming from

multiple sensors by explicitly modeling the noise and uncertainty of data so as to improve confidence and reliability of the reasoning process (Khaleghi et al., 2013).

The chapter adds another relevant feature to the system, i.e., the capability of adaptively selecting the best subset of sensors to be used in the data fusion process. Such selection is performed to activate only the sensors that are strictly necessary on the basis of the current needs of the system, rather than exploiting all the available data sources.

This approach meets several requirements of intelligent pervasive systems. It decreases the computational burden of the information fusion process, and consequently the system response time (Zhang and Ji, 2006). Moreover, it reduces the energy consumption of the sensory infrastructure; this aspect plays a relevant role in pervasive systems, such as Ambient Intelligence, considered here as a case study.

Although such an approach minimizes the number of sensors used during the inference process, it still guarantees a high degree of accuracy. The system is designed to autonomously modify the state of the sensory infrastructure by switching sensors to a low-power mode which disables data gathering and transmission functionalities in order to minimize their energy consumption, whenever this does not compromise the inference accuracy.

The proposed architecture, depicted in Fig. 4.1, consists of three different tiers characterized by increasing abstraction levels. The *lowest* tier (sensory) is in charge of the observation of external phenomena occurring in the environment through the pervasive sensory infrastructure that manages only raw data. The *intermediate* tier (data fusion) copes with the uncertainty of gathered data and tries to infer the external world conditions, which contribute to define the current context, by performing a multi-sensor data fusion with the integration of further available context information, as described in Chapter 3. Finally, the *highest* tier (self-optimization) aims at finding an acceptable trade-off between costs (e.g., energy consumption) and system performance (e.g., accuracy of reasoning). Fig. 4.1 illustrates the main modules within each tier of the proposed architecture. The sensory and data fusion tiers present a simple structure exploiting respectively the following two functionalities:

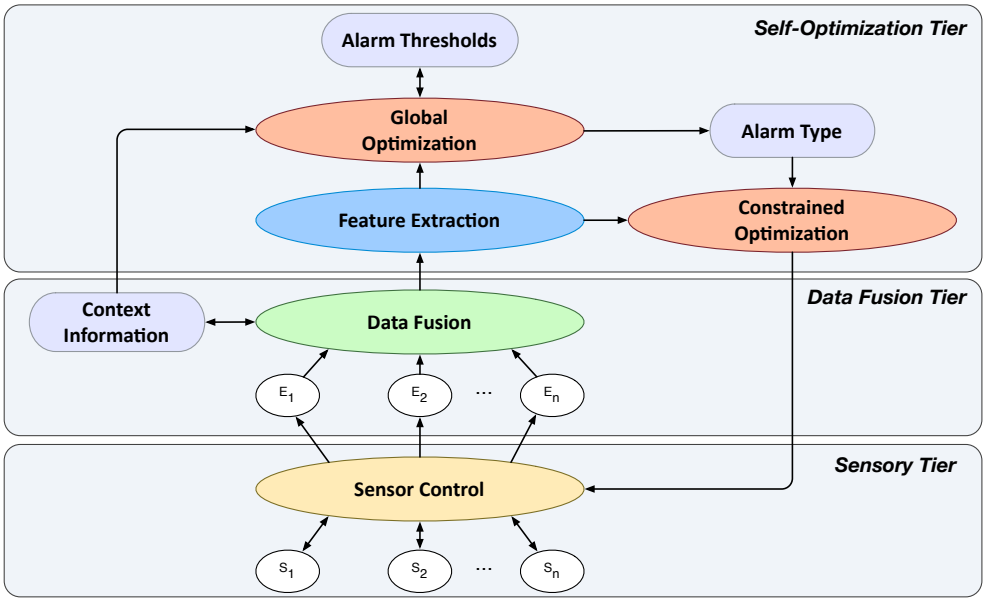


Figure 4.1: Block Diagram of the three-tier architecture.

- *Sensor Control*: collects raw data coming from the sensors, sends them to the *Data Fusion* module and modifies the state of sensory devices, by activating or deactivating them according to the policy indicated by the modules of the self-optimization tier;

- *Data Fusion*: exploits the available context information, performs probabilistic inference on the raw data coming from sensors, and combines them into describing the world through higher-level concepts.

The self-optimization tier exhibits the highest complexity, since it considers the state of the system and the current context conditions, and selects the actions to be performed in order to optimize the system behavior. The main functionalities of this tier are performed by the following three modules:

- *Feature Extraction*: selects the relevant context information about the external world and the internal state of the system in order to perform self-optimization. Such features are inferred both from the outcome of the *Data Fusion* module, and from the meta-analysis of the system internal behavior. The set of features is specific and depends on the particular application

scenario. In order to adapt the system to a specific scenario, it is necessary to identify the meaningful information needed for dynamically reconfiguring the sensory infrastructure.

- *Global Optimization*: exploits context information, in the form of features selected by the *Feature Extraction* modules, to dynamically modify the cost and uncertainty thresholds triggering the self-configuration performed by the *Constrained Optimization* module.

- *Constrained Optimization*: this module is activated by a set of alarms triggered by the *Global Optimization* module, when cost and uncertainty of the information fusion process exceed their respective thresholds. The cost and uncertainty quantities are two of the features selected by the *Feature Extraction* module for the specific scenario considered here.

A more accurate description of the role played by the modules of the self-optimization tier is given in the following sections.

4.2 Context-aware Self-Optimization

The self-optimization modules aim to dynamically optimize the state of the sensory infrastructure, in order to find the best trade-off between performance and costs. To this end they need to exploit the maximum of available information on the external world and on the internal system conditions. Useful information can be obtained by the analysis of the behavior of the *Data Fusion* module, in addition to the other information acquired from the external world.

The *Feature Extraction* module produces a feature vector containing meta-information about the world state and the system state, which, as a whole, defines the current context. Such information are then exploited by the *Global Optimization* and *Constrained Optimization* modules to update the alarm thresholds and reconfigure the sensory infrastructure respectively, as will be described in the following.

4.2.1 Feature Extraction

The *Feature Extraction* module produces, for each time slice t, a feature vector composed of the following elements:

$$FV_t = [\boldsymbol{conf}_t, \boldsymbol{WSB}, T_E, RUE, SC, \boldsymbol{SIE}], \tag{4.1}$$

where $\boldsymbol{conf}_t$ is the configuration of the sensory infrastructure, ***WSB*** (World State Belief) is the belief distribution about the surrounding environment, as inferred by the *Data Fusion* module, T_E is the elapsed time since the last estimated transition of the world state, *RUE* (Reasoning Uncertainty Estimation) is an estimation of the reasoning uncertainty, SC is the cost associated to the active sensors, and ***SIE*** (Sensor Importance Estimation) is an estimation of the importance of each sensor for the inference.

The sensory configuration, which fully describes the state of each sensor, is defined as a binary vector, $\boldsymbol{conf}_t = [s_t^1, s_t^2, \ldots, s_t^n]$, where $s_t^i \in \{0, 1\}$. Each element s_t^i specifies whether the corresponding sensor E^i is *off* or *on*, respectively, in the time slice t.

The *belief* about the state of the world, ***WSB***, is obtained directly from the *Data Fusion* module at each time slice, as calculated by Eq. (3.7). The *Global Optimization* module exploits such information along with T_E, the elapsed time since the last transition of the world state as perceived by the system. Let x_t^* represent the most probable world state, defined as follows:

$$x_t^* = \arg\max_{x_t} \Big(Bel(x_t) \Big). \tag{4.2}$$

The *Feature Extraction* module evaluates the number of time slices elapsed since the last transition of the world state with the following equation:

$$T_E(t) = \begin{cases} T_E(t-1) + 1 & \text{if } x_{t-1}^* = x_t^*, \\ 0 & \text{otherwise.} \end{cases} \tag{4.3}$$

The uncertainty and cost indices associated with a specific sensory configuration are used by the *Global Optimization* module to decide whether to trigger an

alarm, thus requiring a reconfiguration of the sensory infrastructure. To measure the uncertainty of the probabilistic reasoning performed by the *Data Fusion* module, this work adopts an index based on the definition of Shannon entropy (Shannon, 2001):

$$RUE \stackrel{def}{=} -\sum_{x_t} Bel(x_t) \log_2(Bel(x_t)). \tag{4.4}$$

Assuming that the cost of acquiring information from node E^i in the time slice t is expressed by the function $f_{cost}(E_t^i)$, the total cost associated to a specific sensory configuration can be evaluated as:

$$SC \stackrel{def}{=} \sum_{E_t^i \in \boldsymbol{E}_t} f_{cost}(E_t^i) \cdot s_t^i. \tag{4.5}$$

The specific cost function closely depends on the application scenarios. The description of the one adopted for the case study presented here is deferred to Section 4.3.

Finally, to support the system in selecting the most useful sensors to activate at any time slice, this work proposes a heuristic that estimates the importance of each sensor in the current situation, based on the Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951), known as relative entropy or *information gain*. The KL divergence of an approximate distribution G, with regard to a *true* distribution F, measures the information that is lost when G is used instead of F. It is defined as:

$$D_{KL}(F \| G) = \sum_i F(x_t) \log\left(\frac{F(x_t)}{G(x_t)}\right). \tag{4.6}$$

The information gain of an active sensory configuration can be computed as the KL divergence of the prediction $P(X_t | \boldsymbol{E}_{1:t-1}, \boldsymbol{C}_{1:t})$ with regard to the belief $Bel(X_t)$.

However, given a sensory infrastructure with n sensors, in order to exactly determine the optimal sensory configuration, the optimization modules should evaluate the information gain for all the possible 2^n sensor states, with an effort that, as n grows, becomes quickly intractable. For this reason, this work defines a heuristic which roughly approximates the information gain of a given sensory configuration with the simple sum of the information gains of each sensor. The

current information gain of a single active sensor can be simply evaluated by comparing the belief obtained by using its last reading with the belief obtained by ignoring such evidence. The belief that would be obtained by ignoring the evidence e_t^i, for a specific state of the world, is defined as a function of the belief $Bel(x_t)$:

$$\overline{Bel}_{e_t^i}(x_t) = \beta \cdot \frac{Bel(x_t)}{P(e_t^i|x_t)}, \tag{4.7}$$

where β is a normalizing factor. It is then possible to compute the information gain of a single active node E^i, in the time slice t, as the KL divergence of $\overline{Bel}_{e_t^i}(X_t)$ with regard to $Bel(X_t)$, as follows:

$$\begin{aligned} inf\_gain(E_t^i) &= inf\_gain(E_t^i = e_t^i) = \\ &= D_{KL}(Bel(X_t) \| \overline{Bel}_{e_t^i}(X_t)). \end{aligned} \tag{4.8}$$

The evaluation of the information gain of an inactive sensor shows a higher computational cost, since the absence of a reading involves the computation of the belief change for all the possible sensory readings. Namely, the information gain of an inactive sensor should be calculated as follows:

$$inf\_gain(E_t^i) = \sum_{e_t^i} P(E_t^i = e_t^i) \cdot inf\_gain(E_t^i = e_t^i). \tag{4.9}$$

However, in order to reduce the computational cost, the heuristic adopts a further simplification that exhibits the advantage of considering not only the instantaneous information gain, but also the past history. Accordingly, the system does not exploit Eq. (4.9) to compute the information gain of an inactive sensor; rather, at each time slice, it evaluates the current information gain only of the active sensors and updates their sample mean and variance. In this way, an estimate of the value of information provided by the whole sensory configuration $\boldsymbol{conf}_t$ can be represented as

$$\boldsymbol{SIE} \stackrel{def}{=} [inf\_gain_{avg}(\boldsymbol{conf}_t), inf\_gain_{var}(\boldsymbol{conf}_t)], \tag{4.10}$$

where

$$\begin{aligned} inf\_gain_{avg}(\boldsymbol{conf}_t) &\stackrel{def}{=} \sum_i inf\_gain_{avg}(E_t^i) \cdot s_t^i, \\ inf\_gain_{var}(\boldsymbol{conf}_t) &\stackrel{def}{=} \sum_i inf\_gain_{var}(E_t^i) \cdot s_t^i. \end{aligned} \tag{4.11}$$

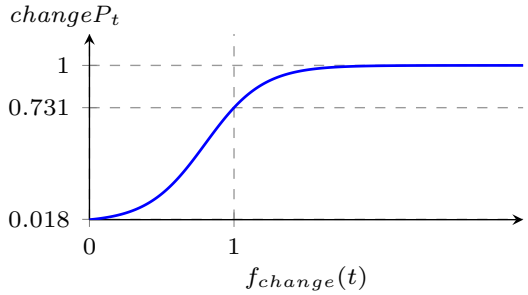
Since the KL divergence does not satisfy the triangle inequality, the proposed approach constitutes only a heuristic, although it performs well in practice, as will be shown in the experimental section. Moreover, storing the sampling mean and variance of the information gain for n sensors is much more efficient than exploring all the 2^n sensory configurations and all possible readings of inactive sensors.

The computational complexity of calculating the feature vector FV_t is dominated by the evaluation of Eq. (4.11). The information gain of a single sensor node can be derived in $O(m)$, where m is the number of possible values of X_t. Their mean and variance can be updated online in $O(1)$, using the recurrence formulas proposed by (Knuth, 1997). Thus, the overall complexity of the operations performed by the *Feature Extraction* module with n sensors is $O(m \cdot n)$.

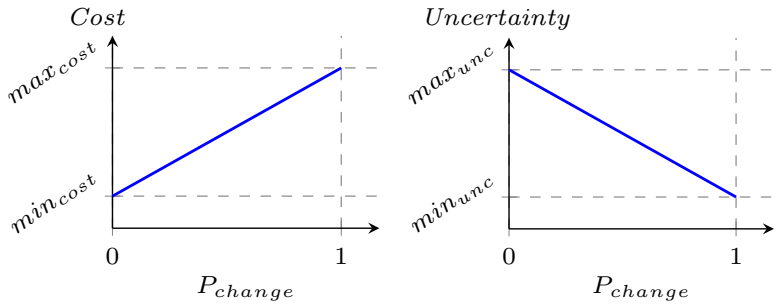
4.2.2 Global Optimization

The goal of the *Global Optimization* is to dynamically drive the behavior of the self-configuring component, according to context changes, since static criteria regarding reconfiguration frequency or global objectives of the system are inappropriate for dynamic scenarios.

The *Global Optimization* module dynamically modifies two alarm thresholds, concerning inference uncertainty and cost, which are used for triggering and constraining the reconfiguration performed by the *Constrained Optimization* module, as depicted in Fig. 4.1. The basic idea is that, in order to obtain significant energy savings, it is appropriate to reduce the set of active sensors when the system expects a reduced variability in context conditions, even if such an action may involve a reduction in the inference accuracy. On the other hand, whenever significant context variability is expected, a better policy might be to increase the sensory activity of the system. To this end, the *Global Optimization* module uses the estimated temporal lengths of world states, as learned during the training phase by the *Data Fusion* module, the inferred belief about the current state of



(a) $changeP_t(f_{change})$ with $\lambda = 5$ and $\nu = 4$.



(b) θ_{cost_t} and θ_{unc_t} functions.

Figure 4.2: Functions used in the *Global Optimization* module.

the world, and the time elapsed since the last state transition observed by the system.

For instance, if the system identifies a state condition with a high probability to persist for a long time, it might switch *off* most of the sensors without a relevant accuracy reduction. Conversely, if the system estimates a sudden transition for current state, it should activate all the sensors necessary to achieve the desired accuracy, even at the cost of higher energy consumption.

