Raisonnement distribué dans un environnement ambiant

Amina Jarraya

université PARIS-SACLAY

NNT : 2019SACLL011

Thèse de doctorat

Raisonnement distribué dans un environnement ambiant

Thèse de doctorat de l'Université Paris-Saclay
préparée à Télécom SudParis et la Faculté des Sciences de Tunis

École doctorale n°580 : Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Informatique

Thèse présentée et soutenue à Tunis, le 16-07-2019, par

Amina Jarraya

Composition du Jury :

Samir Ben Ahmed Professeur, Faculté des Sciences de Tunis, Tunisie	Président
Amal El Fallah Seghrouchni Professeure, Sorbonne Université, France	Rapporteure
Abderaazak Jemai Professeur, INSAT, Tunisie	Rapporteur
Karine Zeitouni Professeure, UVSQ, France	Examinatrice
Amal Bouzeghoub Professeure, Télécom SudParis, France	Directrice de thèse
Amel Borgi Professeure, ISI Ariana, Tunisie	Co-Directrice de thèse
Khedija Arour Professeure, INSAT, Tunisie	Invitée

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Raisonnement distribué dans une environnement ambiant

Mots clés : intelligence ambiante, perception, analyse de données, raisonnement distribué, identification de situation/d'activité, données imparfaites.

Résumé : L'informatique pervasive et l'intelligence ambiante visent à créer un environnement intelligent avec des dispositifs électroniques et informatiques mis en réseau tels que les capteurs, qui s'intègrent parfaitement dans la vie quotidienne et offrent aux utilisateurs un accès transparent aux services partout et à tout moment.
Pour garantir ce fonctionnement, un système doit avoir une connaissance globale sur son environnement, et en particulier sur les personnes et les dispositifs, leurs intérêts et leurs capacités, ainsi que les tâches et les activités associées. Toutes ces informations relèvent de la notion de contexte. Cela passe par la collecte des données contextuelles de l'utilisateur pour déterminer sa situation/son activité courante ; on parle alors d'identification de situations/d'activités. Pour cela, le système doit être sensible aux variations de son environnement et de son contexte, afin de détecter les situations/les activités et de s'adapter ensuite dynamiquement. Reconnaître une situation/une activité nécessite alors la mise en place de tout un processus : perception des données contextuelles, analyse de ces données collectées et raisonnement sur celles-ci pour l'identification de situations/d'activités.
Nous nous intéressons plus particulièrement aux aspects liés à la modélisation distribuée de l'environnement ambiant et à ceux liés au raisonnement distribué en présence de données imparfaites pour l'identification de situations/d'activités. Ainsi, la première contribution de la thèse concerne la partie perception. Nous avons proposé un nouveau modèle de perception permettant la collecte des données brutes issues des capteurs déployés dans l'environnement et la génération des évènements. Ensuite, la deuxième contribution se focalise sur l'observation et l'analyse de ces évènements en les segmentant et extrayant les attributs les plus significatifs et pertinents. Enfin, les deux dernières contributions présentent deux propositions concernant le raisonnement distribué pour l'identification de situations/d'activités ; l'une représente la principale contribution et l'autre représente sa version améliorée palliant certaines limites. D'un point de vue technique, toutes ces propositions ont été développées, validées et évaluées avec plusieurs outils.

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

Title : Distribued Reasoning in Ambient Environment

Keywords : ambient intelligence,, perception, data analysis, distributed reasoning, situation/activity identification, imperfect data.

Abstract : Pervasive Computing and Ambient Intelligence aim to create a smart environment with networked electronic and computer devices such as sensors seamlessly integrating into everyday life and providing users with transparent access to services anywhere and anytime.
To ensure this, a system needs to have a global knowledge of its environment, and in particular about people and devices, their interests and their capabilities, and associated tasks and activities. All these information are related to the concept of context. This involves gathering the user contextual data to determine his/her current situation/activity ; we also talk about situation/activity identification. Thus, the system must be sensitive to environment and context changes, in order to detect situations/activities and then to adapt dynamically.
Recognizing a situation/an activity requires the definition of a whole process : perception of contextual data, analysis of these collected data and reasoning on them for the identification of situations/activities.
We are particularly interested in aspects related to the distributed modeling of the ambient environment and to those related to distributed reasoning in the presence of imperfect data for the identification of situations/activities. Thus, the first contribution of the thesis concerns the perception part. We have proposed a new perception model that allows the gathering of raw data from sensors deployed in the environment and the generation of events.
Next, the second contribution focuses on the observation and analysis of these events by segmenting them and extracting the most significant and relevant features. Finally, the last two contributions present two proposals concerning the distributed reasoning for the identification of situations/activities ; one represents the main contribution and the other represents its improved version overcoming certain limitations. From a technical point of view, all these proposals have been developed, validated and evaluated with several tools.

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

À mes parents Noemen et Safia

Je suis particulièrement reconnaissante pour votre amour et votre soutien continu.

Merci de toujours croire en moi.

Je sais que vous attendez ce jour spécial depuis ma naissance.

À mon mari Omar

Je te suis particulièrement redevable pour ta confiance

inconditionnelle, ta patience, ton soutien et ton amour sincère.

À mes soeurs Mouna et Eya

À ma nièce Aicha et mes neveux Adam, Youssef et Yassine

Je vous aime !

Pour toute ma famille...

Je dédie particulièrement ce travail **à mon bébé Nadine** qui m'a accompagné dans mon ventre tout au long de la rédaction de ce manuscrit.

Je vous adore tout simplement !

Remerciements

Le travail que j'ai réalisé au cours de ma thèse n'aurait certainement pas été possible sans l'aide, les encouragements et les conseils de nombreuses personnes qui y ont été impliquées de près ou de loin. C'est pourquoi je tiens à leur exprimer par ces quelques lignes toute ma reconnaissance, en particulier :

Je remercie vivement Mr. Samir BEN AHMED pour m'avoir fait l'honneur d'accepter de présider ce jury.

Je suis très reconnaissante envers Mr. Abderrazak JEMAI, et Mme. Amal El FALLAH pour avoir accepté de rapporter mes travaux et de faire partie de mon jury.

Je tiens à exprimer ma gratitude à Mme. Karine ZEITOUNI pour avoir accepté d'examiner mes travaux et faire partie du jury.

Je tiens à exprimer mes remerciements les plus sincères à Mme. Amel BOUZEGHOUB et Mme. Amel BORGI, mes directrices de thèse, pour m'avoir conseillée, encouragée et soutenue tout au long de la thèse avec patience et disponibilité, et pour la confiance qu'elles m'ont accordée. Je les remercie également de m'avoir donné l'opportunité d'assister à des congrès internationaux et d'exercer des stages de recherche en France.

J'adresse mes remerciements les plus chaleureux à ma co-encadrante de thèse Mme. Khedija AROUR pour tous les conseils techniques et scientifiques qu'elle m'a apportés et qui m'ont beaucoup aidé à accomplir mes travaux.

Merci également à mes collègues et amies Amina Houari, Fatma Dhaou, Thoraya Ben Chattah et Leila Haddad qui m'ont toujours soutenue. Je leur exprime ma profonde sympathie et leur souhaite beaucoup de succès.

Résumé

Raisonnement distribué dans un environnement ambiant

L'informatique pervasive et l'intelligence ambiante visent à créer un environnement intelligent avec des dispositifs électroniques et informatiques mis en réseau tels que les capteurs, qui s'intègrent parfaitement dans la vie quotidienne et offrent aux utilisateurs un accès transparent aux services partout et à tout moment.

Pour garantir ce fonctionnement, un système doit avoir une connaissance globale sur son environnement, et en particulier sur les personnes et les dispositifs, leurs intérêts et leurs capacités, ainsi que les tâches et les activités associées. Toutes ces informations relèvent de la notion de **contexte**. Cela passe par la collecte des données contextuelles de l'utilisateur pour déterminer sa situation/son activité courante ; on parle alors d'**identification de situations/d'activités**. Pour cela, le système doit être sensible aux variations de son environnement et de son contexte, afin de détecter les situations/les activités et de s'adapter ensuite dynamiquement. Reconnaître une situation/une activité nécessite alors la mise en place de tout un processus : perception des données contextuelles, analyse de ces données collectées et raisonnement sur celles-ci pour l'identification de situations/d'activités.

Nous nous intéressons plus particulièrement aux aspects liés à la modélisation distribuée de l'environnement ambiant et à ceux liés au raisonnement distribué en présence de données imparfaites pour l'identification de situations/d'activités. Ainsi, la première contribution de la thèse concerne la partie perception. Nous avons proposé un nouveau modèle de perception permettant la collecte des données brutes issues des capteurs déployés dans l'environnement et la génération des évènements. Ensuite, la deuxième contribution se focalise sur l'observation et l'analyse de ces évènements en les segmentant et extrayant les attributs les plus significatifs et pertinents. Enfin, les deux dernières contributions présentent deux propositions concernant le raisonnement distribué pour l'identification de situations/d'activités ; l'une représente la principale contribution et l'autre représente sa version améliorée palliant certaines limites. D'un point de vue technique, toutes ces propositions ont été développées, validées et évaluées avec plusieurs outils.

Mots clés : intelligence ambiante, environnement ambiant, contexte, perception, analyse de données, raisonnement distribué, identification de situation/d'activité, données imparfaites.

Abstract

Distributed reasoning in ambient environment

Pervasive Computing and Ambient Intelligence aim to create a smart environment with networked electronic and computer devices such as sensors seamlessly integrating into everyday life and providing users with transparent access to services anywhere and anytime.

To ensure this, a system needs to have a global knowledge of its environment, and in particular about people and devices, their interests and their capabilities, and associated tasks and activities. All these information are related to the concept of **context**. This involves gathering the user contextual data to determine his/her current situation/activity ; we also talk about **situation/activity identification**. Thus, the system must be sensitive to environment and context changes, in order to detect situations/activities and then to adapt dynamically. Recognizing a situation/an activity requires the definition of a whole process : perception of contextual data, analysis of these collected data and reasoning on them for the identification of situations/activities.

We are particularly interested in aspects related to the distributed modeling of the ambient environment and to those related to distributed reasoning in the presence of imperfect data for the identification of situations/activities. Thus, the first contribution of the thesis concerns the perception part. We have proposed a new perception model that allows the gathering of raw data from sensors deployed in the environment and the generation of events. Next, the second contribution focuses on the observation and analysis of these events by segmenting them and extracting the most significant and relevant features. Finally, the last two contributions present two proposals concerning the distributed reasoning for the identification of situations/activities ; one represents the main contribution and the other represents its improved version overcoming certain limitations. From a technical point of view, all these proposals have been developed, validated and evaluated with several tools.

Keywords : ambient intelligence, ambient environment, context, perception, data analysis, distributed reasoning, situation/activity identification, imperfect data.

