# Real time intelligent decision making from heterogeneous and imperfect data

Hela Sfar

université PARIS-SACLAY

NNT: 2019SACLL013

Thèse de doctorat

TELECOM SudParis

# Real time intelligent decision making from heterogeneous and imperfect data

Thèse de doctorat de l'Université Paris-Saclay
préparée à Télécom SudParis

École doctorale n°580 Science et Technologie de l'Information et de la Communication (STIC)
Spécialité de doctorat: Informatique

Thèse présentée et soutenue à Evry, 9 Juillet 2019 , 14h, par

**Hela Sfar**

Composition du Jury :

| | |
|---|---|
| Nazim Agoulmine, Professeur, Université d' EVRY val d'Essonne | Président |
| Mehdi Ammi, Professeur, Université Paris8 | Rapporteur |
| Farah Chehade, Maitre de conférences HDR, Université de Technologie de Troyes | Rapporteur |
| Gérard Dray, Professeur, IMT Mines Alès | Examinateur |
| Amel Bouzeghoub, Professeur, Télécom SudParis | Directeur de thèse |
| Jérôme Boudy, Professeur, Télécom SudParis | Encadreur de thèse |
| Yves Permantier, Ingénieur de recherche, Université d'Orléans | Invité |
| Pascal Doré, Chef de projet recherche, Legrand | Invité |
| Hassane Essafi, Coordinateur de programme, CEA | Invité |

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences chimiques : molécules, matériaux, instrumentation et biosystèmes

**Titre :** La prise de décision intelligente en temps réel à partir de données hétérogènes et imparfaites

**Mots clés :** Environnement intelligent, reconnaissance d'activité, détection d'anomalies, segmentation de données de capteurs

**Résumé :** De nos jours, l'informatique omniprésente fait face à un progrès croissant. Ce paradigme est caractérisé par de multiples capteurs intégrés dans des objets du monde physique. Le développement d'applications personnelles utilisant les données fournies par ces capteurs a conduit à la création d'environnements intelligents, conçus comme un framework de superposition avancé qui aide de manière proactive les individus dans leur vie quotidienne. Une application d'environnement intelligent collecte les données de capteurs deployés d'une façon en continu , traite ces données et les analyse avant de prendre des décisions pour exécuter des actions sur l'environnement physique. Le traitement de données en ligne consiste principalement en une segmentation des données pour les diviser en fragments. Généralement, dans la littérature, la taille des fragments est fixe. Cependant, une telle vision statique entraîne généralement des problèmes de résultats imprécis. Par conséquent, la segmentation dynamique utilisant des tailles variables de fenêtres d'observation est une question ouverte. La phase d'analyse prend en entrée un segment de données de capteurs et extrait des connaissances au moyen de processus de raisonnement ou d'extraction. La compréhension des activités quotidiennes des utilisateurs et la prévention des situations anormales sont une préoccupation croissante dans la littérature, mais la résolution de ces problèmes à l'aide de données de petite taille et imparfaites reste un problème clé. En effet, les données fournies par les capteurs sont souvent imprécises, inexactes, obsolètes, contradictoires ou tout simplement manquantes. Par conséquent, l'incertitude liée à la gestion est devenue un aspect important. De plus, il n'est pas toujours possible et trop intrusif de surveiller l'utilisateur pour obtenir une grande quantité de données sur sa routine de vie. Les gens ne sont pas souvent ouverts pour être surveillés pendant une longue période. Évidemment, lorsque les données acquises sur l'utilisateur sont suffisantes, la plupart des méthodes existantes peuvent fournir une reconnaissance précise, mais les performances baissent fortement avec de petits ensembles de données.

Dans cette thèse, nous avons principalement exploré la fertilisation croisée d'approches d'apprentissage statistique et symbolique et les contributions sont triples: (i) DataSeg, un algorithme qui tire parti à la fois de l'apprentissage non supervisé et de la représentation ontologique pour la segmentation des données. Cette combinaison choisit de manière dynamique la taille de segment pour plusieurs applications, contrairement à la plupart des méthodes existantes. De plus, contrairement aux approches de la littérature, Dataseg peut être adapté à toutes les fonctionnalités de l'application; (ii) AGACY Monitoring, un modèle hybride de reconnaissance d'activité et de gestion des incertitudes qui utilise un apprentissage supervisé, une inférence de logique possibiliste et une ontologie permettant d'extraire des connaissances utiles de petits ensembles de données; (iii) CARMA, une méthode basée sur les réseaux de Markov et les règles d'association causale pour détecter les causes d'anomalie dans un environnement intelligent afin d'éviter leur apparition. En extrayant automatiquement les règles logiques concernant les causes d'anomalies et en les intégrant dans les règles MLN, nous parvenons à une identification plus précise de la situation, même avec des observations partielles. Chacune de nos contributions a été prototypée, testée et validée à l'aide de données obtenues à partir de scénarios réels réalisés.

**Université Paris-Saclay**
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France

**Title :** Real time intelligent decision making from heterogeneous and imperfect data

**Keywords :** Intelligent environment, Activity recognition, Anomaly detection, Sensor Data segmentation

**Abstract :** Nowadays, pervasive computing is facing an increasing advancement. This paradigm is characterized by multiple sensors highly integrated in objects of the physical world.The development of personal applications using data provided by these sensors has prompted the creation of smart environments, which are designed as an overlay advanced framework that proactively, but sensibly, assist individuals in their everyday lives. A smart environment application gathers streaming data from the deployed sensors, processes and analyzes the collected data before making decisions and executing actions on the physical environment. Online data processing consists mainly in data segmentation to divide data into fragments. Generally, in the literature, the fragment size is fixed. However, such static vision usually brings issues of imprecise outputs. Hence, dynamic segmentation using variable sizes of observation windows is an open issue. The analysis phase takes as input a segment of sensor data and extracts knowledge by means of reasoning or mining processes. In particular, understanding user daily activities and preventing anomalous situations are a growing concern in the literature but addressing these problems with small and imperfect data is still a key issue. Indeed, data provided by sensors is often imprecise, inaccurate, outdated, in contradiction, or simply missing. In this thesis, we mainly explored cross-fertilization of statistic and symbolic learning approaches and the contributions are threefold: (i) DataSeg, an algorithm that takes advantage of both unsupervised learning and ontology representation for data segmentation. This combination chooses dynamically the segment size for several applications unlike most of existing methods. Moreover, unlike the literature approaches, Dataseg is able to be adapted to any application features; (ii) AGACY Monitoring, a hybrid model for activity recognition and uncertainty handling which uses supervised learning, possibility logic inference, and an ontology to extract meaningful knowledge from small datasets; (iii) CARMA, a method based on Markov Logic Networks (MLN) and causal association rules to detect anomaly causes in a smart environment so as to prevent their occurrence. By automatically extracting logic rules about anomalies causes and integrating them in the MLN rules, we reach more accurate situation identification even with partial observations. Each of our contributions was prototyped, tested and validated through data obtained from real scenarios that are realized.

**Université Paris-Saclay**
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France

# Acknowledgements

This thesis has been an enriching and tough journey, full of ups and downs. It felt like chasing down different rabbit holes. Fortunately, I was surrounded by many supportive people, who have given me strength to go through the difficult times. So, it is a great pleasure to thank them all for their positive impact during this quest.

Naturally, my first thanks come back to my supervisors, Amel and Jérôme, who had the patience and the will to push me forward over those three years. It is obvious that the thesis would not be what it is without their work and help, for the research, but also personally. Moreover, I want to thank all the partners of the project CoCaps who's worked with me. Thank you for the collaborative work that has been done.

I would like to thanks also the jury of my defence, for their remarks, questions and the evaluation of my work. I am also grateful for the lab with all its members, that welcomed me.

I thank the team and the persons that worked with me: Nathan, Badran, and others. I warmly thank my PhD-mates, and now friends, Monika, Wafa, Fethi, Safa, Mohammed, Asma and Wided that cheered me up, and with whom I shared numerous memories.

I am forever grateful to my parents, Radhia and Taher, and my brothers, Walid, Mohammed Ali, and Anouar for their love, support and care, that where more than necessary during this thesis. Thank you for having faith in me even when doubting myself. I'm also thankful for my second family, my parents in law Aidouda and Azouz, my sister in law Hanouta, and my brother in law Khaled to be in my side for every difficult moment that I faced. I am deeply thankful to my soul-mate and loving husband Khalifa for his help, patience and understanding. Thank you for putting up with my temper and for helping me getting through the tough times. Thank you my son Malek to be the best gift in my life.

I also thank my best friend Chaima for all the good times that boosted my morale.

# Abstract

Nowadays, pervasive computing is facing an increasing advancement. This paradigm is characterized by multiple sensors highly integrated in objects of the physical world. The development of personal applications using data provided by these sensors has prompted the creation of smart environments, which are designed as an overlay advanced framework that proactively, but sensibly, assists individuals in their every day lives. A smart environment application gathers streaming data from the deployed sensors, processes and analyzes the collected data before making decisions and executing actions on the physical environment. The analysis allows to extract knowledge from the sensor data by means of reasoning or mining processes. In particular, understanding user daily activities and preventing anomalous situations are a growing concern in the literature but addressing these problems with small and imperfect data is still a key issue. Indeed, data provided by sensors is often imprecise, inaccurate, outdated, in contradiction, or simply missing. Hence, handling uncertainty became an important aspect. Moreover, monitoring the user to obtain a large amount of data about his/her life routine is not always possible and too intrusive. People are not often open to be monitored for a long period of time. Obviously, when the acquired data about the user are sufficient, most existing methods can provide precise recognition but the performances decline sharply with small datasets.

Before the online analysis of a streaming sensor data, the latter should be processed and then segmented. Data segmentation is to divide data into fragments. Generally, in the literature, the fragment size is fixed. However, such static vision usually brings issues of imprecise outputs. Hence, dynamic segmentation using variable sizes of observation windows is an open issue.

In this thesis, we mainly explored cross-fertilization of statistic and symbolic learning approaches and the contributions are threefold: (i) AGACY Monitoring, a hybrid model for activity recognition and uncertainty handling which uses supervised learning, possibilistic logic inference, and an ontology to extract meaningful knowledge from small datasets; (ii) Caredas, a method based on Markov Logic Networks (MLN) and causal association rules to detect anomaly causes in a smart environment so as to prevent their occurrence. By automatically extracting logic rules about anomalies causes and integrating them in the MLN rules,

we reach a more accurate situation identification even with partial observations; (iii) DataSeg, an algorithm that takes advantage of both unsupervised learning and ontology representation for data segmentation. This combination chooses dynamically the segment size for several applications unlike most of existing methods. Moreover, unlike the literature approaches, DataSeg is able to be adapted to any application features. Each of our contributions was prototyped, tested and validated through data obtained from real scenarios that are realized.

# Résumé

De nos jours, l'informatique omniprésente est en constante progression. Ce paradigme est caractérisé par la présence de multiples capteurs intégrés dans des objets du monde physique. Le développement d'applications personnelles utilisant les données fournies par ces capteurs a conduit à la création d'environnements intelligents pour aider de manière proactive les individus dans leur vie quotidienne. Ces applications collectent de façon continue les données des capteurs deployés, traitent ces données et les analysent avant de prendre des décisions pour exécuter des actions sur l'environnement physique.

L'analyse de données pour la reconnaissance des activités quotidiennes des utilisateurs et la prévention des situations anormales représentent des défis non encore complètement relevés en parti. En effet, la résolution de ces problèmes à l'aide de données de petite taille et imparfaites reste un problème clé. Effectivement, les données fournies par les capteurs sont souvent imprécises, inexactes, obsolètes, contradictoires ou tout simpement manquantes. Par conséquent, l'incertitude liée à la gestion est devenue un aspect important. De plus, il n'est pas toujours possible et trop intrusif de surveiller l'utilisateur pour obtenir une grande quantité de données sur sa routine de vie. Les gens ne sont pas souvent ouverts pour être surveillés pendant une longue période. Évidemment, lorsque les données acquises sur l'utilisateur sont suffisantes, la plupart des méthodes existantes peuvent fournir une reconnaissance précise, mais les performances baissent fortement avec de petits ensembles de données.

Parmi les étapes de traitement de données en ligne est la segmentation des données pour les diviser en fragments. Généralement, dans la littérature, la taille des fragments est fixe. Cependant, une telle vision statique entraîne généralement des problèmes de résultats imprécis. Par conséquent, la segmentation dynamique utilisant des tailles variables de fenêtres d'observation est une question ouverte.

Dans cette thèse, nous avons principalement exploré la fertilisation croisée d'approches d'apprentissage statistique et symbolique et les contributions sont triples : (i) AGACY Monitoring, un modèle hybride de reconnaissance d'activité et de gestion des incertitudes qui utilise un apprentissage supervisé, une inférence de logique possibiliste et une ontologie permettant d'extraire des connaissances utiles de petits ensembles de données ; (ii) Caredas, une méthode basée sur les réseaux de Markov et les règles d'association causale pour détecter les causes d'anomalie dans un environnement intelligent afin d'éviter leur apparition. En extrayant automatiquement les règles logiques concernant les causes d'anomalies et en les intégrant dans les règles MLN, nous parvenons à une identification plus précise de la situation,

même avec des observations partielles ; (iii) DataSeg, un algorithme qui tire parti à la fois de l'apprentissage non supervisé et de la représentation ontologique pour la segmentation des données. Cette combinaison choisit de manière dynamique la taille de segment pour plusieurs applications, contrairement à la plupart des méthodes existantes. De plus, contrairement aux approches de la littérature, DataSeg peut être adapté à toutes les fonctionnalités de l'application. Chacune de nos contributions a été prototypée, testée et validée à l'aide de données obtenues à partir de scénarios réels réalisés.

# Thesis publications

**1. Journal papers**

1. Hela Sfar, Amel Bouzeghoub, Badran Raddaoui. Early Anomaly Detection in smart home: a causal association rules-based approach. Artificial intelligence in Medicine (ArMED), vol. 91, Elsevier, 2018

2. Hela Sfar and Amel Bouzeghoub, Activity Recognition for Anomaly Detection. Innovation and Research in Biomedical Engineering (IRBM), Elsevier, 2019 To APPEAR

**2. Conference papers**

1. Hela Sfar, Anja Habacha Chaibi, Amel Bouzeghoub, Henda Ben Ghézala. Gold Standard based ontology evaluation for ontology learning techniques. Symposium on Applied Computing (SAC), ACM, 2016

2. Hela Sfar, Amel Bouzeghoub, Nathan Ramoly, Jérôme Boudy. AGACY Monitoring: A Hybrid Model for Activity Recognition and Uncertainty Handling. Extended Semantic Web Conference (ESWC), Springer, 2017

3. Hela Sfar, Nathan Ramoly, Amel Bouzeghoub, Béatrice Finance. CAREDAS: Context and Activity Recognition Enabling Detection of Anomalous Situation. Artificial Intelligence in Medicine in Europe (AIME), Springer, 2017

4. Hela Sfar, Amel Bouzeghoub, Nathan Ramoly, and Jérôme Boudy. Novel Hybrid Model for activity recognition. Modeling Decisions for Artificial Intelligence (MDAI), Springer, 2017

5. Nathan Ramoly, Hela Sfar, Amel Bouzeghoub, and Béatrice Finance, LEAF: Semantic Based Experience to Prevent Task Failures. Field on Service Robotics (FSR), Springer, 2017

6. Hela Sfar, Badran Raddaoui, and Amel Bouzeghoub, Reasoning under conflicts for Smart Environment. International Conference On Neural Information Processing (ICONIP), Springer, 2017

7. Hela Sfar and Amel Bouzeghoub. Dynamic Streaming Sensor Data Segmentation in smart environments. International Conference On Neural Information Processing (ICONIP), Springer, 2018

8. Hela Sfar and Amel Bouzeghoub. DataSeg: Dynamic Streaming Sensor Data Segmentation for Activity Recognition. Symposium on Applied Computing (SAC), ACM, 2019

# Contents

# List of Figures

# List of Tables

# Acronyms

**ADL** Activity of Daily Life.

**AGACY** Activity AGgregation under unCertaintY.

**AI** Artificial Intelligence.

**ANN** Artificial Neural Networks.

**AR** Association Rule.

**CAR** Causal Association Rule.

**CARMA** Causal Association Rule Mining Algorithm.

**CBN** Causal Bayesian Network.

**DgMLN** Dynamic ground Markov Logic Network.

**DS** Dempster-Shafer Theory.

**DT** Decision Tree.

**FCD** Fuzzy Context Data.

**FSCEP** Fuzzy Semantic Complex Event Processing.

**HMM** Hidden Markov Model.

**IoT** Internet of Things.

**KB** Knowledge Base.

**KNN** K-Nearest Neighbor.

**LTF** Long Term Feature.

**MLN** Markov Logic Network.

**NB** Naive Bayes.

**NN** Neural Network.

**OWL** Web Ontology Language.

**PDP** Probabilistic Dynamic Predicate.

**PHP** Probabilistic Hidden Predicate.

**PSP** Probabilistic Static Predicate.

**RF** Random Forest.

**RFID** Radio Frequency Identification.

**STF** Short Term Feature.

**SVM** Support Vector Machine.

**UCB** Upper Confidence Bound.

# Chapter 1

# Introduction

## 1.1 Research Context

These last years, the trend of computing is moving toward pervasive computing, also called ubiquitous computing, which aims to provide services according to the user needs. This can be ensured by means of integration, cooperation and communication of multiple devices and software agents that are connected with the user and the environment in which they operate. These pervasive environments are generally perceived with the help of sensors of heterogeneous sources, the collected data is interpreted and a high level context is derived. This context information is then used by context-aware applications to adapt their behaviour to accommodate user desires and activities. In this setting, smart environments are particularly interested by context-awareness and proactivity to make smart decisions depending on user information and the surroundings. Indeed, smart environments are conceived as an overlay digital infrastructure that proactively, but sensibly, support people in their every daily lives. The deployed sensors are outstretched on any objects or human bodies. They gather data including user's location, motion, bio-medical information, environment temperature, humidity, or ambient noise level to extract useful information. This process follows a life cycle to convert raw data to meaningful information. The most important stages are (1) the collection which consists of obtaining data from available sources and verifying the quality and the accuracy of the gathered data. Actually, a bad quality of data heavily impacts the data analysis by producing highly misleading results; (2) the data processing which includes diverse means and methods associated with the preparation of the collected data; (3) the data analysis which is in charge of exploring and interpreting processed data to provide meaningful information that will help future decisions-making. One of the tasks of data processing is data segmentation i.e. divide the data stream into intervals or windows in order to be analyzed to extract knowledge. Depending on the target application, this knowledge

can be used to either recognize human Activities of Daily Living (ADL) (e.g. “watching TV”), actions (e.g. “take cup” or “open door”) and motion gestures or to detect anomalies. Both activity recognition and anomaly detection issues received a great attention these last years by numerous research projects among which we can cite:

- Dem@Care (Dementia Ambient Care) project: it is funded by the European Union, that aims to develop a proper installation of multiple sensors in the home, on the monitored person in an unobtrusive manner (wearable, portable devices), in tandem with recording of utilities usage for activities monitoring and analysis.
- E-Monitorage [1] (Nursing home systems applications and resident supervision data analysis) project: it is funded by the French Interministerial Fund (FUI) where the objective being to improve the comfort and safety of the residents in helping staff to be more efficient.
- SECURE [2](Intelligent System for Early Diagnosis and Follow-up at Home) project. it is funded by a grant of Lombardy Region and Italian Ministry of Education, which proposes an activity recognition system integrated with a novel method to detect abnormal activity routines of elderly people living at home.
- Heart project: it is an European Union’s Horizon 2020 (H2020) project. Heart is aiming to release a health integrated activity recognition platform able to detect activities from heterogeneous data, using scalable algorithms, while safeguarding the privacy of the persons. A priority for the non Academic beneficiary’s competitiveness is to deliver wearable technology for health monitoring, primary dedicated to healthy people (of +40 age) to penetrate the Chinese market.
- TENSOR project: it is an European Union’s Horizon 2020 (H2020) project. TENSOR is aiming to provide a powerful terrorism intelligence platform offering LEAs fast and reliable planning and prevention functionalities for the early detection of terrorist organised activities, radicalisation and recruitment. The platform integrates a set of automated and semi-automated tools for efficient and effective searching, crawling, monitoring and gathering online terrorist-generated content from the Surface and the Dark Web.
- datACRON project: it is an European Union’s Horizon 2020 (H2020) project. datACRON is introducing novel methods for threat and abnormal activity detection in

[1] https://www.pole-scs.org/en/projects/e-monitorage/
[2] http://secure.ewlab.di.unimi.it/

very large fleets of moving entities spread across large geographical areas. Specifically, datACRON aims to develop novel methods for real-time detection and prediction of trajectories and important events related to moving entities, together with advanced visual analytic methods, over multiple heterogeneous, voluminous, fluctuating, and noisy data streams from moving entities, correlating them with archived data expressing, among others, entities' characteristics, geographical information, mobility patterns, regulations and intentional data (e.g. planned routes), in a timely manner.

A new projected named CoCAPS has been started since May 2016. This project finances my thesis. CoCAPS is funded by the French Public Investment Bank (BPI France) and coordinated by Legrand. Legrand is a French industrial group historically located in Limoges, Limousin and one of the world leaders in products and systems for electrical installations and information networks. CoCAPS is aiming at designing low cost sensors to provide enriched information on the behaviour(s) of person(s) inside a building, in the service of energy efficiency and autonomy. Through the collaboration with different industrial(Legrand, id3, and EMKA) and academic (UTC[3], Prisme[4], Vallorem[5] ,and Samovar[6]) partners. COCAPS will allow an energy saving in the smart environment by keeping user satisfaction considering her/his activity. Another aim of COCAPS is to provide autonomy to elderly. By knowing his/her activity, COCAPS may recommend actions to prevent anomalies or also to help the elderly in realizing his/her activities. CoCAPS differs from the previous projects in different aspects. Firstly, CoCAPS goes forward just realizing a prototype to provide a real industrial product: set of sensors. Moreover, unlike most of previous projects different use cases are handled by CoCAPS (e.g. enegrgy management, anomaly detections, etc.)

The final product of CoCAPS will be, then, different sensors that ensure the mentioned functionalities according to the target use case. In order to reach this objective, the tasks are distributed to the different partners of the project. Prisme, the laboratory of Orleans university, starting by acquiring the deployed sensor data in the smart environment, is in charge of fusion these data. It allows, for example, to interpret the data of PIR sensors to a probability distribution of localization in the different areas of the environment. The laboratory Vallorem of the university Orleans-Tours participates in the project for the task of user's gesture recognition. It allows for example to recognize whether the user is sitting or stand up. On the other side, UTC, the university of technology of Compiègne, is in charge of the voice recognition inside a room in the environment with the goal of distinguishing the different persons in the room. At the bottom of these different process, Samovar the

[3]University of Technology of Compiègne https://www.utc.fr/

[4]Prisme laboratory of Orléans university http://www.univ-orleans.fr/en/group/234/home

[5]Vallorem laboratory of university of Tours https://vallorem.fr/

[6]Samovar laboratory of Télécom SudParis school http://samovar.telecom-sudparis.eu/spip.php?article2

laboratory of Télecom SudParis, participates in COCAPS through this thesis. Based on the different given information from the different partners, the role of this thesis in COCAPS is to recognize user daily activity and then to make decisions for the service of energy efficacy and autonomy. For recognizing user activities, labelled data are supposed to be provided which help to build a model for supervised learning. Moreover, real data can be easily obtained through realizing different scenarios in the smart environments related to CoCAPS as for instance the GIS MADONAH. It is a smart apartment equipped with different types of sensors (e.g. PIR, CO2, COV, Temperature, and luminosity sensors) and used as a nursing home for elderly. The real data acquired from this smart environment serve to prove to viability of this thesis proposals in real scenarios.

Hence, as we mentioned above, this thesis is carried out as a part of the COCAPS project and focuses mainly on the data processing and analysis phases by addressing in particular the following issues: activity recognition, anomaly detection, and data segmentation. These issues are difficult in different aspects. Firstly, for intrusiveness concerns, the use of camera is not allowed. This constraint makes these issues even more complex. Secondly, these issues must be processed online because of the constantly incoming sensor streams. Thirdly, incoming sensor data are often imperfect (like missing values, noise or imbalanced distribution), due in general to hardware failures, energy depletion, etc. This problem will increase difficulties to process and analyze the data.

For a clearer understand, in the next section, we show an illustrative scenario about the tackled problems and their requirements.

## 1.2 An illustrative scenario

Mary is an elderly women living alone in her smart home. This smart home contains various sensors: movement sensors, beacons, RFID tags, window and door open sensors, Mary's smart phone, connected devices (stove, TV, fridge, etc.), thermometers, gas detectors, air quality sensors, switches, and microphones. It also possesses various actuators: lights, smart devices (TV, fridge, heaters, etc.), motorized doors, alarms (i.e. speakers), motorized windows and shutters, and user's smart phone. Neither cameras nor intrusive sensors are used. Mary wants a more secure and comfortable life. She suffers from low remembering ability. She often forgets the stove open after cooking or the window open while she is sleeping. She feels now afraid about the problems that may occur: A fire may broke out if the stove is still open for a long time or she may be sick if it was cold and the bedroom's window remained open while she was sleeping. Therefore, she wants to be alerted before the

occurrence of such anomalies related to her daily activity's surrounding context. Yet, she refuses to be monitored for a long period of time.

Mary has different daily routines. Hence, her activity may change according to her life's routine. For example, when she has an appointment she can have a quick lunch in ten minutes otherwise she can take it during half an hour.
This scenario outlines the problems that should be tackled and serves to highlight the requirements to satisfy at different stages in the process as follows:

**Activity recognition**

- *R1: Online recognition.* The ADL recognition should be done online.
- *R2: Precise recognition.* Sensor data may be uncertain. In order to provide precise results, this uncertainty should be considered for recognizing the user ADL.
- *R3: Multiple life routines.* The user can have different routines for his/her daily activities. This criterion should be considered for the recognition of ADL: One ADL may have different duration.
- *R4: Small data processing.* Experiments involving real people have in general small data to avoid their monitoring for a long time period. This is a limiting issue for statistical learning algorithms.

**Anomaly detection**

- *R5: Context anomaly.* The detected anomaly should be related to the context surrounding the activity to detect whether the activity is realized in abnormal conditions.
- *R6: Anomaly forecasting.* The anomaly should be detected in the early stages. A successful method for anomaly detection usually requires solutions for predictions in order to be able to warn of future emergency or abnormal conditions.
- *R7: Online anomaly prediction.* The early detection should be done online.
- *R8: Precise anomaly prediction.* Sensor data may be uncertain. In order to provide precise results, this uncertainty should be considered for predicting the anomalies.
- *R9: Insufficiency of data.* Only a small amount of data about the user daily life routine could be provided for predicting the anomalies.

**Sensor data segmentation**

- *R10: Flexible segmentation.* In order to online recognize ADL and then detect anomalies, data should be before segmented into a set of time window. Afterwards, each window is analyzed separately. Usually, data segmentation is achieved with a fixed window size. For a higher precision, the window size has to be adjusted dynamically. Moreover, for the seek of generality, the segmentation method should be adapted to several processing purposes.
- *R11: Online segmentation.* The segmentation should be realized online based on the received streaming sensor data.

## 1.3 Challenges

Throughout this subsection, we aim to define the different challenges related to each of the aforementioned problems. In the following, we review them by giving key insights.

**Activity Recognition**

The activity recognition task aims to infer from the given segment of sensor data the user's ADL. This process is still challenging despite the multitude of works proposed for this aim [47, 46, 52, 30, 62]. Previous methods are either using learning methods which require a large amount of data to setup the model or based on knowledge techniques (i.e ontology, logic) which claim regular experts intervention for the definition of the model. Both types of methods are able to handle sensor data uncertainty to provide precise results. However, on the one hand, regarding the requirement *R4*, with small amount of data, learning techniques may not provide satisfying results. On the other hand, knowledge techniques require effort and time from the experts to define and update the model which is a laborious task depending on the experts availability. To overcome these issues, hybrid methods, that combine both techniques, have shown recently an increasing interest. Thanks to the high level knowledge provided by the knowledge technique, a hybrid method is independent of the data amount. Moreover, an important part of knowledge is learned which facilitates the expert task and reduces his/her intervention. In order to satisfy the requirement *R2*, sensor data uncertainty should be considered in the whole recognition process. However, hybrid models are complex since several components from heterogeneous fields collaborate to provide the adequate recognition. In addition, two more constraints must be considered: the recognition process should be performed online and the user life routines diversity has to be integrated (requirements *R1* and *R3*). However, as mentioned previously an hybrid method holds two extremely different techniques. For an online recognition, a seamless communication between these two techniques should be ensured to handle streaming sensor data. Fulfilling these requirements leads to the following new challenges:

**Ch1: How to hybridize learning and knowledge based techniques while dealing with data uncertainty and its propagation during the activity recognition process?**

**Ch2: How to achieve online activity recognition through an hybrid method and considering multiple user routines?**

**Anomaly detection**

Anomaly detection is the problem of finding deviation regarding the normal behaviour. In this thesis, based on the requirements *R5*, *R6*, and *R7*, we focus on online anomaly prediction. The anomalies that we target are occurred when the activities are realized in a particular context. Hence, the normal behaviour is the normal conditions where an activity should be performed. Generally, in the literature [58, 39, 39], the anomaly targets only the activity, a group of activities or the activity under some context. Smart environments are dynamic, the user context is changing over time. Therefore, we think that considering the context for detecting the anomalies and extracting their causes is an important topic. Most previous works in this field, do not analyze the context in order the extract the anomaly causes. Moreover, small data is not compatible with the application of statistical learning and may involve uncertainty and missing data (requirements *R8* and *R9*). The problem becomes more complex and leads to the following challenges:

**Ch3: How to early detect anomalies and extract their causes in the activity's context using only a small amount of data?**

**Ch4: How to manage data uncertainty in the anomaly prediction process?**

**Sensor data segmentation**

Before the online analysis of sensor data to recognize user activities and early detect the anomaly occurrence, these data should be segmented. The segmentation aims to divide the data into a set of fragments. It is not possible for any online analysis process (e.g. activity recognition, anomaly detection) to handle the data by a one-pass. The segmentation can be realized either statically or dynamically. Static segmentation means that the segments (or windows) keep always the same size which can be measured in terms of time or sensor data number. This is the simplest way to segment the data that does need neither effort nor investigation. It requires only an empirical evaluation of the target process with different window sizes. The size providing better results is chosen. However, this entails testing each method with all possible sizes. Hence, finding the suitable size is not guaranteed. Moreover, when the size is badly chosen the method's precision can be decreased. The data analysis can be confused when the segment contains too much data (i.e the size is too large) or rather

the opposite, has insufficient information to be precise (i.e. the size is too short). Therefore, dynamic segmentation is needed in order to overcome this issue by adapting the segment size according to the target analysis. However, the dynamic segmentation is already not trivial problem since it requires enough flexibility and precision to correctly choose the best size. According to the requirements *R10* and *R11*, the segmentation should be also adapted for both activity recognition and anomaly detection issues and done online. Thus, the dynamic segmentation with these requirements becomes more difficult. The first challenge is then stated as follows:

**Ch5: How to define an online method able to dynamically segment the streaming sensor data and having the ability to be adapted for different analysis goals?**

## 1.4 Contributions

With current research advancement, numerous challenges are still remaining unsolved, as we can deduce from the previous subsection. In this computer science thesis, we proposed multiple contributions to cope with those challenges and to provide solutions for data processing and analysis in smart environments. Through these contributions, we explored the usage of various techniques and combinations, including learning and knowledge techniques.

### 1.4.1 AGACY Monitoring for activity recognition and uncertainty handling

The online activity recognition handles the received segment of data and decides which activity the user is currently doing. As stated previously, handling uncertainty in hybrid methods for activity recognition is a challenging task ( challenge **Ch1**). In order to tackle this problem, we proposed a novel hybrid method, named AGACY Monitoring, to recognize user activities into smart environment considering sensor data uncertainty belonging to the given segment. The method combines ontology modelling, possibilitic logic reasoning, and supervised learning. The modelling and reasoning part allows to infer knowledge about the realised user events. These events are inferred together with their uncertainty values computed based on those of the received sensor data. The learning part exploits this knowledge in order to learn the user activities and compute their uncertainty values considering those of the given events. AGACY Monitoring recognizes online the user ADL and is able through its hybrid model to consider different user routines (challenge **Ch2**).

### 1.4.2 Caredas for early anomaly Detection

In order to tackle the challenges **Ch 3 and 4**, we proposed, Caredas, a hybrid model combining Markov Logic Network (MLN) [69] and causal association rules mining (CAR) [51] for the early detection of anomalous situations. The CAR serves to extract, offline, logic rules about anomaly causes regarding the context surrounding the user activities. Afterwards, these rules model the logic part of MLN. CAREDAS receives as input, online, a segment of sensor data containing user activity and his/her current context data that we call a situation. Using MLN, it decides if this situation risks to be anomalous or not, if it is the case it recognizes the anomaly. During the whole process, the sensor data uncertainty is considered and integrated in the risk computation of the anomalies by modifying the MLN rules weight calculus. Through this hybridization, the method is able to precisely predict the anomalies even with small amount data.

### 1.4.3 DataSeg: Dynamic streaming sensor data segmentation

As a first step befor the online analysis, the streaming data must be divided into segments. However, as stated previously, dynamically segmenting the sensor data independently of the target use is still problematic. In order to tackle the challenge **Ch5**, we proposed DataSeg, a hybrid method combining clustering and ontology to a dynamic segmentation of streaming sensor data. The method is conceived to be adapted for different target applications. To achieve this aim, as input, it receives a set of features and annotated training dataset that are related to the target use. The clustering groups the dataset into a set of clusters according to the given features. The ontology is automatically updated to semantically represent the provided clusters. The proposed method, using the knowledge provided by the ontology, handles online the incoming streaming data in order to decide which better size for the current segment. Once the size is decided, it becomes possible to send the segment to the target process.

### 1.4.4 Conflict Detection and Resolution

In this thesis, as stated above, we are mainly interested in the hybridization of artificial intelligence models, that is to say, statistical and symbolic learning. In such models and more precisely in the reasoning part, the definition of logic rules is mandatory. However conflicts can occur among these rules. Several experts can collaborate for the definition of the rules. For instance, when one expert updates a rule and forgets to delete the previous one conflicts can occur. Particularly in smart environment applications, the contrariness

conflict may often occur when two rules have the same conclusion but different antecedents. This type of conflict is neglected in the related works, instead they focus on inconsistency conflict of rules. In order to overcome this issue, we proposed, through a collaboration with other members of the team, a method able to detect and resolve conflicts (contrariness and inconsistency) among logic rules. The novel method is based on a new adjustment of the argumentation theory to be able to detect and resolve the conflicts among logic rules.