To this end, the *Global Optimization* module tries to predict if the state of the world will change in the next time slice, by estimating the probability of a state condition transition with the following soft-threshold function:

$$changeP_t(f_{change}) = 1 - \frac{1}{1 + e^{\lambda f_{change}(t) - \nu}}, \tag{4.12}$$

where

$$f_{change}(t) = \frac{elapsedTime_t}{expectedDuration(X_t)}, \tag{4.13}$$

and λ and ν are parameters of the soft-threshold function. This implies the probability of transition of a state condition increases as $f_{change}(t)$ value grows, as shown in Fig. 4.2a. The function $expectedDuration(X_t)$ in Eq. (4.13) is computed as the expected value of the current state duration, as follows:

$$\begin{aligned} expectedDuration(X_t) &= \sum_i w_t^i \cdot avgDuration(x^i), \\ w_t^i &= Bel(x_t^i), \end{aligned} \tag{4.14}$$

where $avgDuration(x^i)$ represents the estimation of the temporal length of a specific state of the world x^i, as learned during the training phase by the *Data Fusion* module, and each weight w_t^i is the belief of a specific state in the time slice t, such that $\sum_i w_t^i = 1$.

The transition probability of the state condition computed by the *Global Optimization* module is used to dynamically modify the cost and uncertainty thresholds, which can vary between minimum and maximum values set by the system administrator. Whenever the uncertainty and cost of the inference performed by the *Data Fusion* module exceed these thresholds, an alarm is triggered and consequently the *Constrained Optimization* module performs the reconfiguration of the sensory infrastructure. The thresholds for cost and uncertainty, i.e., θ_{cost_t} and θ_{unc_t}, are modified according to the following equations:

$$\begin{aligned} \theta_{cost_t} &= min_{cost} + \\ &\quad + (max_{cost} - min_{cost}) \cdot changeP_t(f_{change}), \\ \theta_{unc_t} &= min_{unc} + \\ &\quad + (max_{unc} - min_{unc})(1 - changeP_t(f_{change})). \end{aligned} \tag{4.15}$$

As shown in Fig. 4.2b, when $changeP_t(f_{change})$ is close to 0, the system enters a power-saving mode, thereby reducing the cost threshold and raising the uncertainty threshold. Conversely, when $changeP_t(f_{change})$ is close to 1, the system gives priority to the accuracy of the inference, raising the cost threshold and reducing the uncertainty threshold.

At each time slice, the *Global Optimization* module updates the thresholds and compares them with the cost and uncertainty computed by the *Feature Extraction*

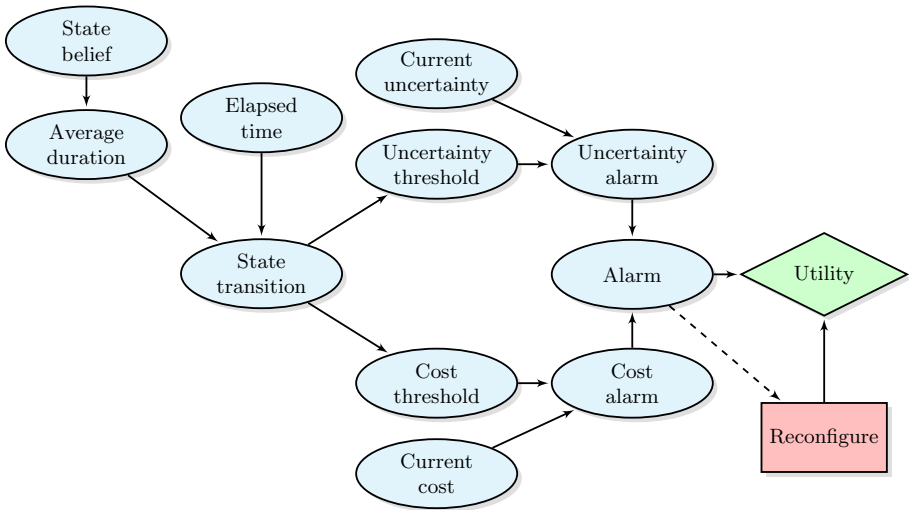


Figure 4.3: The *Global Optimization* module influence diagram.

module, checking whether any alarm is triggered. If both the thresholds are exceeded, priority is given to the reduction of uncertainty, since the main goal of the system is to infer the state of the world in a precise and reliable way.

The computational complexity of the *Global Optimization* module is dominated by Eq. (4.14), which can be computed in $O(m)$, where m is the number of possible values of X_t.

The *Global Optimization* module is implemented as an Influence Diagram (Koller and Friedman, 2009), a generalization of a Bayesian network, capable of supporting probabilistic decision-making. Such Influence Diagram is shown in Fig. 4.3. It merges two types of information: (i) information describing the state of the world, captured by the *Data Fusion* module, and (ii) meta-information about the behavior of the *Data Fusion* module, extracted by the *Feature Extraction* module. All such information represents the context from the point of view of the *Global Optimization* module.

The *Alarm* node and the *Reconfigure* decision node represent binary variables that indicate, respectively, whether an alarm is triggered and whether the system should dynamically modify the state of its sensors.

The conditional probability distributions of the *State transition* node and of the thresholds nodes are based on Eq. (4.12) and Eq. (4.15), respectively. Finally, the utility function of the Influence Diagram is computed as the *XNOR* (exclusive

NOR) of the *Alarm* variable and the *Reconfigure* decision variable, in order to trigger a reconfiguration when an alarm condition holds.

4.2.3 Constrained Optimization

In order to achieve the best possible trade-off between cost and reasoning accuracy, the *Constrained Optimization* module leverages the dynamic thresholds set by the *Global Optimization* module to dynamically modify the state of the sensory infrastructure. For instance, it can reduce the energy consumption of the sensory infrastructure at the cost of a sacrifice in accuracy, by switching redundant devices to low-power mode, thus suspending their data gathering and transmission functionalities, as explained in Section 4.1.

To manage the above conflicting goals, this work adopts a multi-objective trade-off analysis based on a Pareto-dominance criterion (De Paola and Gagliano, 2014; De Paola et al., 2011), since the traditional approach of using multi-attribute utility theory (Padovitz et al., 2005), capable of maximizing a single expected utility that summarizes all the considered attributes, has several drawbacks. Namely, it is often difficult to formulate an utility function which correctly models the optimization problem. Moreover, such an approach would require a precise assessment of the relative relevance of the goals to achieve.

In order to guarantee a steady adaptation of the sensory infrastructure, and to limit the computational complexity of the optimization, the system considers only *atomic* reconfigurations, i.e., actions that modify the state of single sensors. Thus, in a system with n sensors, the *Constrained Optimization* module chooses among n possible sensory configurations when an alarm is raised. If neither of the thresholds is exceeded by the system, no reconfiguration will be performed. The cost and information gain of each achievable configuration can be derived in constant time. Thus, the overall complexity of computing these quantities for all n achievable configurations is $O(n)$.

The *Constrained Optimization* module categorizes the sensory configurations achievable in the time slice t as two disjoint classes: *dominated* configurations and *non-dominated* configurations. A sensory configuration $\boldsymbol{conf}_t^*$ is *non-dominated*, or Pareto-optimal, if no other solution has better values for all the objectives

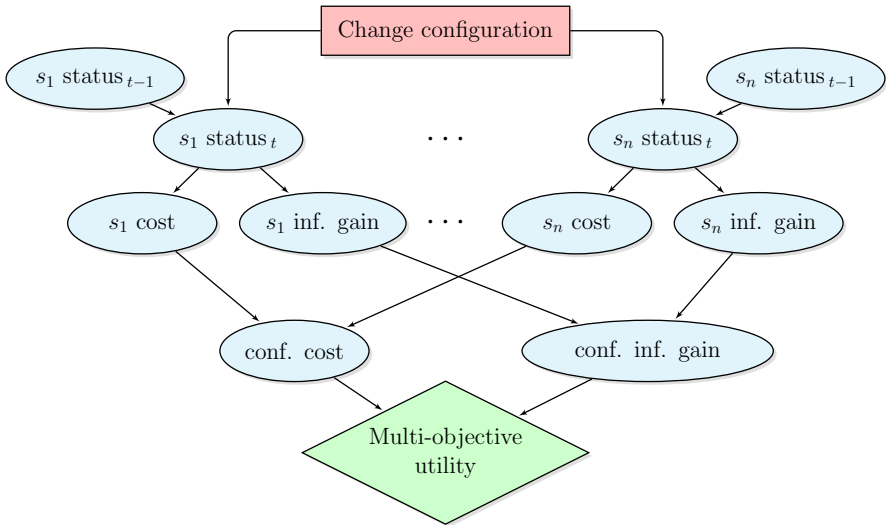


Figure 4.4: The *Constrained Optimization* module multi-objective influence diagram.

considered, so that the following holds $\forall i \in \{1, \ldots, n\}$:

$$\begin{cases} cost(\boldsymbol{conf}_t^*) \leq cost(\boldsymbol{conf}_t^i) \\ \qquad\qquad \text{or} \\ inf\_gain_{avg}(\boldsymbol{conf}_t^*) \geq inf\_gain_{avg}(\boldsymbol{conf}_t^i). \end{cases} \tag{4.16}$$

In order to determine the action that achieves the best trade-off between costs and performance, the *Constrained Optimization* module considers only the set of *non-dominated* configurations, also known as Pareto front. The selected sensory configuration depends on the alarm raised by the *Global Optimization* module, as follows:

- if only the cost threshold was exceeded, the sensory configuration minimizing the expected $cost(\boldsymbol{conf}_{t+1})$ is selected among the *non-dominated* solutions;
- otherwise, if only the uncertainty threshold or both thresholds were reached, the sensory configuration maximizing the expected $inf\_gain(\boldsymbol{conf}_{t+1})$ is selected among the *non-dominated* solutions.

The Pareto front can be derived in $O(n^2)$ time, and the final selection of the configuration to choose requires $O(n)$ time, which yields a total complexity of

$O(n^2)$. The operations performed by the *Constrained Optimization* module are implemented as a multi-objective influence diagram (Diehl and Haimes, 2004). This approach extends traditional influence diagrams, which only allow a single utility node or a single combined utility node that is sum or product of other utility functions (Tatman and Shachter, 1990). To overcome this limitation, a new kind of utility node was defined in (Diehl and Haimes, 2004) as a vector of objectives, together with an algorithm for the solution of such multi-objective influence diagrams.

This work proposes the influence diagram shown in Fig. 4.4, where the *Change configuration* decision node indicates the single sensor whose state must be changed, depending on its previous state. Finally, the best sensory configuration is chosen among the optimal front, as explained above, and a control message is sent to the appropriate sensor, thus triggering the dynamic reconfiguration of the sensory infrastructure.

4.3 Case Study: Activity Recognition in AmI Scenario

The application scenario chosen to prove the effectiveness of the proposed approach is an Ambient Intelligence (AmI) environment. In particular, an implementation of the system was simulated in a smart home system enriched with a pervasive sensory infrastructure based on WSNs. Such scenario is suitable for the experimental evaluation for several reasons: first of all, AmI contexts are characterized by a high dynamism, due to their continuous and unpredictable interaction with users. Second, the use of WSNs as sensory infrastructure poses severe constraints on device energy consumption, justifying the necessity of finding the best trade-off between inference accuracy and energy consumption of sensory devices. Finally, since observed data are not collected by specialized sensors, but rather by sensors whose readings are only partially correlated with the environmental features of interest (De Paola et al., 2011), it is necessary to perform a multi-sensor data fusion.

The system is designed for inferring any hidden characteristic of the observed world. In the case study proposed here, activities performed by the user were chosen as context features to be inferred, since such information allows AmI systems to be fully responsive to user needs. Moreover, recognizing user activities such as eating, cooking, sleeping, or working (Cook, 2010; Krishnan and Cook, 2012), is one of the major challenges of AmI systems.

The approaches to address this challenge vary greatly depending on the kind of activities to classify, the data fusion method adopted, and the sensors used. For example, inertial sensors such as accelerometers and gyroscopes, also installed on mobile devices, are commonly used to recognize activities that involve physical movements, e.g., walking, running, sitting down and standing up (Gao et al., 2014; Kwapisz et al., 2011). The authors of (Gao et al., 2014) compared the advantages and disadvantages of single-sensor and multi-sensor wearable systems, and proposed an approach based on a decision tree classifier to find a trade-off between recognition accuracy and computational and communication complexity. In recent years, several works, such as (Kwapisz et al., 2011), have considered the possibility of exploiting the increasing pervasiveness of smartphones to recognize user activities, by merging the raw data coming from the sensors embedded in these devices. The data collected by wireless sensors which are pervasively deployed in the environment are often used to recognize a wider range of Activities of Daily Living (ADLs), such as eating, cooking, sleeping, or working (Cook, 2010; Krishnan and Cook, 2012).

The implementation of the proposed system in a specific application scenario implies the characterization of (i) the components of the feature vector which are related to the state of the world, (ii) the context information acquired directly from the external world, and (iii) the cost function adopted by the self-configuration modules.

In the scenario considered here, i.e., user activity recognition in a smart home, the feature vector contains the belief about the activity currently performed by the user and the time elapsed since the last activity transition, in addition to features which do not depend on the application scenario.

The best available context information for improving the accuracy of the system proved to be the period of day, as demonstrated in Section 3.4.3. The duration

of the activities also play an important role in the Global Optimization module, highly influencing the performance of the system, as described in Section 4.2.2.

The cost function was defined with the goal of enforcing an energy saving policy that aims to extend the devices' lifetime; thus, the value of the proposed cost function increases when the residual battery charge of the corresponding sensor decreases, according to the following equation:

$$f_{cost}(E_t^i) = baseCost(E_t^i) \cdot \left(1 + \gamma \left(1 - charge(E_t^i)\right)\right), \tag{4.17}$$

where $0 \leq charge(E_t^i) \leq 1$ is the remaining charge of sensor E^i in the time slice t, and γ is a parameter that controls the relative importance of the charge level. When the value of γ increases, the system is more inclined to use all the sensors in a uniform manner, instead of preferring only the best ones, and this greatly extends the sensory infrastructure lifetime with little impact on accuracy, as will be shown in the next section.

4.4 Experimental Evaluation

The experimental results compare the performance of three different systems. The first one, called *All-On*, is the Data Fusion system presented in Chapter 3; it does not exploit the optimization modules to adaptively self-configure its behavior at runtime, and uses all the available sensors in each time slice, thus minimizing the uncertainty of reasoning regardless of energy consumption. The second one, called *Subset-On*, gives priority to energy savings, leveraging only a small subset of sensors, i.e., 10 of the 39 available sensors, and does not exploit the optimization modules to improve its performance. The last system, finally, is the context-aware self-optimizing system proposed in this chapter, and it fully exploits the *Global Optimization* and *Constrained Optimization* modules described in the previous sections. The first two systems are considered as baseline for comparison with the proposed one.

The experimental setting and metrics are the same as the ones presented in Section 3.4.1 and Section 3.4.2. In particular, the experiments presented here

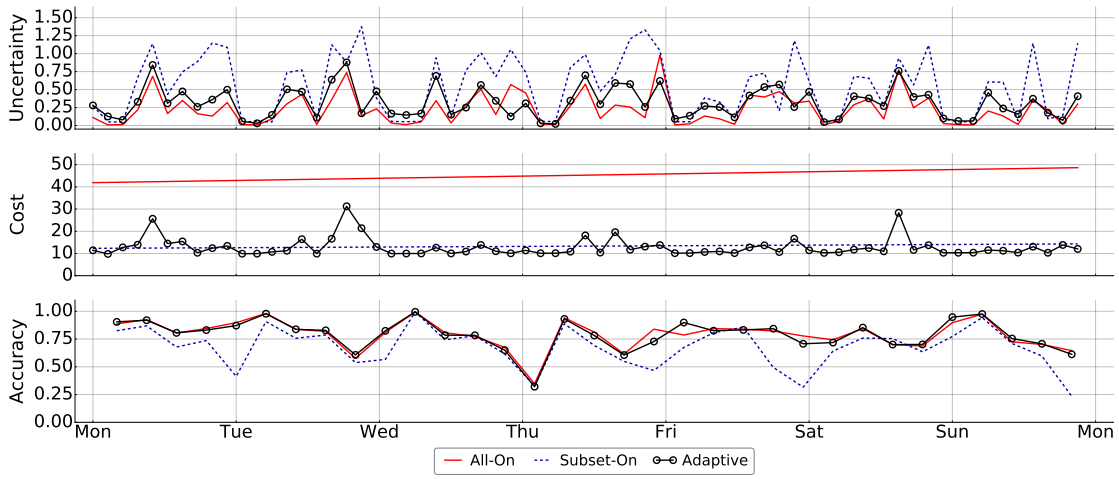


Figure 4.5: Uncertainty, cost, and accuracy trends of the three systems considered during a given week.

compare the uncertainty, cost, and accuracy of each system, as well as precision, recall, and F-score of each activity, as detailed in the following.