Publications dans le cadre de la thèse

[Jarraya et al., 2016a] **Jarraya, A.**, Arour, K., Borgi, A., and Bouzeghoub, A. (2016a). *Distributed cooperative reasoning in ambient environment.* In 30th IEEE International Conference on Advanced Information Networking and Applications, AINA 2016, Crans-Montana, Switzerland, 23-25 March, 2016, pages 1085–1092. **[Regular paper, indexed IEEE, DBLP], CORE2018 rank B conference.**

[Jarraya et al., 2016b] **Jarraya, A.**, Ramoly, N., Bouzeghoub, A., Arour, K., Borgi, A., and Finance, B. (2016b). *FSCEP : A new model for context perception in smart homes.* In On the Move to Meaningful Internet Systems : OTM 2016 Conferences - Confederated International Conferences : CoopIS, C&TC, and ODBASE 2016, Rhodes, Greece, October 24-28, 2016, Proceedings, pages 465–484. **[Long paper, indexed SPRINGER, DBLP], CORE2018 rank A conference.**

[Jarraya et al., 2016c] **Jarraya, A.**, Ramoly, N., Bouzeghoub, A., Arour, K., Borgi, A., and Finance, B. (2016c). *A fuzzy semantic CEP model for situation identification in smart homes.* In ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016), pages 1678–1679. **[Short paper, indexed SPRINGER, DBLP], CORE2018 rank A conference.**

[Jarraya et al., 2017] **Jarraya, A.**, Arour, K., Bouzeghoub, A., and Borgi, A. (2017). *Feature selection based on choquet integral for human activity recognition.* In 2017 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2017, Naples, Italy, July 9-12, 2017, pages 1–6. **[Regular paper, indexed IEEE, DBLP], CORE2018 rank A conference.**

[Jarraya et al., 2018] **Jarraya, A.**, Bouzeghoub, A., Borgi, A. and Arour, K. (2018). *Distributed Collaborative Reasoning for HAR in Smart Homes.* In 2018 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2018, Stockholm, Sweden, July 11-13, 2018. **[Short paper, indexed IEEE, DBLP], CORE2018 rank A* conference.**

Table des matières

Table des figures

Liste des tableaux

Liste des Algorithmes

Introduction générale

Nous introduisons le domaine de recherche abordé dans cette thèse. Nous commençons par délimiter le cadre de notre travail au sein de l'informatique pervasive, puis nous évoquons les motivations qui sont à l'origine de notre travail. Nous donnons enfin un résumé des contributions proposées et présentons l'organisation de ce document.

Cadre de la thèse

L'informatique émerge vers des environnements ambiants, pervasifs ou encore ubiquitaires dans lesquels les dispositifs intelligents sont censés se connecter d'une manière transparente et collaborer les uns avec les autres pour aider l'être humain, anticiper ses besoins et fournir l'information adéquate à la bonne personne, et ce quelques soient le lieu et le moment.

L'enjeu de cette évolution est de rendre l'informatique invisible [Weiser, 1991] : *"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it"*. [Weiser, 1991] oppose l'informatique invisible à la réalité virtuelle consistant à recréer le monde dans une machine et non pas à intégrer des machines dans la réalité. Les avancées en termes de dispositifs mobiles et de communication sans fil ont rendu possible cette intelligence ambiante. L'espace ambiant devient capable de percevoir des changements à travers une multiplicité de dispositifs intelligents et mobiles dont l'accumulation fournit une puissance de calcul substantielle. Les interactions ne se font plus via un utilisateur humain mais directement entre les objets connectés. L'informatique ambiante a alors pour objectif de gérer ces objets et leurs interconnexions afin de rendre des services de manière transparente pour l'utilisateur. Ainsi, les systèmes d'intelligence ambiante visent à fournir la bonne information aux bons utilisateurs, au bon moment, au bon endroit et sur le bon dispositif. La Figure 1 donne une vision de l'évolution de l'informatique ubiquitaire depuis l'informatique mobile vers l'intelligence ambiante en partant de

la mobilité (la capacité pour l'utilisateur à interagir n'importe où), à l'intelligence ambiante (l'analyse du contexte et l'adaptation dynamique aux situations) puis à l'implication de cette dernière avec l'Internet des objets.

Pour atteindre ce dernier objectif, un système doit avoir une connaissance globale sur son environnement, et en particulier sur les personnes et les dispositifs, leurs intérêts et leurs capacités, ainsi que les tâches et les activités associées. Toutes ces informations relèvent de la notion de ***contexte***. Cela passe par la perception du contexte de l'utilisateur pour déterminer sa situation/son activité ; on parle alors également d'**identification de situations/d'activités**. Pour cela, le système doit être sensible aux variations de son environnement, aux variations de son contexte, afin de détecter les situations/activités et de s'adapter ensuite dynamiquement. Reconnaître une situation/activité nécessite alors la mise en place d'un processus de collecte puis d'analyse de ces informations contextuelles pour l'identification des situations/d'activités.

L'environnement ambiant est un environnement où les dispositifs tels que les capteurs sont répartis et hétérogènes. En outre, les données issues de ces capteurs peuvent être imparfaites et donc la situation/l'activité identifiée à partir de ces données imparfaites, peut être incorrecte. Ainsi, les décisions concernant l'adaptation dynamique suite à l'identification de cette situation/activité, seront inappropriées [Bikakis and Antoniou, 2010]. Dans le cadre de l'intelligence ambiante, ce travail de thèse s'attache à proposer une architecture globale distribuée pour l'identification de situations/d'activités en tenant compte de ces deux contraintes : d'une part l'hétérogénéité et la répartition des capteurs et d'autre part l'imperfection des données.

Motivations

L'intelligence ambiante est une discipline émergente qui apporte de l'intelligence à nos environnements quotidiens et rend ces environnements soumis aux utilisateurs pour répondre à leurs besoins. La recherche dans l'intelligence ambiante s'appuie sur les progrès techniques en matière de réseaux de communication sans fil, d'équipements mobiles personnels, de capteurs et de logiciels embarqués. Ces derniers rendent aujourd'hui possibles des services aux usagers dépendants du contexte. Les opportunités d'applications sont nombreuses : par exemple, la téléphonie mobile où le téléphone sait comment réagir aux appels selon la situation de l'utilisateur, ou encore les parkings indiquant aux conducteurs où trouver une place

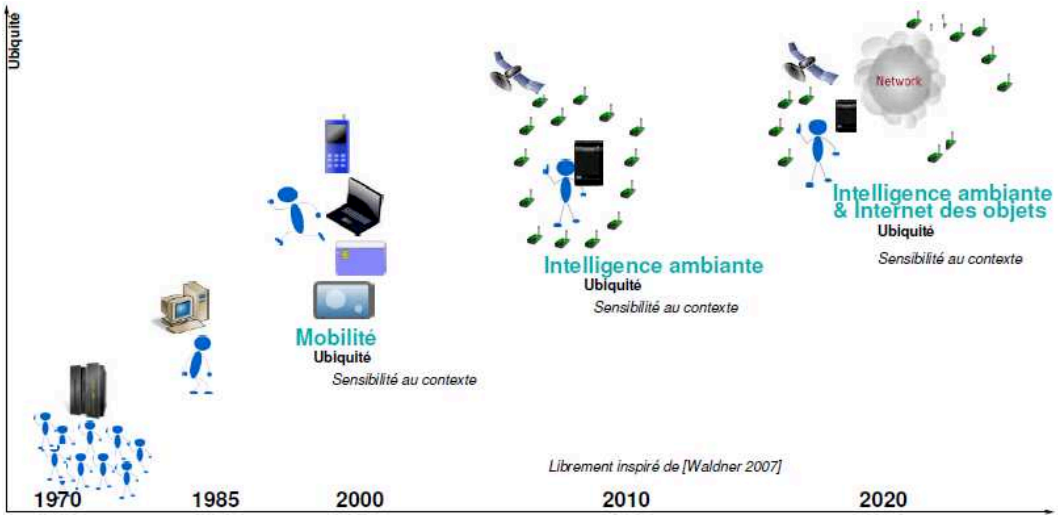


FIGURE 1. Évolution de l'informatique ubiquitaire [ABID, 2012]

libre, les soins de santé, l'assistance à domicile pour les personnes âgées et handicapées, la détection d'urgence et les loisirs [Ye et al., 2012]. La Figure 2 illustre quelques exemples d'applications ubiquitaires.

FIGURE 2. Les applications en environnement ubiquitaire (http ://www.uidcenter.org)

Prenons l'exemple d'une personne âgée vivant seule dans la maison : "Une nuit, sentant le besoin d'utiliser les toilettes, elle s'est réveillée et s'est dirigée vers les toilettes. Dans l'obscurité, elle a trébuché et est tombée par terre. Incapable de se lever, elle a passé la nuit allongée sur le sol froid". La situation "allongée sur le sol toute la nuit" est une situation anor-

male et nécessite une action immédiate telle qu'effectuer un appel d'urgence. Dans de tels scénarios, les environnements intelligents et dans ce cas particulier, les maisons intelligentes sont une solution prometteuse pour les personnes âgées qui sont dépendantes et ont besoin d'aide dans leurs activités de la vie quotidienne. Il s'agit d'un environnement résidentiel enrichi d'une diversité de capteurs multi-modalités, d'actionneurs et d'appareils ainsi que des services et des systèmes basés sur les Technologies de l'Information et de la Communication (TIC) [Ding et al., 2011]. Avant de proposer les services de recommandation et les alertes dans de tels environnements, il faut tout d'abord identifier/reconnaître la situation/l'activité courante de la personne en question. Ainsi, reconnaître les situations/les activités humaines à partir des capteurs intégrés dans un environnement intelligents ou portés sur des corps humains est un sujet de recherche important et stimulant en informatique pervasive. L'identification de situations/d'activités s'effectue après une perception et une analyse des données issues des capteurs présents dans l'environnement en question. Cette identification est face à deux grands défis :

1. Raisonner et identifier les situations/activités dans un contexte imparfait.
2. Raisonner et identifier les situations/activités dans un contexte distribué.

Dans cette thèse, nous nous sommes intéressés à l'identification de situations/d'activités séquentielles mono-utilisateur à partir de séquences de capteurs et avons considéré les deux défis mentionnés ci-dessus comme étant des objectifs à atteindre.

Contributions

Afin d'aboutir à l'identification de situations/activités, une phase de perception des données de capteurs puis une observation et une analyse de ces données sont primordiales.

L'architecture générale que nous avons proposée pour l'identification de situations, est inspirée des travaux de [Daniello et al., 2015] (voir Figure 3). Elle vise à faciliter le processus de prise de décisions dans le domaine des applications hétérogènes.

En effet, il s'agit d'une architecture hiérarchique distribuée ou chaque niveau comporte un ensemble de nœuds ayant des fonctionnalités similaires. Elle est composée de trois principaux modules :

— **Perception** : plusieurs nœuds sont déployés pour collecter les données brutes issues des différents capteurs, appelés *les collecteurs*.
— **Observation** : ce niveau contient un ensemble de nœuds observateurs qui analysent les données brutes collectées issues du niveau *Perception* et les transforment en des

données de haut niveau d'abstraction. Ce niveau comprend trois sous étapes qui sont la segmentation, l'extraction d'attributs et la sélection des attributs.