### 1.4.5 Overall view about this thesis contributions

For a clearer understanding, Figure 1.1 shows an overview of this thesis contributions.

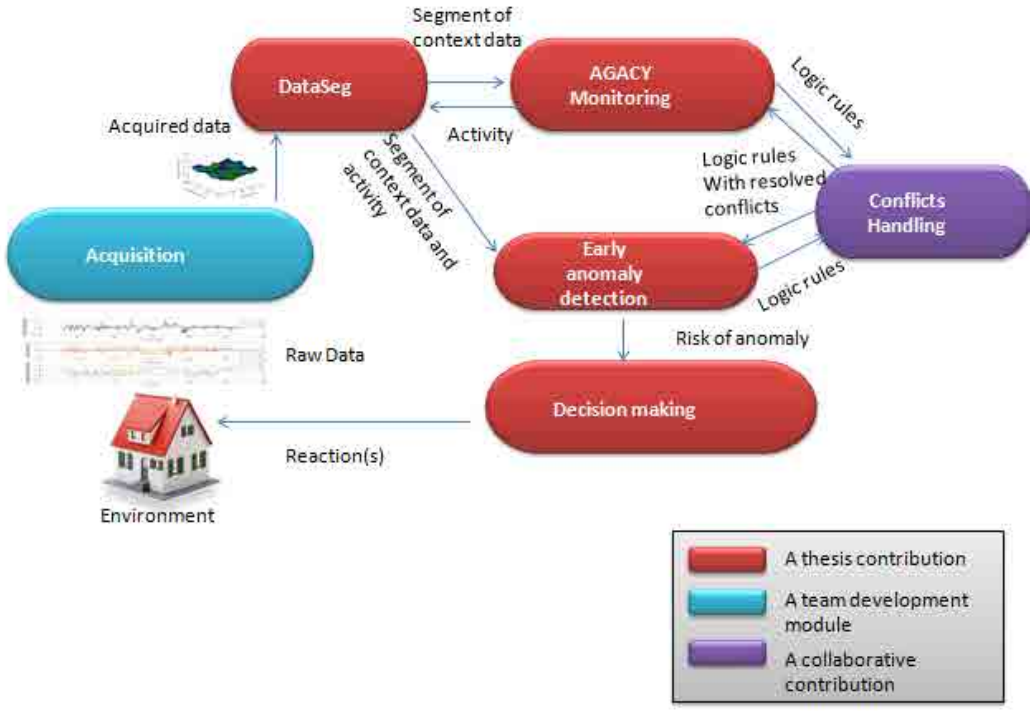



Figure 1.1 Overview of the thesis contributions

To tackle the whole sensor data life cycle, the acquisition of the data is required as a first step, as depicted in the Figure. During this thesis, the sensor data acquisition is achieved by a the FSCEP [5] system that is developed in our team. Afterwards, DATASEG receives a streaming of acquired sensor data, divides this stream into segments with different sizes. Each segment is sent to AGACY MONITORING which provides the knowledge about the user current ADL. This activity is provided to DATASEG which adds it into a segment combined with the current sensor data about the environment context. The final segment is sent to the CAREDAS process that analyzes it to detect possible risk of anomaly. This detected risk

will be exploited afterwards by the decision-making process to generate an action. AGACY MONITORING and CAREDAS are hybrid methods which employ (among others) logic rules for reasoning. Thus, they ask the CONFLICTS HANDLING method that have been proposed in collaboration with other team members. This method detects and resolves the conflict if it occurs among the received rules.

## 1.5 Thesis organization

This manuscript describes the work conducted during my thesis and is divided into six chapters. The next chapter reviews the state of the art about data life cycle in smart environment and identifies the current and open problems. I then address our contributions that start by recognizing user activities in Chapter 3, and early detecting the anomalies in Chapter 4, to finally dynamically segment streaming sensor data in Chapter 5. Chapter 6 concludes the thesis in which the main contributions are summarized and the future perspectives are announced. In the Appendix A, the contribution about conflicts handling, that have been proposed through a collaboration, is presented.

# Chapter 2

# State of the art

## 2.1 Introduction

Decision making in smart environments is an important topic that emerged with a wide range of applications to provide energy saving, security and autonomy of the elderly and disabled people. Each of these applications carries a tremendous amount of challenges and topics at different levels of the process starting from the acquisition of sensor data to decision including the processing and the analysis. Despite the plethora of decision making applications, the employed techniques for the processing and the analysis of sensor data are mainly classified as data-driven or knowledge based depending on the underlying algorithms and formalisms.

This chapter presents a review of the literature and evaluate the current state of the research related to the processing and the analysis of sensor data in smart environments. Moreover, we aim to identify the limitations to tackle in order to achieve our objectives.

As the domain is vast and various, we first introduce smart environments and sensor data life cycle in smart environments. We review, then, the different techniques used in the literature for sensor data processing and analysis. Sensor data processing and analysis fields are very large including several problems and challenges. On the one hand, the analysis aims to extract knowledge from sensor data data. This knowledge, depending on the application, can be the user ADL, his/her daily events, her/his motion gestures, and even an anomaly. In particular, ADL and anomaly recognition problems have gained a lot of attention in smart environment applications due to their important impact on improving the user life quality and sometimes saving her/him life. On the other hand, for online recognition sensor data need to be segmented. Data segmentation is one of data processing. For these reasons, in this thesis, we mainly focus on the problems of activity and anomaly recognition for the analysis of data and on the segmentation problem regarding data processing. Thus, in this chapter, we

identify the types of problems encountered by the previous work about activity recognition, anomaly detection, and sensor data segmentation.

This chapter is organized as follows: in the next section, we define the notion of smart environments and their most important issues. Afterwards, in Section 2.3, we review the techniques that could be applied for the data processing and analysis. In the following Sections 2.4, 2.5, and 2.6 we review respectively previous work about the activity recognition, the anomaly detection, and the sensor data segmentation fields. This chapter ends with a conclusion summarizing our analysis of the literature.

## 2.2 Smart Environments

Smart environments that are aware of the current user's situation and can react to provide proactive decisions are expected to become a part of our everyday life. Furthermore, it is widely believed that computing in smart environments will move from desktop computers to a multiplicity of embedded computers present in the smart devices around us. A smart environment is a small world where different types of smart devices are continuously working to make inhabitants' lives more comfortable. Formally it is defined as follows [25]:

**Definition 1** *Smart Environments are varied physical worlds typically used in human daily life; those are seamlessly embedded with tiny devices capable of pervasive sensing, actuating and computing. These physically embedded tiny devices are all connected through a continuous network for data collection, in order to enable various pervasive applications and services. The Smart Environments include Smart Home, Smart Building, Smart Office, Smart Farm, Smart Hospital, and Smart Meeting Room etc.*

Sensors are embedded in the smart environment for sensing spatial-temporal states of physical or environmental situations. Table 2.1 shows a list of the common types of sensors that are used in various applications. Each sensor is used for measuring one of the physical properties like Temperature, Resistance, Capacitance, Conduction, Heat Transfer, IR presence, etc.

Raw data provided by the sensors has a life cycle, as shown in Figure 2.1, starting from the acquisition, then being processed and analyzed to leverage knowledge models handled by the application to provide proactive decisions. The latter aims to provide actions that can be related directly to the user and also the smart environment. In smart environments the most common issues to tackle for the sensor data are as follows:

- Data are uncertain. Sensor devices have hardware restrictions making provided data always subjected to some level of imperfection. Moreover, the quality of sensor data is

| Sensor | usefulness |
|---|---|
| Temperature Sensor | measures the amount of heat energy or even coldness that is generated by an object or system |
| Accelerometer sensor | measures acceleration, which is the rate of change of the velocity of an object. It measures in meters per second squared (m/s2) or in G-forces (g) |
| Infrared (IR) Sensor | almost always used for remote control detection by emitting IR pulses |
| Pressure Sensor | measures the pressure, typically of gases or liquids. It generates a signal related to the pressure imposed. |
| Light Sensor | measures illuminance, which can be used to measure more than the brightness of a light source |
| Ultrasonic Sensor | measures distance. It is a cheap sensor that can measure 2cm to 400cm of non-contact measurement functionality with a ranging accuracy that can reach up to 3mm. |
| Smoke and Alcohol Sensor | is useful module for detecting alcohol or smoke concentration. It measures the alcohol concentration precisely. It can also measure smoke and alcohol with low sensitivity. |
| Touch Sensor | enables touch detection by measuring capacitance, exhibiting a change in capacitance in response to a change in surrounding materials. |
| Color Sensor | detects the color of the surface, usually in the RGB scale. Color is the result of interaction between a light source, an object and an observer. |
| Humidity Sensor | measures and reports both moisture and air temperature. The ratio of moisture in the air to the highest amount of moisture at a particular air temperature is called relative humidity. |

Table 2.1 List of the most common sensors types

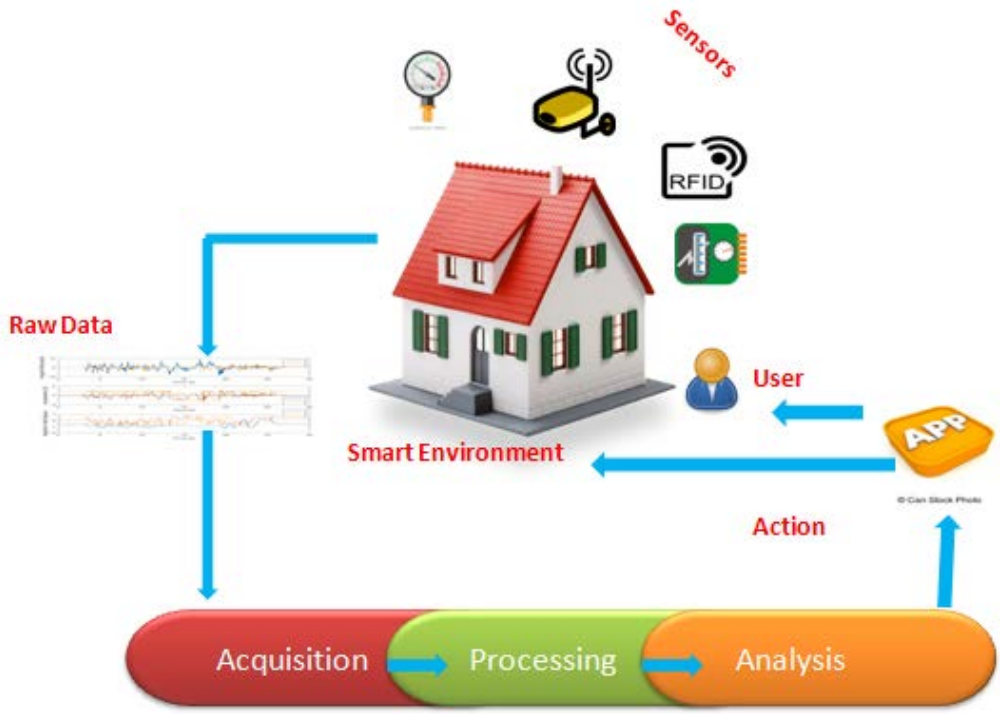



Figure 2.1 Smart environments architecture for decision making applications

often decreased by sensor failures or malfunctions. Thus, when deficiencies on sensor are tackled, it allows to reduce information misunderstandings [13].

- Data are heterogeneous. Normally, each type of sensors (see Table 2.1) is used to provide one kind of data (i.e temperature, humidity, etc.). Usually, a variety of sensors are generated in a smart environment where each one is used for a particular aim (i.e

providing user location, getting status of an object, etc.). Hence, these multitude of sensors provide heterogeneous data.

- Data streaming. Sensor data are generally provided as time series, that is to say a streaming sensor data. Hence, this stream can be either stored on computer to be treated afterwards offline or treated immediately by an online process. In this thesis we are interested by the online handling of sensor data. We believe that results can loose their effect if they come too late.

In the next subsections we provide an overview of each step of the sensor data life cycle.

### 2.2.1 Acquisition

Data acquisition allows mainly the transformation of sensor data into information that could be treated in computers. This task includes the collection and the fusion of the sensor data from the smart environments. Moreover, the acquisition step is responsible of data cleansing from noisy data and computing their uncertainty values which vary between 0 and 1. Uncertain data have almost 0 as uncertainty value. This value increases with the data quality. Nowadays, sensor data acquisition field is a quite popular topic. The corresponding literature around it is rich and there are several works for various applications. The Fuzzy Semantic Complex Event Processing (FSCEP) [5, 4] system, that is developed in our team, is able to acquire data from several sensors and compute their uncertainty values. One advantage of this system is the ability to consider the different types of uncertainty, compute their values, and merge them into one value for each data. Several other systems have been proposed in the literature for the same aim such as CEP2U [38] and SCADA[1]. The CEP2U is a model for dealing with uncertainty in complex events. The SCADA framework is a more evolved system. One of its main functionality is the ability to monitor, gather, and process real-time data and to record events in log file. The main difference between these systems and the FSCEP is, on the one hand, the lack of dealing with fuzzy data for these systems and, on the other hand, they do not handle different uncertainty dimensions.

The acquisition field is mature enough in the state-of-the-art. Several efficient methods have been proposed allowing to achieve this step. In this thesis we have chosen to use the FSCEP system for data acquisition. Firstly, because it is available easily. Secondly, the FCSEP provides more advantages regarding the other systems. Therefore, we focus on the processing and the analysis of sensor data as main contributions of this thesis.

[1] https://inductiveautomation.com/resources/article/what-is-scada

### 2.2.2 Processing

After collecting the sensor data, cleansing it, and computing its uncertainty, we obtain a model of information that can be handled by computers. Hence, it becomes possible to process this information to prepare it for the analysis. Data segmentation is considered as one of the important fields of data processing. In fact, any process cannot handle the acquired data in one-pass. Therefore, data needs to be segmented. The segmentation process divides the data into a set of segments, called also windows. Afterwards, each segment can be analyzed separately. The sensor data segmentation is still an open research question due to its important effect on the analysis process and also on the application output. Thus, the segment size must be properly chosen. Otherwise, the segment may contain either too much or insufficient data which decreases the application accuracy. In the literature, there are two strategies of segmentation: (1) Static or (2) Dynamic [78, 12]. On the one hand, static segmentation means that the segment size is fixed and cannot be changed. The application will be evaluated using different segment sizes. Then, the size providing best results is chosen. On the other hand, the dynamic segmentation allows to divide the data into a set of segments with different size. Each size is chosen dynamically during the application process.

Data segmentation is a challenging task that has a great effect as well as for analysis and also for the application. Therefore, in this thesis we have studied this problem.

### 2.2.3 Analysis

After processing the sensor data and receiving a segment of data, the analysis aims at extracting knowledge from the sensor data belonging to the given segment. As examples of knowledge we can cite user daily activities [61, 77, 56] which are considered as a high level knowledge, the user generated events [19], his/her motion gestures in the smart environment [88], or the occurrence of anomalies [58]. Particularly, activity and anomaly recognition fields are well researched problems and have known a great attention this last decade due the importance of the provided knowledge for a wide range of applications across a variety of domains, such as healthcare, social networks, safety, environmental monitoring, and transportation. On the one hand, recognizing the user activity allows the application to provide a personalized outcome according to the user's activity needs. On the other hand, detecting anomalies is a crucial topic particularly for health-care applications. The anomalies are considered as the deviations from the the normal behaviour. Ideally, these anomalies can be used to better assess the individual's current state in smart home, rate of change, and the potential need for assistance, and ultimately a reduction in the costs of care and medical emergencies [89].

Accordingly, due to the importance of these fields, in this thesis we focus on the problems of activity recognition and anomaly detection for the analysis of sensor data.

### 2.2.4 Applications

A wide range of applications have been proposed in the literature for decision making in smart environments using the knowledge given by the analysis. For a mobile context-aware recommender application, the authors in [29] design a method that suggests to the user the suitable information depending on her/his situation and interests. Indeed, the application is dynamically adapted according to the evolution of the user's interest. Nowadays, energy management applications based on user's activity have gained a great attention in smart environments [79]. Such application allows to propose the best strategy for energy consumption without reducing the user's comfort according to his/her activity. A related medication remainder is another example of known applications for user health monitoring in smart homes [90, 96]. It aims, principally, to remind an elderly person to take his/her medicine. For such applications, forgetting taking medicine is considered as anomaly. It serves then to provide action, a recall, that consists of reminding the user about the forgotten medicine.

Applications in smart environments are considered as the end of sensor data life cycle. We let this avenue for future researches. Mainly, in this thesis we are interested about the sensor data processing and analysis in order to prepare the decision making module associated to an application. The point that comes before studying works into these topics, is to review the different techniques that could be applied for the sensor data analysis and processing. This study is the goal of the next section.

## 2.3 Techniques for data analysis and processing

Throughout this section we present the techniques that could be applied for data analysis and processing, in general, for smart environments. These techniques can be classified into three categories: data driven, knowledge based, and hybrid techniques. On the one hand, data driven techniques, that is to say machine learning, include supervised, unsupervised methods and reinforcement learning. On the other hand, knowledge based techniques mostly concern ontology and logic reasoning. Finally, hybrid techniques combine between methods from both categories. In the next subsections we review each category, identify its benefits and limitations. .

### 2.3.1 Data driven techniques

As the name suggests, data driven techniques are based on analyzing the data of a system, in particular finding connections between the system state variables (input, internal and output variables) without explicit knowledge of the physical behaviour [28]. A data driven algorithm is used, then, to determine the relationship between a system's inputs and outputs using a training dataset that is representative of all the behaviour found in the system. Once the model is trained, it can be tested using an independent dataset to determine how well it can generalize to unseen data.

Data driven methods can be classified into supervised, unsupervised, or reinforcement learning as illustrated in Figure 2.2.

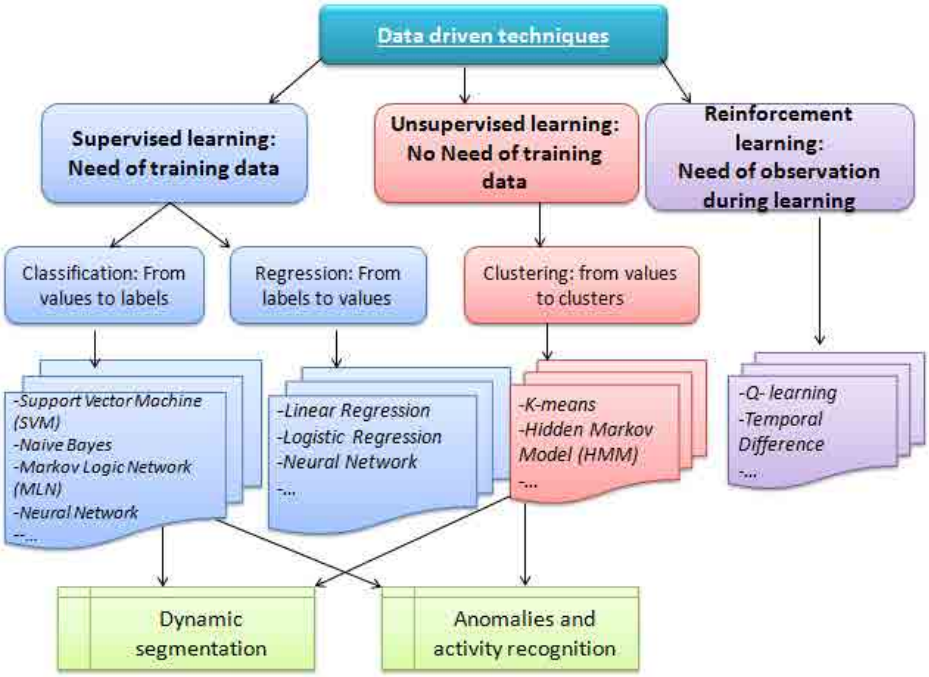



Figure 2.2 Data driven techniques overview

- The supervised learning algorithms build a model based on a labelled training dataset according to a given set of features. This model allows to represent relationships and dependencies between the target prediction output and the input features. It is possible then to predict the output values for new data based on those relationships which are learned from the previous datasets. Formally, the training data is a finite sequence $S = (x_1, y_1), \ldots, (x_n, y_n)$ of pairs where each $x_i$ is some measurement or set of measurements of a single example data point, and $y_i$ is the label for that data point. The supervised learning algorithm is a function $h$ called hypothesis, that aims at predicting

$y$ for arbitrary $x$, especially for those not contained in the training data. Most popular supervised learning algorithms are the Naive Bayes[93], Decision Trees[65], Support Vector Machines (SVM) [64], and Neural Networks[81].

- The unsupervised learning algorithms are the family of machine learning algorithms where no label is attached to the data. Thus, the task is to describe the data structure. These algorithms use techniques on the input dataset to mine for rules, detect patterns, and summarize and group the data points which help in deriving meaningful insights and describe better the data to the users. In unsupervised algorithms we are learning a function $f$ which helps to characterize the unknown distribution $P(Z)$ of the input data ($Z$). Sometimes $f$ is directly an estimator of $P(Z)$ itself (this is called density estimation). In many other cases, $f$ is an attempt to characterize where the density concentrates. Clustering algorithms, for example, divide up the input space in regions (often centered around a prototype example or centroid). Some clustering algorithms create a hard partition (e.g. the k-means algorithm) while others construct a soft partition (e.g. a Gaussian mixture model) which assign to each point of $Z$ a probability of belonging to each cluster.

- Reinforcement learning algorithms aim at using observations gathered from the interaction with the environment to take actions that would maximize the reward or minimize the risk. A reinforcement learning algorithm (called the agent) continuously learns from the environment in an iterative fashion. In the process, the agent learns from its experiences of the environment until it explores the full range of possible states in order to optimize a quantitative reward over time. The most used algorithms are UCB [86] and Q-Learning [102].

On the one hand, supervised methods allow the user to be very specific about the definition of the labels. In other words, it is possible to train the algorithm to distinguish different classes where an ideal decision boundary is set. This characteristic provides this type of learning the highest accuracy regarding the other types. However, it requires effort and time from expert in order to label the data. On the other hand, the unsupervised algorithms act without human guidance and less complexity in comparison with supervised learning algorithms. Nevertheless, they are less accurate regarding the results. This is also because the input data is not known and not labeled by human expert in advance, which means that the computer will need to do it alone. Finally, the reinforcement learning algorithms learn based on the interactions between the model and the user. Different from the unsupervised and supervised, the reinforcement learning allows to make a balance between trying what has worked in the past and trying new things to seek further improvement. However, it can also

be quite difficult to implement, particularly to perform efficiency these interactions which requires much expertise.

Generally, it is difficult to choose the best type of algorithm. Each one has its advantages and disadvantages over the others. Selecting one algorithm among the others is closely related to the context of the target issue and the available datasets. For example, if it is possible to monitor the user and provide a training dataset, supervised algorithms can be good candidates to be used due to their highest accuracy. In the context of the CoCAPS project, labelled datasets for training can be provided. Therefore, supervised learning could be employed for the target problems of this thesis: Activity recognition, anomaly detection, and data segmentation.

In section 2.2 we have presented a set of general issues related to the sensor data which can be uncertain, heterogeneous, and can be provided on streaming. Besides these issues, in the following, we show more specific requirements for the learning algorithms:

- A classification problem. The learning algorithms are used for two kinds of problems: Classification and Regression. Fundamentally, regression and classification are both related to prediction. While regression predicts a value from a continuous set, classification predicts the 'belonging' to a class. In our work, we are more interested in predicting a class (i.e activity and anomaly) than predicting a value. Therefore, we focus on the algorithms that are dedicated to classification.

- Handling uncertainty. As we mentioned in the previous section, in smart environments data can be uncertain. Hence, the data provided by sensors are not always precise and well aimed. Therefore, the algorithm should be able to handle uncertainty to provide more precise results.

- Dealing with heterogeneity. As stated above, one of the data issues in smart environments is their heterogeneity. For instance, it is possible to have different types of sensors in the smart environments. Hence, data can be provided by different sources of information. Thus, the algorithm must be able to treat such example of heterogeneity.

- Dealing with multi-class output. The provided recognition should not be restricted to a binary output. Instead, it must be large enough to hold the different class labels. For example, it is preferable to precise the anomaly's than just indicating if there is an anomaly.

- Efficiency of the results. Obviously, it is always required to apply an efficient algorithm. More the results are precise more the recognition is trustful.

- Time series data handling. The algorithm should be able to handle time series data in order to allow online processing and analysis.
- The training and prediction speed. When we deal with online processing and analysis, the notion of time is very important. The results can loose their purpose when coming too late. Therefore, the speed of the prediction and training is important.
- Results that can be interpreted by a human facilitates the interpretation of the outcome.This criteria is the extent to which a human (including non-experts in machine learning) can understand the choices taken by models in their decision-making process (the how, why and what).

We have selected the most known supervised learning algorithms to compare them based on the above criteria (Table 2.2).

| Algorithm/criterion | Problem | Uncertainty | Heterogeneity | Multi-Class | Performance | Speed | Interpretability | Time series |
|---|---|---|---|---|---|---|---|---|
| Support Vector Machine (SVM) [64] | Classification | Good | Fair | Good | Good | Fast | Fair | Good |
| K Nearest Neighbor (KNN)[48] | Either | Bad | Good | Good | Poor | Good | Good | Good |
| Naive Bayes (NB) [93] | Classification | Good | Bad | Bad | Poor | Good | Fair | Good |
| Random Forest (RF) [9] | Either | Good | Good | Good | Excellent | Poor | Fair | Good |
| Decision Tree (DT) [65] | Either | Bad | Good | Good | Poor | Excellent | Fair | Bad |
| Neural Network (NN) [81] | Either | Good | Fair | Good | Excellent | Excellent | Bad | Good |

Table 2.2 Comparison among a set of supervised learning algorithms

As we can see in the Table 2.2, most of the selected algorithms are able to handle heterogeneity of data even if not directly, for example with SVM, but it is possible with some transformations. Some of the algorithms do not treat uncertainty such as KNN and Decision Tree. Regarding efficiency, SVM, Neural Network and Random forest are good candidates. Random Forest is slow for prediction, while Neural network provides results that can not be interpreted. Accordingly, we can suppose that SVM, and Neural Network can be used for sensor data analysis and processing regarding the mentioned requirements.

### 2.3.2 Knowledge based techniques

The knowledge based techniques formalize unstructured data with the guidance of domain experts. Ontologies and logic-based reasoning are the most used knowledge based techniques in the literature for analysis and processing data, as shown in Figure 2.3, mainly due to their ability to infer high level knowledge and handle uncertain and heterogeneous data. Each of these techniques can be used either solely or combined with other. On the one hand, ontology is considered as a powerful model to formalize a domain. On the other hand, logical reasoning provides an efficient mechanism to infer new knowledge based on given input.

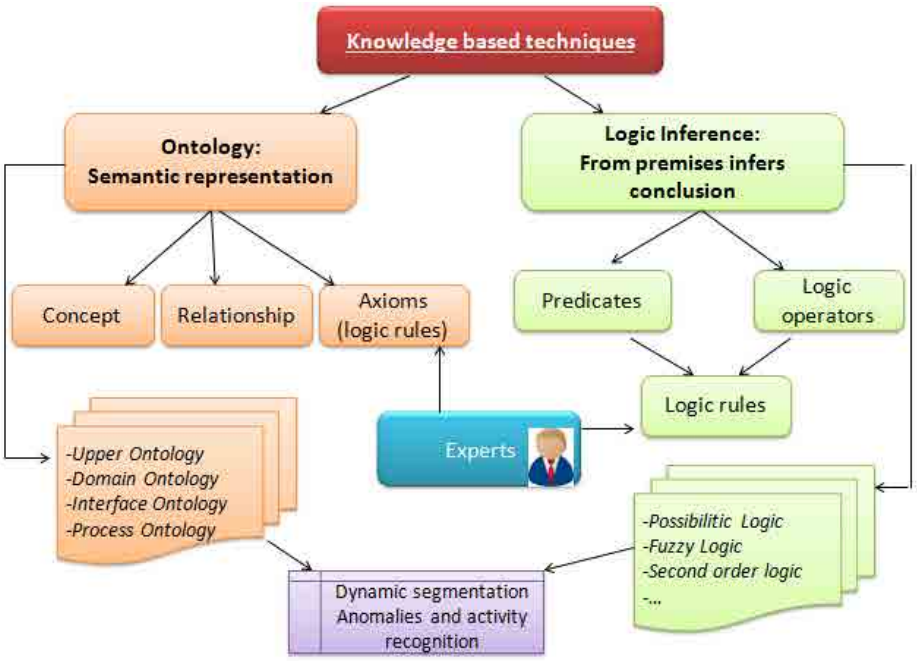



Figure 2.3 Knowledge based techniques overview

- Ontology is a mean to formally model the structure of a system, i.e., the relevant entities and relations that emerge from its observations, and which are useful to the defined purposes. The ontologies can be classified using the degree of abstraction. Hence, we can find upper ontology representing a meta-ontology, domain ontology which is defined to a particular topic, interface ontology that corresponds to the juncture of two disciplines, and finally the process ontology that is involved in business or engineering processes. Domain ontologies are the most used due to their formal abstraction of a particular domain. As examples of domain ontologies we cite: The MarineTLO ontology [2] which is a top-level ontology for the marine domain (also applicable to the terrestrial domain), and the Friend Of A Friend (FOAF) ontology[3] which is used to describe people and social relationship on the Web. It is mostly focused on people's existence in the virtual world, with many properties related to online activity or identity.

  Whatever its type, the ontology is basically composed of a set of concepts, relations, and axioms [80].

  - A concept can represent a variety of things. It can be an object of any sort: person, car, building, can describe an activity or state: swimming, being busy or available, abroad. It can also represent abstract concepts like time or value.

[2] http://www.ics.forth.gr/isl/MarineTLO/docs
[3] http://xmlns.com/foaf/spec/

There is no strict restriction what can express as a concept in the ontology. The only restriction is the real world which the ontology tries to reflect.

- A relationship in an ontology represents a way in which two concepts can be connected to each other. The connection may represent some allegiance: Dog is best friend of Man, Train needs Rails, characteristics of objects: Children are Young, Apples are Juicy, or activities: Policemen chase Criminals, etc.
- An axiom specifies expert's knowledge about the domain. They are generally defined through logic rules. For example, $Parent \equiv Human \cap \exists hasChild.\ Parent$ and $Human$ are concepts and $hasChild$ is a relationship.

- Logic and logical reasoning. Logic can be considered as a language able to infer conclusions from the given preconditions (i.e premises) by executing a set of rules. All of these elements are formalized using predicates and logical operators. The rules are defined by experts to express their knowledge. Several instances of logic language can be used according to the defined requirements such as Fuzzy logic to deal with fuzzy data and Possibilitic Logic to handle and propagate uncertainty, and so on.

Generally, this kind of techniques requires the assistance of experts for the definition of the ontology's elements and the logic rules. The logic rules are mandatory for both ontology and logic. They are the basis of logic languages and used to define axioms in the ontology. However, as rules are defined by human experts, conflicts may occur during their update or management, when different human experts contribute to their definition or when there is a contradiction in the sources of information. As a result, a conflict decreases the precision of the results and changes the behavior of the application. In order to study this issue, we classify conflicts into two main categories: (1) *Related rules conflicts* [7, 35] and (2) *Related events conflicts* [57, 10]. The *related rules conflicts* category is about detecting and resolving conflicting logical rules within the system. Resolving conflicts coming from rules evolution and change is recently studied in [7]. The authors proposed a dynamic system for nutrition recommendation for elderly people based on the recognized activities within their smart home. The system is built based on ontologies and logical rules. In order to resolve the conflicts that can occur during the rules evolution, the authors presented an algorithm that is triggered whenever any change of rules has been made. In the same vein, another work has been proposed by [35]. The authors designed a new method for detecting and resolving conflicting logical rules in wireless sensor networks. The proposed resolution method is based on the Feature Interaction (FI) techniques. In this work, different strategies for conflicts detection and resolution according to the nature of conflicts have been studied.

The category of *related events conflicts* states that the source of conflicts is not the rules format but the events in the environment that are the content of the rule. For example, this category of conflict can concern resolving conflicting activities of multiple users within a smart environment. To tackle this issue, the work in [10] introduced a method for resolving conflicts of multiple users who share context-aware applications within a smart home. Then, the authors proposed an ontology-based technique for conflicts resolution. Furthermore, another work proposed a reminder system for elderly person living in a smart home and a method to resolve the conflicts in the schedule coming from different events [57]. Conflicts in user contexts within ambient intelligent systems has been also studied in [83]. In fact, the authors introduced a formal model that enables the specification of an intelligent environment's components, whose interaction can lead to conflicts. these works have been resumed through the following table:

| Category | Interest | Issue |
|---|---|---|
| Related events conflicts | Conflicts come from events in the environment that are the contents of the rule | Does not handle conflicts coming from the format of the rules |
| Related rules conflicts | -Detection and resolution of conflicting rules in the system -The conflicts come from the rule formats | Manages conflicts in the rules format, but does not handle contrariness conflicts |

Table 2.3 Categories of conflicts handling

*Related rules conflicts* has not been studied enough in the literature. Each of the studied work previously mentioned tackles the problem of conflicting rules and proposes a solution that can be applied only into its particular case of study that is for example the wireless sensors network in [7]. Therefore, the surveyed works suffer from the lack of a general model to recognize and resolve the conflicting rules in rule-based systems. Moreover, previous works of *related rules conflict* do not tackle the problem of *contrariness* conflict which often occurs in smart environment.

### 2.3.3 Hybrid techniques

Hybrid techniques consist of combining both knowledge based and data driven techniques. The aim of this combination is to take advantages of both techniques in order to provide better results as confirmed in the survey [53].

This collaboration can be done through three different manners:

Firstly, some previous works have started the combination by a knowledge based technique in order to infer knowledge from the sensor data. Then, they apply a data driven algorithm to learn from the knowledge instead of learning directly from sensor data. This way is beneficial particularly when applying a supervised learning algorithm: The fact of transforming data to knowledge reduces the amount of data and accordingly reduces the effort of the expert for the labeling process.

Secondly, it is possible to start with the learning technique directly from data. Afterwards, the knowledge based method is used in order to extract or represent knowledge from the learning results. In this sense, we can say that unsupervised learning algorithms can be applied since they do not need expert intervention. Then, since they provide lower accuracy than the supervised ones, knowledge based technique may improve the results by extracting more knowledge.

Finally, the both categories of techniques can collaborate in an iterative way by taking input from each other. This way of collaboration fits perfectly with the reinforcement learning. Interactions are the core of reinforcement learning algorithms. When this interaction is realized with a knowledge based method, results may be improved due to the high level of the knowledge that can be given.

To conclude, hybrid techniques are the best candidates for processing and the analysis of sensor data in smart environments depending on how the hybridization is done.