4.4.1 Experimental Results

The first experiment presented in this section is the comparison of uncertainty, cost, and accuracy trends of the three systems during a given week, when also the *Other* activity class is considered. In this first test, temperature readings were discarded, considering the low correlation between such data and the activity performed by the user. Fig. 4.5 shows the simple moving averages (SMAs), i.e., the unweighted mean of the values collected in the previous hour. As expected, the *All-On* system exhibits the lowest uncertainty and the highest energy cost. Conversely, the *Subset-On* system presents the highest uncertainty and a lower power consumption than the *All-On* system. The *Adaptive* system shows an uncertainty close to that of the *All-On* system, with a lower power consumption. It is worth noticing that the average cost of the *Adaptive* system increases very slowly over time, since the optimization modules are smart enough to exploit sensors in a uniform manner, so as to maximize the life span of the WSN and minimize power consumption. With regards to the accuracy, the *All-On* and *Adaptive* systems perform similarly, and occasionally the *Adaptive* system overcomes the *All-On*. The *Subset-On* system,

Table 4.1: Average accuracy of the two baseline systems compared with the proposed *Adaptive* system, in each test of the cross-validation experiment based on the CASAS dataset (Cook, 2010).

All-On	Subset-On	Adaptive
0.780	0.659	0.755
0.802	0.704	0.818
0.788	0.670	0.796
0.792	0.677	0.803
0.810	0.672	0.815
0.803	0.657	0.811
0.790	0.539	0.777
0.782	0.652	0.788
0.811	0.687	0.817
0.769	0.644	0.773

on the other hand, behaves poorly, and its accuracy is behind those of the other two systems.

Table 4.1 reports the average accuracy of the two baseline systems compared with the proposed *Adaptive* system, in each test of the cross-validation experiment. Table 4.2 summarizes the results of the cross-validation tests, reporting the average accuracy (both considering and excluding the *Other* activity class), uncertainty, power consumption, number of active sensors, and F-score of the three systems in all tests. It can be observed that the proposed *Adaptive* system achieves an accuracy slightly higher than that of the *All-On* system. This may seem a bit unexpected, but it can be explained by considering the particularity of the dataset used, which mainly contains motion sensors. In such a scenario, using an optimal subset of sensors can lead to higher accuracy than keeping all sensors on, since it allows the system to exclude some false positive readings. For instance, when the user is sleeping, the *Adaptive* system exploits only sensors near the bedroom, while the *All-On* system will leave all the sensors on, possibly incurring in a greater number of false positive readings coming from sensors in other rooms, which may have a non-negligible noise level. The *Subset-On* system shows a lower accuracy, i.e., almost 14% less than the *Adaptive* system. The average costs of the two approaches, on the other hand, are very similar, with only a 1% difference. The

Table 4.2: Cross-validation results reporting the average accuracy, uncertainty, power consumption, number of active sensors, and F-score of the two baseline systems compared to the proposed *Adaptive* system.

	All-On	**Subset-On**	**Adaptive**
Accuracy	0.793	0.656	0.795
Accuracy without *Other*	0.889	0.705	0.878
Accuracy with temperature	0.749	0.534	0.786
Uncertainty	0.231	0.573	0.349
Power Consumption	44.771	13.168	13.494
Active Sensors	34.000	10.000	12.124
F-score	0.416	0.271	0.408
Execution Time	1.00x	0.76x	0.81x

average cost of the *All-On* system is more than three times that of the proposed *Adaptive* system.

Table 4.2 confirms that the uncertainty of the *Adaptive* system is comparable to that of the *All-On* system, whilst the *Subset-On* system performs worse than the other two, with an average uncertainty that is 64% higher compared to that of the proposed system. Similar considerations apply to the F-score of the three systems, which is similar for the *All-On* and *Adaptive* system, and more than 30% lower on the *Subset-On* system. To evaluate the optimization overhead, Table 4.2 also reports the average execution time, normalized by the same factor, so as to have 1 computation unit in the case of the *All-On* system. It is worth noticing that the *Adaptive* system gets a 19% speed-up with regard to the *All-On* system. Thus, the speed-up due to the fact of using fewer sensors in the data fusion process overweights the optimization overhead. Considering that the *Adaptive* and *Subset-On* systems use a similar number of sensors, on the average (12 and 10, respectively), the small difference in execution time of the two systems (0.81x and 0.76x, respectively) gives an idea of the optimization overhead.

These results indicate the advantage of using the *Adaptive* approach, since it performs better than the baseline systems, finding an optimal trade-off between performance and consumption. As expected, the accuracy of all systems improves significantly if the *Other* activity class is ignored, increasing by almost 10% in the *All-On* and *Adaptive* systems, and by about 5% in the *Subset-On* system. Table 4.2

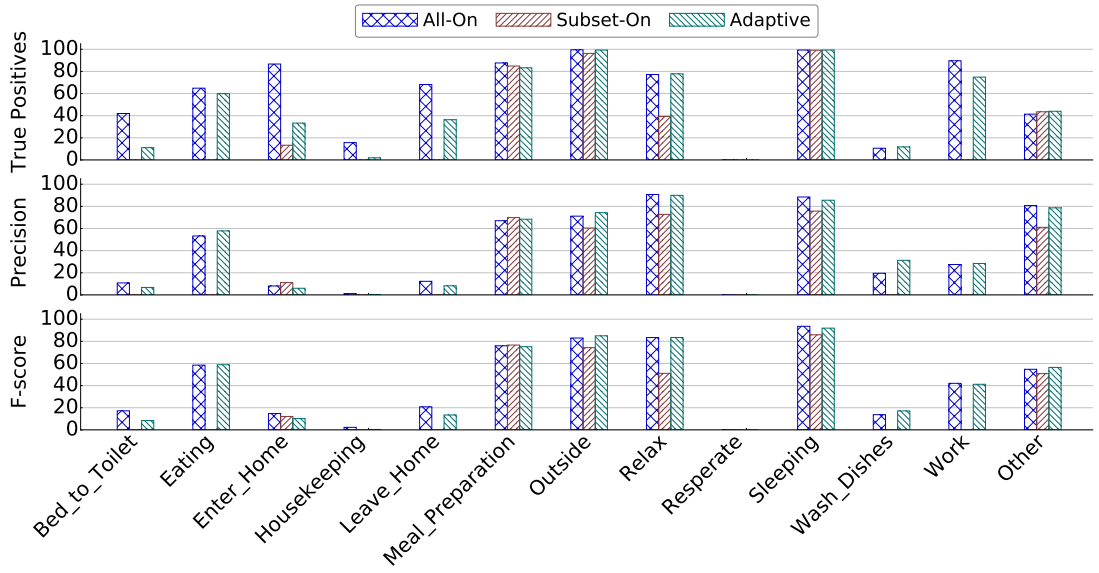


Figure 4.6: True positives, precision and F-score of each activity.

also shows the accuracy of the systems when using temperature data. In the case of the *Subset-On* system, the five temperature sensors were added on top of the fixed 10 normally used. The accuracy of all systems decreases when using temperature data, since there is a low correlation between this information and the activities performed by the user. However, the accuracy of the *Adaptive* system decreases by less than 1%, whilst the other two systems show a more noticeable reduction, since the *Adaptive* system detects the low importance of the temperature sensors, and often turns them off.

Fig. 4.6 compares the true positives rate, precision, and F-score of each activity in all systems, without considering temperature sensors. The performance of the *All-On* and *Adaptive* systems regarding many activities is largely comparable, and the *Adaptive* approach actually obtains better results than the *All-One* system in some of them. However, there are a few activities that are better recognized by the *All-On* system, due to a very low average duration that does not allow the *Global Optimization* module to correctly estimate the probability of an activity transition. As expected, the performance of the *Subset-On* system is visibly worse than the other two approaches. Nevertheless, it can be observed that some activities are easily recognized by all the systems, i.e., *Sleeping*, *Outside* and *Meal Preparation*,

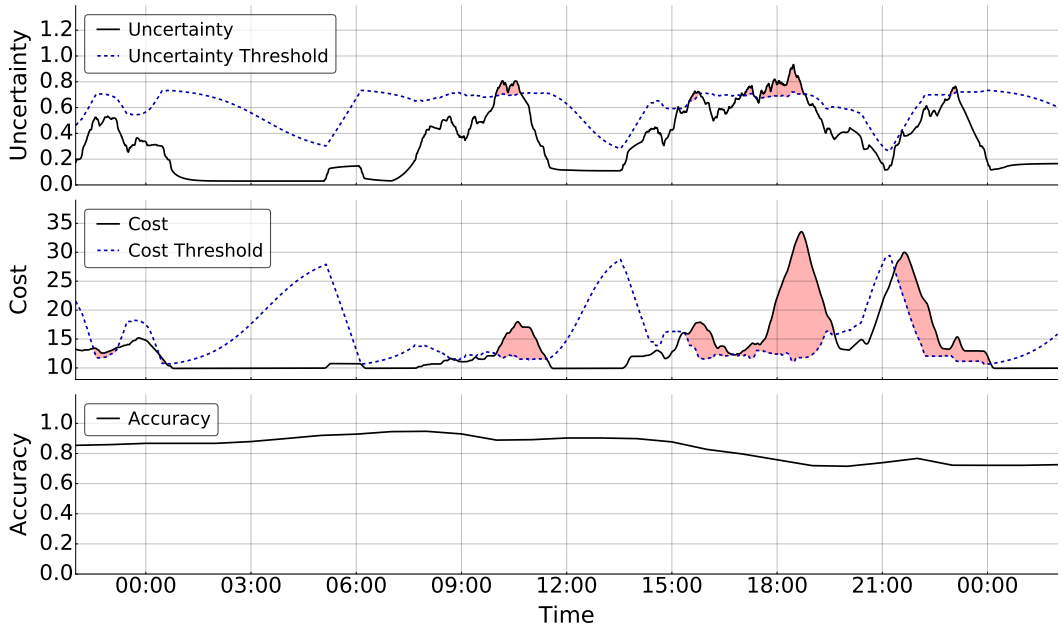


Figure 4.7: Uncertainty, cost, and accuracy trends of the *Adaptive* system in a single day, showing the adaptive thresholds.

while other activities are difficult to handle, regardless of the approach considered, i.e., *Housekeeping* and *Wash Dishes*. This can be explained by considering that some activities are performed in well-defined locations and times during the day, and therefore are better recognized using only motion sensors, while other activities are less structured, and some heterogeneous sensors should be installed in order to recognize them in a satisfying manner, as already noted in Section 3.4.3.

The experimental evaluation includes also a deep analysis of the effect of the *Global Optimization* module on the behavior of the *Adaptive* system. Fig. 4.7 shows the uncertainty and the cost trends of the *Adaptive* system, with the related alarm thresholds, during a given day. The uncertainty and cost alarms triggered by the system in the same day are also highlighted. As explained in Section 4.2.2, when the *Global Optimization* module expects a reduced variability in context conditions, it changes the alarm thresholds in order to switch off most of the sensors without a relevant reduction of accuracy, switching them on when it believes that the current activity will change in a short time, so as to ensure adequate accuracy, even at the expense of higher energy consumption. *Sleeping* is the most regular activity

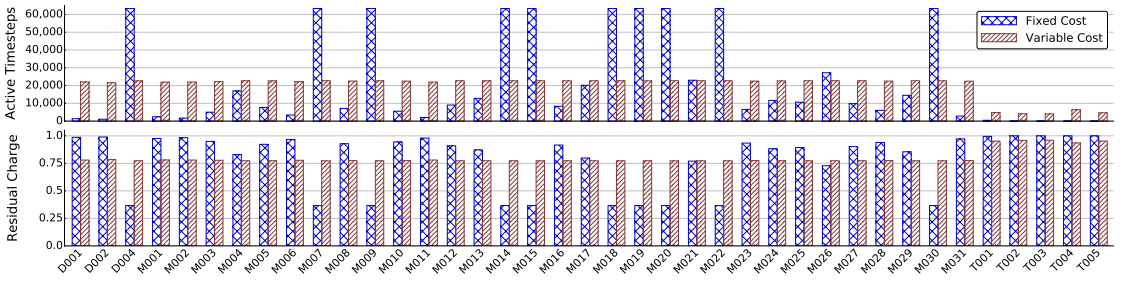


Figure 4.8: Sensor statistics usage with a fixed and variable cost function. The x axis enumerates all the available sensors: door sensors (Dxxx), motion sensors (Mxxx) and temperature sensors (Txxx).

in the dataset considered, and it is also the one with the longest average duration. This regularity makes the *Sleeping* activity ideal to analyze the way in which the system dynamically changes the alarm thresholds. The cost threshold is low at the beginning of the night, when the user goes to sleep, whilst the uncertainty threshold is high. This means that the system believes as unlikely a context transition in a short time, and thus enters a power-saving mode, reducing the cost threshold and raising the uncertainty threshold. During the night, the probability of an activity transition, as defined in Eq. (4.12), increases, and therefore the system gradually reduces the uncertainty threshold and increases the cost threshold. In the morning, when the probability of an activity transition is close to 1, the system gives priority to the accuracy of the inference, thereby increasing the cost threshold to its maximum value, and reducing the uncertainty threshold to its minimum.

Finally, the experimental evaluation compares the effectiveness of using the variable cost function, defined in Eq. (4.17), against an approach that leverages a fixed cost function, that does not increase when the residual battery of the corresponding sensor decreases. The goal of choosing a variable cost function is to enforce an energy saving policy that extends the sensory infrastructure lifetime by using sensors in a uniform manner. Fig. 4.8 compares the two approaches, showing the number of time slices in which each sensor has been powered on during the simulations, and the remaining energy charge at the end of the simulations. In this experiment, temperature sensors were also included, so as to show that the system correctly avoid sensors with a very low information gain, which are not deemed useful. It is evident that with a variable cost function the system uses

sensors more uniformly than the fixed cost approach, greatly extending the sensory infrastructure lifetime. However, it is worth noticing that, despite the energy policy, sensors which show a very low information gain, such as the temperature sensors (i.e., the last five sensors in Fig. 4.8), are seldom used by the system. Finally, comparing the accuracy of the variable cost approach against that of the fixed cost approach, the decrease is less than 1%.

Chapter 5

Filtering out unreliable data

To minimize the impact of feeding unreliable information to the system, the proposed architecture is enriched by a filtering module that aims to accurately classify reliable and unreliable sensing reports. Such a module can pre-process raw data coming from sensors and users, before feeding them to data fusion and reasoning modules in the higher levels of the AmI architecture.

The filtering module proposed in this chapter is validated in the challenging scenario of Smartphone Crowdsensing (SC) in a smart city, in which users send sensing reports about events of interest via smartphone applications. Classifying unreliable data is especially important in a smart city scenario, since it might not be feasible to monitor an entire city using only traditional sensors.