— **Identification** : Ce niveau comporte un ensemble de nœuds *identificateurs* capables d'utiliser les données de haut niveau générées depuis le niveau *Observation* pour l'identification de situations/activités.

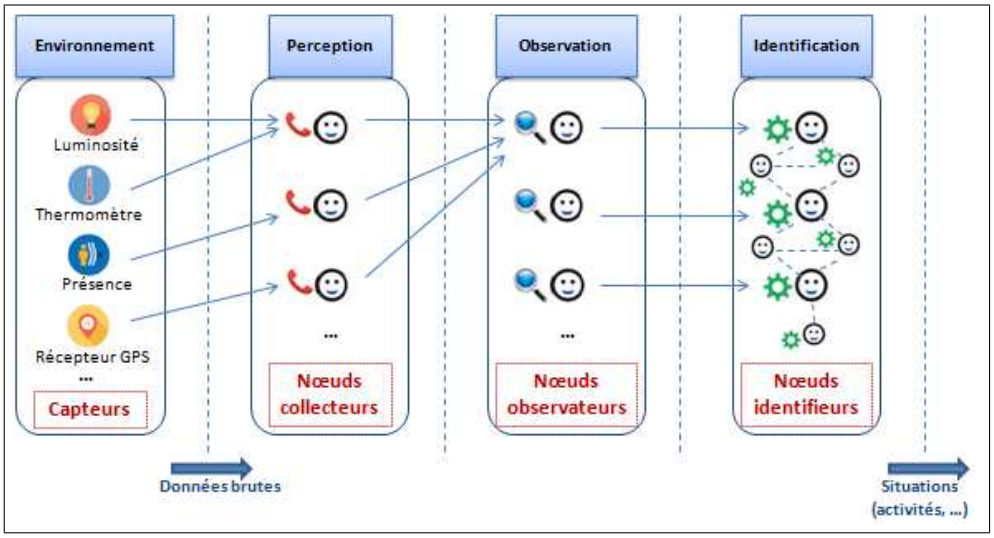


FIGURE 3. Architecture distribuée pour l'identification de situations/d'activités

Dans cette thèse, nous avons contribué aux différents niveaux de l'architecture (voir Figure 4) et en particulier :

— Niveau *Perception* : un nouveau modèle de perception **FSCEP (Fuzzy Semantic Complex Event Processing)** pour le traitement des évènements complexes sémantiques flous.

— Niveau *Observation (étape sélection des attributs)* : une nouvelle méthode de sélection des meilleurs attributs basée sur l'intégrale de Choquet **FSCI (Feature Selection based on Choquet Integral)** et une étude théorique sur sa version floue **Fuzzy-FSCI**.

— Niveau *Identification* : une principale approche de raisonnement distribué est proposée, elle se base sur le modèle multi-agents pour la reconnaissance d'activité humaine et est nommée l'approche **DCR (Distributed Cooperative Reasoning)**. Deux autres versions de l'approche *DCR* sont proposées : 1- une étude théorique sur l'extension de l'approche *DCR* à des données d'entrée floues, l'approche est nommée **Fuzzy-DCR** ; 2- l'intégration et la mise en place d'un apprentissage en ligne dans *DCR*, l'approche est nommée **DCR-OL**.

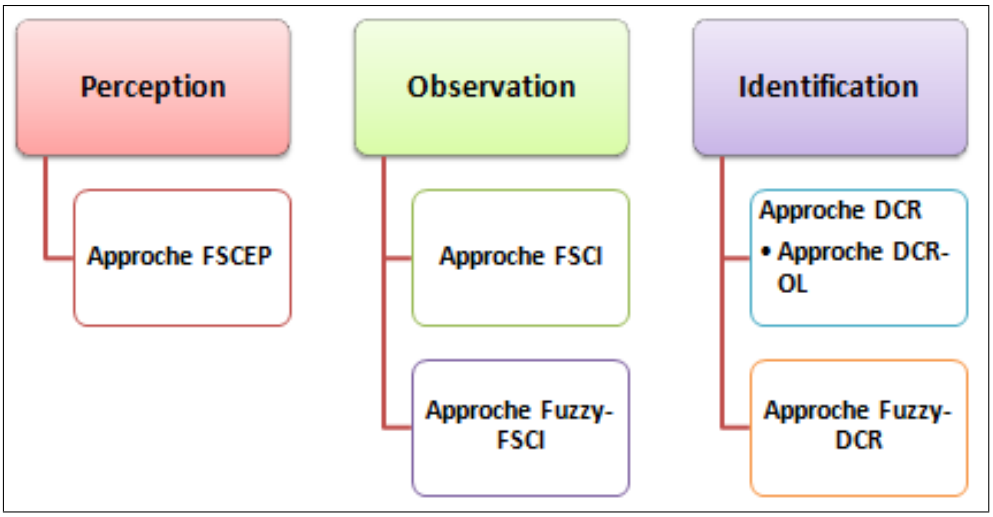


FIGURE 4. Les différentes contributions pour chaque niveau de l'architecture

Organisation du manuscrit

Cette thèse est composée de cinq chapitres, le premier chapitre aborde le contexte dans lequel se situe nos travaux et les quatre chapitres qui suivent, présentent nos différentes contributions comprenant les expérimentations effectuées.

Dans le premier chapitre, nous présentons l'état de l'art de l'identification de situations dans un environnement ambiant. Nous commençons par définir les concepts manipulés et la notion de situation/activité puis nous détaillons les différentes étapes du processus d'identification ainsi que les méthodes et les techniques de raisonnement associées. Enfin, nous discutons des limites du raisonnement dans les systèmes d'intelligence ambiante et les défis relevés dans cette thèse.

Dans le deuxième chapitre, nous présentons notre première contribution *FSCEP* concernant le niveau *Perception* de l'architecture présentée ci-dessus (voir Figure 3). Cette contribution consiste en un nouveau modèle de perception basé sur un système orienté évènement permettant de détecter les changements de l'environnement, de les transformer en des évènements simples puis de générer des évènements complexes sémantiques flous.

Dans le troisième chapitre, nous présentons notre deuxième contribution concernant le

niveau *Observation* de l'architecture et en particulier l'étape sélection des attributs. Cette contribution introduit une nouvelle méthode de sélection des attributs les plus pertinents basée sur l'intégrale de Choquet pour l'identification de situations/activités. Deux versions de cette méthodes sont présentées : la première qui prend en entrée des données non floues (l'approche *FSCI*) ; la deuxième qui prend en entrée des données floues (l'approche *Fuzzy-FSCI*).

Dans le quatrième chapitre, nous passons au niveau de l'*Identification*. Une étude théorique est proposée sur la principale contribution *DCR* concernant le raisonnement distribué pour l'identification d'une situation. Deux autres versions de cette contribution sont présentées qui sont *DCR-OL* et *Fuzzy-DCR*.

Dans le cinquième chapitre, nous présentons une étude expérimentale des approches du niveau *Identification* et en particulier les approches *DCR* et *DCR-OL*.

Enfin, nous concluons ce manuscrit en synthétisant les principales contributions et proposons différentes pistes de recherche.

Chapitre 1

État de l'art : les systèmes ambiants

Sommaire

1.1 Introduction

L'informatique pervasive et l'intelligence ambiante visent à créer un environnement intelligent avec des dispositifs électroniques et informatiques mis en réseau s'intégrant parfaitement dans la vie quotidienne, répondant aux informations fournies par les capteurs dans l'environnement et offrant aux utilisateurs un accès transparent aux services partout et à tout moment. Cet environnement implique un certain nombre d'entités de détection/calcul qui interagissent à la fois avec les utilisateurs et avec l'environnement dans lequel ils opèrent. Basé sur ces entités, un système d'intelligence ambiante peut fournir des services personnalisés aux utilisateurs de manière contextuelle lorsqu'ils interagissent et échangent des informations avec l'environnement. De nos jours, l'informatique pervasive se développe en tant que sujet de recherche universitaire lié à une réalité commerciale [Henricksen and Indulska, 2006]. De nombreuses applications potentielles se présentent telles que les espaces de travail intelligents, les maisons intelligentes et même dans les domaines des soins de santé, les jeux, les systèmes de loisirs et les transports en commun [Cook and Das, 2007]. Ces applications ont un potentiel significatif, impactant les vies humaines.

Les capteurs de l'informatique pervasive peuvent être déployés n'importe où et sur n'importe quel objet ou rattachés à des corps humains. Ils génèrent des données, y compris la localisation de l'utilisateur, le mouvement, les informations biomédicales, la température de l'environnement, l'humidité ou le niveau de bruit ambiant. Les applications qui fournissent des services personnalisés aux utilisateurs, sont basées sur ces données qui sont issues de ces capteurs. Les données de capteurs présentent une grande complexité (modalités différentes, volumes énormes rapides et relations d'interdépendance entre les sources) et un dynamisme (mise à jour en temps réel). Un système d'intelligence ambiante ne devrait pas se contenter de se préoccuper des données individuelles des capteurs (dans quelle pièce se trouve l'utilisateur, quelle est sa fréquence cardiaque ou sa tension artérielle). Ces informations devraient plutôt être interprétées et transformées en des informations de haut niveau telles que "l'utilisateur est en train de regarder la télévision" ou bien "l'utilisateur souffre d'une crise cardiaque". Ce concept d'information de niveau supérieur est appelé **une situation**, qui est un état abstrait et utile pour les applications [Costa et al., 2006]. La possibilité pour les applications d'avoir des situations, réside dans leur capacité à fournir une représentation simple et compréhensible des données de capteurs et à offrir ainsi une utilisation efficace.

Cependant, dans les systèmes d'intelligence ambiante, il peut y avoir des dizaines ou des centaines de situations que les applications doivent reconnaître et auxquelles elles doivent

répondre. Pour réaliser ceci, un nombre important de capteurs est nécessaire pour **l'identification de la situation**. Un système d'intelligence ambiante a une tâche importante pour définir et gérer ces situations qui va du choix des situations à détecter et à reconnaître. Le système d'intelligence ambiante doit savoir, par exemple, quelles situations sont en train de se produire (par exemple, si un utilisateur *regarde la télévision* ou *prend une douche*).

Ce chapitre dresse un panorama et une compréhension globale des situations dans l'informatique pervasive. Après une introduction des concepts manipulés et une définition des termes liés au domaine d'étude, nous présentons la notion de l'identification de situations dans un environnement ambiant. Nous continuons ensuite à détailler le processus d'identification de situations ainsi que les méthodes de raisonnement utilisées pour cette identification. Nous terminons ce chapitre par une analyse des limites de raisonnement pour l'identification dans les systèmes d'intelligence ambiante.