After studying the different techniques that could be used for the processing and analysis of data, the next sections focus on the three problems tackled in this thesis: Activity recognition, anomaly detection, and sensor data segmentation. We review previous researches in these fields and outcome their limitations.

## 2.4 Tackled problems of the sensor data processing and analysis

After reviewing the different techniques used for the processing and analysis of sensor data, in this subsection we study the tackled problems in this thesis. In the following, we review previous works dealing with these problems, comparing them, and underlying their drawbacks.

### 2.4.1 Activity recognition

Recognizing user activity means knowing what the user is doing during his/her daily routine. Previous works in this field are classified into three main categories based on the applied techniques [53]: knowledge-based, Data-driven, and hybrid methods.

*knowledge-based methods*. Through these methods logic rules and reasoning engines are used in order to express expert knowledge to infer user activities based on current sensors input. These approaches have developed from earlier attempts in first-order logic [98, 6] towards a more formal logic model [63] that points to support effective thinking whereas keeping expressive control, to support formal analysis, and to preserve the soundness and

completeness of a logical system. Ontologies have also broadly applied to infer usual activities thanks to theirs powerful semantic representation of real world and reasoning capabilities [52]. Ontologies can provide a standard vocabulary of concepts to represent domain knowledge, specifications and semantic relationships of activities defined in formal logic approaches. They can also provide fully fledged reasoning engines to reason on them following axioms and constraints specified in formal logic approaches.

Even though these methods have achieved high recognition they are restricted to a limited number of sensors due to the difficult creation of antologies and definition of logic rules

*Data driven methods*. A set of Bayesian derivatives models are famously used for activity recognition (i.e naïve Bayes [30, 33] and Bayesian networks [97]) due to their ability to encode causal (dependence) and temporal relationships (i.e. Hidden Markov Models [11, 70] and Conditional Random Fields [31]). Inspired from the language modelling, grammar-based approaches (stochastic) like context free grammars are applied in representing the complex structural semantics of processes in hierarchical activities [23, 73, 84]. Decision trees [62], Neural Networks [46], and Support Vector Machine [64] as another branch in machine learning techniques, which are built on information entropy, have also been used to classify sensor data into activities based on features extracted from sensor data. Despite the good results achieved through the above data-driven techniques in activity recognition, they need a large amount of training data to set up a model and estimate their model parameters [53]. ,

Accordingly, nowadays, hybrid methods have faced an increasing interest for activity recognition due to their ability to overcome issue in both knowledge based and data driven techniques.

*Hybrid methods*. COSAR [20, 21] is a context-aware mobile application that combines machine learning techniques and an ontology. As a first step, the machine learning method is triggered in order to predict the most probable activities based on a provided training data. Then, an ontology reasoner is applied to refine the results by selecting the set of possible activities performed by a user based on his/her location acquired by a localization server. Despite the fact that the sensor data are supposed to be certain, COSAR deals with the uncertainty of the transformation of the localization from a physical format to a semantic one. Another hybrid model that combines a machine learning technique, an ontology, and a log-linear system has been proposed in [92]. The aim of this approach is to recognize a multilevel activity structure that holds 4 levels: atomic gesture (Level 4), manipulative gesture (Level 3), simple activity (Level 2), and complex activity (Level 1). The atomic gestures are recognized through the application of a machine learning technique. Moreover, using a probabilistic ontology defined by the log-linear, and standard ontological reasoning tasks, manipulative gestures, simple activities, and complex activities are inferred. Each

level is deduced based on a time window that contains elements from the previous level. Even though the work in [92] is similar to the previous one regarding the absence of sensor data uncertainty's handling, the inference of the 4 levels activities is based on a probabilistic reasoning that represents a type of uncertainty. FallRisk [26] is another pervasive system that combines data driven and knowledge based methods. Its main objective is to detect a fall of an elderly person living independently in a smart home. FallRisk is a platform that integrates several systems that use machine learning methods for fall detection. It filters the results of these systems thanks to the use of an ontology that stores the contextual information about the elderly person. The main advantage of this system is that it is extensible to integrate several fall detection systems. Moreover, the contextual information of the elderly is taken into account. However, this work does not consider any kind of uncertainty. FABER [22] is another pervasive system used to detect abnormal behavior of a patient. Firstly, it deduces events and actions from the acquired sensor data. This is done based on simple ontology inference methods. Then, these events and actions are sent to a Markov Logic Network (MLN) [69] to analyze the event logs and infer the start/end time of activities. The inferred activity boundaries are communicated together with actions and events to the knowledge based inference engine. This engine evaluates the rules modeling abnormal behaviors and detected abnormal behaviors are communicated to the hospital center for further analysis by the doctors. Nevertheless, similarly to previous works this system does not handle uncertainty of sensor data. SmartFABER [19] system is an extension and an improvement of FABER [22]. These two frameworks share the same aims. Regarding SmartFABER [19], instead of communicating the inferred events and actions to MLN classifier, the system sends them to a module in charge of building feature vectors based on the received events. Then, these features are communicated to a machine learning module for the classification of activities. Next, a proposed algorithm called SmartAggregation is applied to infer current activity instances. For deducing an activity's instance from a sequence of events classified to an activity, the algorithm verifies whether each event satisfies a set of conditions. These conditions are defined by a human expert after a deep analysis of the activity's semantic. If all events satisfy all conditions, then an activity instance could be inferred. This work is proved to outperform FABER [22]. However, it suffers from two main drawbacks: (1) There is no uncertainty handling for sensor data; (2) The performance of the SmartAggregation algorithm depends heavily on the defined conditions. It can suffer from time consuming if there is a huge number of conditions that need semantic verification.

Intuitively, as presented in Table 2.4, hybrid methods provide better recognition than using separately data driven or knowledge based techniques. However, most of hybrid methods in the state-of-the-art are missing the sensor data uncertainty handling.

| Approach | knowledge based technique | Data driven technique | Handling uncertainty of sensor data | Remarks |
|---|---|---|---|---|
| SmartFaber [19] | OWL2 ontology | Any supervised learning technique | No | This is the first paper that proposes a new algorithm to detect activity instances based on hybrid model |
| Faber [22] | First order logic | Markov Logic Network | No | — |
| FallRisk [26] | Ontology | Any data driven technique | No | This system aims specially to detect the fall of the user. It integrates several Fall detection systems and it filters their results |
| [92] | OWL2 ontology + logLinear + Description logic | Any data driven technique | No | This paper handles the uncertainty merely occurring in the knowledge based technique |
| COSAR [21] | Ontology | Any supervised learning technique | No | The combination of data driven and knowledge based techniques serves to filter the result of the data driven method |

Table 2.4 A resume of previous hybrid methods about activity recognition

### 2.4.2 Anomaly detection

The elderly constitute a group that deserves special attention to support their daily life. Even one fall can lead to a dangerous situation as the loss of consciousness and mobility. If not controlled frequently, a person may lay on the ground for many hours or even days without any help. A prompt action may save the life. Smart-home systems, which are currently getting very popular, may act as detectors of such anomalous situations and therefore be supportive for relatives and carers. The benefit is on both sides, the elderly would feel safer and the relatives would not be under such pressure to control the beneficiaries very frequently.

The general definition of an anomaly is a deviation from usual behavior. The concept of "behavior" differs from one application to another. For example, on the one hand, for daily activities, behavior is defined as the user's usual activities in his/her daily routine [2]. An anomaly can therefore be any abnormal activity. On the other side, for maritime situational awareness, behavior is defined as the normal conditions that should exist in the vessel [59]. An anomaly is then any unusual situation on the vessel. Another example can be that in some health-care applications [58, 39], behavior can be considered as the normal response to a treatment taken by a patient. An anomaly is then any abnormal reaction of a patient to the treatment. Hence, we can indicate that the decisions too are different in each case and depend on the purpose and the degree of the anomaly. For instance, in health-care applications, some anomalies need rapid intervention from doctors and others only require careful follow up of the patient. As we can see, what constitutes an anomaly varies widely depending on the target aim. In this study, we focus on detecting the anomalies for daily in-home activities. More precisely, behavior in our work is defined as the normal conditions that a user (e.g. elderly person) must verify in a smart home. For instance, when the user is on the phone, the stove must be closed, and when he/she is sleeping and it is cold, the window must be closed. Otherwise, an anomaly may be detected (fire or sickness, respectively). In the next paragraph, we will provide an overview of anomaly detection in previous studies on daily in-home activities. In the reviewed literature, various methods have been proposed to detect anomalies in daily in-home activities. These approaches can be grouped into three classes: point, collective, and context anomalies [34] .

*Point anomalies*: This class considers each activity independently and decides whether it is anomalous or not with respect to normal behavior. For detecting point anomalies, Han et al. [103] use the mean of various features for different activities and apply a classification method to define regular behavior. Then, they look for anomalous activities based on predefined thresholds of deviation. In [104, 67, 68] the authors learn which rooms the resident is in during different times of day. Using circadian rhythms and room location, they monitor anomalies regarding room occupancy. In [19, 22], the authors propose new systems of anomaly detection for mild cognitive impairment. In order to automatically calculate anomalies, they represent them as propositional logic. Then, according to rules defined by experts, the anomaly is detected as an activity containing a deviation from normal behavior. The work presented in [106] is an extension of [22], in which the rules describing anomalies are automatically generated through a new classification method.

*Collective anomalies*:This class considers groups of activity instances together to determine whether the group is normal or not. To do so, Anderson et al. [27] use an automaton-based approach to define a sequence of activities as normal behaviors and learn those behaviors. They also support combining multiple days of activities to detect anomalies that occur over time. In [101], the authors use unsupervised pattern-clustering techniques to identify the behavior model of the resident. Afterward, they apply a supervised machine learning technique to detect anomalous sequences of activities.

*Contextual anomalies*: This class, also named conditional anomalies, considers activities under some context [100, 42, 34]. In other words an activity can be normal in some context and anomalous in other ones. Holmes [34] uses a typical approach, combining point, context, and collective anomaly detection. The context considered in [34] is the day of week. In fact, Holmes starts by constructing a hierarchical normal behavior. At its bottom level are several regular behaviors grouped by day. They are then gathered to model the temporal correlations between activities. After training, anomalous activities are detected by computing their distance from the normal behavior. The context has been also considered for AD in other fields, including health-care applications like in [58]. In [58] the proposal is aimed to explain the patient's response to treatment. By taking into account the context, any abnormal response is considered as anomaly.

**Discussion**

In order to handle anomalies detection, we distinguish two cases. The first case is treated in the literature in contrast of the second case:

- **Case 1:** The activity and its context are considered as an abnormal pattern without making analysis of its causes. In other words, methods belonging to this case provide knowledge about the anomaly occurrence without the reason behind its occurrence.

- **Case 2:** The activity and its surrounding context are analyzed to detect both the anomaly and its causes. Such methods are helpful particularly for decision making process by providing knowledge about the anomaly and its causes. This knowledge allows decision making process to better decide about the suitable reactions.

For a clearer understanding of these two cases, let us consider the following situation: *The user is sleeping in his/her bedroom, it is cold and the bedroom's window is open*. The user's activity is sleeping, the context is the temperature (cold), the user location (bedroom), and the status of the bedroom's window (open). Through this situation, a context anomaly can be detected. Methods under *Case1* are limited to provide information about anomaly occurrence. The activity surrounding context is not analyzed, then it is not possible to know the exact causes of such anomaly. However, the methods belonging to *Case 2* analyze the activity and its context, detect the anomaly and extract its causes. Thus, they can infer that the anomaly occurred because the window is open. Hence, later this knowledge will allow the decision making process to provide more suitable reactions that is to close the window instead other reactions such as to change the user activity or his/her location.

To the best of our knowledge, most of previous researches about context anomaly detection for daily-in home activity focus on the first case where the notion of environment context is slightly considered without a deep analysis of it.

### 2.4.3 Sensor data segmentation

As we mentioned before, data segmentation is a required process before analyzing the data. Naturally, for each online application data are continuously generated. This stream can not be processed in one-pass. It needs to be divided into a set of segments. Afterwards, each segment can be separately analyzed. Basically, the segmentation can be based on the time or the number of sensor data. Time segmentation means that the size of the segment should be computed on time scale. It is simply the time occurrence difference between the last sensor data in the segment and the first one. On the other hand, the size of the segment can be the number of the sensor data. The time based segmentation is considered more suitable for online analysis than the sensor data number based segmentation. As a matter of fact, in this last case a segment can be very long waiting for the arrival of the next sensor data until the required size. In order to avoid this problem, we have chosen a time based segmentation of sensor data. Basically, there are two methods for data segmentation [100]. The first approach uses a fixed time segment (i.e. time-window) size as illustrated in Figure 2.4. As it is shown in the figure, we consider a data stream of pressure, temperature, and luminosity sensors is divided into a set of equal time window sizes (e.g. 5 min).

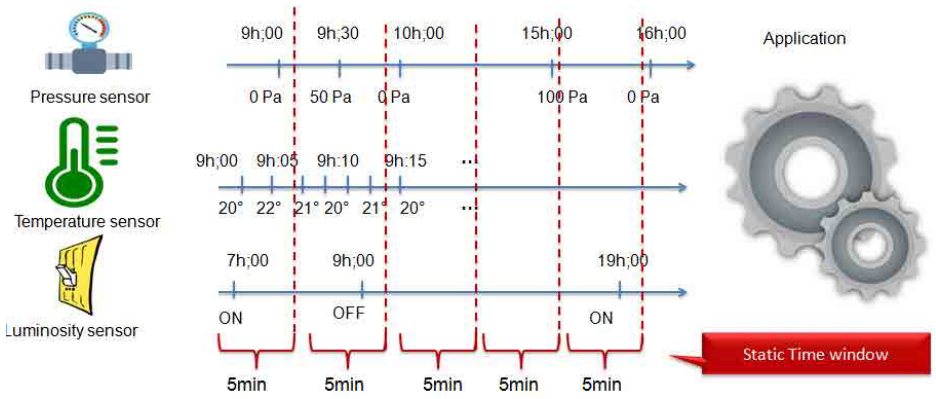



Figure 2.4 Static Segmentation example

This fixed size is selected after an empirical evaluation of the application with different segment size values. The value providing best accuracy is chosen. The main drawback of this method is that without empirical evaluation of the application, a bad choice of the time-window size (i.e. too short or too long) can lead to wrong results. For instance, when time-window size is too short or too long then the time-window either contains insufficient data to make right decisions or may hold so much data making the system confused.

The second method for data segmentation uses a dynamic time window size. In other words, the time window size can be changed according to a set of defined parameters. This flexibility regarding the choice of the segment size provides adapted size of data. Therefore, the dynamic data segmentation has proven better results than the static one [100] since it identifies the time points in a more flexible way. However, it is still a challenging problem. In this chapter we classify previous works of dynamic streaming sensor data segmentation into three main classes based on the techniques used so far: (1) metric-based methods, (2) data driven methods, and (3) knowledge-based methods.

*Metrics-based methods*: the general principle of these methods is that the window size is chosen according to the result of computed metrics such as mutual information [49, 78]. For activity recognition process, the authors in [49] propose a segmentation method that consists of dividing the sensor events into chunks according to the incidence of activity. To achieve this aim, they use the Pearson Product Moment Correlation (PMC) metric to compute the correlation between any pair of sensor events in the window. The authors in [78] propose to compute the mutual information between each couple of sensor events within a window and calculate a feature's weight. Only highly related sensor events are supposed to be in the same activity. Another work [12] proposes a similarity measure to segment a motion stream. The proposed measure is highly related to the body motion attributes. For accelerometer streaming data segmentation, the authors in [99] propose to classify the data according to

a set of features which are used as metrics such as the variance, mean, etc. The method is designed to be specific to accelerometer data.

*knowledge-based methods*. This class applies mainly semantic representation (e.g ontology) and semantic reasoning (e.g logic inference) to derive dynamically the window size. For example, the work in [37] proposes an ontology representing the activities that can be realized by the residents, the types of installed sensors in the smart home, etc. The dynamic segmentation consists of either expanding or shrinking the time-window by queering the ontology at the same time as recognizing the activity. The work in [24] proposes a knowledge based approach for segmenting sensor data series using ontologies to perform a terminology box (TBOX) and a assertion box (ABOX) reasoning, along with logical rules to infer whether the incoming sensor event is related to a given sequence of the activity. As in the previous work, the segmentation method is also integrated into the activity recognition method.

*data driven methods*. Under this class, methods use machine learning in order to learn the best window size regarding the coming sensor data. Generally, a pre-segmented dataset is required for the training phase (i.e a dataset containing the ground truth of time window sizes for the segments). The authors in [76] propose to use a probabilistic learning approach to identify, as a first step, the possible window size using a pre-segmented dataset. As a second step, they propose to learn the most likely window size for an activity based on the computed probabilities of the possible window sizes.

| Works | Category | Comments | Aim |
|---|---|---|---|
| [99] | Metric based | The used metrics are based only on the accelerometer data | Accelerometer data segmentation |
| [75] | Data driven | A segmented training data is supposed to be given for supervised learning | Activity recognition |
| [61] | Knowledge based | The segmentation and the activity recognition are integrated in the same process | Activity recognition |
| [49] | The results of the study based on an analysis of the literature shows that two cases can be distinguished based | The choice of the segmented size depends on primarily events related to the activity | Activity recognition |
| [12] | Metric based | The measure is highly related to the body motion attributes | Body motion recognition |

Table 2.5 A resume of previous works of dynamic streaming sensor data segmentation

To the best of our knowledge, as shown in Table 2.5, all the previous methods are designed to be used for a one type of application such activity or gesture recognition. However, it is more useful to conceive a flexible method that can be applied for different analysis goals (activity recognition, anomaly detection, etc.).

## 2.5 Conclusion

Throughout this chapter we have presented an overview about the smart environment and its data life cycle. In this thesis we focus mainly on the sensor data processing and the analysis. Hence, we have reviewed the different techniques that could be used for these two tasks: knowledge based, data driven, and hybrid techniques are the main techniques that could be applied.

Particularly we are interested in the problems of activity and anomaly recognition to analyze the data and the dynamic sensor data segmentation to process the data and. Hence, we discussed previous work about these three problems. Then, we have deduced the challenges being the motivations of our contributions.

In the next chapter, we present our contribution about activity recognition.

# Chapter 3

# AGACY Monitoring: a hybrid model for activity recognition and uncertainty handling

## 3.1 Introduction

Activity recognition methods serve to analyze a given segment of sensor data in order to provide knowledge about user daily activity (ADL). This ADL information is used to monitor and track the residents. A good number of on-going projects in smart environment for activity recognition such as the CASAS project [15], Opportunity [82], and CARE [8] make a declaration to the significance of this research field. The requirement for the advancement of such innovations is underscored by the maturing population, the expense of medicinal services and the significance that people place on staying independent in their own homes. People need to be able to complete ADLs such as cooking, watching-TV, and taking medicine, to lead an independent life. Thus, one of the important steps in providing a functional health of the smart home resident is the automation of the recognition and the monitoring of his/ her ADLs. This is the essential incitement driving a significant part of researches to the activity recognition in smart environments.

A precise activity recognition method is one of the important inspection of the end user. This precision can not be achieved without considering uncertainty of sensor data during the recognition process. In fact, sensor data are not always well-aimed and precise [53] due to hardware failure, energy depletion, etc. Therefore, the uncertainty is one of the issues of sensor data in smart environments. Thus, the activity recognizer should handle sensor data uncertainty in order to provide precise results. To recognize activities and handle sensor

data uncertainty, the main solutions can be generally classified as *semantic based*, *data driven based*, and *hybrid* approaches. According to the analysis realized in Chapter 2, a hybrid approach takes the "best of both worlds" by using a combination of methods. Such an approach is able to provide a formal, semantic and extensible model with the capability of dealing with uncertainty of sensor data and reasoning rules [53]. Therefore, proposing hybrid models has been the motivation of recent works for activity recognition including [19, 92, 26]. Nevertheless, the lack of sensor data uncertainty consideration is the main drawback of the aforementioned hybrid approaches. To overcome this limitation, this chapter proposes a new hybrid model combining data driven and semantic based methods for activity recognition and sensor data uncertainty handling. The main contributions of the chapter are as follows:

1. Proposition of the AGACY Monitoring hybrid model that integrates semantic and data driven based techniques for activity recognition. This novel approach handles uncertain sensor data and exploits these uncertainty values to compute the produced activities uncertainty values.

2. Improvement of an existing method for features extraction [75] in order to deal with more time distant complex events and their uncertainty values.

Before going further into the subject, it is worth briefly giving some definitions to better understand the approach.

## 3.2 Definitions

Event, complex event, and feature (long term and short term) are the main important terms of this chapter that should be formally defined for a better understanding of the approach

**Definition 2** ***Event Instance.*** *Let E be the set of event labels; T a set of all possible time-stamps; and C a set of uncertainty values. An event instance is a triple ev($e_i$, $t_i$, $c_i$) where $e_i \in E$, $t_i \in T$ and $c_i \in C$.*

**Definition 3** ***Complex Event Instance.*** *Let CE be the set of complex events labels, T the set of all possible timestamps, C the set of uncertainty values. A complex event instance is a triple cev($ce_i$, $t_i$, $c_i$) where $cev_i \in CE$, $t_i \in T$ and $c_i \in C$.*

**Definition 4** ***Short Term Features (STF)*** *is a set of features that holds only features having duration less than ten minutes. Let $Dur(f)$ be the duration of feature $f$: $f \in STF \Leftrightarrow Dur(f) <= 600s$*

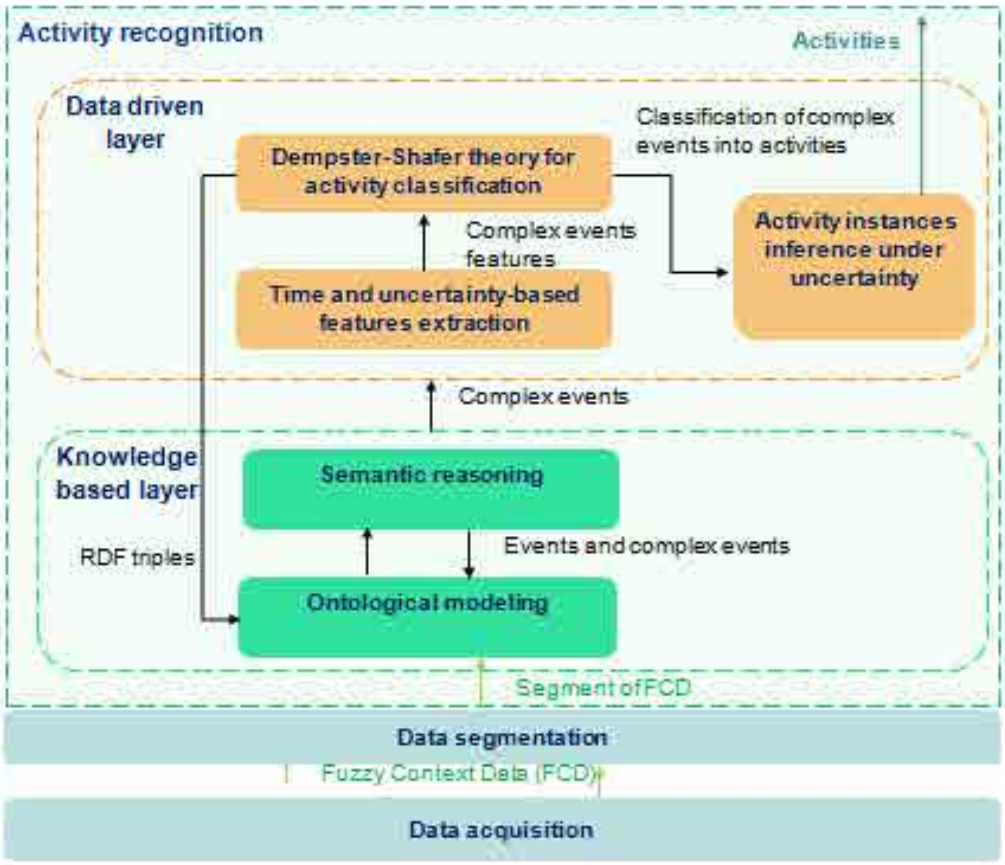



Figure 3.1 The architecture of the AGACY Monitoring

**Definition 5** ***Long Term Features (LTF)*** *is a set of features that holds only features having duration more than ten minutes. Let Dur(f) be the duration of feature f:* $f \in LTF \Leftrightarrow Dur(f) > 600s$

It is important to mention that the value $600s$ is chosen according the dataset used to evaluate the method. However, for the seek of generality, this threshold can be selected after a set of empirical evaluations with different value of thresholds.

## 3.3 The AGACY Monitoring architecture overview

The overall architecture is composed of two layers: the Knowledge based layer and the Data driven layer, as depicted in Figure 3.1. One of the data acquisition steps is to collect the sensor data in order to be stored in computers and compute their uncertainty (i.e. FSCEP system [5]). This process is out of the scope of this chapter. We assume that the sensor data together with their uncertainty values are provided to the segmenta-

tion step. The latter, divides the coming data into segments and sends each one to the knowledge-based layer. In this chapter we apply a static segmentation (i.e. the segment size is fixed). This layer represents semantically the sensor data, belonging to the given segment, together with their uncertainty values (`Ontological modeling step`). Afterwards, it infers complex events and events from the modeled sensor data (`Semantic reasoning step`) and computes their uncertainty's values. The obtained complex events and the computed uncertainty values are then sent to the data driven layer. This layer is responsible for: (1) extracting features from the given complex events (`Time & Uncertainty-based features extraction step`), (2) classification of complex events into activities regarding the extracted features (`Dempster Shafer theory for activity classification`), and finally (3) inferring activities' instances (`Activities instances inferring under uncertainty step`). In the following, further explanation of each layer is provided.

## 3.4 Knowledge-based layer

### 3.4.1 Ontological modeling

The ontological modeling module allows sensor data to be formally conceptualized. This conceptualization, as the ontology is defined for, serves to provide a semantic model from the data. However, the native ontological representation is known to poorly handle uncertainty. Therefore, some attempts for uncertainty integration into ontological models have been realized [1, 41].

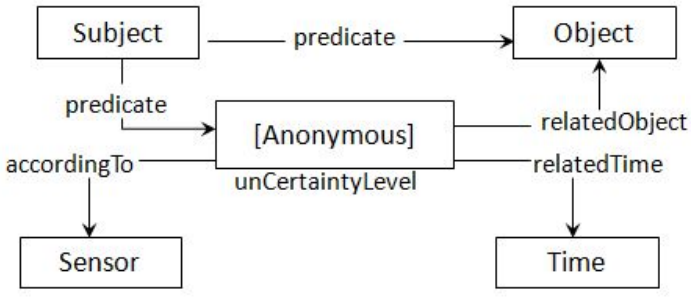



Figure 3.2 The extended uncertainty representation model

In this work, we adopt the model proposed in [41] since it could be attached to any existing ontology without the need for redesigning it. However, the problem of this model is the lack of temporal representation. In order to distinguish two similar sensor data that have the same uncertainty value but come in two different timestamps, it is highly important to

associate an uncertainty value (assigned to a sensor data) with the unique time. Hence, to overcome this problem, we extended the model in [41] by adding a temporal element (the class Time). Our proposed model is depicted in Figure 3.2.

*Anonymous resource* is related to the class: *Subject* through the ObjectProperty: *predicate* and to the *Object* through the ObjectProperty: *relatedObject*. It holds the uncertainty level of the triple $< Subject; predicate; Object >$, according to the model in Figure 3.2, using its dataProperty: *unCertaintyLevel*. It is also linked to the source of uncertainty – namely a sensor or a set of sensors that derive uncertainty– with the objectProperty: *accordingTo*, and to class: *Time* via the objectProperty: *relatedTime*.

Through this model, the uncertainty can be easily integrated into any ontology. To do so, we have designed an ontology to represent sensors, sensor data, sensor data uncertainty values, complex events, events, persons, time, and so on. Figure 3.3 shows an excerpt of this ontology, in which the model from Figure 3.2 is used in order to represent the uncertainty. As Figure 3.3 depicts, among the basic high level concepts in the ontology we find: *Sensor*, *ComplexEvent*, *Event*, *Activity*, *Person*, *Object*, and *Time*.

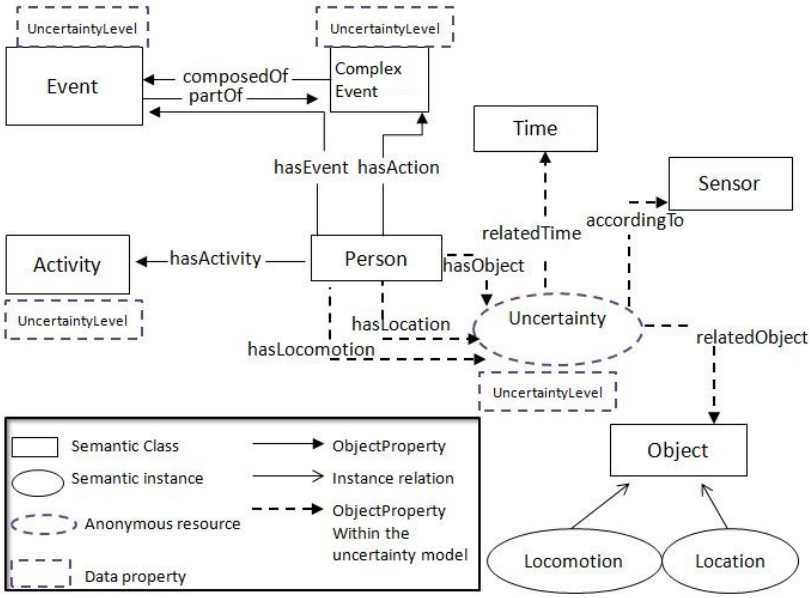



Figure 3.3 An excerpt of the ontology with uncertainty integration

Events and complex events have also values of uncertainty that are represented as dataProperties. Next subsection will describe how the events and complex event are inferred and how their uncertainty values are computed.

$\forall$ $se_1$, $se_2 \in \{Sensors\}$, $t_1$, $t_2 \in \{Time\}$, and p $\in \{Person\}$
(p hasLocomotion **[ a Uncertainty; uncertaintyLevel $n_1$; relatedObject SitOn; relatedTime $t_1$; accordingTo $se_1$ ]**
$\wedge$
(p hasObject **[ a Uncertainty; uncertaintyLevel $n_2$; relatedObject Chair; relatedTime $t_2$; accordingTo $se_2$]**)
$\longrightarrow$ ev(SitOnChair, max($t_1$,$t_2$), min($n_1$, $n_2$))

Example of rule for inferring the event instance sitOnChair

### 3.4.2 Semantic reasoning

This module is mainly responsible for the inference of events and complex event and their uncertainty values. Events are deduced from sensor data and complex event are inferred from events.

The events are deduced by combining the *logic with possibilities* [18] with the *modified semantic logic rules* in [41]. On the one hand, in [41] the traditional semantic logic rules are modified in order to integrate the uncertainty in a clause's definition. This is done according to the ontological model in Figure 3.2 (without the concept Time). Then, this modified rule is able to propagate the uncertainty value from the premise to the conclusion of the rule. However, the propagation is done from only one clause within the premise. Thus, the case when the rule contains more than one clause is omitted. On the other hand, the *logic with possibilities* has the advantage to propagate values of possibilities (in our case values of uncertainty (denoted $Unc$ in Equation 3.1) in a rule from a premise with multiple clauses to the conclusion. This is done through the following rules:

$$\begin{aligned} Unc(p \wedge q) = min(Unc(p), Unc(q));\ Unc(p \vee q) = max(Unc(p), Unc(q)) \\ Unc(\neg p) = 1 - Unc(p);\ Unc(p \longrightarrow q) = max(1 - Unc(p), Unc(q)) \end{aligned} \tag{3.1}$$

Accordingly, in order to deduce the events, we combine the *logic with possibilities* with the *modified semantic logic rules* where the premise is a set of clauses linked with logic operators. A clause is a RDF triple translated in logic [41]. This RDF triple models the sensor data with their uncertainty according to the ontology depicted in Figure 3.3. The conclusion is an event. To propagate the clauses' uncertainty towards the conclusion, the rules in Equation 3.1 are applied.

In this example, we demonstrate how to infer an instance of event labeled *sitOnChair*:

As we can see, the premise of this rule contains 2 clauses. Each one is a RDF triple representing an uncertain sensor data according to the ontology depicted in Figure 3.3. The first clause means "*the resident p is observed to have the locomotion SitOn at time $t_1$ with uncertaintyLevel $n_1$ according to the sensor $se_1$*" while the second clause translates the

information "*the resident p is observed to get the object Chair at time $t_2$ with uncertaintyLevel $n_2$ according to the sensor $se_2$*". The rule's output always creates a new event with an uncertainty value deduced according to rules described in Equation 3.1. Then, a production of the event with label *sitOnChair* is deduced. Since the premise is a conjunction of clauses, its uncertainty value is the minimum between both values of clauses' uncertainty. Accordingly, this new event will be modeled in the ontology as an instance of the concept *Event* and the uncertainty value as a dataProperty of the instance.

After deducing events, the complex events are inferred. The complex events and their uncertainty values are inferred by simply applying the logic with possibilities rules. The complex event is the conclusion of a rule, the clauses of its premise are the events or some defined restrictions.

This example is to illustrate how we can deduce the production of a complex event's instance with label *SitOnChairAtKitchenTable*:

ev(*SitOnChair*, $t_1$, $n_1$) $\wedge$ ev(*PresenceAtKitchenTable*, $t_2$, $n_2$) $\wedge$ ($t_1 \geq t_2$) $\wedge$ (($t_1$-$t_2$) $\leq$ 5s ) $\longrightarrow$ (*SitOnChairAtKitchenTable*, $t_2$, min($n_1$,$n_2$))

The previous rules are examples of rules that are managed by the system. However, a set of required rules are defined by experts according to the semantic of the activities to be monitored and the types of sensors in the smart environment. Other rules can be inferred through an ontological inference engine according to the axioms defined in the ontology.

The deduced complex events are then communicated to the module `Time & Uncertainty - based features extraction` described in the following section.