According to the SC paradigm, it is possible to use humans *as sensors* (Wang et al., 2014a), designing pervasive sensing applications to confirm the occurrence of complex events, which are difficult to detect by exploiting only traditional sensors. Reports may be sent manually or automatically, by leveraging the plethora of sensors equipped on today's smartphones, that provide information such as location, acceleration, temperature, and noise. However, the central role of people in the sensing process implies that the success of such systems is strictly dependent on the reliability of the information sent by participants.

This chapter proposes a novel *Framework for optimizing Information Reliability in Smartphone-based participaTory sensing* (FIRST), that increases the accuracy

in classifying sensing reports with the help of mobile trusted participants (MTPs), which are special users hired to send reliable reports to the system.

The remainder of this chapter is organized as follows. Related work on SC systems is summarized in Section 5.1. Section 5.2 presents the application scenario considered, while Section 5.3 introduces the concept of trusted participants. Section 5.4 describes the three components of the filtering module. Finally, Section 5.5 presents the experimental results.

5.1 Related Work

Related work can be divided into two main approaches: *trusted platform modules* (TPMs) and *reputation-based systems*. TPMs are hardware chips that reside on the participants' devices, and ensure that the sensed data is captured by authentic and authorized sensor devices within the system (Dua et al., 2009; Gilbert et al., 2011; Saroiu and Wolman, 2010). The main drawback of this approach is that TPMs require additional hardware not currently available on off-the-shelf devices, implying such solutions are not readily deployable. Moreover, TPM chips are tailored to verify data coming from physical sensors (e.g., accelerometer, camera). Thus, they are not applicable to SC systems in which the information is directly supplied by the participants.

Most of related work has focused on developing reputation-based systems to increase information reliability (Mousa et al., 2015). Specifically, they associate each user with a *reputation level*, which is estimated and updated over time. Among related work, (Restuccia and Das, 2014; Luo et al., 2014; Wang et al., 2014c; Huang et al., 2014) are the most relevant. In (Wang et al., 2014c), the authors proposed *ARTsense*, a reputation-based framework that includes a privacy-preserving provenance model, a data trust assessment scheme, and an anonymous reputation management protocol. The main issue of (Wang et al., 2014c) is that user reputation is updated by considering contextual factors, such as location and time constraints. Given user location and timestamp of reports are easily forgeable quantities, the solution proposed in (Wang et al., 2014c) may not perform well in practical SC systems, where malicious users may voluntarily tamper with their GPS location and timestamp of reports.

Recently, in (Huang et al., 2014), the authors proposed a reputation framework which implements an improved version of majority vote. The main limitations of this framework are (i) the assumption of constant sampling rate, which is not realistic in asynchronous SC systems, and (ii) the poor resilience to a large number of malicious users, as the framework uses a modified version of majority vote to update user reputation levels. To overcome such limitation, in (Restuccia and Das, 2014) the authors proposed *FIDES*, a reputation-based framework that used mobile security agents to classify sensing reports. Similarly to the filtering module presented in this work, FIDES is also resilient to a large number of malicious users. However, as in (Huang et al., 2014), the necessity to set a significant number of parameters makes the actual performance of the framework hardly predictable in reality. On the other hand, the work presented here does *not* depend on specific parameters.

A number of frameworks aimed to recruit participants in order to maximize the coverage of the sensing area have been recently proposed (Khan et al., 2015; Zhang et al., 2014; Xiong et al., 2015; Zhang et al., 2016; Liu et al., 2016; Ueyama et al., 2014). In (Khan et al., 2015), the authors propose a framework to ensure coverage of the collected data, localization of the participating smartphones, and overall energy efficiency of the data collection process. (Zhang et al., 2014) proposed *CrowdRecruiter*, a framework that minimizes incentive payments by selecting a small number of participants while still satisfying probabilistic coverage constraint. In (Xiong et al., 2015), the authors proposed a framework aimed to maximize the coverage quality of the sensing task while satisfying the incentive budget constraint. In (Ueyama et al., 2014), the authors formulate the problem of sensing given points of interest as a gamification problem, and devise a heuristic algorithm for deriving the set of users to which requests are sent and appropriate reward points for each request. The approach proposed here is different because it relies on MTPs to compute the trustworthiness of participants, which ultimately improves information quality significantly, as will be shown in the remainder of this chapter. Furthermore, this work considers also the problem of modeling and optimizing the information reliability (Mousa et al., 2015).

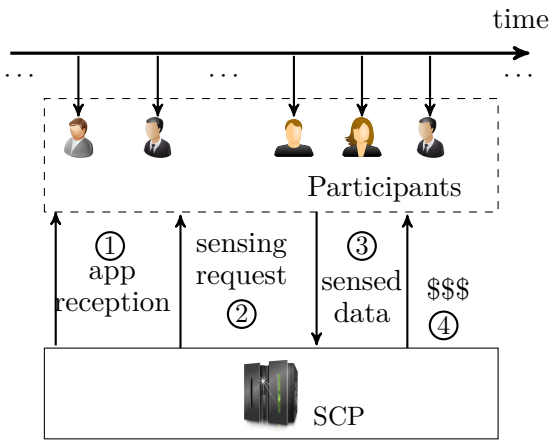


Figure 5.1: SCP system architecture.

5.2 Application Scenario

The scenario considered in this chapter is that of a smartphone crowdsensing architecture, as depicted in Figure 5.1, consisting of a platform (SCP) which can be accessed through 3G/4G or WiFi Internet connection. Participants download through common app markets, like *Google Play* or *App Store*, the smartphone crowdsensing app, which is responsible for handling data acquisition, transmission, and visualization (step 1). Then, the SCP sends (periodically or when necessary) sensing requests through the cloud to registered participants (step 2). The participants can answer such requests by submitting their sensed data (step 3), and eventually receive a reward for their services (step 4). Hereafter, the terms *participant* and *user* will be used interchangeably.

As far as the sensing application is concerned, the phenomenon being monitored has to be (i) quantifiable, (ii) dynamic (i.e., varies over time), and (iii) not subject to personal opinion. This includes phenomena measurable with physical sensors, such as air/noise pollution levels (D'Hondt et al., 2013), but also quantities such as occupancy level of parking lots (Nawaz et al., 2013), gas prices (Dong et al., 2011), traffic events (e.g., car crashes and traffic jams), and so on. Furthermore, this work assumes that the *range* of the sensing quantity being monitored may be divided up into *intervals* or *categories*, which are specific to the SC application but are properly defined before deployment. For example, in a gas price monitoring system,

the range of possible values could be from $2 to $3 dollars per gallon, divided into intervals of 10 cents each. In a traffic monitoring application, a different category for each traffic event (e.g., "Car Crash", "Road Closure", "Traffic Jam") could be specified. A sensing report is considered *reliable* if the quantity being reported falls into the interval the phenomenon is currently in (or belongs to that category). For example, if the actual gas price at a station is $2.46, a report is considered reliable if the reported value falls into the range [$2.40, $2.50).

As far as security assumptions are concerned, the SCP is considered trustworthy in terms of its functionality (such as user registration, issuing credentials, receiving, processing, and redistributing data). Furthermore, confidentiality, integrity, and non-repudiation are assumed to be addressed by using standard techniques such as cryptography and digital signatures. This chapter concentrates on tackling the inappropriate behavior of participants, and assume they may exhibit *malicious* or *unreliable* behavior:

- *Malicious:* these users are willingly interested in feeding unreliable reports to the system; their purpose is to either creating a disservice to other users (e.g., fake road traffic lines), or gaining an unfair advantage w.r.t. other users.
- *Unreliable:* these users are not willingly submitting false information, but they still do it because of malfunctioning sensors or incapability in performing the sensing task (Restuccia and Das, 2014).

The proposed filtering module provides a general approach to determine the reliability of each user depending on his/her behavior. Section 5.5 experimentally studies three types of attacks, namely the corruption, on-off and collusion attacks previously defined in (Mousa et al., 2015), and prove that FIRST is able to quickly detect the malicious behavior and discard unreliable reports. The remainder of this chapter will focus only on the issue of information reliability. Other threats, such as DoS-based attacks, are out of the scope of this work. Also, incentivizing users' participation is out of the scope of this chapter; solutions such as (Yang et al., 2012) may be integrated. Finally, the output of the filtering module can be used by higher-level modules of the architecture proposed in this dissertation to merge the collected data and drive the process of knowledge abstraction up to high-level concepts.

5.3 Mobile Trusted Participants

This work takes the same approach used by the successful *National Map Corps* (McCartney et al., 2015) project, and use mobile trusted participants (MTPs) to tackle the attacks described in the previous section. Specifically, MTPs are individuals who are able and willing to submit regularly reliable reports regarding the phenomenon being monitored or observed. These reports are used to *validate* users' sensing reports coming from nearby, and ultimately estimate the reliability of those participants. Such estimate is used to classify reports generated where MTPs are currently not present, as explained in the next sections.

To allow mathematical formulation, the sensing area is logically divided into $\mathcal{S} = \{s_1, \ldots, s_n\}$ sectors, which may have variable size and shape and represent the *sensing granularity* of the application. For example, in a gas price app, each sector can represent a gas station. In an air pollution monitoring application, a sector may be as large as a neighborhood of a city, whereas in a traffic monitoring application, sectors may be as large as a city block. Let also $\mathcal{U} = \{u_1, \ldots, u_z\}$ be defined as the set of users contributing to the sensing application.

The MTP report validation process is modeled as follows. In order to validate user reports, the module assumes that reports sent by MTPs are valid for a time period of T units. The value of T is a system parameter that is dependent on the variance over time of the sensing quantity being measured. For example, in a traffic monitoring application, a good value of T could be 5-10 minutes, while in a gas price monitoring app T can be much longer. Section 5.5.2 evaluates the impact of T on the system performance.

Definition 5.1. ***Validation of sensing reports****. Whenever a sensing report q is received from a user u_i in sector s_j, the platform checks whether a report from an MTP in sector s_j was received in the previous T time units. If yes, then the report is cross-checked with that coming from the MTP. If q is reliable (i.e., falls into the range of the report sent by the MTP), q is marked as* validated *and classified as reliable. Instead, q is rejected if unreliable. If q is not validated, it is classified as reliable or unreliable depending on an algorithm discussed in Section 5.4.*

Figure 5.2 illustrates an example in which an MTP is moving over a sensing area comprising three sectors. The locations at which the MTP submits a sensing

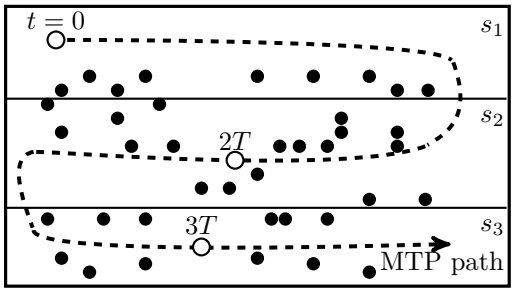


Figure 5.2: Example of an MTP moving over the sensing area.

report are marked as white circles, while users are depicted as black dots. The user reports from sector s_1 between $t = 0$ and T units are validated by using the MTP report sent at $t = 0$. Meanwhile, the MTP moves to sector s_2 and generates a new report at time $2T$, which then validates users reports from sector s_2 in the next time window. Similarly, the MTP report at $3T$ validates the user reports from sector s_3 in the time interval $[2T, 3T]$.

Examples of MTPs in urban sensing scenarios include, but are not limited to, professional drivers (i.e., taxi/bus), policemen, employees of the SC application, or people commuting on a daily basis to their workplace. Henceforth, the MTPs will be considered as reliable, in the sense that it is implied that their reports reflects the actual status of the event being monitored. This also implies that MTP reports originating from the same sector during the same time window are supposed to be equivalent. The case in which trusted participants can be, up to some extent, unreliable has already been studied (Restuccia and Das, 2014).

5.3.1 MTP Optimization Problem

It is intuitive that the number of validated sensing reports (and therefore, information reliability) increases as the number of recruited MTPs increases. However, in practical implementations, the budget to recruit MTPs is not unconstrained; the number of MTPs that can be used by the system will be limited and therefore, insufficient to guarantee perfect information reliability. The question this chapter aims to answer is the following: it is possible to find, *before deployment*, a good estimation of the minimum number of MTPs that will allow the SC system to achieve desired classification accuracy? To this end, the MTP Optimization Prob-

lem (MOP) is defined in the next section. Before that, the metric of classification accuracy is defined as follows:

Definition 5.2. ***Classification accuracy.*** *Let A define the event of the system considering a report as reliable, and let F define the event of a user submitting an unreliable sensing report. Let E define the event of erroneously deeming reliable (resp. unreliable) an unreliable (resp. reliable) report. By definition, it follows that the probability of event E, denoted as $\mathbb{P}\{E\}$, can be computed as*

$$\mathbb{P}\{E\} = \mathbb{P}\{F\} \cdot \mathbb{P}\{A \mid F\} + \mathbb{P}\{\overline{F}\} \cdot \mathbb{P}\{\overline{A} \mid \overline{F}\} \tag{5.1}$$

where $\overline{X}$ is defined as the complement of event X. Thus, $1 - \mathbb{P}\{E\}$ represents the classification accuracy of the SC system, and will henceforth be used to evaluate its performance.

The proposed framework will provide the mathematical tools to relate the number m of MTPs to the error probability $\mathbb{P}\{E\}$ and the mobility of users.

Let ϵ^{max} be the desired maximum classification error probability. The MTP Optimization Problem (MOP) is then defined as follows.

Definition 5.3. MTP Optimization Problem.

$$\textit{Minimize } m \textit{ such that } \mathbb{P}\{E\} \leq \epsilon^{max}$$

5.4 The FIRST Filtering Framework

Figure 5.3 illustrates the main components of the framework, defined as follows.

- **Likelihood Estimation Algorithm** (LEA): it provides an approximation of the mobility of users and MTPs. LEA is based on an *image processing* technique that produces an approximate likelihood based *only* on geographical information (i.e., the map of the sensing area).
- **Computation of Validation Probability** (CVP): this component derives the probability $\mathbb{P}\{V\}$ of the event V that a sensing report will be validated

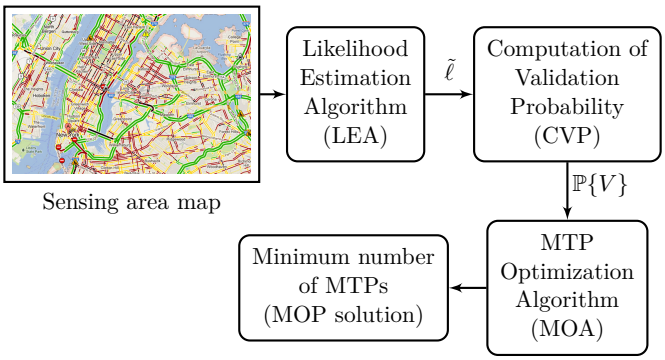


Figure 5.3: Components of FIRST.

by at least one MTP, as a function of the number of MTPs deployed and the approximate mobility of users.

- **MTP Optimization Algorithm** (MOA): it takes $\mathbb{P}\{V\}$ and computes $\mathbb{P}\{E\}$, so as to provide a solution to the MTP Optimization Problem to achieve desired maximum error ϵ^{max}.

For better clarity, Section 5.4.1 describes the CVP component assuming that the actual mobility distribution is known. Then, Section 5.4.2 explains how to obtain an approximate distribution of the mobility by using LEA, when the real mobility is unknown. Finally, Section 5.4.3 describes the MTP Optimization Algorithm, and Section 5.4.4 discusses how FIRST can be implemented in real-world SC systems.

5.4.1 Computation of Validation Probability

This section derives the probability $\mathbb{P}\{V\}$ of the event that a sensing report will be validated by at least one MTP. Let $\mathcal{Q}$ be the set of MTPs competing for offering their sensing services, and $\mathcal{U}$ be the set of users of the application. Let $u(i,z,t)$ be the distribution over the sector set $\mathcal{S}$ of the random variable (r.v.) U_z^t describing the location of user z at time t. Let also $q(i,z,t)$ be the distribution over the sector set $\mathcal{S}$ of the random variable (r.v.) Q_z^t describing the location of MTP z at time t.