1.2 Définitions

Nous présentons un ensemble de définitions permettant de délimiter le domaine de recherche traité dans cette thèse. Nous commençons par l'informatique ambiante et continuons par la notion de l'intelligence ambiante qui est au coeur de ces nouvelles formes d'informatique pour ensuite introduire la notion de contexte et le principe de sensibilité au contexte. Nous présentons enfin le raisonnement distribué, thème que nous développons plus en détail dans la suite de ce chapitre et qui est central pour cette thèse.

— **Informatique pervasive** aussi appelée *informatique ubiquitaire* ou *informatique ambiante* [Coutaz and Crowley, 2008], est un nouveau concept de l'informatique qui vise à doter les objets physiques présents dans notre environnement d'une intelligence et de capacités de communication et d'interaction entre eux et avec les personnes et l'environnement dans lequel ils évoluent. Ainsi, l'intelligence, au lieu d'être centralisée, devient répartie dans les objets présents dans l'environnement. De plus, les objets de l'informatique ambiante sont capables de reconnaître et d'intégrer automatiquement tout nouvel objet.

— **Intelligence ambiante** : elle correspond à l'amélioration, voire l'augmentation du monde réel pour offrir un tout adapté en toute circonstance à l'Homme [Coutaz and Crowley, 2008].

Autrement dit, il s'agit de créer des services et des dispositifs intelligents capables de répondre à des besoins individuels, collectifs et sociaux.

— **Contexte** : Il existe plusieurs définitions de la notion de contexte dans le domaine de l'informatique ubiquitaire. La définition du contexte dans les premiers travaux [Schilit and Theimer, 1994] où le mot contexte apparaît, se réfère à un ensemble décrit d'éléments : la localisation, les identités des personnes autour de l'utilisateur, des objets dans l'environnement physique et les différents changements ayant lieu dans cet environnement. [Dey, 2001] propose une définition généralisée du contexte la plus couramment utilisée : *"Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves."*. En intelligence artificielle, Brézillon [Brézillon, 2002] définit le contexte comme étant *"ce qui n'intervient pas directement dans la résolution d'un problème mais contraint sa résolution"*.

— **Sensibilité au contexte** : Un système est sensible au contexte s'il peut utiliser et interpréter les informations issues du contexte et adapter sa réponse en fonction du contexte d'utilisation [Schilit and Theimer, 1994].

— **Raisonnement distribué** [Zimmermann, 2008] : il peut être étudié à deux niveaux, l'un au niveau de la procédure de déduction, l'autre au niveau de la sémantique. Une procédure distribuée consiste à utiliser séparément différents systèmes de raisonnement, mis en commun par un protocole d'échange de messages. Définir une procédure distribuée pour le raisonnement requiert que la sémantique soit connue. Quant à la sémantique distribuée, elle consiste à affecter une interprétation distincte à chaque nœud d'un système, avec des règles de compatibilité entre interprétations.

1.3 Identification de situations dans un environnement ambiant

Dans cette section, nous définissons les deux termes suivants qui apparaîtront fréquemment plus tard ainsi que les thèmes de recherche évoqués sur l'identification de situations.

— *Les données de capteurs* sont des données brutes récupérées à partir des capteurs

physiques. Ces données sont agrégées pour former le *contexte*-l'environnement dans lequel le système évolue.

— *Une situation* est une abstraction des informations dérivées du contexte courant, se produisant dans le monde réel.

1.3.1 Capteurs et données de capteurs

Le capteur est la source dont nous avons besoin pour collecter des données brutes dans les systèmes d'identification de situations [Su et al., 2014]. Différents capteurs produisent différents types de données. Par exemple, les données collectées sur la plupart des capteurs portables tels que l'accéléromètre ou le gyroscope sont des séries temporelles, les capteurs ambiants tels que les capteurs de mouvement produisent des données numériques ou catégoriques et les caméras enregistrent des données de type image/vidéo [Rashidi and Mihailidis, 2013]. Les méthodes de collecte des données de capteurs varient selon la nature des données brutes [Kim et al., 2010].

La diversité des capteurs conduit à une grande complexité dans l'interprétation de leurs résultats, y compris d'énormes volumes de données, différentes modalités, l'interdépendance et la mise à jour en temps réel. Dans le monde réel, ces capteurs produisent généralement des données imparfaites. Les données de capteurs bruitées peuvent entraîner une mauvaise compréhension de l'état de l'utilisateur ou de l'environnement, ce qui entraînera un comportement incorrect de l'application. Ces capteurs ont également leurs propres limites techniques, sont confrontés à la panne, ou peuvent être déconnectés du réseau de capteurs ou être vulnérables aux interférences de l'environnement. Cela conduit à la question de l'incertitude des données de capteurs, qui peuvent également être obsolètes, inconsistantes, incomplètes, imprécises et contradictoires entre elles [Henricksen and Indulska, 2004]. Un défi majeur est de savoir comment les utiliser pour reconnaître et identifier les situations qui pourraient nous donner une meilleure compréhension des interactions humaines avec l'environnement [Atallah and Yang, 2009].

Les capteurs peuvent être classés en deux catégories : *les capteurs de la vision* et *les capteurs de l'environnement* [Chen et al., 2012]. La première est basée sur l'utilisation de collecte visuelle telle que des caméras vidéo pour surveiller le comportement d'un utilisateur et les changements environnementaux. Les données de capteurs générées depuis les capteurs de l'environnement sont principalement des séries temporelles de changements d'état tels que les capteurs binaires : les capteurs de mouvements, les capteurs de proximité, les capteurs de présence, etc. Les capteurs de l'environnement comprennent aussi les capteurs portables

qui sont des capteurs mobiles de petite taille conçus pour être portés sur le corps humain dans les activités quotidiennes. La plupart des capteurs mobiles sont rattachés à des téléphones mobiles intelligents [Su et al., 2014] tels que les accéléromètres, GPS, capteurs de lumière, capteurs de température, gyroscope, baromètre, etc.

Les contributions présentées dans cette thèse se situent dans la catégorie d'identification de situations basée sur les capteurs de l'environnement. Le principal avantage des approches basées sur les capteurs de l'environnement par rapport aux approches basées sur la vision, est lié à la confidentialité et à l'éthique [Yilmaz et al., 2006] car les caméras sont généralement considérées comme des dispositifs d'enregistrement.

1.3.2 Situations et activités

Une situation est souvent définie comme étant un aperçu de l'état de l'environnement physique à un moment donné. [Ye et al., 2012] définit une situation comme suit : "*A situation is defined as an external semantic interpretation of sensor data*". L'interprétation (*interpretation*) signifie que les situations attribuent des significations aux données de capteurs. L'externe (*external*) signifie que l'interprétation s'effectue au niveau des applications, plutôt qu'au niveau des capteurs. La sémantique (*semantic*) signifie que l'interprétation attribue un sens aux données de capteurs.

Une situation peut être définie en collectant des contextes pertinents, en cherchant des corrélations significatives entre eux, et en leur assignant un nom descriptif. Le nom descriptif peut être appelé *une définition descriptive* d'une situation, il reflète la réflexion de l'être humain en définissant l'état des choses dans la réalité. Une expression logique de prédicats de contexte corrélés s'appelle une spécification logique d'une situation. Ainsi, une situation relie les données et les applications du capteur. Les données du capteur sont extraites d'une certaine situation en évaluant sa spécification, et cette situation déclenchera des applications qui correspondent à son nom descriptif.

[Endsley, 1995] définit la reconnaissance de situations comme étant :"*the perception of elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future*". C'est un processus cognitif qui consiste en trois modules opérationnels. Premièrement, il implique la détection et la reconnaissance des différents éléments de l'environnement, y compris en particulier les caractéristiques et les comportements de haut niveau. Deuxièmement, il a besoin de l'interprétation et de la compréhension de la signification associée aux éléments perçus dans l'environnement. Et troisièmement, il faut pouvoir anticiper les actions des éléments et pré-

dire les futurs états de l'environnement. Pour les entités, que ce soit des êtres humains ou des robots ou des systèmes logiciels, opérant dans des environnements complexes, dynamiques et incertains, la reconnaissance de la situation est le facteur déterminant pour prendre de bonnes décisions au bon moment et au bon endroit.

Ce qui différentie une activité d'une situation, c'est que l'activité fait partie d'une situation. Une situation est un état global de l'environnement (e.g, une salle est occupée, un téléphone sonne, etc.) alors qu'une activité est une interprétation du comportement humain (e.g, préparer le déjeuner, être en train de travailler, etc.).

Une situation comprend des aspects temporels riches et d'autres aspects structurels, y compris *l'heure du jour* - une situation ne peut survenir qu'à un moment donné de la journée ; *la durée* - elle ne peut durer qu'un certain temps ; *la fréquence* - elle peut se produire plusieurs fois par semaine par exemple, et *la séquence* - différentes situations peuvent se produire dans une certaine séquence. La situation peut être un état simple et abstrait d'une certaine entité (e.g, la salle de réunion est occupée), ou une action humaine se déroulant dans un environnement (e.g, travailler ou cuisiner). Une situation peut également être composée par des situations plus fines ; e.g, une situation de "séminaire" inclut les situations plus fines comme "présentation", "questionnement" et "discussion de groupe".

Dans cette thèse, nous avons considéré qu'une situation est une activité humaine et nous nous sommes intéressés aux cas simples c-à-d la situation/l'activité est un état simple non composé. La reconnaissance de situation/d'activité concurrente et/ou bien multi-utilisateurs ne fait pas partie des objectifs de la thèse.

1.3.3 Thèmes de recherche sur l'identification de situations

Dans l'informatique pervasive, les principaux sujets de recherche sur l'identification de situations/d'activités impliquent les problèmes suivants :

- *Représentation* : comment définir les primitives logiques utilisées pour construire la spécification logique d'une situation ?
- *Spécification* : comment former une spécification logique d'une situation qui peut être donnée par des experts ou apprise à partir des données d'apprentissage ?
- *Raisonnement* : comment déduire des situations à partir d'une grande quantité de données de capteurs imparfaites ? Comment raisonner sur les relations de situations et comment maintenir la cohérence et l'intégrité des connaissances sur les situations ?

Les situations dans l'informatique pervasive sont fortement liées aux données de cap-

teurs, au domaine de la connaissance dans l'environnement et des utilisateurs individuels et aux applications. Comme discuté dans les sections précédentes, les données de capteurs se génèrent en quantités importantes, dans des modalités différentes et sont fortement interdépendantes, dynamiques et incertaines. Les situations suivent une relation structurelle et temporelle riche. En outre, la complexité du domaine de la connaissance et des applications rend les situations un sujet d'étude très difficile.

Dans cette thèse, nous nous sommes intéressés à la partie *raisonnement*. L'un des principaux défis est **l'identification de la situation** - dériver une situation en interprétant ou en combinant plusieurs parties du contexte. La performance du raisonnement est généralement minée par la complexité des données de capteurs.