## 3.5 Data driven layer

### 3.5.1 Time & Uncertainty-based features extraction

For each received complex event cev($ce_i$, $t_i$, $c_i$), this module is in charge of building a feature vector representing the sequence S of the recent actions in a time window with a size equal to n. S =$\langle$cev($ce_{i-n+1}$, $t_{i-n+1}$, $c_{i-n+1}$),..., cev($ce_{i-1}$, $t_{i-1}$, $c_{i-1}$), cev($ce_i$, $t_i$, $c_i$)$\rangle$

In this work, we improve the technique proposed in [75] for features extraction. We have chosen this method since it is proved to be effective in recognizing activities based on streams of sensor events or actions instead of streams of sensor data compared to traditional features extraction techniques. This technique builds a vector of features for each events sequence S. The produced vector of feature of a sequence of events $S_i$ holds these information: (i) The label of the feature, $K_i$; (ii) The time $t_0$ of the first event in $S_i$; (iii) The time $t_i$ of the last event in the sequence $S_i$; (iv) The list of events under $S_i$; (v) A weight value fine-tunes the

contribution of each event in $S_i$, so that recent events participate more than the older ones. This weight value is computed as follows (Equation 3.2):

$$F_{k_i}(S_i) = \sum_{ev(e_j,t_j)\in S_i} exp(-\chi(t_i - t_j)) \times f_{k_i}(ev(e_j,t_j)) \tag{3.2}$$

where the factor $\chi$ determines the time-based decay rate of the events in $S_i$; $t_i$-$t_j$ is expressed in seconds and $f_{k_i}(ev(e_j, t_j))$ is the time-independent participation of $ev(e_j, t_j)$ in the computation of the $F_{k_i}$ value. The other way around if $ev(e_j,t_j)$ participates in the execution of $k_i$ then $f_{k_i}(ev(e_j,t_j))$ = 1 else 0.

As we can see in Equation 3.2, the greater difference is of the time distance between $ev(e_j, t_j)$ and $ev(e_i, t_i)$), the less $ev(e_j, t_j)$ participates in the computation of $F_{k_i}$.

In our understanding, this may be true when the approximate duration of the feature is short (e.g. in terms of seconds or some minutes). In contrast, when it is long (e.g. in terms of hours), this hypothesis is not always valid. Let us have the example of the feature "stove usage" in [19]. The duration of the execution of this feature, for a particular recipe, may be equal to a number of hours. We suppose that the vector of this feature contains the following two events *openStove* and *closeStove*. Since the duration of the feature "stove usage" is in terms of hours, the value of ($t_{closeStove}$-$t_{openStove}$) may be equal to 1 or 2 hours. Accordingly, by applying Equation 3.2 the event *openStove* does not participate in the weight calculus of the feature "stove usage". Therefore, incorrect value of weight may be obtained. Accordingly, the execution of the event *openStove* must have a high impact in the execution of the feature. Intuitively, the stove can not be used if it is not opened. Moreover, this technique assumes that the only factor that may have impact on the computation of the feature's weight is the time distance between events. However, when information about actions and events uncertainty values is provided, this information should be taken into consideration in the computation of the feature's weight. Therefore, we made the following assumptions:

**A1.** The uncertainty values must have an impact on the calculus of the feature weight: the higher uncertainty value of event or complex event is, the more the weight of the feature increases.

**A2.** A product containing the uncertainty value of an event or complex event, must be higher or equal than this uncertainty value: The uncertainty value must not be decreased when it is multiplied.

To overcome the problem mentioned above, we propose a new version of Equation 3.2 that meets the assumptions A1 and A2. Hence, to compute the weight of the feature, we propose

the following Equation 3.3.

$$F_{k_i}(S_i) = \sum_{ac(a_j,t_j,c_j)\in S_i} c_j \times Fact_{\chi,\delta t_{ij}} \times f_{k_i}(ac(a_j,t_j,c_j)) \tag{3.3}$$

$$Fact_{\chi,\delta t_{ij}} = \begin{cases} exp(-\chi * \delta^h t_{ij}) & If\ (k_i \in LTF) \\ \frac{1}{\chi * \delta^s t_{ij}} & If((\chi * \delta^s t_{ij} \succ 1)\ \&\ (k_i \in STF)) \\ 1 & Otherwise \end{cases}$$

It is worth mentioning that in our work the features vectors are built from complex events instead of events, in contrast to [75] and the size n of the time window is chosen according to the nature of the feature (STF or LTF). We note that $c_j \times Fact_{\chi,\delta t_{ij}}$, in Equation 3.3, is a sort of uncertainty where $c_j \times Fact_{\chi,\delta t_{ij}} \leq c_j$ (Assumption A2). $\delta t_{ij} = t_i - t_j$. $\delta^h t_{ij}$ means that $\delta t_{ij}$ is expressed in hours and $\delta^s t_{ij}$ means that $\delta t_{ij}$ is expressed in seconds. To fix the problem of the time delay in Equation 3.2, we distinguish three cases: (1) The feature is a LTF ($k_i \in LTF$). Then, to compute $Fact_{\chi,\delta t_{ij}}$ the same function ($exp(-\chi * \delta t_{ij})$) in Equation 3.2 is used. However, $\delta t_{ij}$ is expressed in hours ($\delta^h t_{ij}$) instead of seconds. Accordingly, the actions that happened earlier could participate in the weight computation in contrast to Equation 3.2; (2) The feature is a STF ($k_i \in STF$) and $\chi * \delta t_{ij} \succ 1$. Then, $Fact_{\chi,\delta t_{ij}} = \frac{1}{\chi * \delta^s t_{ij}}$. The quotient function is chosen to compute $Fact_{\chi,\delta t_{ij}}$ since it has a less decreasing shape than the exponentiation function. (3) If $(\chi * \delta^s t_{ij}) \prec 1$, then $\frac{1}{\chi * \delta^s t_{ij}} \succ 1$ and accordingly $c_j \times Fact_{\chi,\delta t_{ij}} \succ c_i$ that does not correspond to Assumption A2. Therefore, $Fact_{\chi,\delta^s t_{ij}} = 1$. This is due to the fact that since $t_i$ and $t_j$ are very close, $ac(a_j,t_j,c_j)$ must have the highest impact on the weight computation, i.e. the value 1 in [0..1] is chosen. As a result, computed weights of the feature are used as their uncertainty values. Afterwards, the actions and the features together with their uncertainty values are sent to `Dempster-Shafer theory for activity classification` module.

### 3.5.2 Dempster-Shafer theory for activity classification

In our work, the Dempster-Shafer theory [105] is applied for activities classification. Recently, this theory is gaining an increasing interest in the field of activity recognition and uncertainty propagation, since it proved to have better results in comparison to Naive Bayes classifier and J48 Decision Tree [36]. Usually, a Directed Acyclic Graph (DAG) is used to represent the sources evidences, their hypothesis, the mass functions, the activities, and to support the distribution and the fusion of evidences. In DAG, evidence sources represent the nodes at the base of DAG. Evidence source readings are mapped to one or more hypothesis. Each one in turn will be mapped to one or more activities. In our work, DAG is used in which an

evidence source is a set of complex events (e.g the complex events belonging the vectors of the features). These complex events represent the frame of discernment in DS. The hypothesis are the labels of the features. Finally, the mass function computes the number of the occurrences of the complex events during the execution of the activity [36]. A complex event having a positive mass value is called a focal element. The mass value of an hypothesis obtained by the propagation rule of DS is discounted with the computed weight of this feature ($F_{k_i}$) (Equation 3.3) after being normalized. The value of this product regroups the uncertainty value of the feature (its weight value) and the uncertainty value of the classification of the features to activities (mass value). This final value of uncertainty is propagated towards activities thanks to the Dempster-Shafer's rule of combination.

Please note that DS could be replaced by any classification method that support uncertainty handling, such as Support Vector Machine [64] (SVM).

### 3.5.3 Activities instances inferring under uncertainty

In this section, we present *AGACY*, our algorithm for activity instances inference. The algorithm aims at improving existing one called *SmartAggregation* [19]. We improve this method since it is accurate and it improves the well known algorithm Naive aggregation [19]. The basic idea of the *SmartAggregation* algorithm is the following: if two consecutive events that occurred respectively at $t_i$ and $t_{i+1}$ are classified with the same activity's class $A_i = A_{i+1}$ and verify the defined conditions of the activity $A_i$, $C^{(A)}$, they are considered as an observation[1] generated by the same activity instance of the class $A_i$. Otherwise, if an event does not satisfy all the conditions, the algorithm tries to integrate it into an observation of a previous inferred instance providing that it satisfies all the conditions defined by human experts. Hence, this algorithm, despite its accuracy, suffers from scalability and time consuming. Assuming a big number of events, the algorithm must check each event for satisfying all the conditions, which is demanding and time consuming. To tackle these problems, we refine our recognition method by the proposed Algorithm 1. It takes advantage of the obtained uncertainty values of the activities to infer the current instances of activity. The basic idea of Algorithm 1 is the following: a group of actions could be considered as an observation of an activity instance providing that (1) all the actions have uncertainty values beyond a defined minimum value, e.g MINUNCERT, (Lines 2-9) and (2) the time distance between every pair of consecutive actions within the group must be lower than $maxDelay_A$. The second condition is always true since the first step done by the algorithm is a segmentation over the output of the DS (Line 1): All the actions associated with the same

[1] An observation is a sequence of events that is generated by an activity instance.

**Algorithm 1:** AGACY

**Input:** G, minUncert, tolUncert, $\beta_{delay}$, $maxDelay_A$, A
**Output :** Inst: a set of activity instances

1: $\alpha \longleftarrow segmentation(G, maxDelay_A)$;
2: $Inst \longleftarrow \phi$;
3: $LowUncert \longleftarrow \alpha$;
4: **for** each g in $\alpha$ **do**
5: **if** $(\forall ac(a_i,t_i,c_i) \in g \longrightarrow c_i \succ minUncer)$ **then**
6: $inst \longleftarrow$activity {instance of A that is generated for the observation g}
7: $Inst \longleftarrow Inst \cup inst$;
8: $LowUncert \longleftarrow lowUncert \backslash g$;
9: $G \longleftarrow G \backslash all\ actions\ in\ g$;
10: **else**
11: $missClass \longleftarrow \phi$;
12: $ratioUncert \longleftarrow ComputeRatioCert(g)$;
13: **if** $ratioUncert \succ tolUncert$ **then**
14: $missClass \longleftarrow missClass \cup GetLowUncert(g)$;
15: $K \longleftarrow 0$;
16: $toBeReplaced \longleftarrow \phi$;
17: **while** $(k \prec (card(G)) \& (card(toBeReplaced) \prec card(missClass))$ **do**
18: **if** $(ac(a_k,t_k,ck) \in G) \& (ac(a_k,t_k,ck) \notin g) \& (c_k \succ minUncert) \& (\forall ac(a_j,t_j,c_j) \in g \longrightarrow |t_j - t_k| \prec maxDelay_A + \beta_{delay})$ **then**
19: $toBeReplaced \longleftarrow toBeReplaced \cup ac(a_k,t_k,c_k)$;
20: **end if**
21: $K \longleftarrow K+1$;
22: **end while**
23: **end if**
24: **end if**
25: **if** card(toBeReplaced) = card(missClass) **then**
26: $g \longleftarrow g \backslash missClass$;
27: $g \longleftarrow g \cup toBeReplaced$;
28: $inst \longleftarrow$ activity{instance of A that is generated for the observation g}
29: $Inst \longleftarrow Inst \cup inst$;
30: $lowUncert \longleftarrow lowUncert \backslash g$;
31: Partition(missClass, $\alpha$, $maxDelay_A$);
32: $G \longleftarrow G \backslash \{all\ actions\ in\ g\}$;
33: **end if**
34: **end for**

activity class A and, that are temporally close (according to $maxDelay_A$), are merged together to obtain a set of groups $\alpha$. G is a set of all actions without segmentation. In this case, there are no conditions to be verified for the actions compared to *SmartAggregation* algorithm. Indeed, the uncertainty value of each complex event is replaced, before the execution of the algorithm, with the uncertainty value of its classified activity class. Therefore, when the uncertainty value $c$ of a complex event *cev* is high, it is almost sure that the complex event *cev* is really produced and belongs to the correct class of activity *act*, hence, must belong to an instance of act.

This process is less time consuming than *SmartAggregation* algorithm. On the other hand, when at least one complex event in the group has an uncertainty value lower than MINUNCERT, the algorithm accepts some time distance shift between actions regarding the uncertainty degree (Lines 10-32). To do so, it checks firstly if the ratio of the number of actions (RATIOUNCERT) in the group that have lower uncertainty value than MINUNCERT is

higher than a defined value called TOLUNCERT (Line 13). If so, the algorithm attempts to replace the actions that have lower uncertainty values with other actions from other groups. A new complex event can be added if it has higher uncertainty value than MINUNCERT and the time distance between the complex event and each complex event in the group is lower than the $MaxDelay_A + \beta_{delay}$ (Lines 18 and 19). $\beta_{delay}$ represents a sort of tolerance about the time distance between two complex events. Thus, we assume that uncertainty value is more important than the time distance. Finally, when all uncertain complex events are replaced with actions having high uncertainty values with some time distance tolerance, this new group can generate a new activity instance (Line 27). Afterwards, the removed complex events having lower uncertainty values than MINUNCERT in the old group are distributed to the rest of the groups according to $maxDelay_A$ (Line 31).

## 3.6 Experimental evaluations

In order to proove the effectiveness of AGACY Monitoring for activity recognition, we have realized a set of experiments. In this section, we first describe the dataset used for the experiments. Then, the experiments and the achieved results are presented.

### 3.6.1 Dataset

We used real-life data collected in highly rich smart environments. The dataset[2] [82] was obtained as a part of the Opportunity project "[3]. This dataset focuses on activities concerning breakfast that holds (from AGACY Monitoring perspective) homogeneous sensor data (level 4), simple events (level 3), complex events (level 2), and activities (level 1) that have been done by three persons S10, S11, and S12 with three different routines each (ADL1-3). In order to test AGACY Monitoring system, this dataset does not contain information about sensor data uncertainty values (level 4). Therefore, we have randomly annotated the level 4 in the dataset with high uncertainty values in [0.8..1]. Moreover, we have injected a set erroneous sensor data in the dataset annotated with low uncertainty values in [0..0.4]. In the experimentation, we have tested the performance of the system with different number of injected erroneous sensor data.

[2]The dataset: http://webmind.dico.unimi.it/care/annotations.zip
[3]http://www.opportunity-project.eu

### 3.6.2 Implementation and experimental setup

We have implemented the AGACY Monitoring system with JAVA. Regarding the data driven layer, for this dataset, we have considered the set of features depicted in Table 3.1 to be treated in the *Time & Uncertainty based features extraction* module.

| Feature Name | STF/LTF | Duration (s) |
|---|---|---|
| PrepareCoffee | STF | 600 |
| Drink | STF | 120 |
| GatherCutlery | STF | 600 |
| GatherFood | STF | 600 |
| Eat | STF | 1200 |
| PutAwayFood | STF | 600 |
| PutAwayCutl | STF | 600 |
| Dishwasher | STF | 300 |
| Resting | STF | 600 |

Table 3.1 List of considered features. STF: Short Term Feature, LTF: Long Term feature

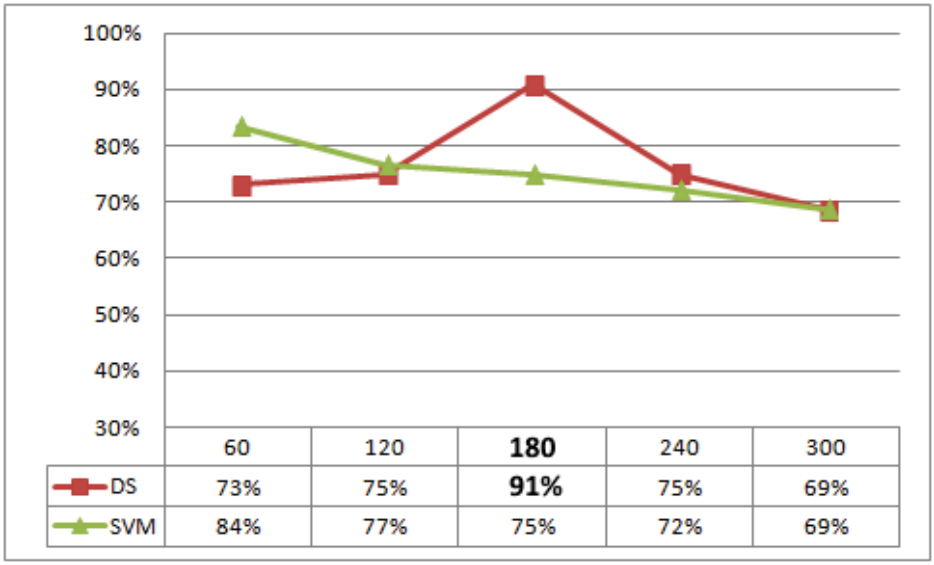
For the *Activity classification* module, the experiments are not limited only on the use of DS – we have also used the SVM for activity classification. The results obtained from this module are compared with those obtained by Rim, H., and al. [92]. Finally, regarding the *Activities' instances inferring under uncertainty* the algorithm AGACY has 4 variables: MINCERT, TOLCERT, MAXDELAY$_A$, $\beta_{delay}$. After an empirical evaluation of these parameters, the most adequate values regarding these experiments are as follows: MINCERT= 0.5; TOLCERT= 1; MAXDELAY$_A$= 30s; $\beta_{delay}$ = 10s;

### 3.6.3 Evaluation and results

#### Activity recognition evaluation

As described previously, the *Activity classification* module outputs the predicted activity class. It is worth mentioning that the AGACY Monitoring has no False Negative result (FN=0): it outputs always at least one activity. Figure 3.4 depicts the average precision measure over the three routines for all subjects by varying the value n of the time window (Tw) in [60s..300s] with one erroneous sensor data for five correct ones (e.g 1/5 erroneous sensor data). The code source of AGACY Monitoring is available online[4]. As it is clearly shown in Figure 3.4, the DS with n=180s reaches 91% of precision recognition rate. For time windows shorter or longer than 180, DS tends to become less efficient: DS is efficient where time window are

[4]https://github.com/Nath-R/AGACY-monitoring



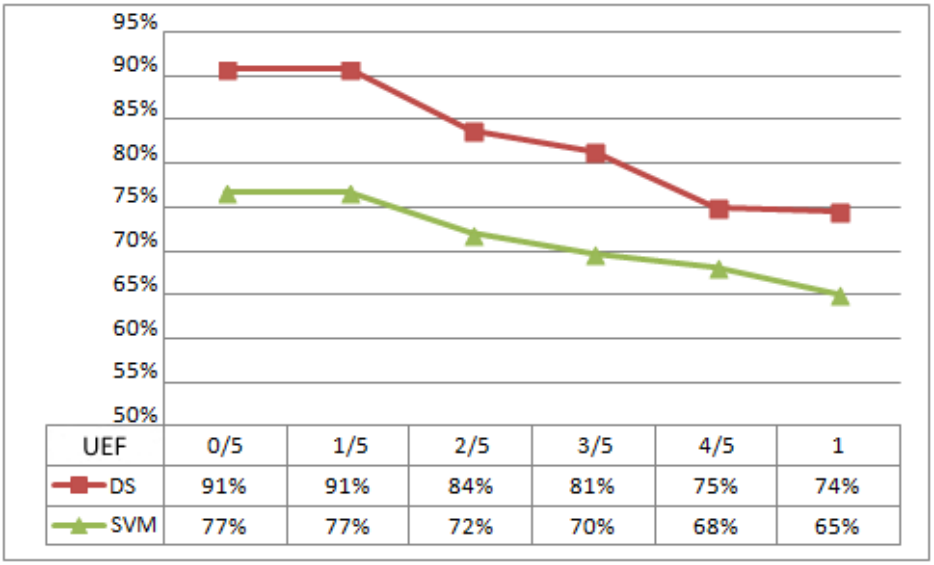

| | 60 | 120 | **180** | 240 | 300 |
|---|---|---|---|---|---|
| DS | 73% | 75% | **91%** | 75% | 69% |
| SVM | 84% | 77% | 75% | 72% | 69% |

Figure 3.4 Average recognition precision for all subjects over the three routines with different values of the size n of the time window (Tw)

properly proportional to the activities: when the time window is too big, there may be conflict between activities, while when it is too small, DS does not have enough data to be efficient. On the other hand, SVM gives better results for short time window (n<=120s), but with the increase of n value, the accuracy of classification gets worse. This maybe explained by the fact that SVM is not as dependent on features weights as DS. In general, DS provides better results than SVM. Figure 3.5 shows the average precision measure over the three routines for all subjects by varying the proportion of the introduced uncertain sensor data compared to the correct sensor data with n=180s. As it is clearly shown in the figure the DS is more efficient than SVM: The precision values of DS are in range [0,74..0,91], however that of SVM are in range [0,65..0,77]. Both methods have a decreasing precision values when the number of uncertain sensor data in the dataset increases. This is an expected result since the methods will have less certain data to make the right decision. However, despite the dataset half contains uncertain sensor data, the system is able to predict the activity with a good precision level (74%).

| All users | AGACY Monitoring | [92] |
|---|---|---|
| Precision | 0,91 | 0,91 |
| recall | 1 | 0,65 |

Table 3.2 Average recognition results over three routines for the three subjects obtained by AGACY Monitoring (with DS) and the system proposed in [92] for n=180s.




| UEF | 0/5 | 1/5 | 2/5 | 3/5 | 4/5 | 1 |
|---|---|---|---|---|---|---|
| DS | 91% | 91% | 84% | 81% | 75% | 74% |
| SVM | 77% | 77% | 72% | 70% | 68% | 65% |

Figure 3.5 Average recognition precision for all subjects over the three routines for varying frequency of uncertain event (UEF). The value $1/5$ means there is one uncertain sensor data for five correct ones. 1 means there is one uncertain sensor data for one correct

We have compared the obtained results with another activity recognizer proposed in [92]. The method has been applied with the same dataset (without erroneous sensor data and uncertainty values).

As it is shown in Table 3.2 the two methods have the same average precision recognition. However, AGACY Monitoring outperforms the second system within recall value. This can be explained by the fact that the system in [92] returns null if it can not infer an activity. However, AGACY Monitoring has the advantage to always predict an activity. Moreover, the proposed AGACY Monitoring system is effective despite the introduced erroneous sensor data in the dataset. This confirms the ability of AGACY Monitoring to effectively handle uncertainty, predict activities, and adapted to the different user routines.

**Evaluation of AGACY algorithm for activities instances inference.**

In this paragraph, we show the evaluation of the AGACY algorithm for inferring the activities instances. Furthermore, we have evaluated the system for time execution. We tested the AGACY algorithm, on the output of the activities classification module with DS (n=180s and the value $1/5$ of erroneous sensor data in the dataset). Moreover, we have implemented and tested the SmartAggregation algorithm [19] with the same dataset without uncertainty annotations. The average time consummation results over the three routines of both AGACY

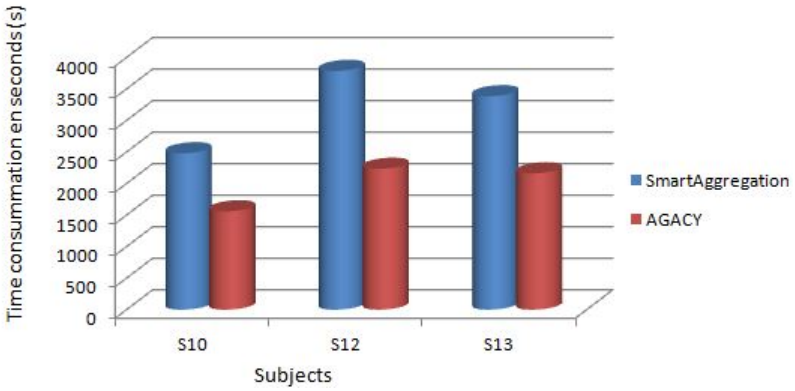



Figure 3.6 Average time execution of activity instances recognition for all subjects over the three routines

and SmartAggregation are presented in Figure 3.6. The results regarding average accuracy of activities instances detection are depicted in Table 3.3.

| All users | AGACY | SmartAgg. |
|---|---|---|
| Precision | 0.916 | 0.854 |
| Recall | 0.830 | 0.872 |
| F-score | 0.871 | 0.862 |

Table 3.3 Average recognition results for activities instances detection over three routines for subjects S10, S11, and S12.

Using above-mentioned experimentation, we obtained two major results: firstly, as it is highlighted in Figure 3.6, AGACY has significantly reduced the time execution for activities' instances recognition regarding *SmartAggregation*. Approximately, for the used dataset, AGACY reduces 40% of *SmartAggregation*'s time consummation. Secondly, in terms of performance both algorithms are efficient with similar F-score value. As a result, the experiments have proved that AGACY is fast and efficient.

## 3.7 Conclusion

In this chapter, we proposed the AGACY Monitoring hybrid model for activity recognition and sensor data uncertainty handling. The main novelty of AGACY Monitoring is that it combines oriented knowledge and learning techniques. Thus, several modules contribute to compute the uncertainty value of the expected output. Unlike the related work, AGACY Monitoring supports the inherent uncertain nature of sensor data and exploits it to compute the uncertainty value of each module's output. Besides, the system is able to integrate

background knowledge with the learning technique for the recognition of current activity instances. Moreover, the experimental results confirm the viability and the performance of the proposed system and its high level precision even during the presence of uncertain sensor data.

Now that the stream data is statically segmented and that for each segment current user activity is recognized, it becomes possible to decide about the risk of anomaly and propose required actions to prevent it. This will be the focus of the next chapter.

# Chapter 4

# Caredas: Early Anomaly Detection in Smart Homes, a Causal Association Rule-based Approach

## 4.1 Introduction

Reliable early anomaly detection (AD) in daily in-home situations is the most important application of many home health-care systems. The general definition of an anomaly in health-care applications is *a deviation from normal behavior* [34, 19, 106]. Normal behavior is a model of the usual user activities classified under features using different machine learning techniques [53]. An anomaly is then detected as a deviation from this model either by the application of logical rules defined by experts, or through the application of various data driven methods [50]. As a result, the detected anomaly can be an unusual activity or group of activities that can be analyzed by considering some context [34]. In the literature review, it is stated that the anomaly detection in daily in-home activities remains immature in the smart home when compared with other fields such as computer security, manufacturing defect detection, medical image processing, etc [74]. Moreover, few works were found that consider the context in addition to the activity in order to detect anomalies [34, 100, 42]. As affirmed in [100], an activity can be abnormal in some context and normal in another. Therefore, context plays an important role in anomaly detection. For example, while cooking, a user may neglect to open the window or switch on the exhaust vent, which could cause breathing problems and leads to a suffocation anomaly for a frail person. In this example, without considering the context, *a closed window and vent* when the user is *cooking* is difficult to detect as an anomaly since the activity is normal if considered without the

context. Another essential issue when dealing with sensor data is uncertainty management. Furthermore, most previous works are detecting the anomaly as an abnormal patter which leads to the inability to explain the reason behind the anomaly occurrence. We believe that it is very important, when detecting the risk of anomaly, to be able to explain the reason behind this risk. This allows quicker decision making to prevent the anomaly. To achieve this aim, the anomaly causes should be extracted firstly and used afterwards to reason about the risk of anomaly and to be given as explanation about the detect risk. Hence, our goal is threefold: (1) introduction of predictive capabilities by early anomaly detection of user situations, (2) recognition of anomaly causes and definition of actions to avoid predicted anomalies, and (3) better exploitation of data uncertainty.

To address these objectives, we first propose to use a hybrid method, namely, *Markov Logic Network* (MLN) [69] which combines probabilistic learning with first order logic. The MLN is known to be an appealing tool to perform calculations with logical rules under uncertainty. In this sense, it seems suitable for achieving our goals. However, a crucial shortcoming is that it requires as input all possible constants, which is a hard constraint in our case where the data is constantly changing. Secondly, to discover anomaly causes and the corresponding actions, we propose a method that mines the history dataset using the causal association rule mining method (CAR) [94]. This method has proven to be an effective and versatile means for causal relationship discovery using local causal structures of a causal Bayesian network (CBN) to explain anomalies. However, the main drawbacks of the studies that use this method are that they generally either focus on a subset of causality structures [94], or the causality structures are learned, which leads to high computational cost [51, 14, 16].

In order to go into more detail and devise solutions to the aforementioned issues, some highly specific research questions must be asked. These questions as well as their answers may overlap; nevertheless, they allow us to lay out our work explicitly:

- Q1 how to improve the MLN method in order to make it more adaptable to highly dynamic data and avoid the need for manual initialization of constants
- Q2 how to consider the various local causality structures of the CBN
- Q3 how to handle data uncertainty in the process

Finding solutions to these research questions relies on the following contributions, which will be explained in detail later in the chapter:

- A new method for CAR discovery named causal association rule mining algorithm (CARMA), which is able to extract causal association rules that can be represented in

the different structures of a CBN. This method is based on a proposed general measure of evaluating the interestingness of a CAR according to its structure. This proposal will be used in order to automatically extract the anomaly causes and corrective actions as the first step (offline). This automatic process will avoid the need for expert intervention to define the causes and the anomalies in the MLN rules.

- An improvement of MLN named DgMLN, for dynamic ground MLN, enabling more flexibility through the creation of the ground MLN automatically from the continuously received data and the extracted CAR instead of being given manually as input.
- Caredas, a method for early anomaly detection and relevant action generation based on DgMLN and the extracted CARs. It takes as input a situation and infers the risk of anomaly. It monitors real-time incoming data using *time windows*.
- A model for data uncertainty management for the entire process, starting from sensor events to causal anomaly detection. In particular, rule weights are dynamically computed according to the data uncertainty.

## 4.2 Background and Related Work

Through this section we review two areas specific to this chapter: the first subsection introduces the MLN method and the second presents the basics of association rules and previous work in CAR mining.

### 4.2.1 Markov Logic Network

Unlike other machine learning methods that are purely probabilistic, an MLN [69] provides many benefits and allows calculation under uncertainty, imperfection, and inconsistency with first order logic, an expressive piece of classical logic, and handles uncertainty. As a matter of fact, first order logic can be seen as a set of hard constraints on the set of possible worlds: if a world violates even one formula, it has zero probability. A world failing to satisfy even a single formula would not be possible. However, in most real-world scenarios, particularly for resident's monitoring in smart homes, logical formulas are not always true since sensor data can be imprecise or uncertain. The advantage of Markov logic over first-order logic is that it allows us to handle this uncertainty to a greater extent [72]. These characteristics make the MLN method an appealing tool for performing calculation to detect a risk of anomalous situations and recommend appropriate actions. On the one hand, our method starts by extracting causal association rules for anomalies offline. Since the MLN combines

logic and probabilistic reasoning, it has sufficient flexibility to process the extracted CARs. On the other hand, our approach has to detect the risk of anomaly occurrence in real time. For this reason, the MLN is useful since the calculation can be done online using a time window and the degree of risk is computed thanks to its ability to handle uncertainty. Moreover, thanks to these advantages, the MLN has been widely used for AD in different contexts. For example, in the context of maritime situational awareness, in [59], the authors use an MLN to detect anomalous conditions that can appear in maritime situations using low-level sensory data. This data is translated into MLN evidence predicates about the *vessel* such as stopped, *overlap* and so on. In another context of dementia detection based on user behavior, the work in [55] uses an MLN to recognize normal user activities and then detect any deviation as an abnormality. Another work in the same context [2] applied an MLN to detect unusual user activity. Generally, the MLN used in these studies lacks dynamicity for its grounding, especially for real-time analysis of incoming sensor data. More details about the MLN and this problem are given in the following paragraphs.

Formally, a MLN is a finite set of pairs ($F_i$,$w_i$); $1 \leq i \leq n$, where each $F_i$ is a function-free first-order logic rule and $w_i \in R$ is its weight. Together with a finite set of constants C = $c_1 ... c_n$ it defines the ground MLN, i.e., the MLN in which logical rules do not contain any free variable. Hence, an MLN defines a log-linear probability distribution over Herbrand interpretations (possible worlds):

$$P(x) = \frac{1}{Z} exp\left(\sum_{i}^{F} w_i n_i(x)\right) \tag{4.1}$$

where F is the set of rules in the MLN, $n_i(x)$ is the number of true groundings of $F_i$ in the possible world x, $w_i$ is the weight of $F_i$, and Z is the normalizing constant. In addition, an MLN is represented as a graph in which nodes are the rules predicates and a link between two nodes means that the predicates are in the same rule.

In general, the MLN suffers from three main shortcomings: (1) scalability, (2) dealing with numerical constraints, and (3) static nature of the ground MLN. Regarding the first issue, the work in [40] proposed a scalable MLN that can be applied to a big social graph. In order to overcome the second issue, the $MLN_{NC}$ [71] extends the MLN with numerical constraints in the logical rules. The third problem of the MLN concerns the static nature of the ground MLN. The ground MLN should be given as input which instantiates the predicates with all possible constants. Moreover, the rule weights are considered as evidence. However, this is unsuitable in our case of smart homes because: (1) The constants in the ground MLN depend on the given context data for the smart home, which changes over time. Therefore, in this case, the ground MLN needs to be continually updating, which is very demanding and results

in continuous expert intervention. (2) The weights of rules depend on the given weights of the contextual data.

### 4.2.2 Association Rules and Causal Association Rules

The first part of this subsection introduces the general ideas of association rules (AR), which is required for understanding CAR, which is detailed in second part.

**Association rules:**

Traditionally, association rule mining is used to find frequent itemsets among historical transactions. Moreover, it allows discovering unknown relationships so as to provide information for decision making or prediction [47]. To assess the strength of an association rule, two conventional criteria are used the *support* and the *confidence*. The support of an association rule, denoted by $S(A \rightarrow B)$, is the ratio of transactions that contain both its antecedent (A) and its consequent (B) while the confidence of $A \rightarrow B$, denoted by $C(A \rightarrow B)$, is the probability $P(B|A)$ of finding the consequent B given the antecedent A. Therefore, the support measures the unexpectedness of the rule, and the confidence measures its reliability. Only association rules that meet the specified minimum user support and confidence are of interest [94].

**Causal association rules:**

AR may not indicate causal relationships; only co-occurrence. However, in our context, on the one hand, activities and context data are causes of anomalies and, on the other hand, anomalies are causes of actions. Therefore, we need to evaluate the rules based on their ability to express causality. This implies that CAR will be more useful than AR in our proposal. In this work, we are interested in using CAR for early anomaly detection combined with an MLN. In fact, using only an MLN for early anomaly detection such as in [44] requires that all possible anomaly causes be defined by expert rules. This is a tedious task that requires effort from experts and good knowledge about the environment in which the user lives. However, this task can be simplified by automatically extracting anomaly causes using CAR from a dataset obtained after monitoring the user for a period of time, and then annotated by experts with user anomalies and required action. Afterward, these causes can be used directly by the MLN.

The CAR extends AR to be able to express a sort of causality among items instead of only co-occurrence. Causal relationships not only indicate that two variables are related, but also specify their interaction. Taking it one step further, the causal mechanism ensures that changes in a causal variable directly cause changes in the effect variable. To achieve this objective, most studies combine the concepts of the causal Bayesian network (CBN) with AR [94, 16, 43, 85, 17]. The idea of this combination is to evaluate the causality expressiveness

of a rule according to the local causality structures of a CBN. For better insight, figure 4.1 shows an example illustrating the local causality structures of a CBN for anomaly causes.