The following equation calculates the probability $\mathbb{P}\{V_z\}$ that a sensing report coming from user u_z is verified by an MTP, conditioned to the fact that user u_z is

currently in sector s_i of the sensing area:

$$\mathbb{P}\{V_z \mid U_t^z = s_i\} = 1 - \prod_{k \in \mathcal{Q}} (1 - q(i, k, t)) \tag{5.2}$$

In the above equation, it is assumed that the mobility of each MTP is independent, which is sound because it is highly unlikely MTPs would influence each other's mobility in any way. The above equation can be explained as follows. The probability that a sensing report is verified is the complement of the probability that no MTP is in the same sector as the user. The probability that a sensing report is verified, irrespective of the location of the user, can thus be computed by using the theorem of total probability, i.e.,

$$\mathbb{P}\{V_z\} = \sum_{i=1}^{n} \mathbb{P}\{V_z \mid U_t^z = s_i\} \cdot u(i, z, t) \tag{5.3}$$

The probability $\mathbb{P}\{V\}$ that on the average a sensing report will be validated can be computed as the average $\mathbb{P}\{V_z\}$ over all the users, which is

$$\mathbb{P}\{V\} = \frac{1}{|\mathcal{U}|} \sum_{z=1}^{|\mathcal{U}|} \mathbb{P}\{V_z\} \tag{5.4}$$

Example. Figure 5.4 shows two sensing areas (S_1 and S_2) divided into the same number $n = 8$ of sectors. A total of $m = 5$ MTPs are present. For simplicity, this example assumes that the mobilities of users and MTPs follow the same distribution and that all users follow the same distribution.

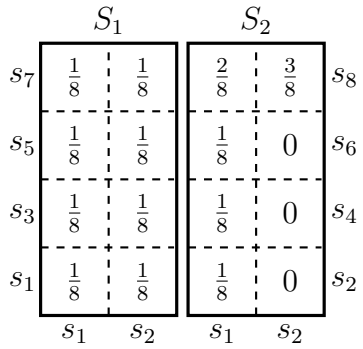


Figure 5.4: Example to illustrate the computation of $\mathbb{P}\{V\}$.

Let ℓ_i^j be defined as the probability that an MTP will be in sensing area j and sector i. The corresponding mobility distributions ℓ_i^1 and ℓ_i^2 are given as: $\ell_i^1 = 1/8$ for $1 \leq i \leq 8$, while

$$\ell_i^2 = \begin{cases} \frac{1}{8} & i = 1, 3, 5 \\ \frac{2}{8} & i = 7 \\ \frac{3}{8} & i = 8 \\ 0 & i = 2, 4, 6 \end{cases} \tag{5.5}$$

To calculate $\mathbb{P}\{V\}$ for both sensing areas, $\mathbb{P}\{V \mid U = s_i\}$ has to be computed for each s_i:

- $S_1 : \mathbb{P}\{V \mid U = s_i\} = 1 - (1 - 1/8)^5 = 0.49$ for every i, since ℓ_i is equal for each sector. Therefore, $\mathbb{P}\{V\} = 1/8 \cdot 8 \cdot 0.49 = 0.49$.
- $S_2 : \mathbb{P}\{V \mid U = s_1\} = \mathbb{P}\{V \mid L = s_3\} = \mathbb{P}\{V \mid U = s_5\} = 1 - (1 - 1/8)^5 = 0.49$. $\mathbb{P}\{V \mid U = s_7\} = 1 - (1 - 2/8)^5 = 0.76$, $\mathbb{P}\{V \mid U = s_8\} = 1 - (1 - 3/8)^5 = 0.90$, $\mathbb{P}\{V \mid U = s_2\} = \mathbb{P}\{V \mid U = s_4\} = \mathbb{P}\{V \mid U = s_6\} = 0$. Therefore, $\mathbb{P}\{V\} = 3/8 \cdot 0.49 + 2/8 \cdot 0.76 + 3/8 \cdot 0.90 = 0.71$.

The above example suggests that the likelihood that some sectors will be more occupied than others significantly impacts on the report validation probability. Indeed, if the mobility of MTPs and participants is more concentrated, the validation probability will increase with respect to the case when the mobility is uniform.

5.4.2 Likelihood Estimation Algorithm

Estimating the mobility distributions u and q is paramount to compute $\mathbb{P}\{V\}$ and, therefore, provide a cost-efficient solution to the MTP Optimization Problem. In cases where information about the mobility of users and MTPs is available, for example, if mobility traces of MTPs and users are available, an exact computation of u and q may be used. However, prior information about MTP and user mobility may not always be available.

This work proposes a heuristic, named *Likelihood Estimation Algorithm* (LEA), to provide an estimate on the mobility of users and MTPs, just knowing the sensing

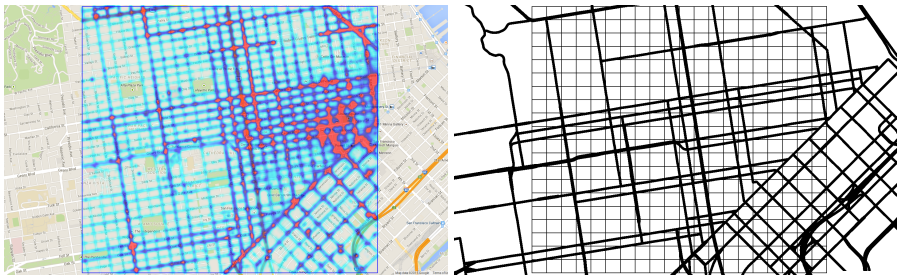


Figure 5.5: (a) Heatmap of mobility traces vs. (b) Arterial roads.

area location. Such heuristic is based on the following rationale: the SC systems considered are deployed in cities, or close to urban areas. This implies that the mobility of users and MTPs will be likely to be almost restricted to the main arterial roads of the sensing areas, or anyway the zones/roads with the greater amount of traffic (both pedestrian and vehicular). By restricting the possible area of movement of the MTPs and users, it is possible to reduce the randomness of their movement, and therefore, provide a better estimate on the likelihood of sectors.

To demonstrate this point, Figure 5.5(a) shows the heatmap of the mobility traces[1] of taxi cabs in a section of Downtown San Francisco, where the intensity of the color indicates the popularity of the place. As the figure points out, the mobility of taxi cabs is definitely not uniform, and mostly concentrated on a few popular places. Furthermore, Figure 5.5(b) shows the main arterial roads provided by Google Maps APIs[2]. From this figure, it emerges that the roads point out (with some degree of approximation) the most popular places as shown in the heatmap of Figure 5.5(a).

The proposed Likelihood Estimation Algorithm works as follows. Let M be the map of the sensing area, divided into n sectors as required by the application, where $\mathcal{S} = \{s_1, \cdots, s_n\}$ is the set of sectors. Then, information about the most popular places (which may be roads/squares/buildings) and the geographical constraints of the sensing area is acquired. By using Google Maps APIs[3], it is possible to highlight

[1]Published in (Piorkowski et al., 2009), available at `http://www.crawdad.org`

[2]APIs publicly available at `https://developers.google.com/maps/documentation/javascript/styling`

[3]Other approaches, such as Open Street Maps (`https://www.openstreetmap.org`), could be also used for such purpose.

Algorithm 5.1 Likelihood Estimation Algorithm (LEA)

Input: M, map of the sensing area
Output: $\tilde{\ell}$, approximate distribution of mobility
1: $\mathcal{S} \leftarrow$ set of sectors $s_1 \dots s_n$
2: $I \leftarrow$ processed image with most popular areas
3: $B \leftarrow 0$ (sum of black pixels in sensing area)
4: **for** each sector $s_i \in \mathcal{S}$ **do**
5: $\quad B_i \leftarrow$ number of black pixels $\in s_i$
6: $\quad B \leftarrow B + B_i$
7: **return** $\tilde{\ell}$

the main arterial roads on a specific location area. This information is leveraged to mark such places in the map M, the background of which is further removed to get a black-and-white image of the sensing area as shown in Figure 5.5(b), where the dark areas represent the popular places.

LEA is described by the pseudocode in Algorithm 5.1. The experimental results in Section 5.5 shows that LEA is remarkably effective in approximating the mobility distribution of users in various settings, by using real-world mobility traces collected in three major cities in three different continents, namely Rome, San Francisco and Beijing.

The implicit assumptions that LEA makes are (i) the mobility of users and MTPs is stationary (i.e., does not change over time); and (ii) users and MTPs follow the same mobility distributions.

Although these are pretty strong assumptions, the experimental evaluation conducted in Section 5.5 shows that LEA provides a pretty good approximation of the likelihood of the sectors, considering that only information from a map are used. Indeed, LEA is not a fine-grained mobility estimation algorithm. Instead, it is a simple heuristic that provides, *before deployment*, an approximate information regarding the likelihood of certain sectors with respect to others. If more reliable information about the mobility is known, it could be used to complement LEA's analysis and achieve better optimization results.

5.4.3 Solving the MTP Optimization Problem

This section describes the methodology adopted by FIRST to solve the MTP Optimization Problem (MOP) defined in Section 5.3.1. In order to solve the MOP, it is necessary to compute the error probability $\mathbb{P}\{E\}$. This implies the need to derive $\mathbb{P}\{A \mid F\}$ and $\mathbb{P}\{\overline{A} \mid \overline{F}\}$, defined in Equation (5.1), as a function of $P\{V\}$.

This section provides the mathematical tools that relate the number m of MTPs to the error probability $\mathbb{P}\{E\}$ and user mobility. Equation (5.4) shows how to compute $\mathbb{P}\{V\}$ given q and u. By applying probability theory, it is possible to obtain $\mathbb{P}\{A \cap F\} = \mathbb{P}\{F\} \cdot \mathbb{P}\{\overline{V}\} \cdot \mathbb{P}\{A \mid \overline{V}\}$ and $\mathbb{P}\{A \cap \overline{F}_j\} = \mathbb{P}\{\overline{F}_j\} \cdot (\mathbb{P}\{V\} + \mathbb{P}\{\overline{V}\} \cdot \mathbb{P}\{A \mid \overline{V}\})$.

The only unknown in $\mathbb{P}\{A \cap F\}$ and $\mathbb{P}\{A \cap \overline{F}\}$ is $\mathbb{P}\{A \mid \overline{V}\}$, which is, the probability of deeming a sensing report reliable in the case it has not been validated by an MTP. Ideally, this probability should be close to 1 when the report being sent is reliable, and close to 0 when the report being sent is not reliable. To this end, FIRST leverages the knowledge provided by the reports submitted by the MTPs, and computes $\mathbb{P}\{A \mid \overline{V}\}$ as follows:

$$\mathbb{P}\{A \mid \overline{V}\} = \mathbb{P}\{V\} \cdot \mathbb{P}\{\overline{F}\} + \frac{1}{2} \cdot \mathbb{P}\{\overline{V}\} \tag{5.6}$$

This formula can be explained as follows. The first part, $\mathbb{P}\{V\} \cdot \mathbb{P}\{\overline{F}\}$, represents the "degree of belief" the system has in the users; it is higher when the user is validated most of the time ($\mathbb{P}\{V\}$ close to 1) and the reports are reliable. The second part, $1/2 \cdot \mathbb{P}\{\overline{V}\}$, represents the "degree of uncertainty" in the users; it is higher when most of the reports have not been validated. Note that, as $\mathbb{P}\{V\}$ increases, the value of $\mathbb{P}\{A \mid \overline{V}\}$ approximates to $\mathbb{P}\{\overline{F}\}$. Also, if $\mathbb{P}\{V\} = 0$ (i.e., no MTPs are present), the system deems as reliable every report with probability $1/2$ (coin tossing), since there is no reason to be more inclined to accept or reject the report if no information is available.

This work proposes the MTP Optimization Algorithm (MOA), which is based on a modified version of binary search algorithm, called Left-most Insertion Point (LMIP). More specifically, LMIP returns the left-most place (i.e., the minimum value) where $\mathbb{P}\{E\}$ can be correctly inserted (and still maintains the sorted order) in the ordered array of the errors corresponding to a particular choice of m. This

Algorithm 5.2 MTP Optimization Algorithm (MOA)

Input: $\tilde{\ell}_i$, $\mathbb{P}\{F\}$, ϵ^{max}, m^{max}
Output: m^*
1: $\epsilon^{min} \leftarrow$ `CalculateError`$(\tilde{\ell}_i, \mathbb{P}\{F\}, m^{max})$
2: **if** $\epsilon^{max} < \epsilon^{min}$ **then**
3: **return** 'infeasible'
4: **return** `LMIP`$(\tilde{\ell}_i, \mathbb{P}\{F\}, \epsilon^{max}, 0, m^{max})$

Algorithm 5.3 Left-most Insertion Point (`LMIP`)

Input: $\tilde{\ell}_i$, $\mathbb{P}\{F\}$, ϵ^{max}, i, j
Output: m^*
1: **if** $j < i$ **then**
2: **return** i
3: $mid \leftarrow \lfloor (i+j)/2 \rfloor$
4: **if** `CalculateError`$(\tilde{\ell}_i, \mathbb{P}\{F\}, mid) \leq \epsilon^{max}$ **then**
5: **return** `LMIP`$(\tilde{\ell}_i, \mathbb{P}\{F\}, \epsilon^{max}, i, mid - 1)$
6: **else**
7: **return** `LMIP`$(\tilde{\ell}_i, \mathbb{P}\{F\}, \epsilon^{max}, mid + 1, j)$

corresponds to the lower (inclusive) bound of the range of elements that are equal to the given value (if any). Note that such algorithm can be applied to solve the MTP Optimization Problem due to the fact that $\mathbb{P}\{E\}$ is a monotonically decreasing function of m.

The algorithm takes as input the approximate distribution $\tilde{\ell}_i$ provided by LEA (equal for participants and MTPs), and also $\mathbb{P}\{F\}$, the desired maximum error ϵ^{max}, and the maximum number m^{max} of MTPs available. It provides as output the optimum number m^* of MTPs to be used to achieve the desired maximum error ϵ^{max}.

In lines 1-3, Algorithm 5.2 checks, with a procedure implementing Equation 5.1), whether the minimum error ϵ^{min} obtained with the maximum number of MTPs available is greater than the desired maximum error ϵ^{max}. If this is the case, then the problem has no feasible solutions and therefore the algorithm terminates immediately. If not, the routine `LMIP` is invoked, which finds m^* by implementing the left-most insertion point algorithm.

$\mathbb{P}\{E\}$	0.29	0.17	0.11	0.07	0.05	0.03	0.02	0.02
m	1	2	3	4	5	6	7	8

Figure 5.6: Example of the `LMIP` algorithm.

As regards the time complexity of Algorithm 5.2, `LMIP` is a variation of binary search, therefore its overall complexity will be $O(x \cdot \log m^{max})$, where x is the complexity of `CalculateError`. Such complexity is $\Theta(n \cdot z)$, given it requires constant time to compute $\mathbb{P}\{E\}$ using Equation (5.1), and $n \cdot z$ iterations to compute $\mathbb{P}\{V\}$ using Equation (5.4), where n is the number of sectors and z is the number of users. Therefore, the overall time complexity of MOA is given by $O(n \cdot z \cdot \log m^{max})$.

Example. In the example of Figure 5.6, the ℓ distribution is assumed equal to ℓ_i^2 presented in Figure 5.4, $\mathbb{P}\{F\} = 0.01$, $m^{max} = 8$ and $\epsilon^{max} = 0.1$. In this case, the LMIP will return $m^* = 4$, since it is the left-most element that provides $\mathbb{P}\{E\} \leq 0.1$.