La diversité des applications complique encore ce grand défi. L'une des principales exigences d'un système informatique pervasif est de fournir des services corrects aux bons utilisateurs aux bons endroits au bon moment et de manière correcte. Il est supposé qu'un système devrait héberger un grand nombre d'applications qui peuvent être dédiées pour différentes situations. Cela nécessite un modèle de situation/d'activité pour prendre en compte l'évolution des spécifications de situations et être en mesure de maintenir la cohérence et la consistance entre les spécifications originales et en évolution.

Ces applications peuvent également avoir différents degrés de signification pour le système pervasif, l'utilisateur ou l'environnement. Certaines applications ne peuvent être déclenchées que si une situation est critique et que la certitude d'identifier cette situation est élevée ; par exemple dans une maison intelligente, une application doit émettre un appel d'urgence lorsqu'il y a un incendie dans la maison ou un accident électrique ou encore lorsque l'habitant subit une crise cardiaque. Ce type d'application sera déclenché si ces situations dangereuses sont identifiées, même si elles sont inférées avec une confiance inférieure par rapport à d'autres situations. Le modèle de situation doit non seulement être capable de gérer l'incertitude, mais aussi être informatif pour les résultats d'inférence ; c'est-à-dire déterminer quelles sont les situations les plus susceptibles de se produire [Loke, 2010].

Dans ce qui suit, nous présentons le processus à suivre pour l'identification de situations/d'activités.

1.4 Processus d'identification de situations/d'activités

Les étapes du processus de reconnaissance de situations/d'activités sont différentes d'un travail de recherche à un autre. [Bulling et al., 2014] utilisent une définition générale appe-

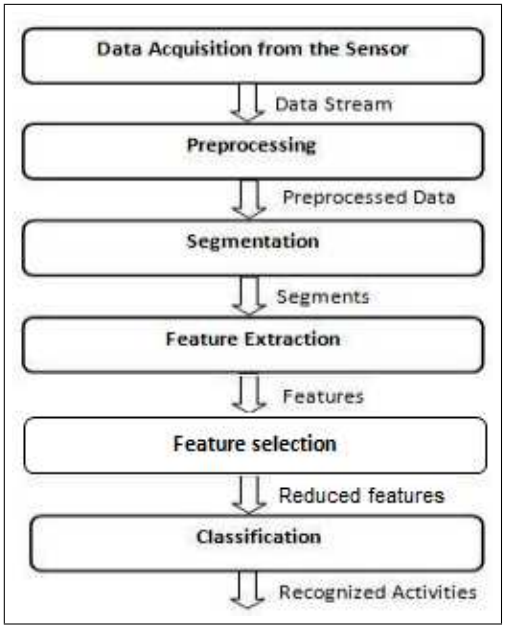


FIGURE 1.1. Processus de reconnaissance de situations/d'activités ([Avci et al., 2010])

lée chaîne de reconnaissance d'activité (*Activity Recognition Chain (ARC)*). Cette chaîne a des étapes spécifiques qui sont : *acquisition*, *pré-traitement et segmentation des signaux*, *extraction et sélection des attributs*, *apprentissage*, *classification*. Alors que dans [Avci et al., 2010], ils définissent le processus en cinq étapes principales : *acquisition*, *pré-traitement et représentation des signaux*, *segmentation*, *extraction des attributs*, *réduction de la dimension et classification*. Nous avons adopté ce processus (voir Figure 1.1) et détaillons les étapes dans les sous sections suivantes. La première étape concernant l'acquisition des données de capteurs a déjà été présentée dans la sous section 1.3.1.

1.4.1 Pré-traitement des données capteurs

Le pré-traitement des signaux ou des données de capteurs est la deuxième étape du processus. Le but de ce pré-traitement est de préparer les données acquises pour la segmentation et l'extraction des attributs. La plupart des systèmes l'utilisent pour réduire le bruit des utilisateurs ou des capteurs eux-mêmes [Shoaib et al., 2013].

Dans [Shoaib et al., 2013], quatre téléphones intelligents sont rattachés au corps du participant. Ces téléphones sont munis de capteurs de type accéléromètre et gyroscope. Le pré-traitement a été effectué en deux étapes. Tout d'abord, il faut supprimer le bruit causé par le retrait des téléphones intelligents des participants pour les arrêter. Deuxièmement, il faut

rajouter une quatrième dimension appelée *magnitude* pour chaque capteur, c'est-à-dire (x, y, z, magnitude) car la *magnitude* est insensible à l'orientation contrairement aux trois autres axes d'un accéléromètre et d'un gyroscope.

1.4.2 Segmentation

La segmentation est la troisième étape de la reconnaissance de l'activité humaine. La segmentation d'une séquence de capteurs continue est une étape primordiale puisqu'elle identifie les segments des flux de données pré-traités susceptibles de contenir des informations sur les activités. L'approche la plus courante de segmentation consiste à partitionner une séquence de données de capteurs en un intervalle de temps fixe ; par exemple, une seconde [Gu et al., 2010] ou une minute [van Kasteren et al., 2008]. Chacune des partitions résultantes partage le même intervalle de temps. Cette technique est connue sous le nom de technique de *fenêtre coulissante statique* ou bien *la segmentation à temps fixe* [Huynh et al., 2007, Bao and Intille, 2004]. Un autre type de technique de fenêtre coulissante statique est *la segmentation à taille fixe*, où chaque fenêtre a le même nombre d'événements de données de capteurs [Krishnan and Cook, 2014].

Toutes ces techniques fonctionnent bien sur des ensembles de données d'activités séquentielles mono-utilisateur [Ortiz Laguna et al., 2011].

1.4.3 Extraction des attributs

Le but de cette étape est de réduire les signaux/les données de capteurs en des attributs qui sont discriminants pour les activités. [Avci et al., 2010] définissent l'extraction des attributs comme étant la conversion de grandes données d'entrées en un ensemble d'attributs avec une représentation réduite, que l'on peut également désigner comme des vecteurs d'attributs. Le vecteur d'attributs est utilisé pour distinguer les différentes activités, puis les attributs seront utilisés comme des entrées pour les algorithmes de classification. Les attributs peuvent être dérivés en fonction des connaissances d'experts ou calculés automatiquement.

Les attributs dans la reconnaissance d'activités sont variés et nombreux. Les données brutes issues des capteurs portables [Avci et al., 2010, Bulling et al., 2014, Su et al., 2014] présentent les attributs les plus populaires utilisés dans la reconnaissance d'activité. Ainsi, [Bulling et al., 2014] classent les attributs en quatre catégories : les attributs basés sur le signal, les attributs du modèle corporel, les attributs basés sur les événements et les attributs multi-niveaux. [Avci et al., 2010] classent les attributs en cinq catégories : les attributs

du domaine temporel, les attributs du domaine fréquentiel, les attributs du domaine temps-fréquence, les attributs heuristiques et les attributs spécifiques au domaine. Pour le calcul des attributs, [Su et al., 2014] extraient les attributs depuis le domaine temporel et fréquentiel. Les attributs du domaine temporel sont la moyenne, le max, le min et la corrélation. Les attributs du domaine fréquentiel sont l'énergie, l'entropie, etc.

L'approche d'extraction d'attributs la plus populaire et la plus utilisée est l'Analyse des Composantes Principales (*Principle Component Analysis ACP* [Cateni et al., 2013]. L'ACP est une méthode non paramétrique simple, utilisée pour extraire les informations les plus pertinentes d'un ensemble de données redondantes ou bruyantes.

1.4.4 Sélection des attributs

Les données de grande dimension génèrent un nombre important d'attributs qui peuvent être non pertinents, bruités ou redondants. La sélection d'un sous-ensemble d'attributs est le processus de sélection des meilleurs attributs parmi tous les attributs existants. Cette étape est importante puisque les attributs les plus pertinents générés vont être les entrées de la phase suivante qui est la classification. Cette étape est aussi connue sous le nom de *réduction de dimension*. En général, la sélection des attributs reflète un problème de recherche selon certains critères d'évaluation.

Les méthodes de sélection d'attributs peuvent être classées en trois catégories : filtres, enveloppes et embarquées. La description détaillée de ces trois catégories est faite dans le Chapitre 3.

1.4.5 Classification et reconnaissance de situations/d'activités

Dans l'apprentissage automatique, la classification vise à étiqueter chaque ensemble d'attributs en l'associant à une classe. Dans notre cas, la classe est la situation/l'activité. Pour cela, on parle alors de reconnaissance de situations/d'activités.

En général, les modèles de reconnaissance de situations/d'activités doivent être construits avant d'être utilisés. Il existe deux approches principales pour la reconnaissance de situations/d'activités dans la littérature : *les approches orientées connaissance* et *les approches orientées données*. La classification en apprentissage automatique est considérée comme une approche orientée donnée. Les données d'entrée sont les vecteurs d'attributs générés depuis l'étape *réduction de dimension* ou bien directement de la phase *extraction d'attributs*. Les méthodes de reconnaissance de situations/d'activités sont détaillées dans la section suivante

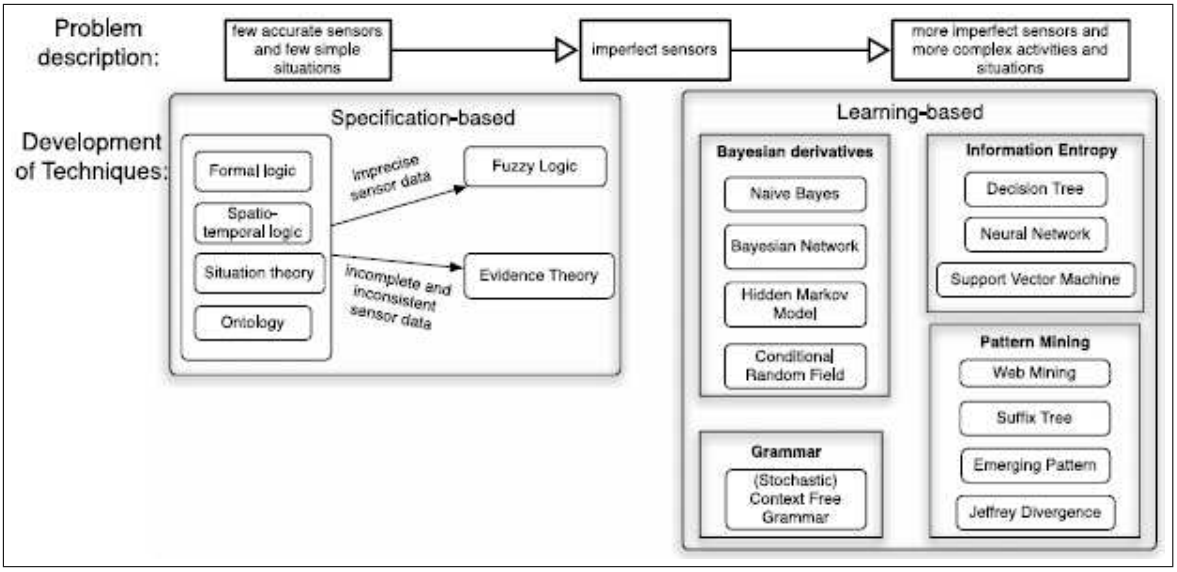


FIGURE 1.2. Les méthodes d'identification de situations/activités [Ye et al., 2012]

(section 1.5).