{c1, c2, c3, c4} is the set of possible causes and *a* is an anomaly. This CBN has four local structures. In Figure 4.1(b,c,d), the variable c3 is conditionally independent from variable c1 (resp. c4) given the variable a (i.e . c1⊥c3|a, c4⊥c3|a). However, Figure 4.1(a) implies a different assertion which is c1 is conditionally dependent on c2 given a, referred to as the V-Structure in CBN. Generally, on the one side, previous works which propose CARM method based on interestingness measure considered only subset of structures [94]. On the other hand, studies considering all structures have high computational cost [51, 14, 16, 43, 85], since causality structures are learned from data. In our opinion, CARM based on interestingness measure is more suitable. However, the measure must be able to evaluate the CAR according to the different structures of the CBN instead of only a subset. Indeed, important information may be provided when the interestingness measure is general enough to evaluate the different causality structures represented by the CAR. For instance, in our case study, as we mentioned before, the CAR will be used to extract anomaly causes and actions. Indeed, in some respects, actions are considered themselves as causes of anomalies, which may produce a conflict. For example, in Figure 4.2, a CBN represents a set of nine variables {c1, c2, c3, c4, c5, a1, a2, ac1, ac2} with different structures ((a), (b), and (c) of Figure 4.1). The set {c1, c2, c3, c4, c5} expresses the anomaly causes from context data and activities, and {ac1, ac2} are the required actions according to the anomalies {a1, a2}.

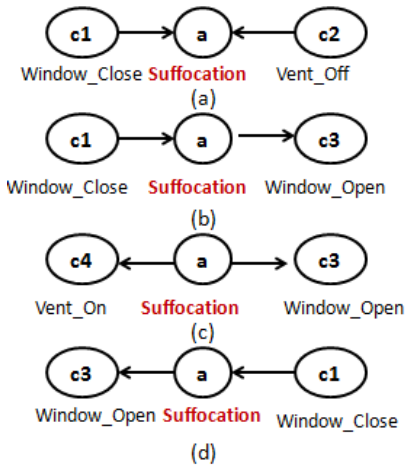



Figure 4.1 Local Causality Structures of CBN: (b–d) conditional independence equivalent; (a) V-structure

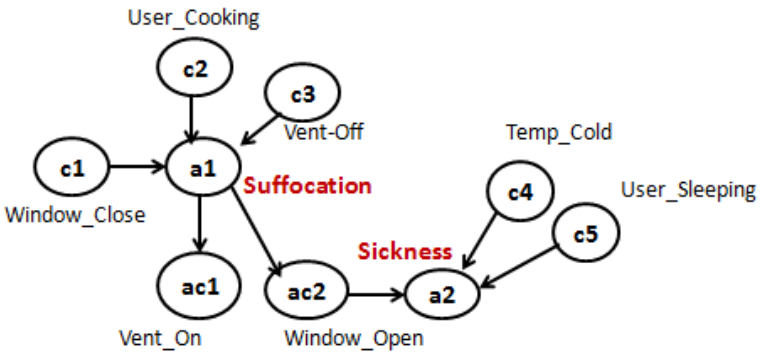



Figure 4.2 Example of CBN representing anomalies, anomalies causes, and actions

As we can see in Figure 4.2, the action *window_Open* which must be done to prevent the anomaly Suffocation, can cause the risk of another anomaly Sickness. In other words, if the user is cooking (c2) and the vent is shut down (c3) and the nearby window is closed (c1), then he/she risks suffocation (a1). Therefore, it is recommended either to open the window (ac2) or to open the vent (ac1). However, if the window is opened and it is cold (c4), then the user risks sickness if he/she is sleeping (c5). Accordingly, thanks to the different local causality structures of the CBN, the method that can detect this type of conflict.

## 4.3 Definitions

In this section, we define some concepts that are required to better understand in the rest of the chapter. Let us start with the notion of fuzzy context data (FCD). An FCD is data obtained from a sensor's observation of the environment. It contains multiple possible values, each associated with a trust weight. More formally, it can be defined as follows:

**Definition 6** *An FCD is a 3-tuple* $(subj, wVal, time)$ *where* $subj$ *is an entity of the environment, and wVal is a set of pairs representing all possible values with their trust weight. Each value of the subject is called context data.* $time$ *is the date observation of this FCD.*

FCD is monitored by FSCEP (Fuzzy Semantic Complex Event Processing) which produces sensor events with uncertainty values [5, 4]. This task concerns the acquisition step of the sensor data life cycle.

**Example 1** *(Temperature,* $\langle(high, 0.5), (medium, 0.38), (low, 0.12)\rangle$*, 2017-11-04 00:03:50) is an FCD in which the* $subj$ *is Temperature, wVal is* $\langle(high, 0.5), (medium, 0.38), (low, 0.12)\rangle$ *representing the possible values of* $subj$ *with their weights. Finally 2017-11-04 00:03:50 is the* $time$*.*

Now, an *Activity (ACT)* is a high-level context point data representing a user task. Formally, an ACT can be defined as follows:

**Definition 7** *An ACT is a 3-tuple act(l,* $t_s$*, u), where l is the label of the ACT,* $t_s$ *is its start time, and u is its uncertainty value.*

***Example 2*** *act(Cooking, 2017-11-04 02:56:55, 0.87) is an activity labeled Cooking that starts at 2017-11-04 02:56:55, and the system recognizes its occurrence with an uncertainty of 0.87.*

Our proposal AGACY Monitoring, presented in Chapter 3, is applied to recognize activities which is able to provide activities with uncertainty values. The uncertainty values of both the FCD and activities are used by the MLN during the probabilistic reasoning.

Next, we define the concept of *Situation* as a set holding an ACT and FCD that occur during a time window given by the segmentation step. A more formal definition is as follows:

**Definition 8** *A situation is a 3-tuple sit(act, fcd, win) where win is the time window size for this situation, act is the activity of the user during win, and fcd is the observed fuzzy contextual data win.*

**Example 3**

$$sit(act(Cooking, 2017-11-0402:56:55, 0.87), \begin{bmatrix} (Stove, <(On, 0.9), (Off, 0.1)>, \\ 2017-11-0400:03:50); (UserLocation, \\ (Kitchen, 1)); (KitchenWindow, (Open, 1)) \end{bmatrix} 4min)$$

*is an example of a situation in which the activity of the user is cooking, the set of fcd provides information about the state of the stove, KitchenWindow, and pinpoints the user location, that is, the kitchen. This information about activity and fcd is acquired during a 4-min time window.*

A situation is considered anomalous, hereby termed *anomalous situation*, if its uncertainty value of its classification as an anomaly, which corresponds to a computed weight of a predicate in the MLN rules, is above a threshold. This threshold is defined by expert knowledge.

The proposed method returns anomalies with weights that are above the threshold. The anomaly having the higher weight is considered. This means that the current situation risks producing this anomaly. When a risk of anomaly is detected, the method also recommends a suitable action to perform for the prevention of this detected anomaly.

Finally, an *action* is the reaction that must be performed in order to avoid the occurrence of an anomaly. Generally, an action produces a change in the context data state, from Window_Close to Window_Open.

## 4.4 Methods

The overall architecture of the proposed method, named Caredas, for risk anomaly detection, as depicted in Figure 4.3, is composed of two layers: the *Perception & Recognition layer* and the *Detection & Decision layer*.

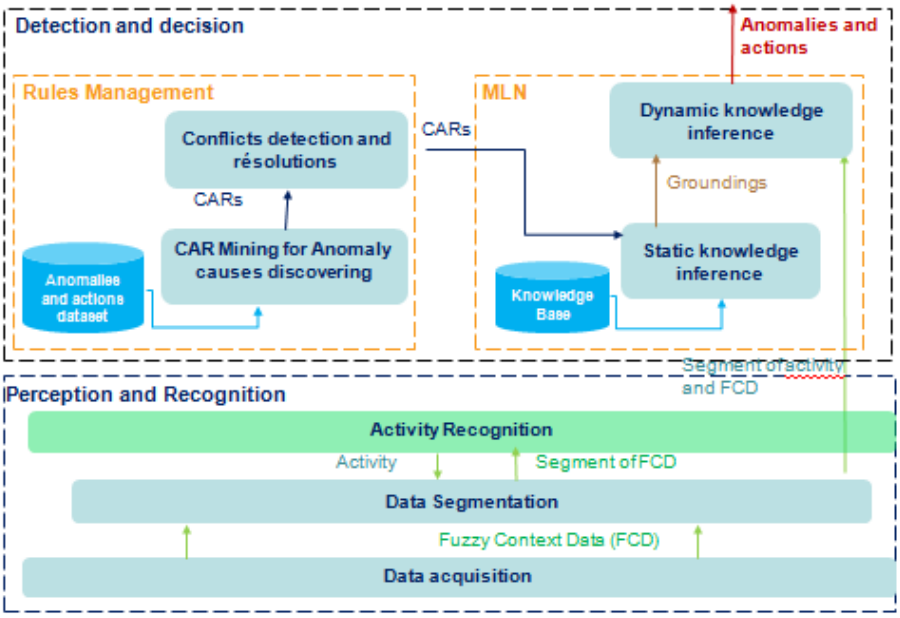



Figure 4.3 CAREDAS architecture

The former is in charge of the acquisition of data received from sensors in the smart home. In addition, it is responsible for the segmentation of data and the recognition of user activities based on the acquired data. On the one hand, similar to AGACY Monitoringn a static segmentation is applied. On the other hand, The activity recognition process is achieved using our proposal AGACY Monitoring system, providing activities as defined in Definition 7. In other words, the input row sensor data is translated into FCD (Definition 6) through the data acquisition process in Figure 4.3. Afterward, these FCDs are statically segmented and each segment is analyzed through the Activity Recognition process (cf. Figure 4.3) in order to recognize daily in-home user activities (cf. Definition 6). This layer produces the input (segment of FCD and activity) for the Detection & Decision layer, which holds the contributions of the present work. After receiving this input from the Perception & Recognition layer and a dataset of anomalies and actions provided by an expert, in the detection & Decision layer we start by mining the anomalies and actions dataset in order to extract CARs through CAR mining for anomaly cause extraction by applying the proposed CARMA method. Afterward, since conflicts may occur in the CAR, we apply our proposal method (see appendix A.1) to detect and resolve these conflicts (conflict detection and resolution). These two actions are performed in the Rules Management process. The latter is performed offline. Indeed, it is executed once a new dataset is given to extract the CAR from and detect and resolve conflicts or when the dataset is updated to refresh the extracted CAR and check the conflict after the change.

At the end of this process, the CARs are used by the MLN process. The latter attempts to compute online the risk of the current situation (see Definition 18) being anomalous. Moreover, it is able to decide which suitable action to perform in case of an anomalous situation. This goal is achieved through two components: (1) static knowledge inference for instantiating a specific type of predicates into MLN rules with knowledge given by CAR, which is almost static and (2) dynamic knowledge inference, which instantiates the other types of predicates into MLN from the dynamic data that holds the current situation (built according to Definition 10). At this stage, the proposed dynamic creation of a Ground MLN (named DgMLN) from the situation and the CAR is completed. Afterwards, it becomes possible to detect the risk of anomalies and make a decision about actions, through the probabilistic reasoning of the MLN and a particular knowledge base containing the MLN rules and predicates. In the following subsections, rule management and MLN processes are detailed.

### 4.4.1 Rules Management

In this subsection, we first explain the CAR for discovering anomaly causes and action mechanisms from the dataset. Afterwards, we present a method for the detection and resolution of conflicts in the CAR.

#### CAR for Discovering Anomaly Causes

Based on the provided dataset of anomalies and actions, this process aims at extracting the CARs that are, on the one hand, about anomaly causes and, on the other hand, about required actions.

**Example 4** *For a clearer understanding of the possible content of such datasets, Figure 4.4 shows an excerpt of a dataset acquired from an intelligent environment platform called HadapTIC*[1]

*As we can see in the Figure, the dataset holds user activities (e.g. sleeping) and context data about user localization, stove state (e.g. off or on), door state (e.g. open or close), etc. Moreover, the dataset is annotated by experts with possible anomalies and the required actions to prevent them. For instance, line 30 in Figure 4.4 is the annotation for the risk of discomfort anomaly and the corresponding action, which is door closed. As we observe in line 30 of the figure, experts agree that when the user is sleeping, the temperature is normal (e.g. 15°), and if the door is open then the user risks to be in an uncomfortable*

[1] http://hadaptic.telecom-sudparis.eu/

```
11214 223 locatedIn_user_bedroom 0.4
299805 227 locatedIn_user_livingroom 0.4
264348 232 temperature_bedroom_13.4 0.5
299805 233 User_locatedIn_livingroom 0.4
299805 244 User_locatedIn_livingroom 0.4
299805 249 User_locatedIn_livingroom 0.4
299805 254 User_locatedIn_livingroom 0.4
299805 259 User_locatedIn_livingroom 0.4
299805 272 User_locatedIn_livingroom 0.4
299805 285 User_locatedIn_livingroom 0.4
907838 289 door_open 0.6
11214 287 User_locatedIn_bedroom 0.4
11214 293 User_locatedIn_bedroom 0.4
263515 299 temperature_bedroom_15 0.8
AGACY User_Sleeping 09
Annotation: AGACY User_Sleeping User_locatedIn_bedroom door_open temperature_bedroom_15: Anomaly_Discomfort Action_door_closed
```

Figure 4.4 An excerpt of dataset of activities and sensor data annotated with anomalies and actions

*situation. Hence, the action to perform in order to avoid this is to close the door. This is an example of an action. We note here that actions depend on the nature of the anomaly. Some anomalies can be avoided by performing simple actions, and some others require emergency intervention. In our case study, calling for an emergency is considered an action even if it is not a direct action that can be performed in the smart home like closing the stove or the door.*

More formally, we define a CAR as follows:

**Definition 9** *(CAR) Let Context = {$c_1$, $c_2$, ..., $c_n$}, Anomalies = {$a_1$,$a_2$, ..., $a_m$}, and Actions = {$ac_1$,$ac_2$, ..., $ac_s$} be the sets of context data (FCD values and activities), anomalies, and actions belonging to the dataset, respectively. Then, a CAR can take three forms:*

$$
\begin{aligned}
&Case1: && c_1, c_2, \ldots, c_p \rightarrow a;\ p \geq 1 \ or \\
&Case2: && a \rightarrow ac_1, ac_2, \ldots, ac_m;\ m \geq 1 \ or \\
&Case3: && c_1, c_2, \ldots, c_l, a \rightarrow ac;\ l \geq 1
\end{aligned}
$$

A CAR may have either multiple antecedents and only one consequent ("Case1" and "Case3") or vice versa ("Case2") conforming to the local causality structures of the CBN. Indeed, "Case1" corresponds to the structure (a) of Figure 4.1, meaning one or more contexts cause an anomaly. "Case2" represents the structure (c) of Figure 5.1, in which one anomaly causes one or more actions. Finally, "Case3" is about the format of structures (d) and (b) of Figure 5.1, where one or more context data points that are the causes of an anomaly imply (or cause) an action. It is important to mention here that these are the possible formats of the CAR that will be extracted by CARMA and do not contain various logical operators like negative and XOR operators. This can be explained by the fact that causal association rule extraction algorithms are classified as a statistical data mining method. Therefore, this kind of method naturally was not conceived to perform logical reasoning. However, it can

be extended either by extra verification or with logical inference to extract more logically complex rules. Even if we find a negation operator in Equation 2, it does not signify a logic operator; it is just used for probability computation. In this work, we do not focus on this aspect of logical reasoning for CAR mining.

To compute the interestingness of a CAR $r$ regarding the three cases, we propose the following new measure called *CausalSignificance(r)* (*CSign(r)*, for short).

$$Csign(r) = \begin{cases} (1)\ P(a|c_1 \wedge c_2) + P(a|c_1) + P(a|c_2) - P(a|\neg c_1 \wedge a_2) - P(a|c_1 \wedge \neg c_2) & if\ r = c_1, c_2 \rightarrow a \\ (2)\ P(ac_1|a) + P(ac_2|a) - P(ac_1 \wedge ac_2|a) - P(ac_1|a) * P(ac_2|a) & if\ r = a \rightarrow ac_1, ac_2 \\ (3)\ P(ac|a) + P(a|c) - P(ac \wedge c|a) - P(c|a) * P(ac|a) & if\ r = c, a \rightarrow ac \end{cases} \tag{4.2}$$

where $c_i \in$ Context, a $\in$ Anomalies, and $ac_i \in$ Actions.

This measure is defined for the simplest condition in a CAR definition when the rule has two antecedents or two consequents. Regarding (1), the measure evaluates the rule $r$ according to the property of a V-structure: both $c_1$ and $c_2$ are highly associated with a, i.e, the role of $P(a|c_1 \wedge c_2) + P(a|c_1) + P(a|c_2)$ ), and conditionally dependent on $a$, i.e, the role of $P(a|\neg c_1 \wedge a_2) - P(a|c_1 \wedge \neg c_2)$. This means that $c_1$ and $c_2$ must occur together with $a$ and the measure decreases the interestingness of the rule otherwise. In (2), each of $ac$ and $c$ is associated with a, i.e, the role of $P(ac|a) + P(a|c)$, and is conditionally independent given $a$, i.e, the role of $P(ac \wedge c|a) - P(c|a) * P(ac|a)$. Lastly, in (3), the measure is defined considering Figure 4.1 (d), which corresponds to the similar definition of (2) but with a different rule structure: both $ac_1$ and $ac_2$ are associated with a, i.e, the role of $P(ac_1|a) + P(ac_2|a)$, and conditionally independent given a, i.e, the role of $P(ac_1 \wedge ac_2|a) - P(ac_1|a) * P(ac_2|a)$.

Because the interestingness of a rule X $\Rightarrow$ y or x $\Rightarrow$ Y having more than two antecedents or two consequents is dependent on the weakness of its sub-rules, the generalized interestingness measure is defined as follows:

$$Csign(R) = \begin{cases} min_{\{x_1, x_2\} \subseteq X} CSign(x_1, x_2 \Rightarrow y) & if\ R = X \Rightarrow y \\ min_{\{y_1, y_2\} \subseteq Y} CSign(x \Rightarrow y_1, y_2) & if\ R = x \Rightarrow Y \end{cases} \tag{4.3}$$

Based on the Csign measure, the proposed causal association rule mining algorithm (CARMA) approach is illustrated in Algorithm 2.

The process of CARMA is inspired by the incremental rule mining strategy [94]. We have chosen this strategy since it has provided good results for causal association rule extraction in the fields of drug-to-drug interaction [94] and gene expression [95]. CARMA takes six inputs, including the anomalies and sample actions set D, and two user-specified parameters:

**Algorithm 2:** CARMA

**Input:** Dataset of anomalies actions D; Anomalies set A, Context set C, Actions set Ac, minimal support threshold minS, minimal Csign threshold minCsign

**Output:** CAR set R

1 **for** $a \in A$ **do**
2 $R_1 \leftarrow \{\}$;
3 **for** $ac \in Ac$ **do**
4 **if** $Supp(a \rightarrow ac) \succ minSupp$ **then**
5 $R_1 \leftarrow R_1 \bigcup \{a \rightarrow ac\}$;
6 $R_1 \leftarrow R_1 \bigcup SubCarma(R_1, a, ac, x \leftarrow Y, minSupp, minCsign)$;
7 **end**
8 **end**
9 $R \leftarrow R \bigcup R_1$;
10 $R_2 \leftarrow \{\}$;
11 **for** $c \in C$ **do**
12 **if** $Supp(c \rightarrow a) \succ minSupp$ **then**
13 $R_2 \leftarrow R_2 \bigcup \{a \rightarrow ac\}$;
14 $R_2 \leftarrow R_2 \bigcup SubCarma(R_2, c, a, X \leftarrow y, minSupp, minCsign)$;
15 **end**
16 **end**
17 **end**
18 $R \Rightarrow R \bigcup R_2$ ;
19 **for** $ac \in Actions$ **do**
20 $R_3 \leftarrow \{\}$;
21 **for** $a \in Anomalies$ **do**
22 **for** $c \in Context$ **do**
23 **if** $Supp(a, c \rightarrow ac) \succ minSupp$ **then**
24 $R_3 \leftarrow R_3 \bigcup \{a, c \rightarrow ac\}$;
25 $R_3 \leftarrow R_3 \bigcup SubCarma(R_3, a, ac, X \leftarrow y, minSupp, minCsign)$
26 **end**
27 **end**
28 **end**
29 **end**
30 $R \leftarrow R \bigcup R_3$ ;
31 Return $R$;

minimal support threshold minSupp and minimal Csign threshold minCsin, the set of actions Ac, the set of anomalies A, and the set context C which includes the FCD values and activities in the sample set. Then, the algorithm returns a set of potential causal association rules R. CARMA is able to discover the different types of CAR formats. It separately extracts CARs according to each case in Definition 9: Case (2) (Lines 1–8) , Case (1) (Lines (10–16)), and

**Algorithm 3:** SubCarma

**Input:** $R_1$, ant, effect, form, minSupp, minCsign
**Output:** CAR set $R_1$

```
1  if (form = x → Y) then
2  |  rulePart → Consequents ;
3  end
4  else
5  |  rulePart → Antecedents ;
6  end
7  for j = 1 to max length of rulePart of rules in R1 do
8  |  elem_j → RulePart(rulePart, R1, j) ;
9  |  for rule(form) in R1 with j element do
10 |  |  if rulePart = Consequents then
11 |  |  |  Generate new rule ras ant ← elm_j ∪ effect;
12 |  |  end
13 |  |  else
14 |  |  |  Generate new rule ras ant ∪ elm_j ← effect;
15 |  |  end
16 |  |  if Support(r) ≻ minSupp then
17 |  |  |  setCsign(r) based on Equation 3 and 2 ;
18 |  |  end
19 |  |  if Csign(r) ≻ minCsign then
20 |  |  |  R1 ← R1 ∪ r;
21 |  |  end
22 |  end
23 end
24 Return R1
```

Case (3) (Lines 18–28). For each case, CARMA proceeds by an incremental strategy by calling Algorithm 3 named SubCarma (Lines 6, 14, and 24). The latter, according of the format of the rule (having several consequents and one antecedent or vice versa), combines the antecedents or the consequents of the extracted rules to discover new rules in a breadth-first manner. For each newly generated rule, the support and Csign are computed; only the ones that pass the support and Csign thresholds are selected. The final output of CARMA is the set of CARs.

**Example 5** *The set of rules in Table 5.1 are examples of CARs obtained after executing Algorithm 2 CARMA on the dataset of anomalies, context data, and activities where an extract of which is shown in Figure 4.4. In the table, we find two anomalies (e.g. FireElectricity and Discomfort). The FireElectricity anomaly can be caused by different combinations. For*

*instance, when the user is using the phone despite being in the kitchen but the stove is on, then there is a risk of FireElectricity, since the call can take a long time.*

Table 4.1 Examples of CAR

| CAR | Types (Definition 9) | Csigni values |
|---|---|---|
| (1) User_LocatedLivingRoom, Stove_On ⇒ FireElectricity | case (1) | 0.92 |
| (2) User_LocatedKitchen, User_Phoning, Stove_On ⇒ FireElectricity | case (1) | 0.92 |
| (3) User_Reading, User_LocatedBedroom, Stove_On ⇒ FireElectricity | case (1) | 0.98 |
| (4) User_LocatedBedroom, User_Sleeping, Stove_On ⇒ FireElectricity | case (1) | 1 |
| (5) User_LocatedBedroom, Door_Open, User_Sleeping ⇒ Discomfort | case (1) | 0.79 |
| (6) FireElectricity ⇒ Stove_Off | case (3) | 0.65 |

### 4.4.2 Markov Logic Network

As we stated above, the aim of this process is to detect the risk of anomalies and decide which suitable actions to perform. To achieve this goal, two processes are required: (1) static knowledge inference and (2) dynamic knowledge inference.

#### Static Knowledge Inference

The aim of this part of the MLN is to instantiate a particular type of predicate in the `MLN knowledge base` with static knowledge coming from the discovered CAR. In this subsection, we first present the content of the knowledge base, which is a set of different types of predicates and rules. Afterward, we will present the algorithm enabling the grounding of this type of predicate in the knowledge base from the CAR.

**MLN Knowledge Base (KB)** Three types of predicates belong to the KB of the MLN: Probabilistic Dynamic Predicate (PDP), Probabilistic Static Predicate (PSP), and Probabilistic Hidden Predicate (PHP). They are defined as follows:

**Definition 10** *Probabilistic Dynamic Predicate (PDP): in our MLN KB, a PDP is the predicate* `Evidence(Object,` $w_{object}$`)` *that is grounded with context data (corresponding to an FCD value or an activity). The distribution truth values (uncertainty value=* $w_{object}$*) of the PDP may be given as evidence. The Dynamic in its name comes from the fact that its grounding changes over time since it depends on the user and environment data.*

**Example 6** $\langle Window1, [Open, 0.3; Close, 0.7], 10:50:55\rangle$ *is an example of an FCD of the Window sensor. As we can see, the uncertainty values are provided. The higher one is for*

*Window1_Close (0.7). In this work, the context data with the highest uncertainty value is considered for grounding. Otherwise, if equal uncertainty values are given, the predicate will be instantiated with all the data. Accordingly, this FCD instantiates the* `Evidence` *predicate as follows:* `Evidence(Window1_Close, 0.7).`

**Definition 11** *Probabilistic Static Predicate (PSP): PSP involves the causes obtained from a given CAR. Indeed, CARs do not change over time but are almost static since they changed whenever the dataset is updated or a new one is given. Therefore, the grounding of this type of predicate is almost static, which explains the Static in its name. The uncertainty value of a PSP is the value of the given CSign (Equation 5.3) of the CAR which it instantiates. In our MLN KB, the PSP predicates are nbAnomalyCause(X, Anomaly,$w_{car}$), nbActionCause(anomaly, Y, $w_{car}$), nbAnAcCause(X, anomaly, Y, $w_{car}$), where X is a set of context data including activities and Y is a set of actions. $w_{car}$ is the Csign value of the corresponding CAR.*

Note that *nb*AnomalyCause(X,Anomaly) can be instantiated with a CAR represented in the V-Structure (e.g Case 1 of Definition 9) in which the antecedent is a set of the context data and activity that cause the anomaly in the consequent. Initially, before instantiation, the number of causes is unknown and varies from one CAR to another which explains the *nb* in the predicate. In other words, *nb* is the size of the common causes belonging to set X. The determination of this number and the grounding is done thanks to the process in the next paragraph. Similarly, the predicates *nb*ActionCause(anomaly, Y) and *nb*AnAcCause(X, anomaly, action) are used to instantiate a CAR having format respectively of cases 3 and 2 of Definition 9, respectively. *nb* in these predicates is the size of the actions set Y in the former one and the size of context set X in the latter.

**Definition 12** *Probabilistic Hidden Predicate (PHP): In the MLN KB, PHP can be referred as the query node: the right side of the rule. Its distribution truth ($p_{pred}$) is unknown. Anomaly (anomaly, $p_{ano}$), Action(Actions, $p_{ac}$), AnoAct(anomaly, and action, $p_{anoac}$) are the PHPs tested in the MLN KB.*

Based on these three types of predicates, we define the abstract level of rules for the MLN KB shown in Table 4.2. As we can see, these rules are abstract since the variable *nb* has not been determined yet (i.e., the role of the next paragraph). This number *nb* plays an important role in the semantics of the rules. For example, in rule (1), to predict the risk of anomaly, all the anomaly causes must be given as evidence (i.e., the role of $\forall x_i, x_i \in X$). Accordingly, the number of Evidence predicates cannot be lower than the number *nb* corresponding to the size of anomaly causes X.

Table 4.2 Rules set in the MLN KB

| Rules | Weights |
|---|---|
| (1) *nb**Evidence($x_i$, $w_{x_i}$), *nb*AnomalyCause(X, y, $w_{car}$) ⇒ Anomaly (y, $w_y$), $\forall x_i, x_i \in X$ | $w_1$ |
| (2) Anomaly(y, $w_y$), *nb*ActionCause(x, Z, $w_{car}$) ⇒ Action(Z, $w_Y$) | $w_2$ |
| (3) *nb**Evidence($x_i$, $w_{x_i}$), Anomaly(y, $w_y$), *nb*AnAcCause(X, y, z, $w_{car}$) ⇒ AnoAct (y,z, $w_{y,z}$), $\forall x_i, x_i \in X$ | $w_3$ |

**Grounding of PSP from CAR** In this subsection, we explain the process of grounding the PSP with the given CAR. This process is achieved thanks to the following algorithm (Algorithm 4):

**Algorithm 4:** PSP Grounding

**Input:** set of CAR sCar, set of anomalies, actions and contexts in the dataset sA, sAc, sC

**Output:** grounding of PSP Gpsp

1 **for** $ruler \in sCar$ **do**
2 $\quad ant \Leftarrow Antecedent(r)$;
3 $\quad consq \Leftarrow Consequent(r)$;
4 $\quad$ **if** $size(ant) = 1$ $\&\forall x \in cons, x \in sAc$ **then**
5 $\quad\quad Gpsp \Leftarrow Gpsp \bigcup Grounding(nbActionCause, r, 1, size(consq))$;
6 $\quad$ **end**
7 $\quad$ **else if** *(size(consq)==1* $\&\forall x \in ant, x \in sC$ **then**
8 $\quad\quad Gpsp \Leftarrow Gpsp \bigcup Grounding(nbAnomalyCause, r, 1, size(consq))$;
9 $\quad$ **end**
10 $\quad$ **else if** $\exists x, y \in an\ wherex \in sA, y \in sC$ & $size(consq) == 1)$ **then**
11 $\quad\quad Gpsp \Leftarrow Gpsp \bigcup Grounding(nbAnAcCause, r, size(an), 1)$;
12 $\quad$ **end**
13 **end**
14 Return $Gpsp$;

As we mentioned before, each PSP in the MLN KB corresponds to a particular format of CAR of Definition 9. Therefore, Algorithm 4 instantiates each PSP according to the criteria of the format. For example, in the case of a rule having one antecedent's element and one or more consequent elements where all of them are actions, the algorithm instantiates the "*nb*CauseAction" PSP by calling the function Grounding (lines 4,5, and 6). For this predicate, this function according to the number of actions in the consequent part of the rule instantiates the predicate by putting these actions in the rule and replacing *nb* with this number. This process is similar for the rest of the formats (lines 7–13).

**Example 7** *In order to instantiate the PSP based on the CAR shown in Table 5.2, Algorithm 4 is called. The result is depicted in Table 5.3. As we can see in the Table and as explained above, the nb in the PSP corresponds to the number of common causes of anomaly in the*

Table 4.3 PSP Grounded from CAR

| PSP grounded | corresponding CAR |
|---|---|
| 2AnomalyCause(User_LocatedLivingRoom, Stove_On, FireElectricity, 0.92) | (1) |
| 3AnomalyCause(User_LocatedKitechen, User_Phoning, Stove_On, FireElectricity, 0.92) | (2) |
| 3AnomalyCause(User_Reading, User_LocatedBedroom, Stove_On, FireElectricity, 0.98) | (3) |
| 3AnomalyCause( User_LocatedBedroom, User_Sleeping, Stove_On, FireElectricity, 1) | (4) |
| 3AnomalyCause( User_LocatedBedroom, Door_Open, User_Sleeping, Discomfort, 0.79) | (5) |
| 1ActionCause(FireElectricity, Stove_Off, 0.65) | (6) |

*predicate nbAnomalyCause, the number of actions caused by the anomaly in the predicate nbActionCause, and finally the number of anomaly causes that cause the action in the predicate nbAnAcCause. For instance, on the one hand, CAR (1) holds two commons causes of the anomaly FireElectricity: User_LocatedLivingRoom and Stove_On. Accordingly, for this CAR, the nb in nbAnomalyCause is replaced by 2. On the other hand, in CAR (6) the anomaly FireElectricity causes just one Action–Stove_Off–, which explains the 1 that replaces the nb in nbActionCause.*

### Dynamic Knowledge Inference

The main role of this step is to find the most probable anomaly for the current situation that represents the data belonging to the given segment. To do so, three main steps are required: (1) dynamic ground MLN creation (2) rule weight calculation, and (3) Probabilistic Hidden Predicates weight calculation.

**Grounding of Probabilistic Dynamic Predicates (PDP)** The PSPs in the MLN KB are grounded from the extracted CARs thanks to the application of Algorithm 6. The PHPs are the query predicates so they are the result of the whole process of the MLN. To produce this result, the grounding of the PDP is mandatory. This grounding is from the current situation representing the given segment. Therefore, some context data belonging to the FCD and the activity in the situation are the grounding of the PDP evidence. As we mentioned in the previous subsection, for each FCD in the current situation, the PDP is instantiated only with the context data having a higher uncertainty value or with all of them in case of uncertainty values being equal in the context data.

**Example 8** *Let us consider that the current situation* $S_1 = \langle\langle Window1, [Open, 0.3; Close, 0.7], 10:50:55\rangle\langle Stove,$
$[On, 0.9; Off, 0.1], 10:52:34\rangle\langle Vent, [On, 0.89; Off, 0.11], 10:52:57\rangle; act(Phoning, 10:53:00, 0.78), 10:50:00--10:55:00\rangle$. *The set of FCD and activities in* $S_1$ *are used to instantiate the PDP evidence. The result after grounding as follows: Evidence(Window1_Close,0.7);*

*Evidence(Stove_On,0.9); Evidence(Vent_On, 0.89); Evidence(User_Phoning, 0.79). As we can see, the context data with higher truth values is considered for the grounding (i.e Stove_On has higher uncertainty value than Stove_Off).*

*At this stage the PSP and PDP are instantiated from the situation and the extracted CAR. Accordingly, the ground MLN can be created. Figure 4.5 shows an excerpt of a ground MLN that is built based on the grounding of the PDP and PSP in Examples 8 and 7.*

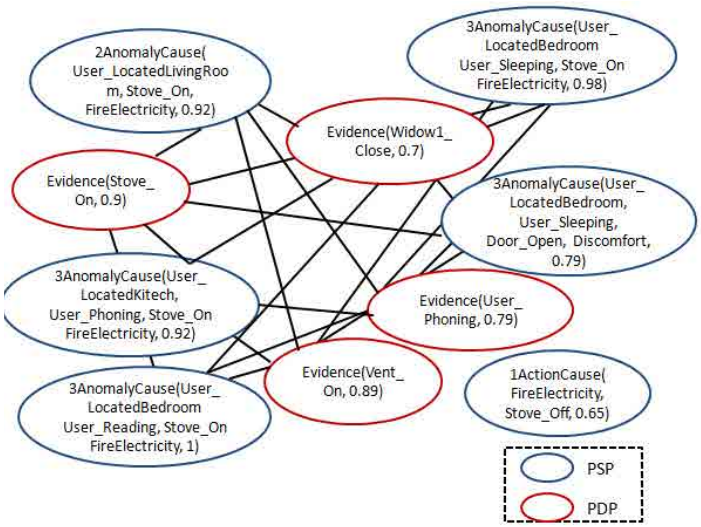



Figure 4.5 Excerpt of Ground MLN

*According to the definition of the MLN (see Section 5.2.1), it is represented through a graph in which two nodes are related if the two predicates are in the same rule. This is clearly represented in Figure 4.5. For example, the sets of PSP are never related with each other since each rule in Table 4.2 has only one PSP.*

At this stage of the whole process for early anomaly detection, the data are acquired and segmented, the user activity is recognized, the CARs have been extracted, the eventual conflict in the rules has been resolved, the current situation has been determined, and the PSP and PDP have automatically been grounded, which allows the dynamic construction of the Ground MLN. Accordingly, the Ground MLN is created automatically based on the extracted CAR and the built situations. The next step is then to compute the rule weights in the MLN KB based on the weights of the PSP and PDP after grounding.