5.4.4 Practical Implementation

This section describes how the system, *after deployment*, handles the case in which a report has not been validated by MTPs (i.e., how $\mathbb{P}\{A \mid \overline{V}\}$ is actually computed). For each user u_i, the system keeps track of the number k_i of sensing reports submitted, the number k_i^v of sensing reports validated by an MTP, and the number k_i^r of reports that have been validated as reliable.

As soon as a report q is sent by user u_i, if the report has not been validated by an MTP, then the report is classified as reliable with probability

$$\mathbb{P}\{A \mid \overline{V}\} = \frac{k_i^r}{k_i} + \frac{1}{2} \cdot \left(1 - \frac{k_i^v}{k_i}\right) \tag{5.7}$$

After being classified as reliable, reports may be subsequently analyzed by additional data fusion algorithms in the higher levels of the proposed AmI architecture, so as to determine the actual status of the sensing area by combining only the information conveyed by reliable reports.

5.5 Experimental Results

This section presents the experimental results obtained by evaluating the performances of FIRST and comparing it with relevant related work. First, Section 5.5.1 reports the performance results obtained by considering an application monitoring vehicular traffic events. Then, Section 5.5.2 describes the Participatory PerCom application and discusses the results obtained during the experiment.

5.5.1 Participatory Traffic Sensing

This experiment considered mobility traces collected from the following datasets:

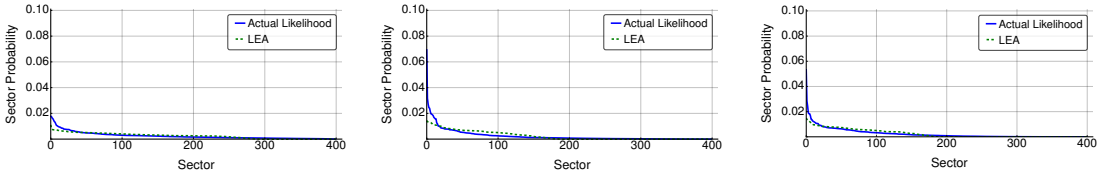
- *CRAWDAD-SanFrancisco* (Piorkowski et al., 2009): This dataset contains mobility traces of approximately 500 taxis in San Francisco, USA, collected over one month's time;
- *CRAWDAD-Rome* (Amici et al., 2014): In this dataset, 320 taxi drivers in the center of Rome were monitored during March 2014;
- *MSR-Beijing* (Yuan et al., 2013): This dataset collected by Microsoft Research Asia contains the GPS positions of 10,357 taxis in Beijing during one month.

The scenario is a traffic sensing application in which taxi cab drivers report traffic anomalies. Consistent with the example mentioned in Section 5.2, the reports are divided into 4 categories such as "Car Crash", "Road Closure", "Traffic Jam", "No Event". The sensing areas are of approximately 4×4km square areas, which characterize the downtown of cities such as San Francisco, Rome, and Beijing. In the chosen scenario, the taxi cabs report every 5 minutes information about their surroundings to the SCP. The application was implemented using the OMNeT++ simulator[4].

Evaluation of FIRST components

This section evaluates the proposed Likelihood Estimation Algorithm (LEA) and MTP Optimization Algorithm (MOA). The goal of the first set of experiments

[4]Available at `https://www.omnetpp.org`



(a) San Francisco (b) Rome (c) Beijing

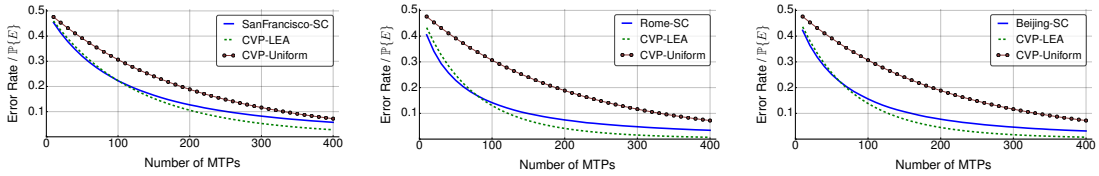
Figure 5.7: Comparison of mobility traces vs. Likelihood Estimation Algorithm.

is to test the efficacy of LEA in computing the likelihood of sectors. To obtain ground-truth information about the actual mobility of taxi cabs, the traces were processed using OMNeT++. To apply LEA, the sensing area was divided into a grid of 20×20 sectors, with sectors having the same size as a city block.

Figure 5.7 shows the distribution of the likelihood of sectors and the one obtained by LEA, respectively. More specifically, the figure shows the actual and estimated probability of a taxi to be in each sector of the sensing area. These experiments conclude that LEA approximates well the likelihood of sectors, considering the scarce information available. This result is extremely significant, as it is necessary to provide very precise estimation of the classification accuracy of FIRST as a function of the number of MTPs.

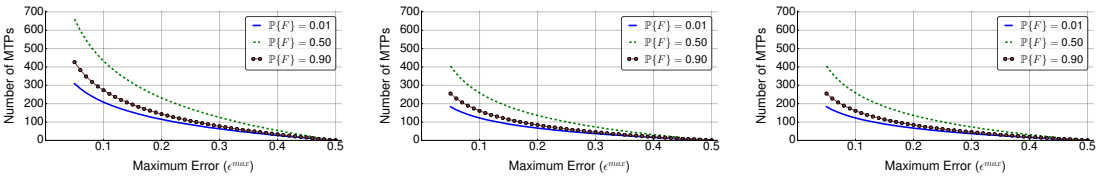
Figure 5.8 shows $\mathbb{P}\{E\}$ as a function of the number of MTPs, calculated analytically by the Computation of Validation Probability (CVP) component of FIRST. For comparison purposes, CVP was evaluated by providing as input (i) the distribution computed by LEA as applied to each considered sensing area (*CVP-LEA*, represented by a dashed line), and (ii) the uniform mobility distribution (*CVP-Uniform*, represented by a dotted continuous line) as a baseline approximation. Such analytical results are compared with the experiments using the traffic datasets.

As shown in Figure 5.8, in all three scenarios, *CVP-LEA* computes $\mathbb{P}\{E\}$ with remarkable precision. In particular, the maximum difference obtained is 3.47%, achieved in the Rome setting. Furthermore, Figure 5.8 shows that the accurate estimation of the mobility provided by LEA translates into an improved prediction accuracy of CVP w.r.t. the uniform distribution, as *CVP-Uniform* yields a maximum difference of 17.02% in the case of Rome. Finally, the figure highlights



(a) San Francisco (b) Rome (c) Beijing

Figure 5.8: Number of MTPs vs error rate / $\mathbb{P}\{E\}$.



(a) San Francisco (b) Rome (c) Beijing

Figure 5.9: Results of the MTP Optimization Algorithm.

that San Francisco requires the largest number of MTPs to achieve a specified maximum error probability.

In Figure 5.9, the MTP Optimization Algorithm (MOA) is applied to analyze the number of MTPs that are necessary to provide maximum desired error probability ϵ^{max}. In this experiment, users sends unreliable reports with three probability values $\mathbb{P}\{F\} = 0.01$, 0.5 and 0.9. The figure confirms that San Francisco requires the highest number of MTPs to achieve given ϵ^{max}. These results also highlight that FIRST is remarkably effective in achieving high accuracy with a low number of MTPs. More specifically, it provides on the average 85% of accuracy with a density of about 32% MTPs per sector in case of Rome and Beijing, and 55% in the case of San Francisco.

Note that higher accuracy values require in general a significant number of MTPs, especially when the behavior of participants becomes hardly predictable (i.e., $\mathbb{P}\{F\} = 0.5$) and the mobility is highly entropic (i.e., in San Francisco). In these cases, only a high number of MTPs can guarantee that a sufficient number of reports are validated and therefore, the desired accuracy may be provided.

Indeed, somehow surprisingly, Figure 5.9 also shows that fewer MTPs are needed when $\mathbb{P}\{F\} = 0.9$ than when $\mathbb{P}\{F\} = 0.5$. Intuitively, this is due to the

Table 5.1: Experimental parameters.

Parameter	Value
Experiment length	240mins
Timestep length	5mins
Location	Rome
Number of users	1000
$\mathbb{P}\{F\}$ for non attackers	0.01
a_r (FIDES)	0.7
a_u (FIDES)	0.9
Initial reputation (FIDES)	0.5
Reputation threshold (FIDES)	0.75
Initial weights (FIDES)	[1, 0, 0]
λ_s (Huang)	0.7
λ_p (Huang)	0.8
a (Huang)	1
b (Huang)	-2.5
c (Huang)	-0.85
Initial reputation (Huang)	0.5
Reputation threshold (Huang)	0.5
Number of MTPs (FIRST, FIDES)	400
Number of attackers	500
Attackers $\mathbb{P}\{F\}$	0.8
On-off steps	(10, 10)
Collusion groups	3

fact that, when participants send reports randomly, it is more difficult to understand their reliability. On the other hand, when their behavior is more "regular" (i.e., consistent over time) it is easier to evaluate their reliability.

Evaluation of attack resiliency

Based on the behavior models defined in Section 5.2, this work takes into account the following security attacks, which were defined in other domains and recently cast in the context of smartphone crowdsensing (Mousa et al., 2015). For simplicity, hereafter the term *attacker* will be used for both malicious and unreliable users, and the terms *threat* and *attack* will be used interchangeably.

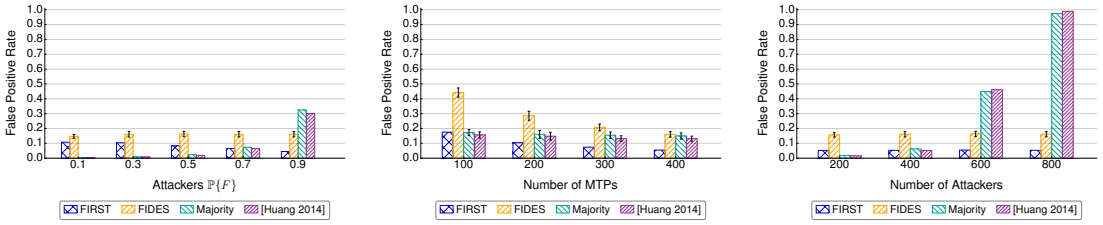
1. *Corruption attack*: for each sensing report, the attacker sends unreliable data with probability p and correct data with probability $1 - p$. This attack can be carried out by unreliable and malicious users alike.

2. *On-off attack*: in this attack, the malicious user alternates between normal and abnormal behaviors to conceal his/her maliciousness. Specifically, the adversary *periodically* sends n reliable reports and then m unreliable reports, and then repeats the process. This attack is extremely easy to carry out but also extremely challenging to detect and contrast (Perrone and Nelson, 2006; Chae et al., 2015; Alzaid et al., 2012).

3. *Collusion attack*: in this attack, two or more malicious participants coordinate their behavior in order to provide the same (unreliable) information to the SCP (Marforio et al., 2011; He et al., 2015). The malicious behavior may also include GPS location spoofing, so as to mislead the SCP into assuming colluding participants are nearby (Restuccia and Das, 2014).

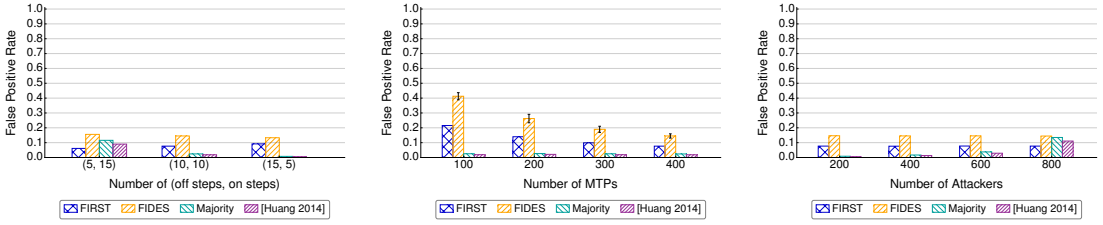
For comparison reasons, the FIDES framework (Restuccia and Das, 2014), and the reputation-based framework proposed in (Huang et al., 2014), hereafter referred to as [Huang 2014], have been implemented. FIDES uses a modified version of Jøsang's trust model to update the reputation of users. This framework inherits from Jøsang's trust model a strong sensitivity to parameter tuning. On the other hand, [Huang 2014] proposes an approach which is a improved variation of majority vote, and its performance also depends on the choice of parameter setting (Gompertz's function's, among others). To obtain baseline performance, a pure majority vote scheme was also implemented. The parameters used in the comparison experiments (as proposed in the respective papers) are reported in Table 5.1. Confidence intervals at 95% are shown only when above 1% of the value.

Figure 5.10 reports the false positive rate (percentage of false reports accepted w.r.t. the total number of reports accepted) obtained by the frameworks when subject to a corruption attack, as a function of the (constant) attack probability, number of MTPs (applicable only to FIDES and FIRST), and number of attackers. Figure 5.10a and Figure 5.10b show that the performance of Majority and [Huang



(a) (b) (c)

Figure 5.10: Corruption attack: False Positive Rate vs. $\mathbb{P}\{F\}$, MTPs, and attackers.



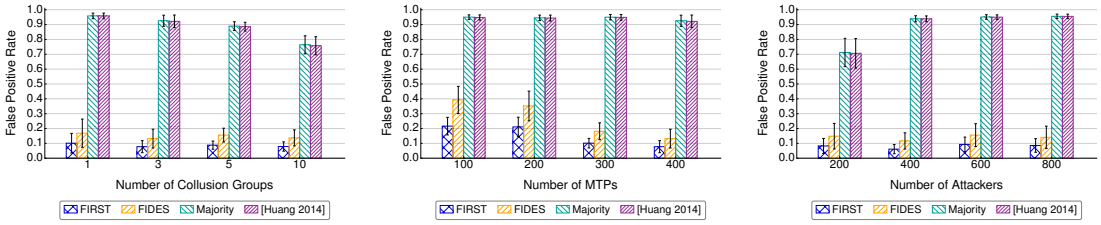
(a) (b) (c)

Figure 5.11: On-off attack: False Positive Rate vs. on-off steps, MTPs, and attackers.

2014] decreases as the number of false reports and attackers increases. This is reasonable, as both schemes are based on data aggregation and therefore not resilient to large number of malicious users and/or unreliable reports.

Figure 5.11 shows the results obtained under the On-off attack by all the considered schemes. As expected, the performance of FIRST is slightly affected by this attack, especially when the percentage of ON steps is less than the OFF one. This is because, the less the ON steps are, the harder it is for FIRST to decrease the accept probability of malicious users. However, FIRST is able to achieve a False Positive Rate of about 10% in the worst case of ON=5. On the other side, [Huang 2014] and Majority are instead more affected when the ON step is greater than the OFF, as it is more likely for them to misclassify sensing reports when the percentage of unreliable reports/number of attackers is higher.

Figure 5.12 shows the results obtained by running the Collusion attack. The experiment has been implemented as follows. There are k collusion groups. An



(a) (b) (c)

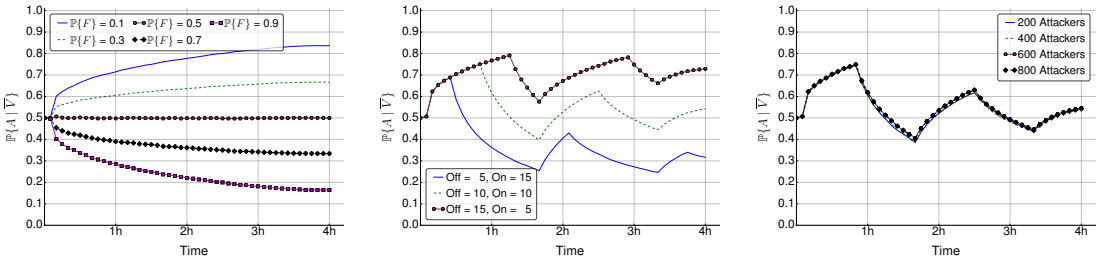
Figure 5.12: Collusion attack: False Positive Rate vs. $\mathbb{P}\{F\}$, MTPs, and attackers.

attacker belonging to the k-th group coordinates with the other attackers belonging to the same group by implementing together an On-off attack. In such attack, during the ON phase the attackers send false reports pertaining to a chosen sector, the same for every user in the k-th group. The results conclude that [Huang 2014] and Majority are severely affected by this attack, while FIRST tolerates well this attack by keeping the False Positive Rate below 10% by using 400 MTPS, regardless of the number of attackers and collusion groups considered. This is because FIRST uses MTPs to validate data and does not rely on data aggregation.