1.5 Les méthodes de raisonnement pour la reconnaissance de situations/ d'activités

La communauté scientifique a développé deux principales approches de raisonnement pour la reconnaissance de situations/d'activités, à savoir les approches orientées connaissance et les approches orientées données. Les approches orientées données utilisent des ensembles de données de capteurs à grande échelle pour apprendre des modèles d'activité en utilisant des techniques d'exploration de données et d'apprentissage automatique. D'un autre côté, les approches orientées connaissance exploitent des connaissances pour construire des modèles d'activité en utilisant des technologies d'ingénierie et de gestion des connaissances. L'ensemble des techniques utilisées dans les deux approches est illustré dans la Figure 1.2 qui sera détaillé dans la suite.

1.5.1 Les méthodes de raisonnement orientées connaissances

Dans les premiers travaux, l'identification de situations a commencé avec peu de capteurs dont les données sont faciles à interpréter et les relations entre les données de capteurs et les situations sont faciles à établir. Ces approches orientées connaissances ou également

appelées approches basées sur les spécifications, utilisent les connaissances sous forme de règles logiques et appliquent des moteurs de raisonnement pour déduire les situations appropriées à partir des données actuelles des capteurs. Ces approches se sont développées depuis la logique du premier ordre vers un modèle logique plus formel tout en gardant un pouvoir expressif du raisonnement [Ye et al., 2012].

La modélisation de l'activité orientée connaissances est basée sur des observations du monde réel. Par exemple, pour préparer du café, il faut un récipient contenant du café et du sucre. Même si différentes personnes peuvent utiliser différentes marques de café, certaines peuvent rajouter du lait et d'autres préfèrent le sucre brun ou le sucre blanc. Certains concepts essentiels sont toujours présents pour chaque activité. L'idée est d'utiliser cette connaissance préalable pour créer des modèles d'activité initiales. Les relations implicites entre les activités, le contexte temporel et spatial associé et les entités impliquées (objets et personnes) fournissent une diversité d'indices et d'heuristiques pour identifier les activités.

La première étape pour les systèmes orientés connaissances est d'acquérir les informations contextuelles nécessaires. Ceci est généralement réalisé en utilisant des approches d'ingénierie de connaissances standards. Ensuite, les structures de connaissances seront modélisées en utilisant un formalisme formel de représentation des connaissances, par exemple, les schémas, les règles ou les réseaux. Selon la nature des connaissances acquises, différentes approches peuvent être distinguées. Certains travaux utilisent des approches logiques pour la reconnaissance des activités. Il existe, en effet, un certain nombre d'approches de modélisations logiques et de techniques de raisonnement basées sur des théories logiques et des formalismes de représentation.

1. **La programmation logique** : elle commence par l'utilisation de la logique des prédicats dans la définition des situations. Les travaux dans [Henricksen and Indulska, 2006, Delir Haghighi et al., 2008], prennent l'initiative de considérer les situations comme des abstractions des données de capteurs qui auraient plus d'influence sur les applications. Ils ont travaillé sur la façon de définir des situations par des formules logiques simples en utilisant des connaissances purement expertes. Nous en citons quelques uns.

   Prenons l'exemple de [Henricksen and Indulska, 2006], qui définit une situation *Occupied* lorsqu'une personne est détectée comme étant engagée dans un événement qui ne devrait généralement pas être interrompu. Par exemple, *in meeting* ou *taking call*. Cette situation *Occupied* est spécifiée par le formalisme décrit dans la Figure 1.3.

$$
\begin{aligned}
Occupied(person) = \exists t_1, t_2, event\,|engagedIn(person, event, t_1, t_2) \\
\cdot((t_1 \leq timenow() \wedge (timenow() \leq t_2 \vee isnull(t_2))) \\
\vee((t_1 \leq timenow() \vee isnull(t_1)) \wedge timenow() \leq t_2)) \\
\wedge(event = \text{'inmeeting'} \vee event = \text{'takingcall'})
\end{aligned}
$$

FIGURE 1.3. Un exemple de situation formalisé par la logique des prédicats [Henricksen and Indulska, 2006]

```
(?user rdf:type socam:Person),
(?user, socam:locatedIn, socam:Bedroom),
(?user, socam:hasPosture, 'LIEDOWN'),
(socam:Bedroom, socam:lightLevel, 'LOW'),
(socam:Bedroom, socam:doorStatus, 'CLOSED')
-> (?user socam:status 'SLEEPING')
```

FIGURE 1.4. Un exemple de règle formalisée pour l'identification de l'activité *sleeping* [Gu et al., 2005]

2. **L'ontologie** : Les ontologies ont de plus en plus attiré l'attention en tant que moyen générique, formel et explicite pour *détecter et spécifier la connaissance du domaine avec sa sémantique intrinsèque à travers la terminologie consensuelle et les axiomes formels et les contraintes* [Ye et al., 2007]. Elles fournissent une manière formelle de représenter les données de capteurs, le contexte et les situations dans une terminologie bien structurée, ce qui les rend compréhensibles, interopérables et réutilisables par les humains et les machines. Les ontologies prennent en charge un ensemble de primitives de modélisation pour définir des classes, des individus, des propriétés d'attributs et des propriétés d'objets (c'est-à-dire, des relations entre des objets). Par exemple, la propriété *is-a* est l'une des propriétés les plus utiles pour modéliser le niveau d'abstraction de deux concepts de domaine : *'Dining Room' is-a 'Eating Activity Space'* et *'Pan' is-a 'Cooking Utensil'* [Tapia et al., 2006]. Les ontologies sont expressives dans la modélisation d'entités dans un environnement intelligent et les concepts de domaine y compris les capteurs [Stevenson et al., 2009], le profil utilisateur, les objets avec lesquels les utilisateurs interagissent, et les informations ambiantes telles que la température, l'humidité et le niveau sonore [Gu et al., 2004]. En se basant sur des concepts modélisés, les experts peuvent définir des spécifications logiques de situations via des règles. Un exemple de règle sur l'activité *sleeping* est extrait depuis [Gu et al., 2005] (voir Figure 1.4).

3. **La logique floue** : La théorie des sous ensembles flous est largement utilisée pour traiter l'imprécision des données. Un sous ensemble flou est représenté par une fonction d'appartenance : elle caractérise à quel point un élément appartient à un ensemble flou [Zadeh, 1965]. Dans l'informatique pervasive, la logique floue est utilisée pour

mettre en correspondance les données de capteurs avec les variables linguistiques (les variables floues) qui ont un sens et sont compréhensibles par l'être humain [Delir Haghighi et al., 2008]. Un degré d'appartenance reflète la valeur de vérité auquel un élément est membre d'un ensemble. Par exemple, pour la température, les attributs flous associés tels que *froid*, *tiède* et *chaud* sont reliés à une plage de degrés de température. Un degré de température de 15°C pourrait être évalué par rapport à ces attributs flous et sera interprété comme étant *froid* avec la valeur floue 0.3, *tiède* avec 0.6 et *chaud* avec 0.1. C'est la méthode privilégiée pour gérer les connaissances numériquement fournies par les dispositifs de mesure et les connaissances symboliquement exprimées par un observateur humain.

4. **La théorie Dempster–Shafer (DS)** : C'est une théorie mathématique basée sur la notion de preuves utilisant les fonctions de croyance et le raisonnement plausible. Elle acquiert un intérêt croissant dans le domaine de la reconnaissance des activités et est adaptée pour détecter l'incertitude du contexte dans le mécanisme de raisonnement. La théorie DS est utilisée pour fusionner les données de capteurs de bas niveau afin de générer des activités de haut niveau. Une structure de modèle d'activité doit être connue pour maintenir la distribution et la fusion des croyances. Dans des travaux récents, certains modèles ont été proposés pour gérer la reconnaissance d'activité à partir des données de capteurs de bas niveau. Un graphe orienté acyclique (*Directed Acyclic Graph ou DAG*) pour l'identification d'une situation est proposé dans [McKeever et al., 2010] (voir Figure 1.5). Dans ce diagramme, les capteurs représentent les nœuds racines dans la base du diagramme. Les données de capteurs sont liées à une ou plusieurs valeurs de contexte. Chaque valeur de contexte sera à son tour liée à une ou plusieurs situations. Ainsi, la propagation et l'agrégation des croyances s'effectuent à partir des données de capteurs jusqu'aux activités.

1.5.2 Les méthodes de raisonnement orientées données

La spécification et l'identification des situations peuvent varier en fonction des facteurs tels que le temps, la localisation, les utilisateurs et les environnements. Cela rend les approches basées sur les spécifications (utilisant des modèles de connaissances a priori) peu pratiques à utiliser. Les techniques d'apprentissage automatique ont été largement appliquées pour apprendre les associations complexes entre les situations et les données de capteurs. Les approches orientées données également appelées approches basées sur l'apprentissage

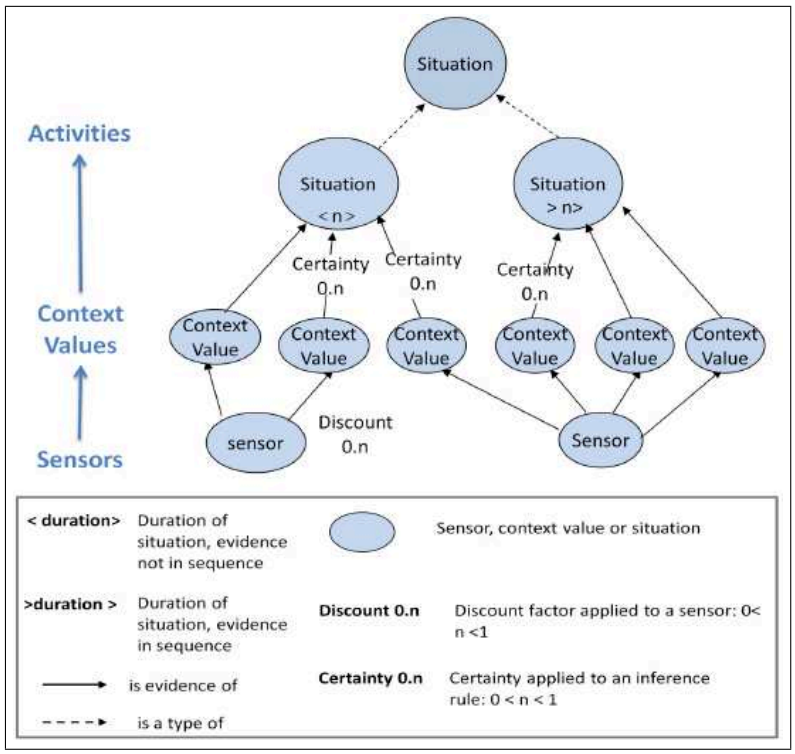


FIGURE 1.5. Le graphe orienté acyclique proposé par [McKeever et al., 2010] pour l'identification d'une situation

utilisent des bases de données de capteurs à grande échelle pour apprendre les modèles d'activités/de situations en utilisant des techniques d'apprentissage automatique. Il existe deux grandes catégories d'apprentissage automatique qui sont l'apprentissage supervisé et l'apprentissage non supervisé [Alpaydin, 2010]. Dans cette thèse, nous avons utilisé l'apprentissage supervisé et présentons dans la suite quelques méthodes utilisées dans ce domaine précis [Alpaydin, 2010].