**Rules weight calculation** To compute the weight ($w_i$) of a logical rule $F_i \in F$ of the MLN model we apply the following proposed formula :

$$w_i = ln \frac{(\sum_j w_{P_{ij}})/n_i}{1 - \sum_j w_{P_{ij}}/n_i} \tag{4.4}$$

where $P_{ij}$ is the predicate (PDP or PSP) number j in the rule $F_i$ and $w_{P_{ij}}$ is its weight, and $n_i$ is the total number of predicates that instantiate $F_i$.

**Computation of the weight of Probabilistic Hidden Predicates** It is possible at this stage to compute the weights of the probabilistic hidden predicates. Let $k \in$ PHP in the ground MLN, then $P_{pred_k}$ is its weight to be computed as follows:

$$P_{pred_k} = Av(P(x = const)) \; \forall const \in Constant_k \tag{4.5}$$

where $Constant_k$ is the set of constants in the Ground MLN that are in the predicates that have an edge with the PHP $k$. P(x=const) is computed by applying the formula in Equation 4.1. The final classification of a given situation is the anomaly that has the maximum weight value in the corresponding PHP. If this weight is above a defined threshold, then the situation is considered anomalous. Afterward, the anomaly of this situation is matched with the actions or set of actions having the highest weights in the corresponding PHP. This weight will be its uncertainty value, *unc*.

## 4.5 Experimental evaluations

In this section, we describe our proof-of-concept implementation for Caredas, the proposed early anomaly detection method. First, we show experiments that have been done to evaluate Caredas before and after the improvement of MLN . The results are then discussed. Then, we evaluate the proposed CARMA algorithm for CAR discovery in the drug-to-drug interaction (DDI) domain using a clinical dataset.

### 4.5.1 Evaluation of Caredas

In order to evaluate the proposal Caredas, for online early anomaly detection, we developed a prototype. We extensively evaluated the method with a dataset acquired out of more than 2 hours of elderly-like routine on the Hadaptic platform[2]. The dataset contains approximately 600 context data points and 10 different activities that occurred 75 times for 15 different scenarios. Afterwards, the dataset was annotated with possible anomalies and required actions for each detected anomaly from experts and with user verification. The smart lab was equipped with motion sensors, beacons, switches, thermometers, and more. The prototype was integrated with an FSCEP [5, 4] implementation for data acquisition, and our proposal

[2]http://hadaptic.telecom-sudparis.eu/

AGACY Monitoring for activity recognition. For the same reason as the evaluation of AGACY Monitoring, this method uses fixed time-window size segmentation. Accordingly, before execution, the method requires a preliminary step, in which the values of the parameter *win*, which corresponds to the time window size of the situation, is experimentally chosen. Therefore, we tested the method with different values of $win \in [60s...300s]$. The constant Z in Equation 4.1, was set to 10; the thresholds for the minimum support and minimum Csign in CARMA were set to 10 and 0.5, respectively. Finally, the threshold for considering a situation anomalous was set to 0.7. The method was evaluated by comparing its output against expected results. For each time window, the precision and recall of the system were computed. Furthermore, we computed the correctness, which is simply the proportion of correct (expected) answers given by the system. Moreover, the complexity of the proposed DgMLN is computed. To test the method for online early anomaly detection, the CARMA algorithm must be first executed offline to extract the anomaly causes from the dataset. Afterwards, the MLN grounds only the PSP predicates. Once this is done, the online process is simulated as follows: for a defined time window size, the system extracts the data from the dataset, without considering the additions that have occurred during this time window and makes an analysis to output the possible risk of anomaly, if it exists. Then, for the evaluation, this output is compared to the expected results, which are provided by the experts.

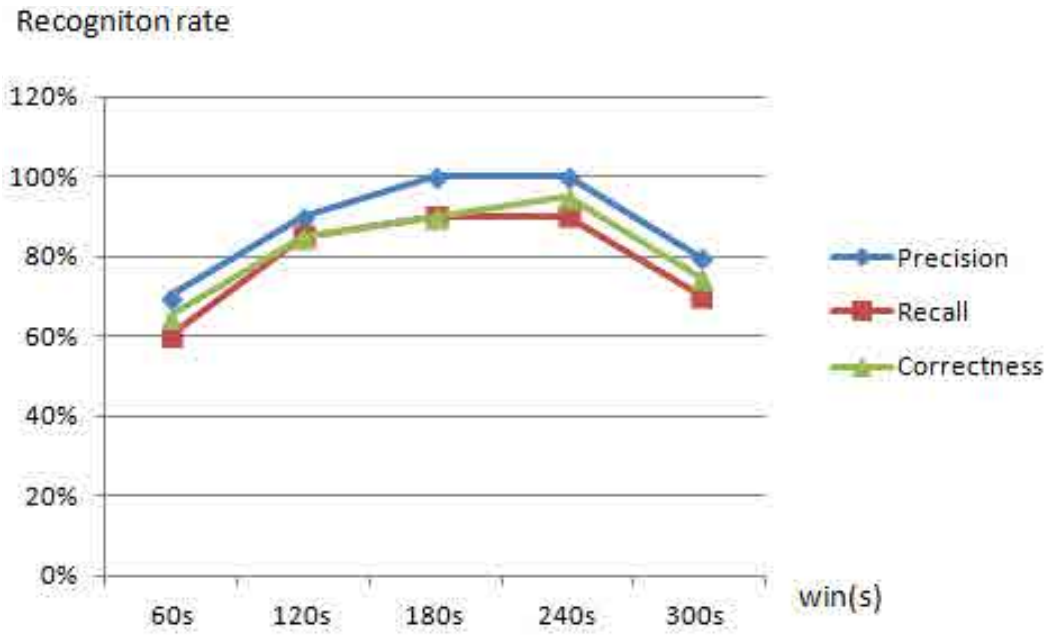



Figure 4.6 Precision, recall, and correctness of the method for different time window *win*

As we can see in Figure 4.6, the method has high precision, recall, and correctness values for all time windows. This means our system rarely detects untimely events, which is an

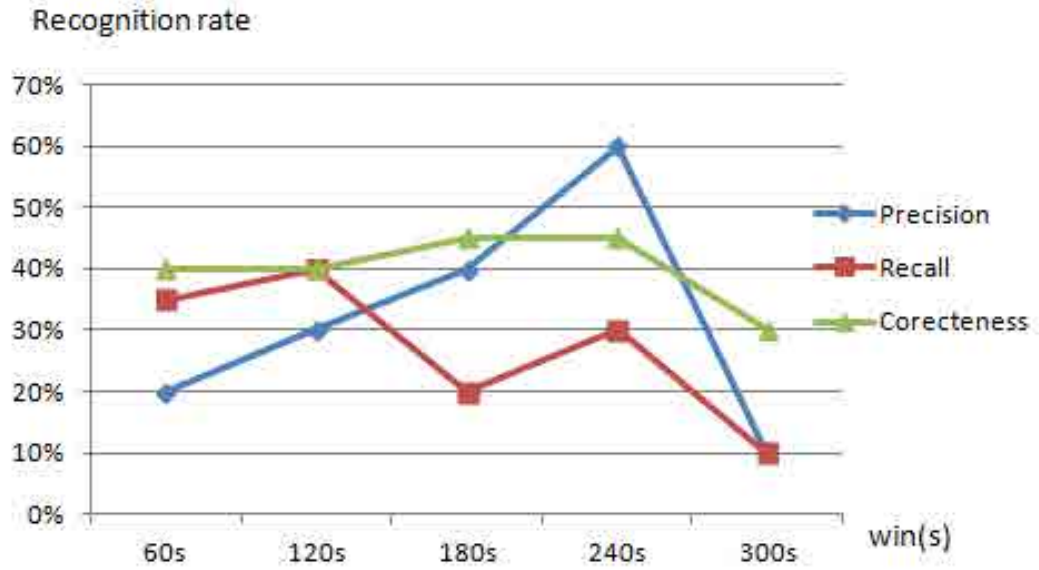



Figure 4.7 Precision, recall, and correctness of the method with static ground MLN for different time window *win*

important feature for the comfort of the user. However, as depicted by the recall, it sometimes misses anomalies, but relatively rarely for time windows shorter than 4 minutes. Overall, we can see that all curves decrease with an increase size in the time window until 4 min and they decrease for higher time windows. On the other hand, when the time window is short (less than 2 minutes), it does not contain sufficient data to make a decision. This is due to the hard constraint defined in the MLN rules that requires that the number of evidences must be equal to or higher than the causes in the PSP predicates. Accordingly, sufficient data is required in the time window to make the right decision. In conclusion, with a suitable time window (3 and 4 min), our proposed approach is highly reliable with the proposed improvement of MLN, rule weights, and the CARMA algorithm to recognize anomalous situations and recommend actions. Furthermore, the proposed algorithm for DgMLN about the grounding of PSP and PDP is not complex. On the one hand, the complexity of the PSP grounding is O(1): the algorithm has only one loop in which the size of the set of rules is fixed (e.g., size of sCar in Algorithm 5.4). On the other side, O(n) is the complexity of the PDP grounding: the algorithm has only one loop in which the size on the data belonging to the situation is variable (e.g., n). In order to also prove the efficiency of the dynamic ground MLN over the static variant, we have also implemented the MLN as a static ground MLN [69] with the same dataset and CAR. However, as the static ground MLN [69] requires, all the predicates are instantiated with all possible constants from the beginning, and rule weight values are fixed to 1. This version of the method was tested and evaluated on the

same dataset and metrics. The evaluation results are presented in Figure 4.7. As we can see in the figure, the static ground MLN does not work well for early anomalous situation detection and action decision as it shows poor results compared to the dynamic variant. It particularly encounters difficulty for 5-minute time windows with only 10% precision and recall. In fact, it is not able to handle a large variety of context data. Since in this work we focus on considering the user's environmental context in knowing her/his activities for early anomaly detection, we studied the impact of the context in the process. Therefore, for each segment of time window belonging to the dataset, we removed a context element. For example, we removed the context data about the stove (e.g., *stove_On*). Afterward, using the updated dataset, we evaluated the system with the same parameters. The result is depicted in Figure 4.8. Obviously, as we can observe, the values of the three measures for different time windows are either the same as that in Figure 4.6 or lower. On the one hand, for each time window, when the removed context element is considered as an anomaly cause by CARMA, then an important element for making the right decision is missing, such as for 60s time windows. Therefore, the system performance decreases. On the other hand, in the case when the removed context element is not considered as anomaly cause, the system may have the same performance as for the time window 240s. This experiment proved that the context plays an important role in increasing or decreasing the system performance. In general, when a context element that is selected as an anomaly cause by CARMA is missing, the system may not make the right decision. However, the advantage is that our proposal is not sensitive to insufficient context data. In other words, the system may produce correct decisions even if context data is missing. However, it is possible only if these context data points are not selected as anomaly causes. This is an advantage of the system that proves its ability to make the right decision even with insufficient data, which is likely to happen in real-time analysis.

### 4.5.2 Evaluation of CARMA with medical dataset

In order to prove the generality of the proposed method for CAR extraction to be used in different domains, we applied the CARMA algorithm into the field of drug-to-drug interaction (DDI). To this end, we ran CARMA using a large-scale clinical dataset. This dataset is provided by the FDA Adverse Event Reporting System (FAERS) that was established by the United States Food and Drug Administration (FDA) to collect ADR reports from health-care professionals, patients, and pharmaceutical manufacturers. The dataset used in this work is the *FAERS XML 2017q2*[3]. It contains total of 29,175 data points about reports, patients, drugs, reactions, etc. This dataset is used by CARMA in order to extract CARs about drugs and

[3]https://www.fda.gov/Drugs/GuidanceComplianceRegulatoryInformation/Surveillance/AdverseDrugEffects/

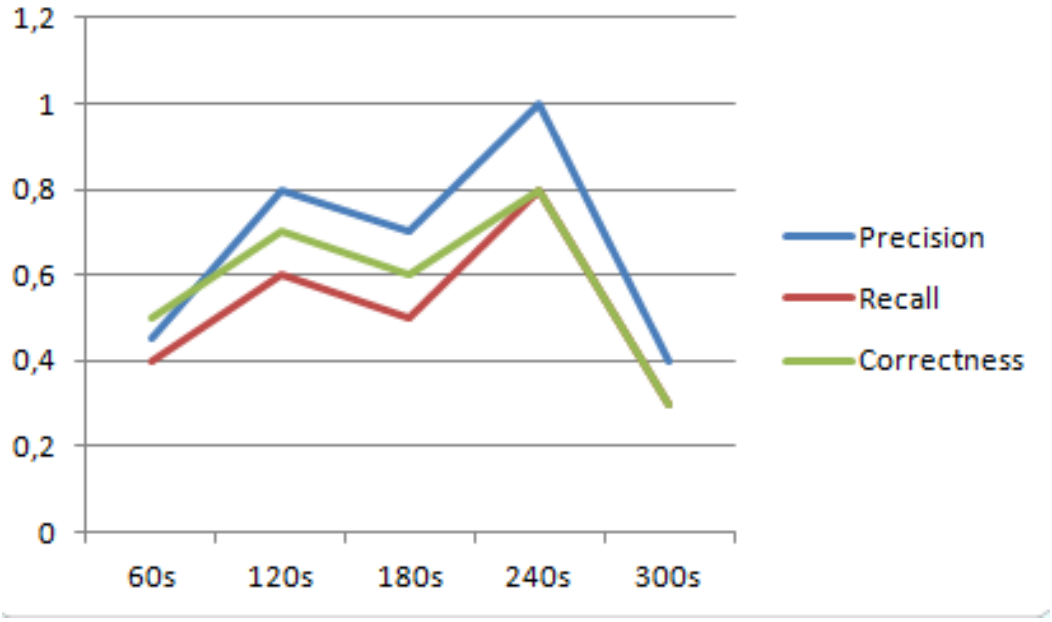



Figure 4.8 Precision, recall, and correctness of the method with dataset after contexts element removal for different time windows *win*

reactions. In this dataset, drugs are causes of reactions, e.g aspirin + warfarin ⇒ bleeding and it is not possible to know from reports if reactions cause reactions. Therefore, in this study, only V-Structure can be used. Accordingly, only case 1 of the rule format in Equation 4.6 is considered, in which the anomaly $a$ is replaced with the reaction *reac* and anomaly causes $c_i$ are replaced with drugs $d_i$. Then, Algorithm 2 is adapted accordingly. The thresholds for minimum Support and Csign were set to 100 and 0.05, respectively. Following common practice of pharmacovigilance studies in evaluating the clinical validity of DDI [66, 54], we randomly selected 100 CARs identified by CARMA and presented them to four doctors and pharmacists for manual review. In this evaluation the doctors and pharmacists were asked to validate the 100 CARs based on four criteria concerning drugs and two concerning associations. The result and the criteria are shown in Table 4.4. Moreover, the obtained result was compared with that in [94] and classic AR using the FAERS dataset.

We can conclude from Table 4.6 that CARMA outperforms the work in [94] and the AR in all criteria. For example, on the one hand, CARMA has identified more drug combination that doctors and pharmaceutics have confirmed to be known to interact (e.g 26% for CARMA vs 20% for [94] and 10% for AR) such as sacubitril + valasartan → Hypotension. As affirmed by doctors, this combination of drugs may produce hypertension for patients but does not imply a definitive stop to the treatment. On the other hand, CARMA has identified fewer drug combinations that aren't known to interact (e.g 42% for CARMA vs 50% for [94] and 79%

Table 4.4 Performance for CAR extraction into FAERS dataset

| Criteria | [94] | CARMA | **AR** |
|---|---|---|---|
| **drugs** | | | |
| 1 Drug combinations known to interact | 20% | 26% | 10% |
| 2 Drug combinations known to be given together or treat the same disease | 13% | 17% | 4% |
| 3 Drug combinations that seem to be due to other confounding issues | 17% | 14% | 7% |
| 4 Drug combinations that are unknown to interact | 50% | 42% | 79% |
| **Associations** | | | |
| 1 One/more drugs in antecedent can cause the adverse reaction in consequent | 77% | 89% | 61% |
| 2 No drug in antecedent can cause the adverse event in consequent | 23% | 11% | 39% |

for AR). For instance, the method in [94] has identified false interactions between celecoxib and metformin, while CARMA has not identified this interaction.

## 4.6 Conclusion

Through this chapter we have presented a novel method, named Caredas, for early anomaly detection. The method combines MLN and CAR for online prediction and anomaly causes extraction. Caredas is evaluated with a dataset acquired through the Hadatptic platform, it has proven a good accuracy for anomaly prediction. Moreover, CARMA has shown a good results for causal association rules extraction event with medical dataset. As we can remark, until this part of the report, a static segmentation is applied for activity recognition and also for early anomaly detection. One of the main conclusion of the evaluation results of both methods is that the performance is changing according to size of the segment. With a good selected size, the methods are accurate. However, with too large or too small segment size the methods are confused. thus, is it possible to propose a method that adapts dynamically the segment size according to analysis goa (activity recognition or early anomaly detection) ? The answer of this question is in the next chapter.

# Chapter 5

# DataSeg: Dynamic Streaming Sensor Data Segmentation for Smart Environment Applications

## 5.1 Introduction

Data segmentation is a crucial problem for data processing that is required before analyzing the data. It aims at dividing the long sequence of sensing records into a set of individual segments. Each segment corresponds to a "specific" concept which can be interpreted differently according to the use case (i.e. activity recognition, anomaly detection, ...).

In the literature, we distinguish two main approaches to segment a streaming sensor data: static approaches where the segment size is fixed and dynamic approaches allowing a flexible choice of segment size. The experiments realized in the two previous chapters with static segmentation have shown the great impact of the segment size on the analysis results. When the segment size is chosen improperly bad results can be obtained. In order to avoid this problem, as explained in Chapter 2, the dynamic sensor data segmentation has proven better results than the static one [100] since it identifies the time points in a more flexible way. However, dynamic streaming sensor data segmentation is still a challenging problem. Generally, previous researches in this field [76, 49, 37], either assume that a pre-segmented dataset is available for learning the suitable time-window size or being conceived to be applied for a particular application. Accordingly, most related works suffer from the generality issue. For these reasons, we claim that the data segmentation method should be as general as possible in order to be adapted according to the use case (i.e activity recognition, anomaly detection). One of the conclusion drawn from Chapter 2 is that hybrid methods

are the best candidates to be employed for the processing and analysis of sensor data in smart environments. Based on this analysis and in order to overcome the generality issue of previous works, we propose in this chapter a novel hybrid method for the dynamic segmentation of streaming sensor data. The proposal combines a clustering method, that is to say, an unsupervised learning method, and a semantic based method (i.e ontology), that relies on high-level "symbolic" representation, in order to dynamically choose the best time-window size. The method is designed to be employed for different applications as long as a dataset of user monitoring can be provided.

Initially, to tackle the cold start problem that usually concerns learning approaches, an ontology with a default classification is created. This task can be done by the designer after the setup of the environment. Afterwards, once a dataset is acquired related to the target application [1], a clustering based classification of the dataset is done offline according to a defined set of features. The proposal dynamically updates offline the ontology whenever any change occurs in the obtained clusters. During the online process, the method uses the ontology to decide which better size for the current time-window. The proposed method thanks to this hybridization neither requires a pre-segmented dataset nor being limited only for a particular type of application's use. In the following, we summarize the main contributions of the chapter:

- A new method for dynamic streaming sensor data segmentation. The proposal is flexible to be used in a wide range of applications once an annotated dataset is provided.
- A novel marriage between semantic and data driven based techniques to provide a flexibility for the time-window size choice.

## 5.2 Method

Through this section, we explain in details the whole process of the method for the dynamic streaming sensor data segmentation.

As we can see in Figure 5.1, the method holds mainly two steps: Model building and Dynamic segmentation. The former starts by building a model for a `Clustering` algorithm in order to classify the given dataset into a set of classes based on given features. The dataset must concern one application and the features are defined accordingly. As mentioned before, the proposed method could be used with different application domains. The model construction of the clustering is executed offline once a dataset is given or updated. Afterwards, the `Ontology updater` dynamically updates the ontology when the clusters

[1] dataset of user activities for activity recognition, of anomalies in case of anomaly detection, etc.

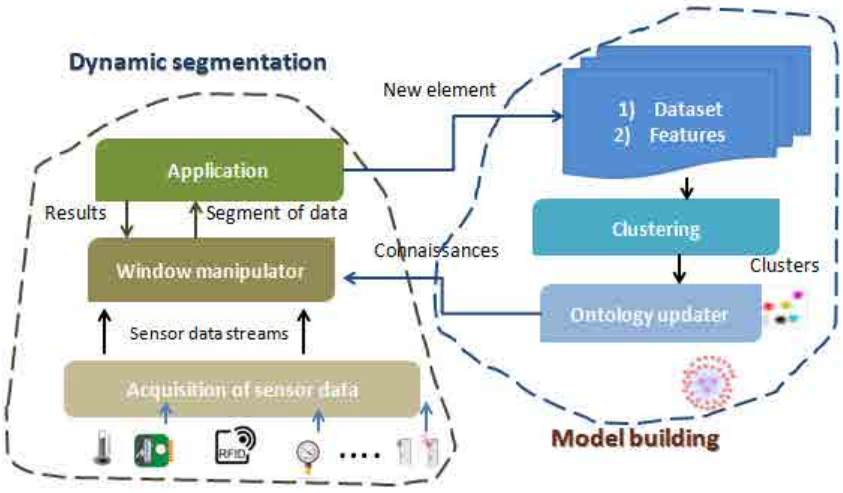



Figure 5.1 DataSeg architecture

are updated or new ones appear. The ontology, that represents the second part of DataSeg's model, is designed to semantically represent the information provided by the built clusters for each application. Hence, using the ontology provides a generality feature to DataSeg. The second step of DataSeg, that is processed online, is to dynamically segment the streaming sensor data into time-windows based on the knowledge provided by the ontology. After choosing the optimal segment size, the segment data is sent to the target application. DataSeg waits the output of the application and checks its correctness based on the given knowledge from the ontology. When the application's output is not satisfied DataSeg extends the segment size. In case of new element is detected by the application, DataSeg learns it using the clustering.

### 5.2.1 Clustering

In order to ensure independence from the given prior knowledge, we use unsupervised learning (clustering) allowing to provide this knowledge. The aim of the clustering in our work is to to predict related information about a new element of the application. More precisely, the clustering allows to group the annotations of a given dataset into a set of classes sharing similar features values related to the target application. For example, *activity duration* and *used sensors* can be the features corresponding to an activity recognition application while the *anomaly start time* can be considered as a feature for an anomaly detection application. Hence, for instance, the result of the clustering is a set of clusters containing similar activities regarding their duration and the used sensors for their occurrences. For example, Figure 5.2,

shows a set of clusters obtained from a dataset annotated with user daily activities. The selected features are the used sensors and the activity's duration.

| Cluster | DurationInterval | Sensors | Activity |
|---|---|---|---|
| 1 | {5min, 30min, >1H} | {M003,M002} | {Sleeping} |
| 2 | {5min} | {D002, D004} | {Enter_Home, Leave_Home} |
| 3 | {10min} | {M014} | {Eating, Wash_Dishes, Meal_Preparation} |
| 4 | {5min, 10min} | {M014, M018, M019} | {Meal_Prepartion} |
| 5 | {10min} | {M014, M020} | {Eating, Wash_Dishes} |

Figure 5.2 An example of a set of clusters about user activities

As we can see in the Figure 5.2, activities sharing similar features' values are in the same cluster (i.e Enter_Home et Leave_Home).

### 5.2.2 Ontology Updater

For the purpose to provide generality of the proposed method, we conceived an ontology to semantically represent the information provided by the clusters for each dataset. Thus, through the ontology we can obtain a common representation of the different clusters about different applications. Morever, in order to allow a quick start of the system even if a dataset is not yet ready, the ontology initially contains similar information that should be provided by the clusters for each application. For a particular application, when the dataset is ready and the clusters are obtained, the ontology is dynamically updated with the content of the given clusters. The dynamic update is insured by using program and SPARQL queries. Algorithm 5 shows the operations performed by the ontology updater. Initially the ontology contains the three classes *Application*, *Dataset*, and *Element*. Moreover, it contains the applications that will be held by the method represented as subclasses of *Application* class. The class *Element* represents the main information provided by the application. For instance, for the activity recognition, the ontology must contain the class *Activity*, a subclass of *Element*, related to the class *activity recognition*, a subclass of *Application*, and related to the classes corresponding to the application's features (i.e Duration). After receiving the application name, the set of features and clusters, and the dataset name, Algorithm 1 starts by executing a rule named *search*, by calling the function *ExecuteRule*, which searches whether the given dataset is already in the ontology (Line 1 and 2). It is important to mention that the algorithm stores the corresponding classes in the ontology for each possible application and their corresponding features. Hence, the function *GetClass* (Line 4) returns the corresponding ontological class for the input application *app*. Afterwards, by applying a logic rule, named *insertInstance*, all the different elements belonging to the given clusters become instances of the corresponding subclass of *Element* related to the given application (Lines 5–8). For

**Algorithm 5:** Ontology Updater

**Input:** Ontology O, Features set sf, Clusters C, Name of the dataset name, Application app
**Output:** Updated ontology O

```
1  if (ExecuteRule(search, O, name) ≠ ∅) then
2  |  ExecuteRule(insertInstance, Dataset, name);
3  end
4  appC ← GetClass(O, app);
5  for (each cluster c_i ∈ C) do
6  |  for (each element el ∈ c_i ) do
7  |  |  ExecuteRule(insertInstance, el, hasElement(O, appC);
8  |  end
9  |  for (each feature f_j ∈ sf) do
10 |  |  fC ← GetClass(O,f_j);
11 |  |  val ← GetValue(c_i, f_j);
12 |  |  ExecuteRule(insertInstance, val, fC);
13 |  |  setObj ← GetObjectProperties(hasElement(appC), fC);
14 |  |  for (each objectProperty obj ∈ setObj) do
15 |  |  |  ExecuteRule(insertRelation, obj, c_i, val);
16 |  |  end
17 |  end
18 end
```

example, if the application is *Activity recognition* the ontology must hold the class *Activity* subclass of *Element*. In this case the activities in the clusters become instances of the *Activity* class. Next, each feature value in the cluster becomes an instance of the corresponding feature's class in the ontology using a logic rule (Line 9–12). For example, 5min is a value in the cluster of the feature Duration. Accordingly, 5min becomes an instance of the class *Duration* in the ontology (see Fig. 4). Finally, the algorithm extracts the set of objectProperties having the feature class as a range (Line 13). Then, it relates the instances corresponding to elements in the clusters with theirs features values using the adequate objectProperty. For example, in the ontology (Fig. 4), the feature's class *Duration* is the range of the objectProperty *hasDuration* where the Class *Activity* is its domain. Therefore, Algorithm 1 links the added activity's instances for each cluster with their corresponding durations.

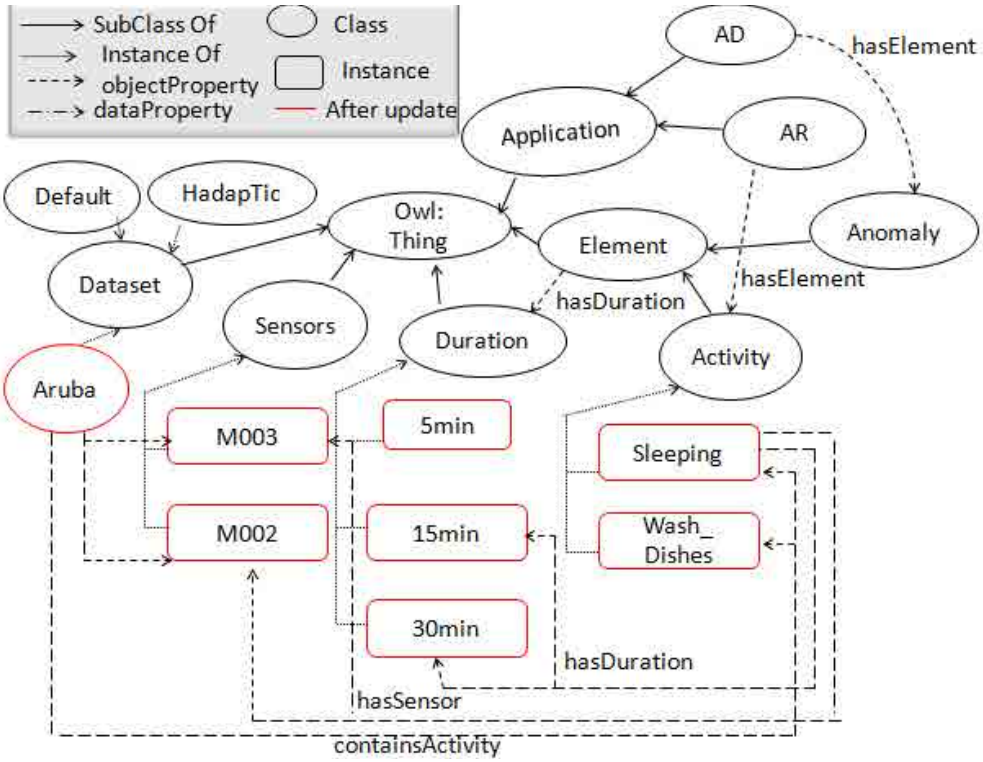



Figure 5.3 An example of the Ontology used in the experiments after being updated with Aruba dataset's clusters. Added elements are in red

### 5.2.3 Window manipulator

Generally, a sensor sequence can be represented as $\{S1, S2...Sn\}$. *Si* refers to the it$^{th}$ sensor event, and each sensor event is encoded in the template of $\{date, time, sensorID, sensorValue\}$. Once the coming sensor data are represented in this template, the main aim of this step is to divide this streaming sensor data into a time window with size "sw". Each chunk "sw" is chosen dynamically using the Window manipulator Algorithm 2.

Initially, the time window size is the minimum duration in the ontology regarding the given application and dataset. It is important to note here that *Duration* is defined as a feature for all applications. For instance, if we are using this method for activity recognition and we set up the method for aruba dataset, then the algorithm sets the initial time window size as the minimum activity's durations belonging to Aruba. This duration should not be used before in the same session (Line 7). Afterwards, the algorithm keeps extracting the set of sensor data (*sensorsData*), that are in the streaming SD, occurred during the defined time window size (*actualSize*). This is achieved using the function *ReadOnline()* (Line 8). The algorithm stops the loop when *sensorsData* are included, in at least one of the feature vectors extracted from the ontology (Lines 12,18) using the *testAppart* function. In other words, based on

**Algorithm 6:** Window Manipulator

**Input:** ontology O, streaming sensor data SD, application app, application's features feats, Dataset's name dName
**Output:** set of possible results possRes, time window size sw

```
1  usedsizes← ∅;
2  sensorsData ← ∅;
3  keep ← true;
4  possRes ← Null ;
5  previousSize ← 0 ;
6  while (keep) do
7      actualSize ← getMinimumDuration(O, usedSizes, previousSize, app, dName);
8      sensorsData ← ReadOnline(SD, actualSize-previousSize, sensorsData);
9      EleFeatVects ← getElementsVectors(O, app, feat);
10     for each appEl ∈ EleFeatVects do
11         if (testAppart(sensorData, appEl) then
12             add(possRes, appEl);
13         end
14     end
15     previousSize ← actualSize;
16     Add(usedSizes, actualSize);
17     if (possRes!= Null) then
18         keep=false;
19         sw← actualSize;
20     end
21 end
```

the given application, the algorithm extracts the features vector for each given application's element. For example, in case of anomaly detection, it extracts for each anomaly its features vector holding the features' values. Then, it checks whether the data in the streaming are belonging to this vector. In the positive case, this element is added to the set of possible results *possRes* (Line 13). Finally, if the algorithm detects at least one possible element then it stops reading (Lines 19, 20). Otherwise, it extends the actual time-window size (*actualSize*) with the difference between it and the next minimum element's duration extracted from the ontology (Line (7)), the actual size becomes the previous size, and this process is repeated until possible results are detected ( Line 17, 18).

## 5.3 Experimental evaluations

To evaluate the usefulness of our proposal DataSeg, we have applied it in case of activity recognition. We tested it with 6 weeks over Aruba dataset[2] and a real dataset dataset acquired from GIS MADONAH[3]. On the one hand, Aruba was acquired in real smart environment by the Center for Advanced Studies in Adaptive Systems (CASAS) [15]. The Aruba dataset contains ground-truthed activities of a home-bound person in a small apartment for 16 weeks. On the other hand, we have acquired a real dataset from the GIS MADONAH smart apartment, that is related to Cocaps projetc. The dataset is about a scenario of 7 hours that we realized. For the activity recognizer method we applied the SVM algorithm. More details about the dataset and the experiments are given in the next subsections.

### 5.3.1 Datasets

Data collected from Aruba dataset was obtained using 31 motion sensors, three door sensors, five temperature sensors, and three light sensors. 11 activities were performed for 220 days (7 months). The dataset is imbalanced, as some of the activities occur more frequently than others. Table 7.1 presents the statistics of the sensor events and activities performed in the 6 weeks over Aruba dataset. “Other activity” class contains events with missing labels. It covers 54% of the entire sensors events sequence.

The real dataset of Gis MADONAH contains sensor data values of an elderly like routine. It contains a set of presence, motions sensors and light detectors. The dataset holds principally four activities: Sleeping, Watching TV, Discomfort, and Eating.

These two datasets were used in DataSeg, on the one hand, as training dataset for SVM and, on the other hand, for the clustering.