Interestingly enough, [Huang 2014] and Majority perform slightly well when the collusion groups are more. This is explained by considering that when the collusion groups are more, less attackers will belong to the same group, and so it is more likely that a scheme based on aggregation may perform better.

Finally, Figure 5.13 reports the probability $\mathbb{P}\{A|\overline{V}\}$ of FIRST (i.e., the probability that a report will be accepted when not validated) as a function of time, in all the considered attacks. In the Corruption attack, as expected $\mathbb{P}\{A|\overline{V}\}$ converges to the $\mathbb{P}\{\overline{F}\}$ probability of the attackers. In the On-off attack, FIRST reacts by decreasing the $\mathbb{P}\{A|\overline{V}\}$ probability and increasing it in the OFF phases. The same behavior is also experimented in the Collusion attack, but in this case, the performance is not affected by the number of attackers, as explained above.

As described in Section 5.4.3, $\mathbb{P}\{A|V\}$ is equal to $P\{\overline{F}\}$, because when reports are verified the probability of misclassification is zero; on the other hand, when the reports are not validated, ideally $\mathbb{P}\{A|\overline{V}\}$ should also tend to $\mathbb{P}\{\overline{F}\}$, and Figure 5.13a shows that FIRST achieves such goal.



(a) (b) (c)

Figure 5.13: Acceptance probability $\mathbb{P}\{A|\overline{V}\}$ in corruption, on-off, and collusion attacks.

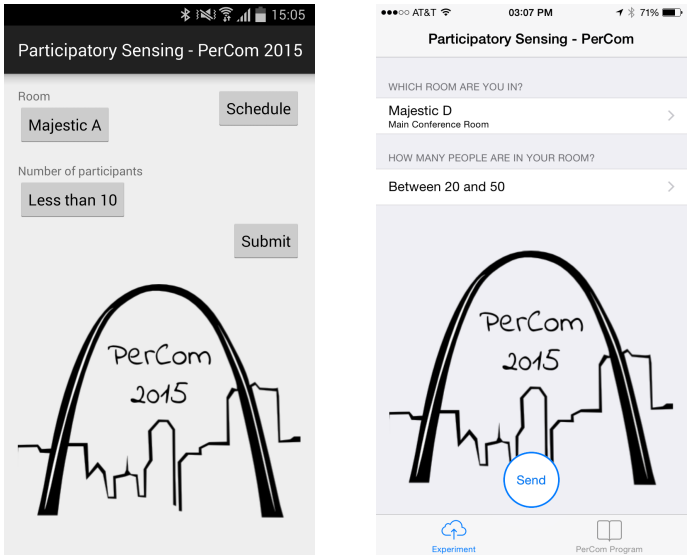
5.5.2 Participatory PerCom

In addition to the traffic sensing application scenario described above, the performance of FIRST has been evaluated by implementing an SC system designed to monitor the attendance of participants at various events during the IEEE PerCom 2015 conference held in St. Louis, Missouri, USA. In such a system, the voluntary participants were asked to regularly submit (i) the conference room they were currently in, and (ii) the (approximate) number of participants in that room. The goal of the experiment was to evaluate the accuracy of FIRST in classifying sensing reports sent by participants in a practical scenario.

Experimental setup

The server-side of the SC system handling the storage of sensing reports was implemented by using a dedicated virtual machine on *Amazon Web Services*. Figure 5.14 shows the screenshots of the Android and iOS apps distributed to the participants. The apps provided a simple interface for the participants to report the room they were in (8 choices, from 'A' to 'H'), and the approximate number of people in that room (5 choices, i.e., 'Less than 10', 'Between 10 and 20', 'Between 20 and 50', 'Between 50 and 100', and 'More than 100').

In order to recruit participants, the conference and workshop attendees were asked if they were willing to install the app and participate in the experimental study. This way, 57 participants attending the entire conference and workshops were recruited, which is significant considering that participants were not incen-



(a) Android app (b) iOS app

Figure 5.14: Screenshots of the Android and iOS apps.

tivized with any kind of reward. To better test the resilience of the system, about 25% of users were asked to purposefully send unreliable reports once in a while, besides the correct ones.

In order to acquire ground-truth information about the location of participants, 20 *Gimbal*™ beacon devices[5] were used, deployed as shown in Figure 5.15. The beacons emitted periodically *Bluetooth* packets that were received by the SC app. Whenever a user sent a report, the location of the nearest beacon was also automatically included in the report by the SC app. This way, it was possible to acquire ground-truth information on user location. Note that FIRST does *not* use such information to verify the reliability of the report. The purpose of such information was to calculate the accuracy of FIRST after the experiment was over, as explained below.

To acquire ground-truth information about the number of people in each room, three people voluntarily acted as MTPs and sent every 5 minutes the actual number of people in each conference room. To evaluate the impact of the MTP reporting

[5]Available at `http://www.gimbal.com`

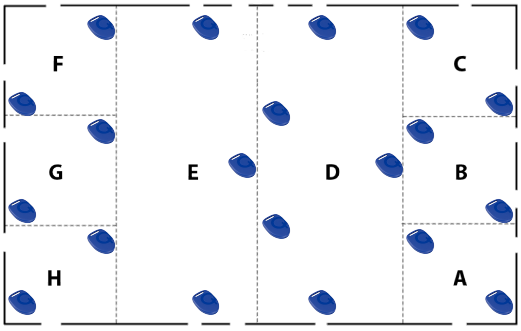


Figure 5.15: Position of the Bluetooth beacons.

interval on the performance of the system, 4 different intervals have been considered (10, 15, 20, and 25 minutes).

During the experiment, the number of people attending a particular event was almost constant during a 10-minute time window. Therefore, each MTP report was used to validate all the reports sent in the following 10-minute time frame. More specifically, a user report was validated as *reliable* if (i) an MTP report r was sent during the 10-minute time frame before the user report was received, *and* (ii) the reported number of people in that room was in the same range as the one sent by the MTP in r. If the number of people in the room reported by the user mismatched the information acquired by the MTP, the report was considered *unreliable*. Otherwise, if no MTP report was available during the previous 10-minute time window, Equation (5.7) was used to decide whether to consider the report as reliable, as explained in Section 5.4.4. After the experiment, the ground-truth information provided by the Bluetooth beacons and the MTPs was used to calculate the classification accuracy of FIRST.

Experimental results

Figure 5.16 illustrates the accuracy of the considered approaches as a function of the MTP reporting intervals implemented in the experiments. Reports sent by MTPs were used by FIRST and FIDES to update the reputation of users and validate their reports, but are not included in the results, which measure the classification accuracy of the compared systems with regard to reports sent by normal users.

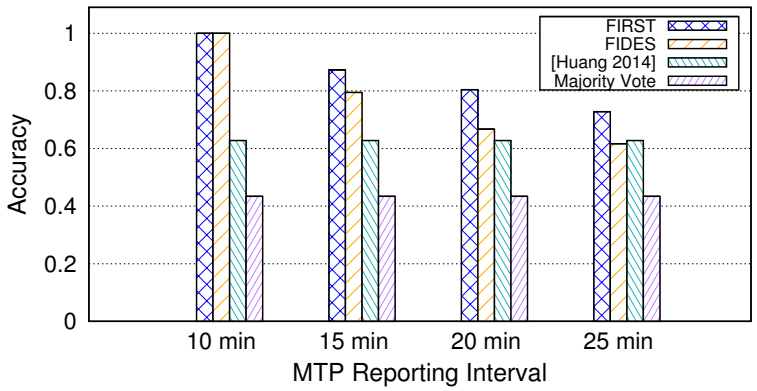


Figure 5.16: Comparison of FIRST vs. FIDES, Majority Vote and [Huang 2014].

These results conclude that FIRST outperforms existing approaches as far as classification accuracy is concerned. In particular, FIRST achieves on the average an accuracy of 80.16%, as compared to FIDES, [Huang 2014] and majority vote which achieve 69.20%, 62.76%, and 43.38%, respectively. The results can be explained as follows. When the MTP reporting interval is 10 minutes, both FIDES and FIRST achieve accuracy of 100%, because each report is validated by MTPs. As the reporting interval increases, FIDES performs worse than FIRST due to the challenge in finding a parameter setting which achieves good performance in all scenarios. In contrast, FIRST does *not* require any parameter setting to be implemented, and it is able to achieve high accuracy in all the considered scenarios.

Note that the majority vote and [Huang 2014] schemes do not rely on MTPs. As a result, the accuracy achieved by majority vote and [Huang 2014] in Figure 5.16 does not depend on the MTP reporting interval. As far as performance is concerned, Figure 5.16 concludes that such approaches do not obtain accuracy values close to FIRST and FIDES.

Conclusions

This work described the design and implementation of a multi-tier cognitive architecture for a robust and resilient AmI system, capable of self-configuring and self-managing its sensory infrastructure. The entire hardware and software system is represented in increasing levels of abstraction, so as to describe the environment and interactions with the system through high-level symbolic concepts. The system is capable of focusing on those aspects of the external environment which are considered the most important at different times, depending on the context, by managing its resources intelligently and minimizing energy consumption.

To achieve such goals, the system relies on heterogeneous data collected by numerous wireless sensors pervasively deployed in the environment. To this end, one of the main modules of the proposed architecture is a context-aware self-optimizing adaptive system for sensory data fusion. The inference subsystem leverages Dynamic Bayesian Networks for inferring the state of the world, exploiting contextual information to increase reasoning accuracy.

Moreover, a self-optimization module chooses dynamically the subset of sensors to use, finding an optimal trade-off to maximize sensing accuracy and minimize energy consumption. The experimental results have confirmed that selecting the right combination of context information is fundamental to maximize the inference accuracy, especially when only few sensors are available, and that exploiting the best context information set greatly improves the accuracy of activity recognition systems. Furthermore, the results demonstrated that the proposed adaptive system performs better than static systems, achieving substantial energy savings compared to a system that statically uses all the available sensors, with only a small increase in inference uncertainty.

The proposed architecture is also enriched by a filtering module to accurately classify reliable and unreliable sensing reports in a scenario with highly uncertain information quality.

In the current study, both training and test data are collected from the same environment. As part of future development, the generalization potential of the proposed approach should be evaluated by considering training and test data coming from different smart homes or offices. Furthermore, it would be interesting to test the system in a real scenario with heterogeneous sensors, including data coming from wearable devices such as smart watches.

Bibliography

H. Alzaid, E. Foo, J. G. Nieto, and E. Ahmed. Mitigating On-Off Attacks in Reputation-based Secure Data Aggregation for Wireless Sensor Networks. *Security and Communication Networks*, 5(2):125–144, 2012.

R. Amici, M. Bonola, L. Bracciale, A. Rabuffi, P. Loreti, and G. Bianchi. Performance assessment of an epidemic protocol in VANET using real traces. *Procedia Computer Science*, 40:92–99, 2014.

F. Amigoni, N. Gatti, C. Pinciroli, and M. Roveri. What planner for Ambient Intelligence applications? *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 35(1):7–21, 2004.

P. Andry, P. Gaussier, J. Nadel, and B. Hirsbrunner. Learning invariant sensorimotor behaviors: A developmental approach to imitation mechanisms. *Adaptive Behavior*, 12(2):117–140, 2004.

J. Bermejo-Alonso, R. Sanz, M. Rodríguez, and C. Hernández. Ontology-based engineering of autonomous systems. In *Proceedings of the 6th International Conference on Autonomic and Autonomous Systems (ICAS)*, pages 47–51. IEEE, 2010.

G. L. Chadderdon. Assessing machine volition: An ordinal scale for rating artificial and natural systems. *Adaptive Behavior*, 16(4):246–263, 2008.

Y. Chae, L. C. DiPippo, and Y. L. Sun. Trust Management for Defending On-Off Attacks. *IEEE Transactions on Parallel and Distributed Systems*, 26(4): 1178–1191, April 2015.

K. Cho, I. Hwang, S. Kang, B. Kim, J. Lee, S. Lee, S. Park, J. Song, and Y. Rhee. HiCon: a hierarchical context monitoring and composition framework for next-generation context-aware services. *IEEE Network*, 22(4):34–42, 2008.

D. Cook. Learning setting-generalized activity models for smart spaces. *IEEE Intelligent Systems*, 2010(99):1, 2010.

D. Cook and S. K. Das. *Smart environments: technology, protocols and applications*, volume 43. John Wiley & Sons, 2004.

D. Cook, J. Augusto, and V. Jakkula. Ambient Intelligence: Technologies, Applications, and Opportunities. *Pervasive and Mobile Computing*, 5(4):277–298, 2009.

P. De Mil, T. Allemeersch, I. Moerman, P. Demeester, and W. De Kimpe. A Scalable Low-Power WSAN Solution for Large-Scale Building Automation. In *IEEE International Conference on Communications (ICC '08)*, pages 3130–3135, 2008.

A. De Paola and L. Gagliano. Design of an adaptive Bayesian system for sensor data fusion. In *Advances onto the Internet of Things*, pages 61–76. Springer, 2014.

A. De Paola, S. Gaglio, G. Lo Re, and M. Ortolani. Multi-sensor fusion through adaptive Bayesian networks. In *AI*IA 2011: Artificial Intelligence Around Man and Beyond*, pages 360–371. Springer, 2011.

A. De Paola, S. Gaglio, G. Lo Re, and M. Ortolani. Sensor9k: A testbed for designing and experimenting with WSN-based Ambient Intelligence applications. *Pervasive and Mobile Computing*, 8(3):448–466, 2012a.

A. De Paola, M. La Cascia, G. Lo Re, M. Morana, and M. Ortolani. User detection through multi-sensor fusion in an AmI scenario. In *Proceedings of the 15th International Conference on Information Fusion (FUSION)*, pages 2502–2509. IEEE, 2012b.

A. De Paola, M. La Cascia, G. Lo Re, M. Morana, and M. Ortolani. Mimicking biological mechanisms for sensory information fusion. *Biologically Inspired Cognitive Architectures*, 3:27–38, 2013.

A. De Paola, M. Ortolani, G. Lo Re, G. Anastasi, and S. K. Das. Intelligent Management Systems for Energy Efficiency in Buildings: A Survey. *ACM Computing Surveys*, 47(1):1–38, 2014.

A. De Paola, S. Gaglio, G. Lo Re, F. Milazzo, and M. Ortolani. Adaptive distributed outlier detection for WSNs. *IEEE Transactions on Cybernetics*, 45(5): 888–899, 2015.

A. K. Dey, G. D. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16(2):97–166, 2001.

E. D'Hondt, M. Stevens, and A. Jacobs. Participatory noise mapping works! An evaluation of participatory sensing as an alternative to standard techniques for environmental monitoring. *Pervasive and Mobile Computing*, 9(5):681–694, 2013. Special Issue on Pervasive Urban Applications.

M. Diehl and Y. Y. Haimes. Influence diagrams with multiple objectives and tradeoff analysis. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 34(3):293–304, 2004.