1. **Le classifieur naïf bayésien (*Naïve Bayes NB*) [Alpaydin, 2010]** : est l'un des classifieurs les plus simples en apprentissage supervisé basé sur le théorème de Bayes. Naïve Bayes a été largement utilisé [Mühlenbrock et al., 2004, Patterson et al., 2003]. Par exemple, [Mühlenbrock et al., 2004] appliquent NB pour l'apprentissage des activités des utilisateurs et aussi de la disponibilité des utilisateurs.
2. **Les arbres de décision (*Decision Tree DT*) [Alpaydin, 2010]** : Un arbre de décision est un modèle prédictif où chaque feuille représente une classe et chaque branche représente une conjonction d'attributs qui mènent aux classes. L'un des principaux avantages de l'utilisation d'un arbre de décision est la possibilité de générer des règles

de classification faciles à comprendre et à expliquer. Ces règles sont utiles pour analyser les performances des capteurs et pour l'extraction des attributs. Ainsi, [Bao and Intille, 2004] ont utilisé des arbres de décision pour classer les activités humaines à partir des données issues des capteurs d'accélération rattachés au corps humain.

3. **Les forêts aléatoires (*Random Forest RF*) [Breiman, 2001]** : Elles combinent des prédicteurs ou estimateurs de base que sont les arbres, donnant lieu à ce que l'on appelle aujourd'hui les méthodes d'arbres. Plus largement ce sont des méthodes d'ensemble dont le principe général est de construire une collection de prédicteurs (un ensemble d'arbres de décision), pour ensuite agréger l'ensemble de leurs prédictions.

4. **Les arbres extrêmement aléatoires (*Extra-Trees ExT*)** : [Geurts et al., 2006] ont proposé les arbres extrêmement aléatoires. Chaque arbre est construit à partir de l'ensemble d'apprentissage au lieu d'un échantillon bootstrap (c'est un échantillon aléatoire simple avec remplacement de n éléments parmi l'échantillon de taille n). Une coupe est effectuée au hasard sur un attribut choisi aléatoirement dans l'ensemble d'attributs sans tenir compte des classes. Ainsi les Extra-Trees sont très rapides pour l'apprentissage et réduisent significativement la corrélation entre les arbres de la forêt.

5. **Les machines à vecteurs de support (*Support Vector Machine SVM*) [Alpaydin, 2010]** : est une méthode relativement nouvelle de classification des données linéaires et non linéaires. Un SVM utilise une mise en correspondance non linéaire pour transformer les données d'apprentissage d'origine en des données de dimension supérieure. Dans cette nouvelle dimension, il recherche l'hyperplan linéaire de séparation optimale qui sépare les données d'apprentissage d'une classe par rapport aux autres. Avec une mise en correspondance non linéaire appropriée à une dimension supérieure, les données de deux classes peuvent toujours être séparées. Les SVM sont adaptés pour gérer de grands espaces d'attributs car ils utilisent une protection contre le surapprentissage *overfitting*, qui ne dépend pas nécessairement du nombre d'attributs. Les SVM ont été appliqués pour la reconnaissance d'activité par [Kanda et al., 2008]. Ces derniers utilisent SVM pour catégoriser les trajectoires de mouvement humain (telles que 'fast walk', 'idle walk', 'wander', et 'stop') en fonction de leurs attributs de vitesse, de direction et de forme.

1.6 Limites du raisonnement dans les systèmes d'intelligence ambiante

L'intelligence d'un système ambiant est principalement déterminée par ses capacités de raisonnement. Le but du raisonnement dans l'intelligence ambiante est d'exploiter les données de capteurs et les traduire en des données de haut niveau d'abstraction (les situations/les activités).

En général, le rôle du système d'intelligence ambiante est de :

— détecter les données de capteurs erronées
— gérer les données de capteurs manquantes
— évaluer la qualité et la validité des données de capteurs détectées
— transformer les données de capteurs en des données haut-niveau d'abstraction
— identifier les situations/activités à partir des données haut niveau d'abstraction.
— s'adapter dynamiquement et prendre la bonne décision lors de l'identification des situations/des activités.

Face à ces exigences, les principaux défis que nous pouvons considéré dans un système d'intelligence ambiante :

— Le raisonnement dans un contexte dynamique, imparfait et ouvert ;
— Le raisonnement distribué dans un environnement où les capteurs sont répartis et hétérogènes ;
— La gestion des données de capteurs volumineuses en temps réel
— L'imperfection des données de capteurs.

La plupart des systèmes actuels de l'intelligence ambiante n'ont pas réussi à gérer tous les défis discutés ci-dessus. Ceci résulte principalement de l'absence d'un modèle de raisonnement qui gère l'imperfection des données et de son architecture centralisée. Cette dernière désigne une entité centrale responsable de la collecte des données de capteurs et du raisonnement sur celles-ci. Ces deux limites sont discutées ci-dessous en détail.

1.6.1 Raisonnement dans un contexte imparfait

Les propriétés des systèmes ambiants ont un impact sur la cohérence des informations de contexte. La composition d'observations réparties provenant de différentes sources peut introduire des incohérences potentielles. De plus, le contexte est un élément dynamique évoluant fréquemment et affectant en continu les applications sensibles au contexte. Par

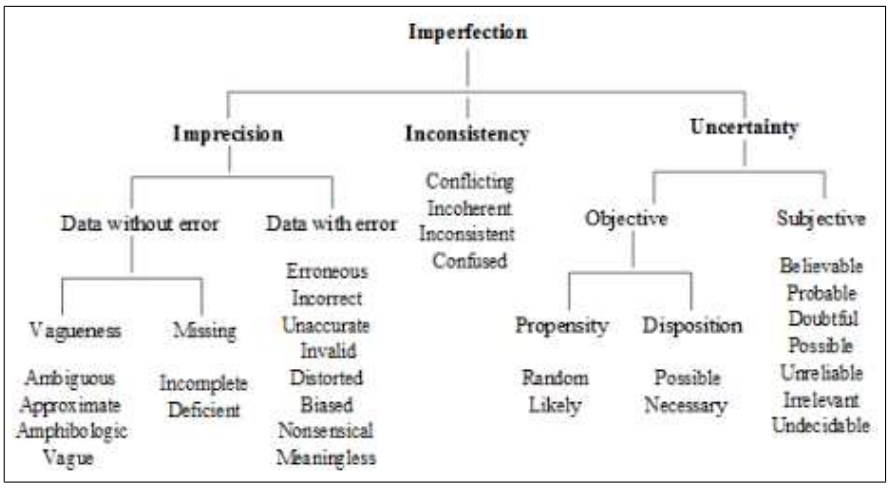


FIGURE 1.6. Imperfection de l'information dans l'intelligence artificielle ([Smets, 1997])

conséquent, la gestion de l'imperfection est indispensable dans la reconnaissance de situations/d'activités. En effet, l'imperfection intervient tout au long du processus de décision. Il y a un niveau d'imperfection associé aux données de capteurs perçues, le degré de compréhension de ces données et leurs capacité à projeter ce qui va se passer dans le futur. Du coup, il existe trois niveaux d'imperfection dans le processus de prise de décision : l'imperfection des données, l'imperfection de compréhension et l'imperfection de projection [Endsley, 2011].

1.6.1.1 L'imperfection des données

Les données de contexte sont par nature imparfaites puisque d'une part elles résultent d'un modèle représentant une abstraction de la réalité ne pouvant être exhaustive et d'autre part elles dépendent de dispositifs tels que des capteurs ayant des limitations physiques intrinsèques. L'imperfection des données est donc associée à des erreurs issues des capteurs.

L'imperfection des données est définie de manière différente selon le domaine d'application. En intelligence artificielle, [Smets, 1997, Snoussi et al., 2012] classent une donnée imparfaite en trois grandes catégories qui sont : *imprécise*, *inconsistante* et *incertaine* (voir Figure 1.6). Dans les systèmes de gestion de bases de données, une donnée imparfaite est classée en 5 classes qui sont : *inconsistante*, *imprécise*, *vague*, *incertaine* et *ambiguë* ([Bosc and Prade, 1997]). Dans l'informatique pervasive, [Ye et al., 2012] définit une donnée imparfaite ou encore incertaine en 5 catégories : *obsolète*, *imprécise*, *incomplète*, *inconsistante* et *contradictoire*.

Dans cette thèse, nous avons adopté la dernière définition d'une donnée imparfaite ou

Techniques	Incomplete	Imprecise	Inaccurate	Contradictory	Out of date
Fuzzy logic	X	√	X	X	√
Evidence theory	√	X	X	√	X
Bayesian model	X	X	√	√	X
Decision tree	√	X	√	√	X

FIGURE 1.7. Techniques de résolution des différentes dimensions de l'imperfection/l'incertitude ([Ye et al., 2012])

incertaine ([Ye et al., 2012]). Ainsi, **(1) obsolète** : si une donnée est trop ancienne, elle peut être périmée ; **(2) imprécise** : les données imprécises sont correctes, mais inexactes. Par exemple, la localisation d'un utilisateur par ses coordonnées est plus précise que la localisation par pièce ; **(3) inconsistante** : une donnée inconsistante est complètement ou partiellement erronée ; **(4) contradictoire** : une contradiction se produit lorsque deux données provenant de deux sources différentes, sont opposées ; **(5) incomplète** : lorsque les données sont manquantes.

Lorsque les données de contexte collectées sont imparfaites, le risque existe de baser une décision sur des données erronées [Brazdil and Clark, 1990]. De plus, le coût est très élevé pour raisonner sur des données incertaines vu la complexité des solutions à mettre en œuvre [Smets, 1997, Motro, 1995, Motro, 1997]. L'imperfection inhérente aux données de contexte peut être traitée par différentes méthodes de raisonnement orientées données ou orientées connaissance (voir Figure 1.7, ✓ : dimension gérée ; ✗ : dimension non gérée).

Nous avons mené une étude sur les travaux existants qui ont proposé des modèles de raisonnement intégrant des dimensions de l'incertitude.