### 5.3.2 Evaluation Result

F-Score measure was used to evaluate DataSeg with the two datasets. Moreover, in order to discuss its performance regarding static time-window size, we firstly tested SVM with fixed time-window size (i.e. 5min). [4] Afterwards we used SVM as an activity recognizer in the method. Furthermore, the results are compared with the dynamic data segmentation process, SWMIex [78]. SWMIex is a metrics-based method which uses the Mutual Information measure to compute sensor correlation. SWMIex, SVM with static time window, and our

[2]http://ailab.wsu.edu/casas/datasets/

[3]http://www.bourges.univ-orleans.fr/madonah/

[4] After doing a set of experiments with different time window sizes, five minutes has shown best results for SVM using 6 weeks over Aruba dataset.

| id | Activity | number of events |
|---|---|---|
| 1 | Bed_to_Toilet | 266 |
| 2 | Eating | 3207 |
| 3 | Enter_Home | 404 |
| 4 | Housekeeping | 2117 |
| 5 | Leave_Home | 384 |
| 6 | Meal_Preparation | 57029 |
| 7 | Relax | 70917 |
| 8 | Resperate | 108 |
| 9 | Sleeping | 6536 |
| 10 | Wash_Dishes | 2092 |
| 11 | Work | 3264 |
| 12 | Other Activity | 174 264 |

Table 5.1 Statistics of six weeks over Aruba dataset

proposal are tested in same conditions (i.e dataset). For the process, we used three quarters of each dataset for the offline training and one quarter for the online testing. Figure 5.4 shows the different F-score values of the three methods for each activity belonging to Aruba dataset.

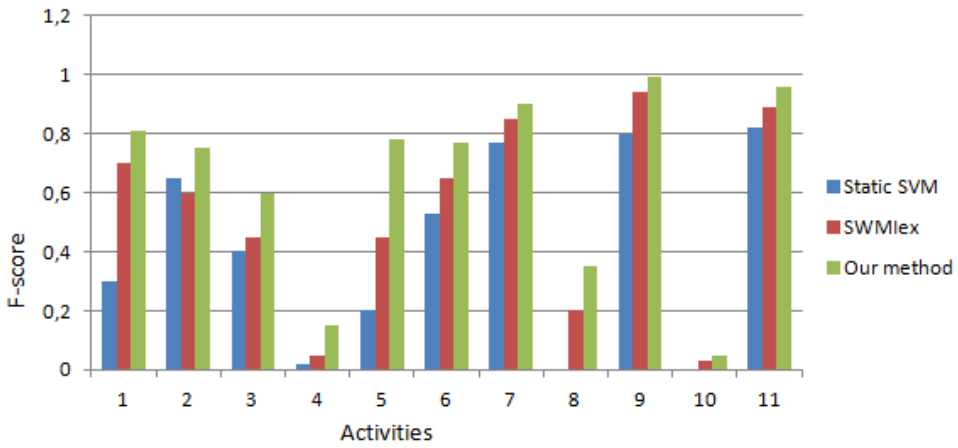



Figure 5.4 F-score values for the different activities in Aruba dataset using only SVM with static time-window length (5 min), DataSeg, and SWMIex methods

As it is shown in Fig. 5.4, DataSeg outperforms the static SVM and SWMIex in terms of F-score. Furthermore, we conclude that the three methods have unbalanced performance. In other words, they have higher efficiency to recognize some activities (i.e. 9. Sleeping, 11. Work) than to detect some others (4.HouseKeeping, 10.Wash_Dishes, 8.Resperate).

| Dataset /method | Static SVM | DataSeg |
|---|---|---|
| Gis MADONAH dataset | 0.75 | 0.87 |
| Six weeks over Aruba | 0.40 | 0.64 |

Table 5.2 Average F-score for all activities

This result, has two possible explanations. On the one hand, since also the data et is unbalanced then infrequent activities have low chance to be recognized such as the *Resperate* activity. On the other hand, the *used sensor* is a feature for the three methods and some activities in the dataset can have different used sensors for each occurrence. The methods are then sometimes confused ( i.e *HouseKeeping*).

The SVM with 5 min as time-window length has the lowest performance (F-score $\in$ [0..0.8]) since it has a static segmentation. In fact, the duration of each activity varies according to the resident routine. For some activities five minutes is sufficient for the recognition and for others this length is either too long or too short. On the other hand, DataSeg has the best performance (F-score$\in$ [0.38..0.99]) thanks to its hybrid dynamic segmentation.

Table 5.2 shows the average F-score values of DataSeg compared to the static SVM using the Gis MADONAH and Aruba datasets. Obviously, as we can see in the table, DataSeg is more performing using these two datasets and outperforms the Static SVM. The better F-score value obtained by DataSeg using GIS MADONAH.

## 5.4 Conclusion

In this chapter we proposed, DataSeg, a hybrid method for dynamic streaming sensor data segmentation. The proposal combines clustering and ontology techniques to provide high flexibility in order to be used in different types of applications. As a matter of fact, the main advantage of our method regarding previous works is its ability to be adapted with different application purposes. Moreover, it is able to dynamically update the ontology whenever any update in the clusters occurs. In order to prove the efficiency of DataSeg, we tested it in case of activity recognition application. DataSeg has proven a good precision level better than static method and a previous related work for the same dataset.

# Chapter 6

# Conclusion

## 6.1 Introduction

Pushed by the advancement of the population, specifically toward more established populace, we are facing an expanding interest for individual and household care. Due to this need, smart environments are presently rising. Indeed, such technology is nowadays affordable and more widespread than ever. It monitors proactive reactions in the environment in order to perform services to the user exploiting the data provided by the deployed sensors. In order to achieve this aim, these data follow a life cycle that consists generally of data acquisition, processing and analysis before making decisions. In this thesis, we mainly focused on the processing and analysis phases, as even if a lot of work in the literature has already been undertaken in this area, there is still room for improvement regarding results precision especially when no large data volumes are available.

By proposing several contributions we have acquired a rich experience in smart environments area. From this understanding, we explore different opportunities for cross-fertilisation of statistical learning and semantic reasoning and we came out with the conviction of the necessity of using both approaches to face various Artificial Intelligence challenges.

This conclusion is worth considering for researchers and future works. In fact, our work opens up multiple perspectives, including both improvement of our proposals, and new research axes. Let us discuss those in this conclusive chapter.

## 6.2 Thesis contributions

After being acquired, the sensor data in a smart environment need to be processed and analyzed in order to provide, afterwards, required decisions by the target application. Through

our literature review, we pointed out five open research challenges. On the one hand, the processing stage, consisting mainly of data segmentation, should be online, dynamic and applicable to multiple application purposes. On the other hand, data uncertainty should be considered in the whole process. Particularly, in this thesis, we have focused mainly on activity recognition and anomaly detection issues for data analysis. Data uncertainty is particularly a topical challenge and it should be considered during the analysis. Indeed, both issues precision is sensible to the data uncertainty and omitting it may lead to wrong results. Yet, most previous approaches require either large amount of data or regular human intervention to provide precise results considering data uncertainty.

Overcoming those limits is essential to enable efficient processing and analysis for applications in smart environments. In this thesis, we proposed the following contributions to tackle them:

- **DataSeg: Dynamic streaming sensor data segmentation**. In order to online process the sensor data, we proposed a method able to first, dynamically segment the streaming sensor data by varying the window size, and second, to be sufficiently generic so as to be capable of supporting various applications.

  In order to achieve this aim, we propose an hybrid approach that takes advantage of both data-based and knowledge-based methods. The general principle of the approach is as follows:

  Given a user-annotated data collection reflecting standard daily living annotated activities and a set of application's features, the annotations are grouped according to the given features through a clustering algorithm.

  The obtained clusters are semantically represented using our defined ontology. During the online process, the ontology is queried by our segmentation algorithm in order to retrieve the knowledge helping the algorithm to choose the optimal size of the current data segment.

- **AGACY Monitoring for Activity recognition and uncertainty handling**. To continuously analyze the streaming sensor data and provide the knowledge about the current user activity, we proposed a hybrid method that uses an ontology as knowledge representation and a supervised learning algorithm. This hybridization considers data uncertainty and provides accurate results even with small amount of data about the user. Moreover, it does not require regular expert intervention. The semantic layer, using an ontology and the possibilistic logic represents semantically the sensor data which allows, on the one hand, to infer knowledge about the user (i.e. his/her realized

actions) and, on the other hand, to propagate sensor data uncertainty values and compute new ones of the inferred events. The supervised data driven algorithm receives the knowledge provided by the semantic layer to learn the user activities and compute their uncertainty values based on those of the inferred events.

- **Early anomaly detection**. To early detect anomalous situations that require proactive reactions to avoid their occurrence, we used MLN along side with CAR mining. By relying on MLN, that combines formal logic and Markovian modelling, the proposal can take into account uncertainty of data modelled through weights. The CAR mining extracts automatically logic rules about anomalies causes from a given dataset in order to be used as MLN rules instead of being manually defined by human experts. Moreover, the extracted anomalies causes allows to give an explanation about the detected risk of anomaly.

**Discussion** As the proposed contributions have several advantages and have proved their viability through different experiments and evaluations, they have also some limits. Firstly, for an important part of our work we have focused on the supervised learning (i.e SVM). Despite the high accuracy that could provide such algorithms, obtaining annotated dataset is still difficult. It requires, on the one hand, the user agreement for being monitored, and expert intervention, on the other hand, to annotate the obtained dataset. Users do not accept always monitoring their lives even for a period of time and experts are not often available. To avoid these problems, focusing on the improvement of some unsupervised learning algorithms could be an alternative to provide precise results without the need of user monitoring and expert intervention. Secondly, most of our contributions have an online operation (i.e DataSeg, AGACY Monitoring). Maybe *online learning algorithms* could be better employed for this aim since they are particularly conceived to learn on an online fashion. Thirdly, our contributions handle the data uncertainty as one value for each data representing its overall uncertainty. However, uncertainty has several dimensions [91], it could be better to study the effect of each dimension on the results. Finally, throughout some of our proposals (AGACY Monitoring and DataSeg) we have designed new ontologies for different aims. Nevertheless, we believe that it could be better to reuse existing ontologies which allows to encourage re-usability and gain effort and time.

## 6.3 The need for both learning and reasoning

Through our contributions we have used several times a combination between data driven and semantic based techniques. Each technique has its strengths and weaknesses for processing

and analysis the sensor data in smart environments. On the one hand, semantic based techniques, including reasoning methods, allow to better understand the environment without a long and possibly erroneous learning phase. On top of that, they are usually compatible with uncertain, complex and heterogeneous data modelling, enabling a complete context understanding. Nevertheless, as environments are open and various, it is impossible to specify everything, for every case. On the other hand, data driven techniques are suitable for learning models from massive volumes of sensor data to infer the current state and predict its future progression which is essential in domestic applications. Yet, learning techniques mostly require large amount of data to be precise, lack depth of understanding, and would require a strong learning phase if used by themselves.

As a result, both semantic and data driven approaches are required for a sensor data processing and analysis in smart environments. The knowledge provided by semantic based techniques provides semantic interpretation of the received sensor data while the learning phase in data driven techniques is a means of adaptation to new situations. In this thesis, we have experimented various combination approaches that are summarized as follows:

- DataSeg: combines an ontology and a clustering algorithm for the dynamic segmentation of streaming sensor data.
- AGACY Monitoring: uses possibilistic logic and an ontology along side with SVM algorithm to continuously recognize user daily activity.
- The early anomaly detection method: uses MLN, which is by nature a combination between formal logic and markov models, and CAR mining approach to extract logic rules that will be directly exploited by MLN.

That being said, however, even both semantic and data driven based techniques are essential for sensor data processing and analysis, the combination should be done in an adequate manner according to users and environments constraints. In fact, a main important factor is the position of each technique in the approach. In other words, the semantic technique can be employed before, after, or in parallel with the data driven based one. We state that **this order defines the combination strategy of these two techniques**. The choice of this order depends of different parameters. For example, AGACY Monitoring starts by representing semantically the sensor data and inferring new knowledge about the user events using ontologies and logic reasoning. Afterwards, it applies a supervised learning algorithm to learn the user daily activities. This combination works well for AGACY Monitoring since a training dataset about the user activities is provided. Despite this dataset is not large, the method was precise thanks to the prior knowledge and the reasoning ability provided by the

ontology. However, whether the user refuses being monitored or not may influence the data collection and then by the same way the combination order.

Moreover, we think that **both techniques should be employed to complement each other**. This states that they must not be used for the same purpose. Indeed, in our contributions, both approaches were used whenever for the purpose of enriching each other. For instance, in DataSeg, the clustering provides knowledge about the annotation of the dataset, the ontology serves then to semantically represent this knowledge and infer a new one. Nevertheless, when semantic and data driven based techniques are used for the same aim, unwanted results may be obtained as affirmed in [91].

## 6.4 Perspectives

Taking feedbacks from experience into account is important for upcoming works. In fact, this thesis has conducted a set of contributions that open up multiple perspectives for further research. We present possible perspectives we identified. Note that they are ordered by the distance, in both time and effort, from the current state of our works.

- **Experiments consolidation**
  Each chapter holds the different experiments that have been realized to evaluate the presented contribution. However, these experiments need some improvements. A first perspective is to consolidate our experiments by evaluating the prototypes with real dataset. Moreover, we plan to compare the results with other methods using the same data. To do so, we can rely on the HadapTIC platform, but also on the EVIDENT[1] platform at Telecom sudParis, that is a complete apartment equipped with diverse smart devices including smart carpets, lights and others. Moreover, we plan to perform real scenarios in the Gis MADONAH smart apartment related to the project COCAPS. This can be performed in a relatively short term.

- **Prototype implementation**
  As detailed in each chapter, our contributions were implemented and experimented on its own. However, since the contributions are related, as a second perspective we plan to implement a framework holding all the prototypes and enabling interactions between them to provide a unified output. This integration will reinforce the contributions by allowing efficient and reliable communications. From this, we aim to perform further experiments, in particular with the framework full deployed.

[1] https://evident.telecom-sudparis.eu/

- **Decision making in smart environments**
  As mentioned in previous chapters, our proposals open research problems and new contributions can be conducted. After analyzing the data and extracting knowledge about user ADL and anomalies, the question that should be asked is *"How to exploit this important knowledge about anomalies and user ADL for making decisions ?"*. To answer this question, as a health care application, our proposal of early anomaly detection integrates a process by exploiting the knowledge about the anomaly it allows to propose actions in order avoid the anomaly occurrence. However, this process needs improvements in order to be a different and separated model able to reason about actions in a more deeper and general way than that we proposed. In the literature, different from the health care applications, several domestic applications have been raised to provide services to the user based on his/her known ADL. Personalized energy consumption, reminder, and recommender systems are examples of such applications. We have started studying the decision making topic of health care and domestic applications and we have observed that besides his/her ADL, the user preferences play an important role for the decision makers by allowing to provide more adapted decisions. Therefore, user's ADL and comfort preferences have recently been explored in different works. Each of these works is designed to achieve a particular purpose (e.g anomaly prevention, energy saving, recommending services, etc.). However, we believe that there is a need of a decision maker able to be adapted to the current user situation for different purposes. For instance, such a system can react in case of observed risk of anomalies by taking into account the user preference about the reactions, and also can act on the environment to reduce the energy consumption considering the user comfort and his/her cost preferences. We state that defining such a system reduces the number of deployed decision making applications in the smart environments. The next steps are then to model this system, study which are the borders of its generality and its development constraints.

- **Activity recognition**
  Our contribution, AGACY Monitoring, for activity recognition has shown promising results for handling sensor data uncertainty while recognizing the current user activity. Nevertheless, this work has also its limitations due to two assumptions that have been made to frame it. Firstly, AGACY Monitoring is a single user's activity recognition, we suppose the user is living alone. This means that when more than one person are in the smart home, AGACY Monitoring is not able to distinguish the activity of each one. In real life, a smart home's multi-residents is very common. Therefore, we believe that it is important to improve AGACY monitoring with ability of multi-user activity

recognition. This is still a challenging problem despite the technology advancement on the activity recognition topic. This task may be facilitated using wearable sensors or cameras observing the users. However, this problem is more difficult when intrusive sensors such cameras are avoided and no constraint about using wearable sensors is made. In this case, the difficulty lies on the fact that it is important to differentiate the activities of residents without using cameras and wearable sensors. Secondly, AGACY Monitoring is only suitable for sequential single user activity recognition. In other word, the system is not able to handle a situation when the user is doing parallel activities such as phoning while cooking or watching tv while eating.

# Bibliography


[1] Aarti Singh, Dimple Juneja, A. S. (2011). A fuzzy integrated ontology model to manage uncertainty in semantic web: the fiom. *International Journal of Computer Science Engineering*, vol. 3.

[2] Aditi Kapoor, K.K. Biswas, M. H. (2017). Unusual human activity detection using markov logic networks. In *IEEE International Conference on Identity, Security and Behavior Analysis*. IEEE.

[3] Alejandro J. Garcıa, G. R. S. (2004). Defeasible logic programming: An argumentative approach. *Theory and Practical Logic Programming*, vol. 4.

[4] Amina Jarraya, Nathan Ramoly, A. B. and al. (2016a). A fuzzy semantic cep model for situation identification in smart homes. In *European Conference on Artificial Intelligence*. IEEE.

[5] Amina Jarraya, Nathan Ramoly, A. B. and al. (2016b). A new model for context perception in smart homes. In *International Conference on Cooperative Information Systems*. Springer.

[6] Anand Ranganathan, Jalal Al-Muhtadi, R. H. C. (2004). Reasoning about uncertain contexts in pervasive computing environments. *IEEE Pervasive Computing*, vol. 3.

[7] Asad Masood Khattak, Wajahat Ali Khan, Z. P. F. I. S. L. (2016). Towards a self adaptive system for social wellness. *Sensors*, vol. 15.

[8] B. J.A. Krose, Tim van Kasteren, C. G. T. V. d. D. (2008). Care: Context awareness in residences for elderly. In *International Conference of the International Society for Geron-technology*. Springer.

[9] Breiman, L. (2001). Random forest. *Machine Learning*, vol. 45.

[10] Choonsung Shin, W. W. (2009). Service conflict management framework for multi-user inhabited smart home. *Journal of Universal Computer Science*, vol. 15.

[11] Christian Wojek, Kai Nickel, R. S. (2006). Activity recognition and room-level tracking in an office environment. In *Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*. IEEE.

[12] Chuanjun Li, B. P. (2005). A similarity measure for motion stream segmentation and recognition. In *International workshop on Multimedia data mining: mining integrated media and complex data*. Springer.

[13] Claudia C. Gutierrez Rodriguez, S. S. (2013). Managing sensor data uncertainty: A data quality approach. *International Journal of Agricultural and Environmental Information Systems*, vol. 4.

[14] Constantin F. Aliferis, Alexander Statnikov, I. T. and al. (2010). Local causal and markov blanket induction for causal discovery and feature selection for classification part i: Algorithms and empirical evaluation. *Journal of Machine Learning Research*, vol. 11.

[15] Cook, D. J. (2010). Learning setting-generalized activity models for smart spaces. *IEEE Intelligent Systems*, vol. 99.

[16] Cooper, G. F. (1997). A simple constraint-based algorithm for efficiently mining observational databases for causal relationships. *Data Mining and Knowledge Discovery*, vol. 1.

[17] Craig Silverstein, Sergey Brin, R. M. J. U. (2000). Scalable techniques for mining causal structures. *Data Mining and Knowledge Discovery*, vol. 4.

[18] D. Dubois, J. Lang, H. P. (1994). Automated reasoning using possibilistic logic: Semantics, belief revision, and variable certainty weights. *IEEE Transaction on Knowledge and data engineering*, vol.6.

[19] Daniele Ribon, Claudio Bettini, G. C. and al. (2016). Smartfaber: Recognizing fine-grained abnormal behaviors for early detection of mild cognitive impairment. *Artificial Intelligence in Medicine*, vol. 67.

[20] Daniele Riboni, C. B. (2009). Context-aware activity recognition through a combination of ontological and statistical reasoning. In *IEEE International Conference on Ubiquitous Intelligence and Computing*. IEEE.

[21] Daniele Riboni, C. B. (2011). Cosar: Hybrid reasoning for context-aware activity recognition. *Personal and Ubiquitous Computing*, vol. 3.

[22] Daniele Riboni, Claudio Bettini, C. C. Z. H. J. R. H. (2015). Fine-grained recognition of abnormal behaviors for early detection of mild cognitive impairment. In *International Conference on Pervasive Computing and Communication*. IEEE.

[23] Darnell Moore, I. E. (2002). Recognizing multitasked activities from video using stochastic context-free grammar. In *Eighteenth national conference on Artificial intelligence, American Association for Artificial Intelligence*. IEEE.

[24] Darpan Triboan, Liming Chen, F. C. Z. W. (2017). Semantic segmentation of real-time sensor data stream for complex activity recognition. *Personal and Ubiquitous Computing*, vol. 21.

[25] De, D. (2014). Sensor networks for smart environments. *Journal of Telecommunications System and Management*, vol. 1.

[26] De Backere F, Ongenae F, V. d. A. F. and al. (2015). Towards a social and context-aware multi-sensor fall detection and risk assessment platform. *Computers in Biology and Medicine*, vol. 64.

[27] Derek T. Anderson, Marıa Ros, a. M. K. and al. (2012). Similarity measure for anomaly detection and comparing human behaviors. *International Journal of Intelligent Systems*, vol. 27.

[28] Dimitri Solomatine, Linda M. See, R. J. A. (2017). Chapter 2 data-driven modelling : Concepts , approaches and experiences. In *Practical Hydroinformatics*. Springer.

[29] Djallel Bouneffouf, Amel Bouzeghoub, A. L. G. (2012). Hybrid-$\varepsilon$-greedy for mobile context-aware recommender system. In *Advances in Knowledge Discovery and Data Mining*. Springer.

[30] Donald J. Patterson, Lin Liao, D. F. and Kautz, H. (2003). Inferring high-level behavior from low-level sensors. In *Proceedings of Proceedings of the 5th International Conference on Ubiquitous Computing*. Springer.

[31] Douglas L. Vail, Manuela M. Veloso, J. D. L. (2007). Conditional random fields for activity recognition. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent System*. ACM.

[32] Dung, P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, vol. 77.

[33] Emmanuel Munguia, TapiaStephen S. Intille, K. L. (2004). Activity recognition in the home using simple and ubiquitous sensors. In *Proceedings of the international conference on Pervasive Computing*. Springer.

[34] Enamul Hoque, Robert F. Dickerson, S. M. P. M. H. A. B. J. A. S. (2016). A comprehensive anomaly detection system for daily in-home activities. In *International Conference on Distributed Computing in Sensors Systems*. IEEE.

[35] Evan H. Magill, J. B. (2016). Exploring conflicts in rule-based sensor networks. *Pervasive and Mobile Computing*, vol. 27.

[36] Faouzi Sebbak, Farid Ben hammadi, A. C. and al. (2014). Dempster–shafer theory-based human activity recognition in smart home environment. *Annal Telecommunication*, vol. 69.

[37] George Okeyo, Liming Chen Hui Wang, R. S. (2014). Dynamic sensor data segmentation for real-time knowledge-driven activity recognition. *Pervasive and Mobile Computing*, vol. 19.

[38] Gianpaolo Cugola, Alessandro Margara, M. M. G. T. (2015). Introducing uncertainty in complex event processing: model, implementation, and validation. *Computing*, vol. 2.

[39] Grando MA, Moss L, S. D. K. J. (2013). Argumentation-logic for creating and explaining medical hypotheses. *Artificial Intelligence in Medicine*, vol. 15.

[40] Haiquan Chen, Wei-Shinn Ku, H. W. and al. (2016). Scaling up markov logic probabilistic inference for social graphs. *IEEE Transactions on Knowledge and Data Engineering*, vol. 29.

[41] Hamdi Aloulou, Mounir Mokhtari, T. T. and al. (2015). Uncertainty handling in semantic reasoning for accurate context understanding. *Knowledge-Based Systems*, vol. 77.

[42] Hayes, M. (2015). Contextual anomaly detection framework for big sensor data. *Journal Of Big Data*, vol. 28.

[43] Heckerman, D. (1997). Bayesian networks for data mining. in: Data mining and knowledge discovery. *Data Mining and Knowledge Discovery*, vol. 1.

[44] Hela Sfar, Nathan Ramoly, A. B. B. F. (2017a). Caredas: Context and activity recognition enabling detection of anomalous situation. In *Artificial Intelligence in Medecinein Europe*. Springer.

[45] Hela Sfar, Amel Bouzeghoub, N. R. J. B. (2017b). Agacy monitoring: a hybrid model for activity recognition and uncertainty handling. In *Extended Semantic Web Conference*. Springer.

[46] Jhun-Ying Yang, Jeen-Shingn Wang, Y.-P. C. (2008). Using acceleration measurements for activity recognition: an effective learning algorithm for constructing neural classifiers. *Pattern Recognition Letter*, vol. 29.

[47] Jiahui Wena, Mingyang Zhong, Z. W. (2015). Activity recognition with weighted frequent patterns mining in smart environments. *Expert Systems with Applications*, vol. 42.

[48] Jianping Zhang, I. M. (2003). Knn approach to unbalanced data distributions: A case study involving information extraction. In *Workshop on learning from unbalanced dataset II, ICML.*

[49] Jie Wan, Michael J. O'Grady, G. O. (2015). Dynamic sensor event segmentation for real-time activity recognition in a smart home context. *Personal and Ubiquitous Computing*, vol. 19.

[50] Jinlong Huang, Qingsheng Zhu, L. Y. J. F. (2016). A non-parameter outlier detection algorithm based on natural neighbor. *Knowledge Based Systems*, vol. 92.

[51] Jiuyong Li, Thuc Duy Le, L. L. J. L. Z. J. B. S. (2013). Mining causal association rules. In *Data Mining Workshops*. Springer.

[52] JUAN YE, LORCAN COYLE, S. D. P. N. (2007). Ontology-based models in pervasive computing systems. *The Knowledge Engineering Review*, vol. 22.

[53] Juan Yen Simon Dobson, S. M. (2012). Situation identification techniques in pervasive computing. *Pervasive and Mobile Computing*, vol. 9.

[54] June Almenoff, Joseph M. Tonning, A. L. G. and al. (2007). Perspectives on the use of data mining in pharmaco-vigilance. *Drug Safety*, vol. 28.

[55] K. S. Gayathri, Susan Elias, B. R. (2015). Hierarchical activity recognition for dementia care using markov logic network. *Personal and Ubiquitous Computing*, vol. 15.

[56] K. S. Gayathri, S. E. S. (2014). An ontology and pattern clustering approach for activity recognition in smart environments. In *International Conference on Soft Computing for Problem Solving*. Springer.

[57] Kejun Du, Daqing Zhang, X. Z. and al. (2008). A hybrid context-aware reminding framework for elders with mild dementia. *Sensors*, vol. 15.

[58] Laura Moss, Derek Sleeman, M. S. and al. (2010). Ontology-driven hypothesis generation to explain anomalous patient responses to treatment. *Knowledge-Based Systems*, vol. 15.

[59] Lauro Snidaro, Ingrid Visentini, K. B. (2015). Fusing uncertain knowledge and evidence for maritime situational awareness via markov logic networks. *Information Fusion*, vol. 15.

[60] Leila Amgoud, P. B. (2013). A formal characterization of the outcomes of rule-based argumentation systems. In *Iternational Conference on Scalable Uncertainty Management*. Springer.

[61] Liming Chen, Chris Nugent, G. O. (2014). An ontology-based hybrid approach to activity modeling for smart homes. *IEEE Transactions on Human-Machine Systems*, vol. 44.

[62] Ling Bao, S. S. I. (2004). Activity recognition from user-annotated acceleration data. In *Proceedings of the Second International Conference on Pervasive Computing*. Springer.

[63] Loke, S. W. (2010). Incremental awareness and compositionality: a design philosophy for context-aware pervasive systems. *Pervasive and Mobile Computing*, vol. 6.

[64] M.A. Hearst, S.T. Dumais, E. O. J. P. B. S. (1998). Support vector machines. *IEEE Intelligent Systems and their Applications*, vol. 13.

[65] M.A.Friedl, C. (1997). Decision tree classification of land cover from remotely sensed data. *Remote Sensing of Environment*, vol. 61.

[66] Manfred Hauben, David Madigan, C. M. G. and al. (2005). The role of data mining in pharmacovigilance. *Expert Opinion Drug Safety*, vol. 4.

[67] Marek Novak, Miroslav Binas, F. J. (2012). Unobtrusive anomaly detection in presence of elderly in a smart-home environment. In *ELEKTRO*. Springer.

[68] Marek Novak, Frantisek Jakab, L. L. (2013). Anomaly detection in user daily patterns in smart-home environment. *Journal of Selected Areas in Health Informatics*, vol. 3.

[69] Matthew Richardson, P. D. (2006). Markov logic networks. *Machine learning*, vol. 62.

[70] Md. Kamrul Hasan, Husne Ara Rubaiyeat, Y.-K. L. S. L. (2008). A reconfigurable hmm for activity recognition. In *Proceedings of 10th International Conference on Advanced Communication Technology*. Springer.

[71] Melisachew Wudage Chekol, Jakob Huber, C. M.-H. S. (2016). Markov logic networks with numerical constraints. In *European Conference on Artificial Intelligence*. IEEE.

[72] Michael A. Hayes, M. A. C. (2017). Contextual anomaly detection framework for big sensor data. In *IEEE International Conference on Identity, Security and Behavior Analysis*. IEEE.

[73] M.S. Ryoo, J. A. (2006). Recognition of composite human activities through context-free grammar based representation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE.

[74] Mukhopadhyay, S. C. (2016). Activity and anomaly detection in smart home: A survey. *Next Generation Sensors and Systems*, vol. 12.

[75] Narayanan C. Krishnan, D. J. (2014). Activity recognition on streaming sensor data. *Pervasive and Mobile Computing*, vol. 10.

[76] Narayanan Chatapuram Krishnan, D. J. C. (2014). Activity recognition on streaming sensor data. *Pervasive and Mobile computing*, vol. 10.

[77] Natalia Díaz-Rodríguez, Olmo León Cadahía, M. P. C. and al. (2014). Handling real-world context awareness, uncertainty and vagueness in real-time human activity tracking and recognition with a fuzzy ontology-based hybrid method. *Sensors*, vol. 14.

[78] Nawel Yala, Belkacem Fergani, A. F. (2016). Towards improving feature extraction and classification for activity recognition on streaming data. *Journal of Ambient Intelligence and Humanized Computing*, vol. 8.

[79] Nguyen, Tuan Anh, A. M. (2013). Energy intelligent buildings based on user activity. *Energy and Buildings*, vol. 56.

[80] Nicola Guarino, Daniel Oberle, S. S. (2009). What is an ontology? In *book on ontology*, chapter 8, pages 1–17. Springer.

[81] P. Salamon, L. H. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12.

[82] Paul Lukowicz, Gerald Pirkl, D. B. and al. (2010). Recording a complex, multi modal activity data set for context recognition. In *Workshop on Context-Systems Design, Evaluation and Optimisation at ARCS*. IEEE.

[83] Paulo Jorge Carreira, Silvia Resendes, A. C. S. (2014). Towards automatic conflict detection in home and building automation system. *Pervasive and Mobile Computing*, vol. 12.

[84] Pavan Turaga, Rama Chellappa, V. S. S. O. U. (2008). Machine recognition of human activities: a survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 18.

[85] Pearl, J. (1994). From bayesian network to causal networksy. *Bayesian Networks and Probabilistic Reasoning*, vol. 1.

[86] Peter Auer, N. C.-B. and Fischer, P. (1992). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, vol. 8.

[87] Philippe Besnarda, A. H. (2001). A logic-based theory of deductive arguments. *Artificial Intelligence*, vol. 128.

[88] Qian Wan, Lubbock, Y. L. C. L. R. P. (2014). Gesture recognition for smart home applications using portable radar sensors. In *36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE.

[89] Ramesh Paudel, William Eberle, L. B. H. (2018). Anomaly detection of elderly patient activities in smart homes using a graph-based approach. In *International Conference on Data Sience*. ISBN.

[90] Ramljak, M. (2017). Smart home medication reminder system. In *25th International Conference on Software, Telecommunications and Computer Networks*. IEEE.

[91] Ramoly, N. (2018). *Contextual Integration of Heterogeneous Data in an Open and Opportunistic Smart Environment: Application to Humanoid Robots*. PhD thesis, Telecom SudParis.

[92] Rim Helaoui, Daniele Riboni, H. S. (2013). A probabilistic ontological framework for the recognition of multilevel human activities. In *International joint conference on pervasive and ubiquitous computing*. ACM.

[93] Rish, I. (2001). An empirical study of the naive bayes classifier. In *International Joint Conference on Artificial Intelligence*. IEEE.

[94] Ruichu Cai, Mei Liu, Y. H. B. M. and al. (2017). Identification of adverse drug-drug interactions through causal association rule discovery from spontaneous adverse event. *Artificial Intelligence In Medecine*, vol. 75.

[95] Ruichu Cai, Anthony K. H. Tung, Z. Z. Z. H. (2013). What is unequal among the equals? ranking equivalent rules from gene expression data. *IEEE Transaction of Knowledge Data Engineering*, vol. 15.

[96] Samir V.Zanjala, G. R. T. (2016). Medicine reminder and monitoring system for secure health using iot. In *International Conference on Information Security and Privacy*. Elsevier.

[97] Tao Gu, Hung Keng Pung, D. Z. D. Z. (2004a). Bayesian approach for dealing with uncertain contexts. In *Proceedings of Advances in Pervasive Computing, coexisted with Pervasive'04. Austrian Computer Society*. Springer.

[98] Tao Gu, Xiao Hang Wang, H. K. P. D. Q. Z. (2004b). An ontology-based context model in intelligent environments. In *Conference on Communication Network and Distributed Systems*. IEEE.

[99] Tian Guo, Zhixian Yan, K. A. (2012). An adaptive approach for online segmentation. In *International Workshop on Data Engineering for Wireless and Mobile Access*. Springer.

[100] VARUN CHANDOLA, ARINDAM BANERJEE, V. K. (2009). Anomaly detection : A survey. *ACM Computing Surveys*, vol. 28.

[101] Vikramaditya Jakkula, D. J. C. (2011). Detecting anomalous sensor events in smart home data for enhancing the living experience. In *Association for the Advancement of Artificial Intelligence*. IEEE.

[102] WATKINS, C. J. (2002). Q-learning. *Machine Learning*, vol. 47.

[103] Yongkoo Han, Manhyung Han, S. L. and al. (2012a). A framework for supervising lifestyle diseases using long-term activity monitoring. *Sensors*, vol. 12.

[104] Yongkoo Han, Manhyung Han, S. L. and al. (2012b). Smart homes for the elderly dementia sufferers: identication and prediction of abnormal behavior. *Journal of Ambient Intelligent of Hymanizied Computing*, vol. 3.

[105] Zadeh, L. A. (1986). A simple view of the dempster-shafer theory of evidence and its implication for the rule of combination. *Artificial Intelligence magazine*, vol. 7.