F. Doctor, H. Hagras, and V. Callaghan. A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 35(1):55–65, 2005.

Y. F. Dong, S. Kanhere, C. T. Chou, and R. P. Liu. Automatic image capturing and processing for petrolwatch. In *Proceedings of the 17th IEEE International Conference on Networks*, ICON '11, pages 236–240, Washington, DC, USA, 2011. IEEE.

A. Dua, N. Bulusu, W.-C. Feng, and W. Hu. Towards trustworthy participatory sensing. In *Proceedings of the 4th USENIX Conference on Hot Topics in Security*, HotSec '09, pages 1–6. USENIX, 2009.

K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J.-C. Burgelman. *Scenarios for Ambient Intelligence in 2010*. Office for official publications of the European Communities, 2001.

M. D. Erbas, A. F. Winfield, and L. Bull. Embodied imitation-enhanced reinforcement learning in multi-agent systems. *Adaptive Behavior*, 22(1):31–50, 2014.

G. Fabeck and R. Mathar. Kernel-based learning of decision fusion in Wireless Sensor Networks. In *Proceedings of the 11th International Conference on Information Fusion*, pages 1–7. IEEE, 2008.

J. F. Ferreira, M. Castelo-Branco, and J. Dias. A hierarchical bayesian framework for multimodal active perception. *Adaptive Behavior*, 20(3):172–190, 2012.

E. Friedman. *Jess in action: rule-based systems in Java*. Manning Publications Co., 2003.

L. Gao, A. Bourke, and J. Nelson. Evaluation of accelerometer based multi-sensor versus single-sensor activity recognition systems. *Medical Engineering & Physics*, 36(6):779–785, 2014.

P. Gilbert, J. Jung, K. Lee, H. Qin, D. Sharkey, A. Sheth, and L. P. Cox. Youprove: Authenticity and fidelity in mobile sensing. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 176–189. ACM, 2011.

T. Gu, X. Wang, H. Pung, and D. Zhang. An ontology-based context model in intelligent environments. In *Proceedings of Communication Networks and Distributed Systems Modeling and Simulation conference*, pages 270–275, 2004.

H. Hagras. Embedding computational intelligence in pervasive spaces. *IEEE Pervasive Computing*, 6(3):85–89, 2007.

D. Hall and S. McMullen. *Mathematical techniques in multisensor data fusion*. Artech House, 2004.

D. He, S. Chan, and M. Guizani. User privacy and data trustworthiness in mobile crowd sensing. *IEEE Wireless Communications*, 22(1):28–34, February 2015.

C. Hernández, I. López, and R. Sanz. The operative mind: a functional, computational and modeling approach to Machine Consciousness. *International Journal of Machine Consciousness*, 1(01):83–98, 2009.

I. Horrocks, P. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A semantic web rule language combining OWL and RuleML. *W3C Member Submission*, 2004.

M. A. Hossain, P. K. Atrey, and A. El Saddik. Learning multisensor confidence using a reward-and-punishment mechanism. *IEEE Transactions on Instrumentation and Measurement*, 58(5):1525–1534, 2009.

K. L. Huang, S. S. Kanhere, and W. Hu. On the need for a reputation system in mobile phone based sensing. *Ad Hoc Networks*, 12:130–149, 2014.

M. C. Huebscher and J. A. McCann. Adaptive middleware for context-aware applications in smart-homes. In *Proceedings of the 2nd Workshop on Middleware for Pervasive and Ad-Hoc Computing*, pages 111–116. ACM, 2004.

M. Jacyno, S. Bullock, N. Geard, T. R. Payne, and M. Luck. Self-organizing agent communities for autonomic resource management. *Adaptive Behavior*, 21(1): 3–28, 2013.

Y. Jiang, H. Qiu, M. McCartney, W. G. Halfond, F. Bai, D. Grimm, and R. Govindan. CARLOG: a platform for flexible and efficient automotive sensing. In *Proceedings of the 12th ACM Conference on Embedded Networked Sensor Systems (SenSys)*, pages 221–235. ACM, 2014.

S. Kang, J. Lee, H. Jang, Y. Lee, S. Park, and J. Song. A scalable and energy-efficient context monitoring framework for mobile personal sensor networks. *IEEE Transactions on Mobile Computing*, 9(5):686–702, 2010.

K. Kawamura, W. Dodd, P. Ratanaswasd, and R. Gutierrez. Development of a robot with a sense of self. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2005)*, pages 211–217. IEEE, 2005.

J. Kephart and D. Chess. The vision of autonomic computing. *Computer*, 36(1): 41–50, 2003.

B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi. Multisensor data fusion: a review of the state-of-the-art. *Information Fusion*, 14(1):28–44, 2013.

A. Khan, S. K. A. Imon, and S. K. Das. A novel localization and coverage framework for real-time participatory urban monitoring. *Pervasive and Mobile Computing*, 23:122–138, 2015.

D. E. Knuth. *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms*. Addison-Wesley, 1997.

D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

N. C. Krishnan and D. Cook. Activity recognition on streaming sensor data. *Pervasive and Mobile Computing*, 2012.

S. Kullback and R. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

N. Kushwaha, M. Kim, D. Kim, and W. Cho. An intelligent agent for ubiquitous computing environments: smart home UT-AGENT. In *Proceedings of the 2nd IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems*, pages 157–159, Piscataway, NJ, USA, 2004. IEEE Press.

J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

A. Lalomia, G. Lo Re, and M. Ortolani. A hybrid framework for soft real-time WSN simulation. In *Proceedings of the 13th IEEE/ACM International Symposium Distributed Simulation and Real Time Applications (DS-RT)*, pages 201–207, 2009.

Y. Liu, B. Guo, Y. Wang, W. Wu, Z. Yu, and D. Zhang. Taskme: Multi-task allocation in mobile crowd sensing. In *Proceedings of the 2016 ACM International*

Joint Conference on Pervasive and Ubiquitous Computing, UbiComp '16, pages 403–414. ACM, 2016.

T. Luo, S. S. Kanhere, and H.-P. Tan. SEW-ing a simple endorsement web to incentivize trustworthy participatory sensing. In *Proceedings of the 11th IEEE Annual Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, SECON '14, pages 636–644. IEEE, 2014.

C. Marforio, A. Francillon, S. Capkun, S. Capkun, and S. Capkun. *Application Collusion Attack on the Permission-based Security Model and its Implications for Modern Smartphone Systems*. Technical Report, Department of Computer Science, ETH Zurich Zürich, Switzerland, 2011.

E. A. McCartney, K. J. Craun, E. Korris, D. A. Brostuen, and L. R. Moore. Crowdsourcing the national map. *Cartography and Geographic Information Science*, 42(sup1):54–57, 2015.

R. J. Meinhold and N. D. Singpurwalla. Understanding the Kalman filter. *The American Statistician*, 37(2):123–127, 1983.

H. Mousa, S. B. Mokhtar, O. Hasan, O. Younes, M. Hadhoud, and L. Brunie. Trust Management and Reputation Systems in Mobile Participatory Sensing Applications: A Survey. *Computer Networks*, 90:49 – 73, 2015.

M. Mozer. The Neural Network House: An Environment that Adapts to its Inhabitants. In *Proceedings of the Intelligent Environments AAAI Spring Symposium*, pages 110–114, Palo Alto, CA, USA, 1998.

K. P. Murphy. *Dynamic Bayesian Networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

S. Nath. ACE: exploiting correlation for energy-efficient and continuous context sensing. In *Proceedings of the 10th ACM International Conference on Mobile systems, applications, and services (MobiSys)*, pages 29–42. ACM, 2012.

S. Nawaz, C. Efstratiou, and C. Mascolo. ParkSense: A smartphone based sensing system for on-street parking. In *Proceedings of the 19th ACM Annual Inter-*

national Conference on Mobile Computing & Networking, MobiCom '13, pages 75–86. ACM, 2013.

V. Nimier. Introducing contextual information in multisensor tracking algorithms. In *Advances in Intelligent Computing (IPMU)*, pages 595–604. Springer, 1995.

A. Padovitz, S. W. Loke, A. Zaslavsky, B. Burg, and C. Bartolini. An approach to data fusion for context awareness. In *Modeling and Using Context*, pages 353–367. Springer, 2005.

M. Paxton and S. Benford. Experiences of participatory sensing in the wild. In *Proceedings of the 11th ACM International Conference on Ubiquitous Computing*, UbiComp '09, pages 265–274. ACM, 2009.

C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context aware computing for the Internet of Things: A survey. *Communications Surveys & Tutorials, IEEE*, 16(1):414–454, 2014.

L. F. Perrone and S. C. Nelson. A Study of On-Off Attack Models for Wireless Ad Hoc Networks. In *Proceedings of the 1st Workshop on Operator-Assisted (Wireless Mesh) Community Networks*, pages 1–10, Sept 2006.

M. Piorkowski, N. Sarafijanovoc-Djukic, and M. Grossglauser. A parsimonious model of mobile partitioned networks with clustering. In *Proceedings of the 1st International Conference on COMmunication Systems and NETworkS*, COMSNETS '09, 2009.

D. Preuveneers, J. Bergh, D. Wagelaar, A. Georges, P. Rigole, T. Clerckx, Y. Berbers, K. Coninx, V. Jonckers, and K. Bosschere. Towards an Extensible Context Ontology for Ambient Intelligence. In *Ambient Intelligence*, volume 3295 of *Lecture Notes in Computer Science*, pages 148–159. Springer Berlin Heidelberg, 2004.

A. Rahmati and L. Zhong. Context-based network estimation for energy-efficient ubiquitous wireless connectivity. *IEEE Transactions on Mobile Computing*, 10 (1):54–66, 2011.

A. Rahmati, C. Shepard, C. Tossell, L. Zhong, and P. Kortum. Practical context awareness: Measuring and utilizing the context dependency of mobile usage. *IEEE Transactions on Mobile Computing*, 14(9):1932–1946, 2015.

A. Ram, G. Boone, R. Arkin, and M. Pearce. Using genetic algorithms to learn reactive control parameters for autonomous robotic navigation. *Adaptive Behavior*, 2(3):277–305, 1994.

P. Remagnino and G. Foresti. Ambient Intelligence: A New Multidisciplinary Paradigm. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 35(1):1–6, 2005.

F. Restuccia and S. K. Das. FIDES: A trust-based framework for secure user incentivization in participatory sensing. In *Proceedings of the 15th IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks*, WoWMoM '14, pages 1–10. IEEE, 2014.

P. Ribino, A. Oliveri, G. Lo Re, and S. Gaglio. A knowledge management system based on ontologies. In *International Conference on New Trends in Information and Service Science. NISS '09*, pages 1025–1033, 2009.

N. Roy, G. Pallapa, and S. K. Das. A middleware framework for ambiguous context mediation in smart healthcare application. In *Proceedings of the 3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMOB)*, pages 72–79. IEEE, 2007.

N. Roy, S. K. Das, and C. Julien. Resolving and mediating ambiguous contexts in pervasive environments. *Smart Healthcare Applications and Services: Developments and Practices*, pages 122–147, 2011.

N. Roy, S. K. Das, and C. Julien. Resource-optimized quality-assured ambiguous context mediation framework in pervasive environments. *IEEE Transactions on Mobile Computing*, 11(2):218–229, 2012.

D. Sanchez, M. Tentori, and J. Favela. Hidden Markov Models for activity recognition in Ambient Intelligence environments. In *Proceedings of the 8th Mexican*

International Conference on Current Trends in Computer Science, pages 33–40. IEEE, 2007.

R. Sanz, I. López, and C. Hernández. Self-awareness in real-time cognitive control architectures. *AI and Consciousness: Theoretical Foundations and Current Approaches*, 2007.

S. Saroiu and A. Wolman. I Am a Sensor, and I Approve This Message. In *Proceedings of the 11th Workshop on Mobile Computing Systems & Applications*, HotMobile '10, pages 37–42. ACM, 2010.

C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

J. Stover, D. Hall, and R. Gibson. A fuzzy-logic architecture for autonomous multisensor data fusion. *IEEE Transactions on Industrial Electronics*, 43(3): 403–410, Jun 1996.

E. M. Tapia, S. S. Intille, and K. Larson. *Activity recognition in the home using simple and ubiquitous sensors*. Springer, 2004.

J. A. Tatman and R. D. Shachter. Dynamic programming and influence diagrams. *IEEE Transactions on Systems, Man and Cybernetics*, 20(2):365–379, 1990.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT press, 2005.

Y. Ueyama, M. Tamai, Y. Arakawa, and K. Yasumoto. Gamification-based incentive mechanism for participatory sensing. In *2014 IEEE International Conference on Pervasive Computing and Communication Workshops*, PerCom '14 Workshops, pages 98–103, March 2014.

T. van Kasteren and B. Krose. Bayesian activity recognition in residence for elders. In *Proceedings of the 3rd IET International Conference on Intelligent Environments (IE)*, pages 209–212, 2007.

D. Wang, M. T. Amin, S. Li, T. Abdelzaher, L. Kaplan, S. Gu, C. Pan, H. Liu, C. C. Aggarwal, R. Ganti, et al. Using humans as sensors: an estimation-

theoretic perspective. In *Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, pages 35–46. IEEE Press, 2014a.

D. Wang, L. Kaplan, and T. F. Abdelzaher. Maximum likelihood analysis of conflicting observations in social sensing. *ACM Transactions on Sensor Networks (ToSN)*, 10(2):30, 2014b.

X. Wang, W. Cheng, P. Mohapatra, and T. Abdelzaher. Enabling reputation and trust in privacy-preserving mobile sensing. *IEEE Transactions on Mobile Computing*, 13(12):2777–2790, 2014c.

World Wide Web Consortium. OWL 2 Web Ontology Language Document Overview. *W3C recommendation*, 2012.

H. Xiong, D. Zhang, G. Chen, L. Wang, and V. Gauthier. Crowdtasker: Maximizing coverage quality in piggyback crowdsensing under budget constraint. In *2015 IEEE International Conference on Pervasive Computing and Communications*, PerCom '15, pages 55–62, March 2015.

D. Yang, G. Xue, X. Fang, and J. Tang. Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing. In *Proceedings of the 18th ACM Annual International Conference on Mobile Computing and Networking*, MobiCom '12, pages 173–184. ACM, 2012.

J. Yick, B. Mukherjee, and D. Ghosal. Wireless Sensor Network survey. *Computer Networks*, 52(12):2292–2330, 2008.

J. Yuan, Y. Zheng, X. Xie, and G. Sun. T-Drive: Enhancing driving directions with taxi drivers' intelligence. *IEEE Transactions on Knowledge and Data Engineering*, 25(1):220–232, Jan 2013.

O. Yurur, M. Labrador, and W. Moreno. Adaptive and energy efficient context representation framework in mobile sensing. *IEEE Transactions on Mobile Computing*, 13(8):1681–1693, 2014.

L. A. Zadeh. Fuzzy sets. *Information and control*, 8(3):338–353, 1965.

D. Zhang, H. Xiong, L. Wang, and G. Chen. Crowdrecruiter: Selecting participants for piggyback crowdsensing under probabilistic coverage constraint. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '14, pages 703–714. ACM, 2014.

M. Zhang, P. Yang, C. Tian, S. Tang, X. Gao, B. Wang, and F. Xiao. Quality-aware sensing coverage in budget-constrained mobile crowdsensing networks. *IEEE Transactions on Vehicular Technology*, 65(9):7698–7707, Sept 2016.

Y. Zhang and Q. Ji. Active and dynamic information fusion for multisensor systems with dynamic Bayesian networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 36(2):467–472, 2006.

Y. Zhang, N. Meratnia, and P. Havinga. Outlier detection techniques for Wireless Sensor Networks: a survey. *Communications Surveys & Tutorials, IEEE*, 12(2): 159–170, 2010.

Y. Zheng and P. Zheng. Multisensor image fusion using fuzzy logic for surveillance systems. In *Proceedings of the 7th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, volume 2, pages 588–592, 2010.