[106] Zaffar Haider Janjua, Daniele Riboni, C. B. (2016). Towards automatic induction of abnormal behavioral patterns for recognizing mild cognitive impairment. In *ACM/SIGAPP Symposium on Applied Computing*. ACM.

# Appendix A

# Reasoning under Conflicts in Smart Environment

## A.1 Introduction

Hybrid methods, that combine between semantic and data-driven based techniques, are considered the best candidates to be employed for the analysis and the processing of sensor data, as demonstrated in Chapter 2. On the one hand, hybrid methods are able to provide good results event with small amount of data unlike data driven based techniques. On the other hand, they do not require regular human expert intervention which is considered as the main drawback of semantic based techniques. Accordingly, in this thesis, we have proposed different hybrid methods for sensor data segmentation, activity recognition, and early anomaly detection that are presented in Chapter 3, 4, and 5.

Obviously, semantic based techniques (i.e ontology and logic reasoning) are an important part of hybrid methods. In such techniques, definition of logic rules is mandatory thus it can be called also rule-based methods. In fact, rule-based methods differ from standard procedural or object-oriented programs in that there is no clear order in which code executes. Instead, the knowledge of the expert is captured in a set of rules, each of which encodes a small piece of the expert's knowledge. However, conflicts may occur during the update or management of the rules, when different human experts contribute to their definition or when there is a contradiction in the sources of information. As a result, a conflict decreases the method performance and changes its behavior. To illustrate the problem, let us consider the following scenario:

**Scenario 1**: *Mary is an elderly woman living alone in her smart house. The house is equipped with multiple sensors and a rule-based is included in the decision maker aiming to reason*

*about Mary's daily activities based on a set of defined rules about her life routine. During the set up of the method and according to the life routine of Mary, the expert defines a logic rule stating that it is possible to conclude that Mary is eating Bread ("EatBread") only if Mary has fetched the Knife ("FetchKnife) and fetched the Bread ("FetchBread"). However, due to some health issues, the life routine of Mary changes: for eating bread she must be sitting also. Therefore, another expert updates the set of rules and adds another rule stating that the "EatBread" can be inferred from the three conditions "FetchKnife", "FetchBread" and "SitOn". This latter, nevertheless, does not remove the previous rule which gives rise to the creation of two conflicting rules in the system. As a result, erroneous activities are provided by the system which decreases its accuracy.*

Scenario 1 illustrates possible conflicts that may often occur in rule-based methods for smart environments. To the best of our knowledge, this type of conflict which we call *contrariness* is not treated in the literature. Indeed, previous works on rule-based methods focus in general on handling conflict only based on *inconsistency* [3, 60]. More precisely, as stated in chapter 2, existing approaches omit this form of conflicts that frequently occur in smart environment where additional preconditions are required for the conclusion of a rule to hold.

In this appendix, we propose to address this problem by adopting argumentation theory to reason over conflicts in rule-based methods. Argumentation theory consists of a set of arguments, attacks between them and a semantics for identifying the acceptable arguments. An argument is a reason for believing a statement, doing an action, etc. One of the key aims of argumentation research is to provide principled techniques for handling conflicts. Various approaches have been proposed to formalize argumentation and there are two major families of structured argumentation systems: deductive argumentation [87] and defeasible argumentation [3, 60]. Despite the popularity of these systems, the attack among arguments is mainly rooted by inconsistency. However, real-life (counter)-arguments do not necessarily appear mutually inconsistent. It is the exploration of this gap, then, that this appendix is aimed at which is the advantage of this work. The appendix presents an argumentation framework in the context of rule-based methods where the notion of counter-argumentation is introduced through two sorts of conflicts, namely, *inconsistency* and *contrariness* among two arguments.

To resume, through this appendix we propose a generic model for reasoning under conflicts in rule-based methods by the exploitation of argumentation theory. The model is general enough to be able to handle conflicts in rules-based methods undependable of the process. Particularly, for smart environments besides *inconsistency* of rules they also may be *contrariness*. The proposal, is designed to treat both types of conflict. This work has been realized in collaboration with other members of the team.

## A.2 Argumentation Framework for rules-based methods

Argumentation is a form of reasoning that studies different ways in which an argument for or against a proposition can be made, and then determine which arguments attack which other arguments in an effort to decide what can be reasonably concluded. In this section, we introduce a formalism, and investigate a natural extension of deductive argumentation's well-known model of argument systems [87].

### A.2.1 Rule-based knowledge representation

A logical rule is usually expressed in the form:"if the precondition $\phi$ holds then the conclusion $\psi$ holds", where the precondition and the conclusion are logical properties. However, such a piece of knowledge cannot simply be formalized through the standard material implication of classical logic: in other words, it is not possible to capture the intended meaning of the above statement by an implication in classical first-order logic of the form $\phi \rightarrow \psi$. In this respect, a very relevant role is played by research in logic programming. In fact, logic program rules are implications with a non-standard semantics. In the sequel, we will consider a simple language $\mathscr{L}_R$ constructed from a set of rules where each rule is of the form $\alpha_1 \wedge \ldots \wedge \alpha_n \rightarrow \beta$ where $\alpha_1$ to $\alpha_n$ and $\beta$ are literals, and let $\vdash$ be the consequence relation for that language. The language $\mathscr{L}_R$ will be used throughout this appendix in order to represent data in smart environment systems.

Since rules in rule-based systems for smart environments may originate from distinct sources and change over time, methods are required to maintain rule consistency. For this, we define the notion of *contrariness* between a pair of rules of $\mathscr{L}_R$ that is intended to encompass both standard inconsistency and a form of disagreement between them. The main idea is to replace classical inconsistency between conflict rules with a more general notion of conflict.

Let us introduce the concept progressively before introducing the formal definition.

Let $\alpha$ and $\beta$ be two rules of $\mathscr{L}_R$. $\alpha$ *is the contrary* of $\beta$ in any of the following cases. First, $\alpha$ and $\beta$ are mutually inconsistent in $\mathscr{L}_R$. Note that this also covers the standard-logic occurrences of conflict. Second, we need to address the case that requires $\alpha$ of the form $\phi \rightarrow \psi$ to contrary $\beta$ of the form $\phi \wedge \gamma \rightarrow \psi$. First, $\alpha$ and $\beta$ must be two rules about the same conclusion, hence $\psi \equiv \psi'$ [1]. Second, we require the precondition of the first rule to entail the precondition of the second one but not conversely; formally, for $\alpha = \phi \rightarrow \psi$ and $\beta = \phi' \rightarrow \psi'$, we should have $\phi \vdash \phi'$ and $\phi' \nvdash \phi$.

[1] The symbol $\equiv$ denotes the logical equivalence, i.e. $\psi \vdash \psi'$ and $\psi' \vdash \psi$.

**Definition 13** *Let $\alpha$ and $\beta$ be two rules of $\mathscr{L}_R$ such that $\alpha = \phi \rightarrow \psi$ and $\beta = \phi' \rightarrow \psi'$. Then, $\alpha$ is the contrary of $\beta$, denoted $\alpha \bowtie \beta$, iff:*

- $\{\alpha, \beta\}$ *is inconsistent, or*
- $\phi \vdash \phi'$, $\phi' \nvdash \phi$, *and* $\psi \equiv \psi'$.

**Example 9** *Let $\alpha = FetchKnife \wedge FetchBread \rightarrow EatBread$ and $\beta = FetchKnife \wedge FetchBread \wedge SitOn \rightarrow EatBread$ be two rules. Then, $\beta \bowtie \alpha$.*

The concept of contrariness of a rule is naturally extended into a concept of contrariness of a set of rules.

**Definition 14** *Let $\Phi$ and $\alpha$ be a subset and a rule of $\mathscr{L}_R$, respectively. $\alpha$ is the contrary of $\Phi$, denoted $\alpha \bowtie \Phi$, iff there exists $\beta$ in $\mathscr{L}_R$ such that $\Phi \vdash \beta$ and $\alpha \bowtie \beta$.*

Obviously, $\bowtie$ is neither symmetric, nor antisymmetric, nor antireflexive. However, it is monotonic as shown by the following proposition.

**Proposition 1** *Let $\Phi$, $\Psi$, and $\alpha$ be two subsets and a rule of $\mathscr{L}_R$, respectively. If $\alpha \bowtie \Phi$, then $\alpha \bowtie \Phi \cup \Psi$.*

It is easy to show the following result.

**Proposition 2** *Let $\Phi$, $\Psi$, and $\alpha$ be two subsets and a rule of $\mathscr{L}_R$, respectively. If $\alpha \equiv \beta$, then $\alpha \bowtie \Phi$ iff $\beta \bowtie \Phi$.*

Next, we define our argumentation framework in the light of the notion of contrariness.

## A.2.2 Arguments

In the literature, deductive argumentation adopts solely consistency to build arguments [87]. However, in the context of rule-based reasoning, we are interested in defining a stronger notion of argument and encompass a specific form of conflict that often occurs in real-life argumentation: i.e., claims that additional preconditions are required for the conclusion of a rule to hold. In what follows, the existence of a finite set of rules $K$ from $\mathscr{L}_R$ is assumed.

**Definition 15** ***Argument*** *An argument for a rule $\alpha$ with respect to $K$ from $\mathscr{L}_R$ is a pair $\langle \Phi, \alpha \rangle$ s.t. the following conditions hold: (1) $\Phi \subseteq K$, (2) $\forall \beta$ s.t. $\Phi \vdash \beta$, $\beta \not\bowtie \Phi$ (3) $\Phi \vdash \alpha$, and (4) $\forall \Phi' \subset \Phi$, $\Phi' \nvdash \alpha$*

For an argument $\langle \Phi, \alpha \rangle$, we call $\Phi$ the *support*, $\alpha$ the *conclusion* of the argument. We also say that $\langle \Phi, \alpha \rangle$ is an argument for $\alpha$. Notice that the usual non-contradiction condition in deductive argumentation expressed by $\Phi \nvdash \bot$ is naturally extended and replaced by a non-contrariness requirement (Condition (2)).

The following example shows that whilst it is clear that taking consistency of supports into account is a valuable development when generating arguments, it is not just consistency, per se, that is important.

**Example 10** *Let consider the set of rules* $K = \{FetchBread \rightarrow EatBread, FetchKnife \wedge FetchBread \rightarrow EatBread, FetchKnife \wedge FetchBread \wedge SitOn \rightarrow EatBread, FetchKnife \rightarrow EatBread\}$. *Then, some arguments of K are:* $\langle \{FetchKnife \wedge FetchBread \rightarrow EatBread\}, FetchKnife \wedge FetchBread \rightarrow EatBread \rangle$ *and* $\langle \{FetchKnife \wedge FetchBread \wedge SitOn \rightarrow EatBread\}, FetchKnife \wedge FetchBread \wedge SitOn \rightarrow EatBread \rangle$. *However,* $\langle \{FetchBread \rightarrow EatBread, FetchKnife \rightarrow EatBread\}, FetchBread \wedge FetchKnife \rightarrow EatBread \rangle$ *is not an argument.*

A notion of *quasi-equivalent* arguments is now introduced as follows. It is intended to capture situations where two arguments can be said to make the same point on the same grounds. Not surprisingly, for an argument, its quasi-equivalent ones can be infinite.

**Definition 16** ***Quasi-equivalence*** *Two arguments* $\langle \Phi, \alpha \rangle$ *and* $\langle \Phi, \beta \rangle$ *are quasi-equivalent iff* $\alpha \equiv \beta$.

**Property 1** *Let* $\langle \Phi, \alpha \rangle$ *be an argument. There is an infinite set of arguments of the form* $\langle \Psi, \beta \rangle$ *s.t.* $\langle \Phi, \alpha \rangle$ *and* $\langle \Psi, \beta \rangle$ *are quasi-equivalent.*

Arguments are not necessarily independent. The definition of *more conservative arguments* captures a notion of subsumption between arguments, translating situations where an argument is in some sense contained within another one.

**Definition 17** ***Dependency*** *An argument* $\langle \Phi, \alpha \rangle$ *is more conservative than an argument* $\langle \Psi, \beta \rangle$ *iff* $\Phi \subseteq \Psi$, $\beta \vdash \alpha$.

Then, a more conservative argument can be seen as more general in the sense that it is less demanding on the support and less specific with respect to the conclusion.

Actually, the concept of being more conservative induces the concept of quasi-equivalent arguments, and conversely.

**Proposition 3** *Two arguments* $\langle \Phi, \alpha \rangle$ *and* $\langle \Psi, \beta \rangle$ *are quasi-equivalent iff each one is more conservative than the other.*

The notions of quasi-equivalence and of being more conservative will be used in the next subsection to avoid some redundancy when counter-arguments need to be generated.

### A.2.3 Conflicts among arguments

Based on the notion of contrariness $\bowtie$ introduced in Subsection A.2.1, let us now explain how arguments can be challenged. To do so, we will distinguish tow kinds of counter-argument: *rebuttal* and *defeater*.

**Definition 18** ***rebuttal*** *An argument $\langle \Psi, \beta \rangle$ is a rebuttal for an argument $\langle \Phi, \alpha \rangle$ iff $\beta \bowtie \alpha$.*

Clearly, the notion of rebuttal defined for arguments in $\mathscr{L}_R$ is thus asymmetric, since the relation $\bowtie$ is not symmetric.

**Example 11** *The argument $\langle \{FetchKnife \wedge FetchBread \wedge SitOn \rightarrow EatBread\}$, $FetchKnife \wedge FetchBread \wedge SitOn \rightarrow EatBread\rangle$ is a rebuttal for $\langle \{FetchKnife \wedge FetchBread \rightarrow EatBread\}, FetchKnife \wedge FetchBread \rightarrow EatBread\rangle$.*

An argument attacks another if its conclusion is the contrary of some rules in the other. This leads to the notion of *defeater* defined as follows:

**Definition 19** ***Defeater*** *An argument $\langle \Psi, \beta \rangle$ is a defeater for $\langle \Phi, \alpha \rangle$ iff $\beta \bowtie \Phi$.*

The next proposition shows a general relation between the two kinds of conflicting arguments, i.e. rebuttal and defeater.

**Proposition 4** *If an argument $\langle \Psi, \beta \rangle$ is a rebuttal for $\langle \Phi, \alpha \rangle$ then $\langle \Psi, \beta \rangle$ is a defeater for $\langle \Phi, \alpha \rangle$.*

As intended, defeaters can exist even if there is no inconsistency involved, as shown by Example 11: the support of the defeater is consistent with the support of the argument $A$ it attacks ; but, the attack relation is mainly based on the second form of contrariness among rules.

Since defeaters are arguments, then they can be ordered from more conservative to less conservative as shown by the following definition.

**Definition 20** ***Maximally conservative defeater*** *An argument $\langle \Psi, \beta \rangle$ is a maximally conservative defeater for $\langle \Phi, \alpha \rangle$ iff $\langle \Psi, \beta \rangle$ is a defeater for $\langle \Phi, \alpha \rangle$ such that no defeaters for $\langle \Phi, \alpha \rangle$ are strictly more conservative than $\langle \Psi, \beta \rangle$.*

We assume that there exists an enumeration which we call canonical enumeration of all maximally conservative defeaters for $\langle \Phi, \alpha \rangle$.

Now, in order to avoid some redundancy among counter-arguments we can ignore the unnecessary variants of maximally conservative defeaters. To do so, we introduce the notion of *rational defeaters* as follows.

**Definition 21** ***rational defeater*** *Let $\langle\Psi_1,\beta_1\rangle,\ldots,\langle\Psi_n,\beta_n\rangle,\ldots$ be the canonical enumeration of all maximally conservative defeaters for $\langle\Phi,\alpha\rangle$. Then, $\langle\Psi_i,\beta_i\rangle$ is a rational defeater for $\langle\Phi,\alpha\rangle$ iff $\forall\ j<i$, $\langle\Psi_i,\beta_i\rangle$ and $\langle\Psi_j,\beta_j\rangle$ are not quasi-equivalent.*

Henceforth, the rational defeaters will gather all the possible attacks of a given argument in the same ones: the ones which are representative of all defeaters for that arguments.

### A.2.4 Argument Graph

Given a rule-based knowledge base, the idea is to gather arguments and counter-arguments in the same structure in order to capture the way arguments can take place as a dispute develops. To do so, in this subsection, we can collect and organize all these arguments and rational defeaters in the same structure. If we construct an argument graph based on the arguments that can be constructed from a rule-based knowledge base, then we can be exhaustive in including all arguments and rational defeaters that can be constructed from that base. An argument graph is a directed graph where each node denotes an argument and each edge denotes an attack by one argument on another. Hence, when one argument is a counterargument to another argument, this is represented by an edge. More formally, given a rule-based knowledge base $K$, we can generate an argument graph $G=(\mathscr{A},\mathscr{R})$ where $\mathscr{A}$ is the set of arguments obtained from $K$ as mentioned previously in Definition 15 and $\mathscr{R}$ is a rational defeater relation.

**Definition 22** ***Argument graph*** *Let $K$ be a rule-based knowledge base. An argument graph for $K$ is a graph $G=(\mathscr{A},\mathscr{R})$ where $\mathscr{A}$ is the set of arguments generated from $K$ and $\mathscr{R}=\{(A,B)|A,B\in\mathscr{A}$ and $A$ is a rational $defeater\ for\ B\}$.*

Now, in order to evaluate the acceptability of arguments of a given argument graph we adopt Dung's argumentation system [32]. Given an argument graph for $K$ and two arguments $A$ and $B$ from $\mathscr{A}$, we say that $A$ *attacks* $B$ if there is $(A,B)\in\mathscr{R}$. Moreover, a set $S\subseteq\mathscr{A}$ *attacks* an argument $B\in\mathscr{A}$ if there is $A\in S$ such that $A$ attacks $B$. A set $S\subseteq\mathscr{A}$ is said to be *conflict-free* if there are no arguments $A,B\in S$ such that $A$ attacks $B$. Moreover, an argument $A\in\mathscr{A}$ is *defended* by a set $S\subseteq\mathscr{A}$ iff $\forall B\in\mathscr{A}$ such that $B$ attacks $A$, there is $c\in S$ such that $c$ attacks $b$.

Using the notions of conflict-freeness and defense, we can define a number of argumentation semantics, each embodying a particular rationality criterion, in order to identify reasonable sets of arguments, called *extensions*.

Given a rule-based knowledge base $K$ and an argument graph $G=(\mathscr{A},\mathscr{R})$. A set $\mathscr{E}\subseteq\mathscr{A}$ of arguments is said to be:

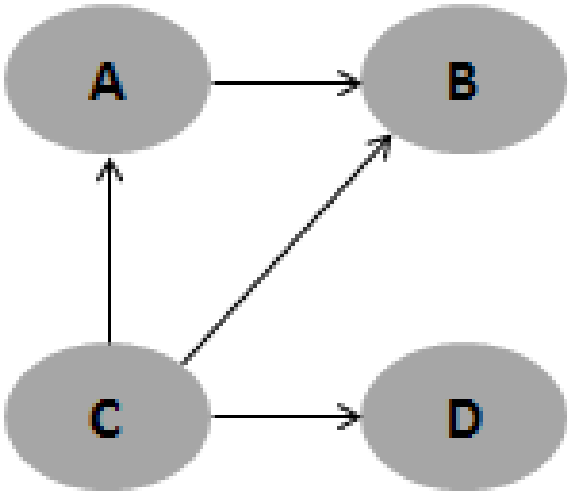



Figure A.1 An argument graph for $K$.

– *admissible* iff $\mathscr{E}$ is conflict-free and all its arguments are defended by $\mathscr{E}$,
– a *complete* extension iff $\mathscr{E}$ is admissible and $\mathscr{E}$ contains all and only the arguments it defends,
– a *grounded* extension iff $\mathscr{E}$ is a minimal (w.r.t. set inclusion) complete set of arguments,
– a *preferred* extension iff $\mathscr{E}$ is a maximal (w.r.t. set inclusion) admissible set of arguments.

Basically, each of these semantics corresponds to some properties which certify whether a set of arguments can be profitably used to support a point of view in a dispute. In the sequel, we consider preferred semantics, which represents the main contribution in Dung's theory, as it allows multiple extensions (differently from grounded semantics), the existence of extensions is always guaranteed, and no extension is a proper subset of another extension (differently from complete semantics).

**Example 12** *Let the four arguments $A,B,C$ and $D$ constructed from the rule-based knowledge base $K = \{FetchKnife \wedge FetchBread \rightarrow EatBread, FetchBread \rightarrow EatBread, FetchBread \wedge SitOn \rightarrow EatBread, FetchKnife \wedge FetchBread \wedge SitOn \rightarrow EatBread\}$. The argument graph associated to K is depicted in Figure A.1 where $A = \langle\{FetchBread \wedge SitOn \rightarrow EatBread\}, FetchBread \wedge SitOn \rightarrow EatBread\rangle$, $B = \langle\{FetchBread \rightarrow EatBread\}, FetchBread \rightarrow EatBread\rangle$, $C = \langle\{FetchKnife \wedge FetchBread \wedge SitOn \rightarrow EatBread\}, FetchKnife \wedge FetchBread \wedge SitOn \rightarrow EatBread\rangle$, and $D = \langle\{FetchKnife \rightarrow EatBread\}, FetchKnife \rightarrow EatBread\rangle$. Let us determine the extensions of the argument graph of K under the preferred semantics. Clearly, $\mathscr{E} = \{C\}$ is the preferred extension of the argument graph.*

## A.3 Empirical Evaluation

AGACY Monitoring for activity recognition and Caredas for early anomaly detection are using logic rules (see Chapters 3 and 4). Hence, in order to demonstrate the effectiveness of the proposal for conflicts resolution, we applied it to resolve added conflicts into AGACY monitoring and Caredas.

### A.3.1 Evaluation of conflict resolution using AGACY Monitoring

In order to evaluate the proposed method for conflict resolution in rule-based systems, we tested our method into our previous work AGACY Monitoring system [45] for activity recognition in smart environments. The system returns the user activity for a defined time window. To achieve this aim it has two layers: (1) A semantic layer and (2) A data driven layer. The former is in charge of the inference of the events and actions that have been done by the user. This inference process requires the definition of logical rules from human experts. The second layer applies the Demspter Shafer Theory (DS) [105] or the Support Vector Machine (SVM) [64] for the activity recognition based on the inferred actions and events. The system was tested in [45] with Opportunity dataset [2]. The dataset holds (from AGACY Monitoring perspective) sensors data, events, actions, and activities that have been done by three persons with three different routines. According to the actions within the dataset, a set of 25 rules has been defined. Since the number of rules is not high and the rules have been defined by one expert, they do not contain any conflict. Consequently, for the same dataset, we have added in the system a number *nbCR* of conflicting logical rules. Then, we have applied our argumentation-based method in order to resolve conflicts among rules. Firstly, we measured the precision of the system with 3 conflicting logical rules for different time windows with DS and SVM. The result of this experiment is depicted in Figure A.2. Then, with a time window equal to 180s, we measured the precision of the system by varying the number of added conflicting rules as shown in Figure A.3. Notice that we choose the value 180s for time window since it is the most suitable time window value to provide the best precision [45]. Finally, we applied the proposed method for conflict resolution and we measured the precision of the system for the different time window values. The result is shown in Figure A.4.

On the one hand, we can observe in Figure A.2 that with 3 conflicting rules the system has an average precision level (66% for DS and 44% for SVM) for TW$\in [60s..180s]$. Then, the precision value decreases with the increasing value of TW to be equal to 11% with TW=240s. On the other hand, by varying the number of conflicting rules with TW=180s we

[2] http://webmind.dico.unimi.it/care/annotations.zip

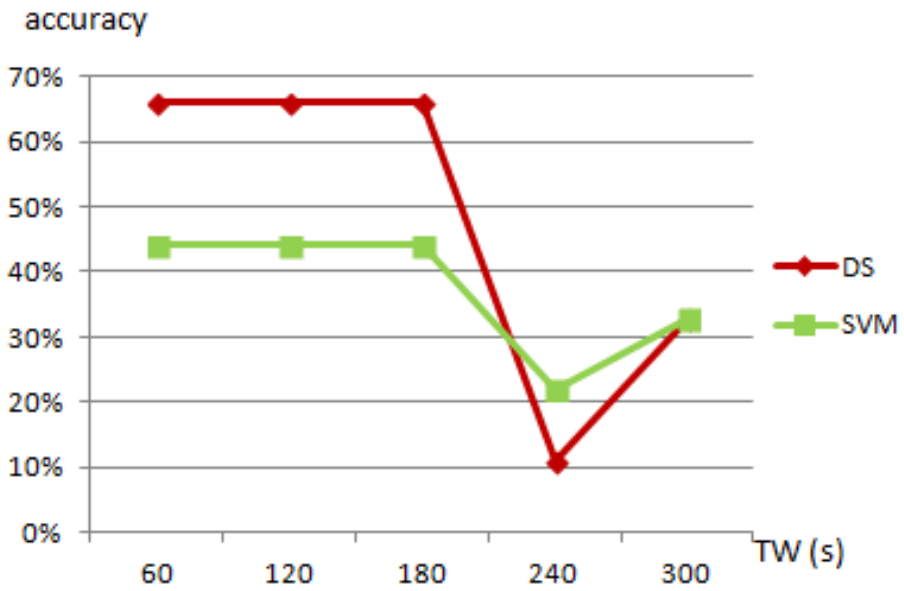



Figure A.2 Average recognition precision of AGACY Monitoring for all subjects over the three routines for three conflicting rules with different values of TW

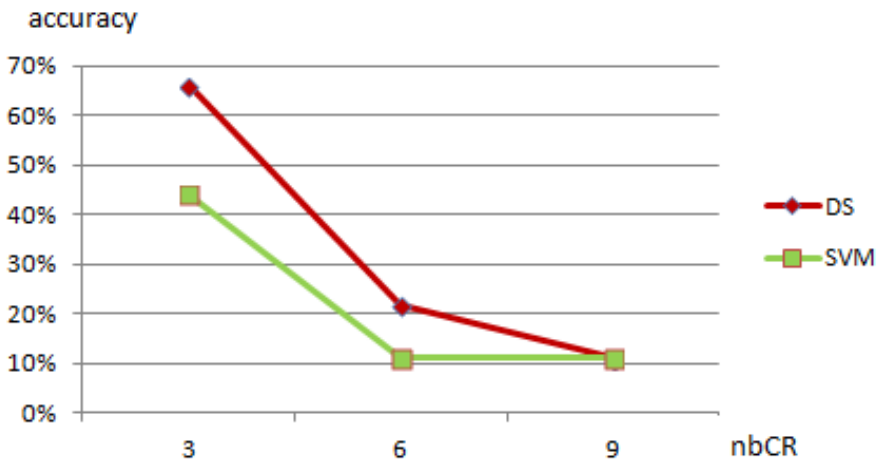



Figure A.3 Average recognition precision of AGACY Monitoring for all subjects over the three routines for TW=180s with different number of conflicting rules (nbCR)

clearly see in Figure A.3 that the system has a poor precision level with 6 and 9 conflicting rules. Consequently, the presence of conflicting rules in such system has a great effect on its performance that decreases with the increasing number of conflicting rules. The resolution of the added 9 conflicting rules through our proposed argumentation based method (Section 3) has obviously increased the precision level of the system as depicted in Figure A.4. As we can see, the system with DS has a precision value of 91% with TW=180s instead of 10% of

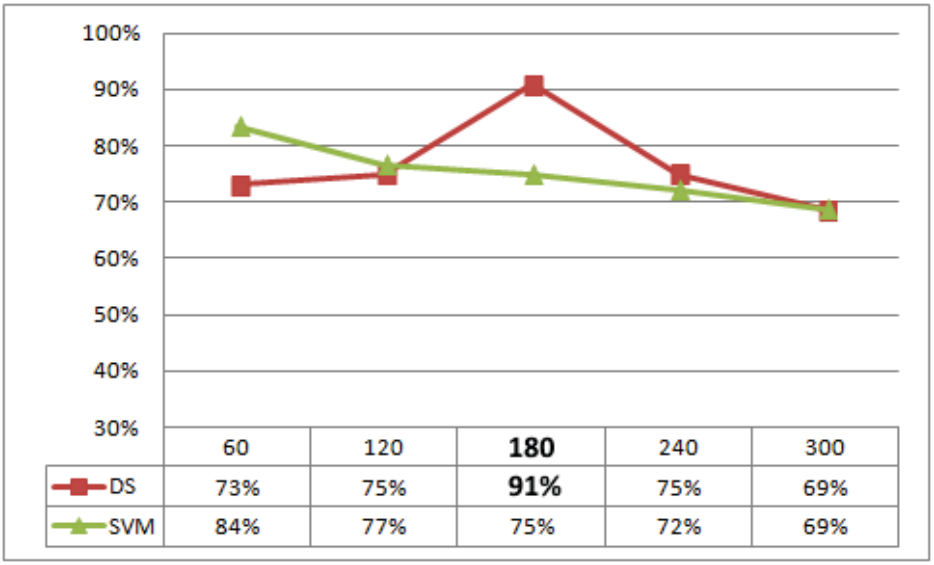



Figure A.4 Average recognition precision of AGACY Monitoring for all subjects over the three routines for different values of TW after conflict resolution

precision with 9 conflicting rules. The accuracy curves of AGACY Monitoring obtained after the conflict resolution are similar to that without conflict [45]. This result proves that our proposed method is perfectly able to resolve all conflicts in the system. Moreover, besides than decreasing the precision level, the conflicting rules have also changed the behavior of the system. As a matter of fact, the Figure A.2 shows that for TW <= 180s, the DS and SVM have the same accuracy values. Nevertheless, for TW > 180s the SVM works better. On the one hand, this behavior is different from the normal one (without conflict): The SVM has better precision than DS for TW <180s otherwise the DS works better than SVM.

### A.3.2 Evaluation of conflict detection and resolution method using Caredas

In order to prove the generality of the proposed method for conflicts detection and resolution we have applied it into Caredas, our proposal of early anomaly detection with the same Hadaptic dataset used previously to evaluate it (see Chapter 4). Hence, we injected two conflicting CARs and then we ran the method prototype with the same conditions with and without resolution of conflicts. The result of the evaluation is depicted in Figure A.5.

As we can see in Figure A.5, with two conflicting CARs, the recognition rate of the system is lower for all metrics (precision, recall, and correctness) for the different time windows. For instance, the method has only 40% recall for a 1 min time window instead of 60% with conflict resolution. Obviously, the recognition rate will be lower if the number of

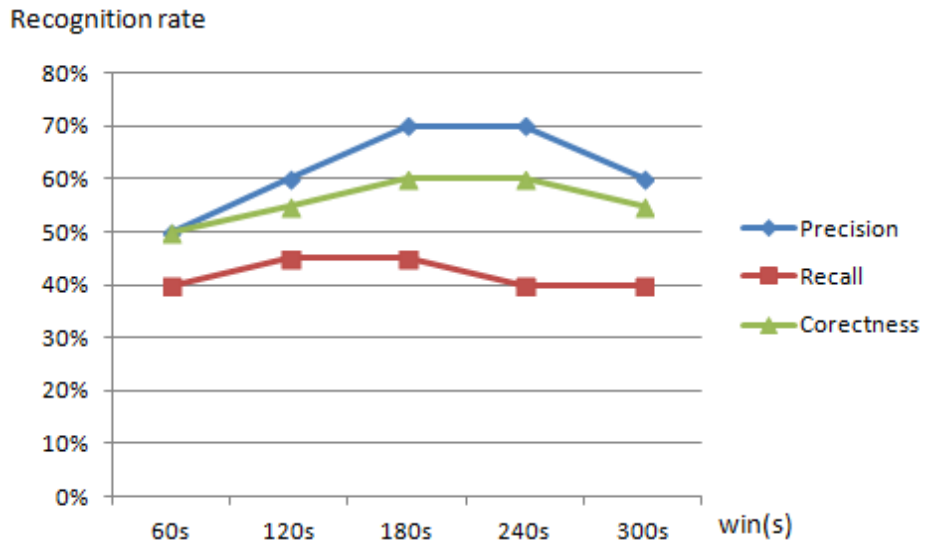



Figure A.5 Precision, recall, and correctness of the method with 2 conflicting CARs for different time windows *win*

conflicting rules is increased. Accordingly, it is very important to resolve conflict in order to increase the system performance.

Moreover, in order to prove the system's ability to resolve the added conflict, we evaluated the proposal with conflict resolution using the rules after adding conflict. The obtained curves of Figure A.6 for the three measures are the same as in Figure 4.6. This result confirms that the system is able to resolve conflicts without any decrease in its performance.

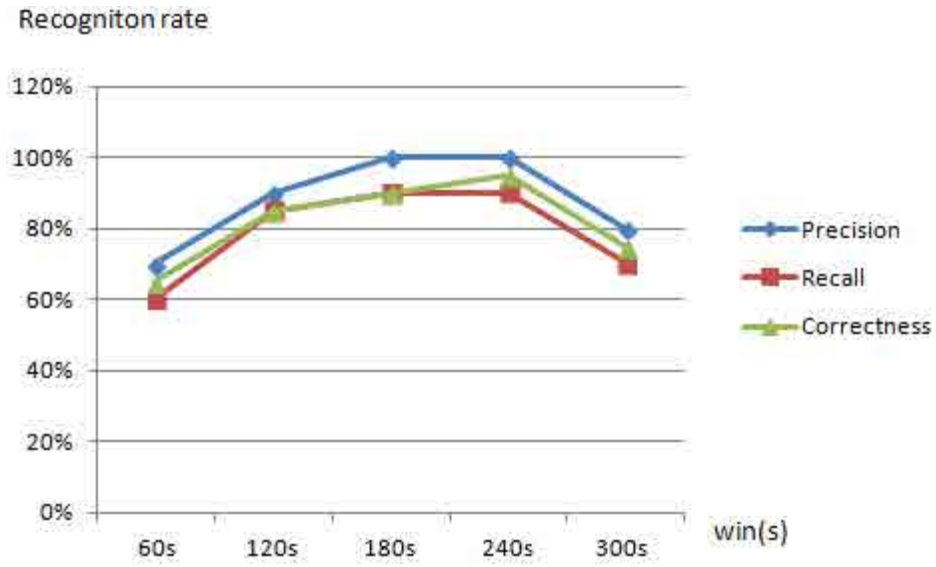



Figure A.6 Precision, recall, and correctness of the method using conflict resolution for different time window *win*

## A.4 Conclusion

In this appendix we proposed a new model for conflicts handling in rule-based methods. The proposal is based on the exploitation of argumentation theory to resolve two types of conflicts that often occur in smart environments. The main benefit of this proposal is its ability to be applied in any rule-based method. Therefore, we have exploited this model to resolve the eventual occurrence of conflicts in the rules-based method used for our hybrid models, AGACY Monitoring and Caredas, presented in the Chapters 3 and 4.