D'après le tableau comparatif (Tableau 1.1, ✓ : dimension gérée ; ✗ : dimension non gérée), nous constatons qu'il n'existe pas de modèle de raisonnement qui a pu gérer toutes les dimensions de l'incertitude des données.

1.6.1.2 L'imperfection de compréhension

Dans la vie réelle, les gens associent un degré de confiance à leur compréhension de ce qui se passe. Prenons cet exemple : dans le domaine médical, le médecin peut avoir 80% de confiance dans un diagnostic particulier après avoir effectué des analyses et des mesures physiologiques.

De même, les systèmes de reconnaissance de situations/d'activités, peuvent combiner la sortie de plusieurs données de capteurs pour produire une localisation cible ou une identification. Cette sortie est accompagnée par un certain degré de confiance basé à la fois sur la nature des données de capteurs et sur la qualité de l'algorithme utilisé pour déterminer

TABLE 1.1. Exemples de travaux utilisant des techniques pour résoudre l'incertitude

Travaux	Approche de raisonnement	Modèle de raisonnement	Les dimensions de l'incertitude prises en compte				
			Contradiction	Imprécision	Inconsistance	Obsolescence	Incomplétude
[Mohd Noor et al., 2016]	orientée connaissance	Ontologie OWL + Théorie de Dempster–Shafer	✗	✗	✓	✗	✓
[Riboni et al., 2016]	hybride	Ontologie OWL + Réseau de logique de Markov	✓	✗	✓	✓	✗
[Meditskos et al., 2015]	orientée connaissance	Ontologie OWL + Logique défaisable	✓	✗	✓	✗	✗
[Díaz Rodríguez et al., 2014]	orientée connaissance	Ontologie floue + raisonneur FuzzyDL + dynamic time warping (DTW)	✗	✓	✗	✗	✓
[Wang and Ji, 2012]	orientée donnée	Réseaux bayésiens dynamiques	✗	✗	✗	✗	✓

la situation/l'activité. Trouver des moyens de représenter de manière pertinente le niveau de confiance associé à la classification et à l'agrégation des données est très important pour garantir le succès des systèmes de reconnaissance.

1.6.1.3 L'imperfection de projection

Projeter ce qui se passera dans le futur est intrinsèquement incertain dans la plupart des systèmes de reconnaissance. Un degré d'incertitude doit accompagner les actions à mener dans le futur. La difficulté de la projection provient non seulement de l'incertitude des données, mais aussi de la capacité du système à prédire avec précision le comportement futur.

Dans cette thèse, la projection n'entre pas dans le périmètre de nos travaux. Nous nous intéressons aux deux premiers types de l'incertitude qui sont : l'incertitude des données et l'incertitude de compréhension.

1.6.2 Raisonnement dans un contexte distribué

La majorité des systèmes ambiants s'appuient sur une approche totalement centralisée. Cette approche désigne une entité centrale responsable à la fois de la collecte des données contextuelles à partir des entités présentes dans l'environnement et du raisonnement sur celles-ci.

Des exemples représentatifs de tels systèmes centralisés pour la reconnaissance de situa-

tions/d'activités sont cités dans le Tableau 1.2 (✓ : défi géré ; ✗ : défi non géré). Ces travaux se sont intéressés principalement à des problématiques telles que la gestion de l'incertitude des données de capteurs, la gestion temporelle des données d'entrée, l'identification de situations/d'activités à partir des données annotées, partiellement annotées ou non annotées, la gestion des activités concurrentes et la reconnaissance hybride de situations/d'activités (utilisation d'une approche orientée connaissance et orientée donnée à la fois).

Ces approches centralisées intègre un modèle de reconnaissance de situations/d'activités déjà construit et identifient les activités au fur et à mesure que l'environnement change. Dans un environnement pervasif, ces approches ont certainement de nombreux avantages. En effet, elles réalisent un meilleur contrôle et une meilleure coordination entre les différentes entités participantes. Néanmoins, les connexions entre les capteurs et le système centralisé ne sont pas garanties. Les communications peuvent être peu fiables ou restreintes dans certaines conditions. En outre, gérer un grand volume de données de capteurs entrants provenant de différentes localisations diminue considérablement la performance du système.

Pour remédier à cela, une approche totalement décentralisée est nécessaire permettant de répartir à la fois les sources des données de contexte et le raisonnement. Le besoin d'avoir une approche décentralisée dans les systèmes d'intelligence ambiante a récemment donné lieu à des nombreux travaux de recherche à déployer les méthodes et les techniques de l'intelligence artificielle [Bikakis and Antoniou, 2010].

Le raisonnement distribué est nécessaire pour résoudre la complexité résultante de la coexistence des différentes entités qui collectent, stockent, traitent, échangent et raisonnent sur les données contextuelles [Bikakis et al., 2008]. Par conséquent, certaines approches de raisonnement distribué tentent de surmonter cette limite, telles que Gaia [Roman et al., 2002], DRAGO [Serafini and Tamilin, 2005], P2P-DR [Bikakis and Antoniou, 2008], P2PIS [Adjiman et al., 2006], DRS [Viterbo and Endler, 2009], DDSR [Lam et al., 2012] et DESIA [Maciel et al., 2015]. Nous discutons des principales caractéristiques de chacune de ces solutions dans ce qui suit et les résumons dans le Tableau comparatif 1.3.

1.6.2.1 Gaia

La plateforme *Gaia* [Roman et al., 2002] vise à fournir un environnement informatique générique permettant d'intégrer les espaces physiques et leurs dispositifs informatiques ubiquitaires dans un système informatique. Elle fournit des services de base, y compris les événements, la présence d'entités (périphériques, utilisateurs et services), la découverte et la dénomination. En spécifiant des interfaces associées à des services, les applications peuvent

TABLE 1.2. Les approches centralisées pour la reconnaissance de la situation/d'activité

Travaux	Approche de raisonnement	Techniques	Gestion temporelle	Gestion de l'incertitude	Données annotées	Situation/ activité
[Attard et al., 2013]	connaissance	Technique de maching + Ontologie (DCON)	✓	✗	✗	Situation
[Okeyo et al., 2014]	connaissance	Logique de description + Logique temporelle + Ontologie (ADL)	✓	✗	✗	Activités séquentielles et concurrentes
[Rodriguez et al., 2014]	connaissance	Logique floue + ontologie (CRISP)	✗	✓	✗	Activités séquentielles
[Daniello et al., 2015]	hybride	FFCA (règles de classification) + ontologies (SSN,SAW,domain)	✗	✓	✗	Situation
[Ye et al., 2015]	hybride	Pyramid Match Kernel + ontologie (domaine, capteur, activité)	✓	✗	✓	Activités séquentielles et concurrentes
[Azkune et al., 2015]	hybride	Clustering + JSON (context knowledge file)	✓	✗	✗	Activités séquentielles
[Wen et al., 2015]	donnée	Extraction des règles d'association	✗	✗	✓	Activités séquentielles
[Wen and Zhong, 2015]	donnée	Mesure de similarité + Clustering	✓	✗	Partielle	Activités séquentielles
[Fahad and Rajarajan, 2015]	donnée	Distance from means + Probability estimation	✗	✗	✗	Activités séquentielles

être construites d'une manière générale et donc elles peuvent s'exécuter dans des espaces actifs arbitraires.

L'infrastructure contextuelle de *Gaia* permet aux applications d'obtenir une variété d'informations. Divers composants, appelés *fournisseurs de contexte*, obtiennent le contexte à partir des capteurs ou d'autres sources de données. Ceux-ci incluent des capteurs qui détectent la localisation des personnes, les conditions de la pièce (par exemple, la température et le son), etc. Les *fournisseurs de contexte* permettent aux applications de les interroger pour obtenir des informations sur le contexte. Certains *fournisseurs de contexte* disposent également d'un canal d'événements pour envoyer des événements de contexte de manière asynchrone. Ainsi, les applications peuvent interroger un *fournisseur de contexte* ou écouter sur le canal des événements pour obtenir des informations sur le contexte.

Les *synthétiseurs de contexte* sont des composants *Gaia* qui collectent des données de contexte auprès de divers *fournisseurs de contexte*, dérivent le contexte de haut-niveau d'abstraction et fournissent ces contextes inférés aux applications. Le *synthétiseur* déduit un changement de contexte inféré, il publie les nouvelles informations.

Gaia adopte deux approches d'inférence de base. Les synthétiseurs basés sur des règles utilisent des règles prédéfinies écrites avec la logique du premier ordre pour inférer différents contextes. Chacune des règles a également une priorité associée, qui permet de choisir une règle lorsque plusieurs règles sont valides en même temps. Toutefois, si toutes les règles valides ont la même priorité, l'une d'elles est choisie au hasard. Certains *synthétiseurs* peuvent également utiliser des techniques d'apprentissage automatique, telles que l'apprentissage bayésien et l'apprentissage par renforcement, pour en déduire des contextes de haut niveau.

1.6.2.2 DRAGO

L'architecture de raisonnement distribué pour une galaxie d'ontologies (*Distributed Reasoning Architecture for a Galaxy of Ontologies DRAGO*) [Serafini and Tamilin, 2005] est un système de raisonnement distribué implémenté sous forme d'architecture pair à pair, dans laquelle chaque pair enregistre un ensemble d'ontologies et des correspondances. Dans DRAGO, les opérations de raisonnement sont effectuées en utilisant un raisonnement local sur chaque ontologie enregistrée et en coordonnant avec d'autres pairs lorsque des ontologies locales sont sémantiquement connectées aux ontologies enregistrées dans d'autres pairs. DRAGO n'implémente pas de couche de contexte, c'est-à-dire qu'il ne dispose d'aucun service de collecte, de stockage ou de distribution de contexte.

DRAGO est implémenté pour fonctionner sur HTTP et accéder aux ontologies et aux

correspondances publiées sur le Web. Il regroupe un réseau d'ontologies réparties sur un réseau pair-à-pair dans lequel chaque pair est appelé un *DRAGO Reasoning Peer (DRP)*. Un DRP est l'élément de base du système et peut contenir un ensemble de différentes ontologies décrivant des domaines d'intérêt spécifiques (par exemple, des ontologies décrivant différentes activités des utilisateurs d'une université). Dans un DRP, il existe également des correspondances sémantiques, chacune définissant des relations sémantiques entre des entités appartenant à deux ontologies différentes, décrites à l'aide de C-OWL [Bouquet et al., 2003]. Comme ces correspondances établissent une corrélation entre l'ontologie locale et les ontologies affectées à d'autres DRP, un DRP peut également demander des services de raisonnement pour d'autres DRP dans le cadre d'une tâche de raisonnement répartie. Parmi les services de raisonnement, DRAGO permet de vérifier la cohérence des ontologies, construire des classifications, vérifier la conformité des concepts et vérifier l'implication.

Le raisonnement avec plusieurs ontologies est effectué en combinant des opérations de raisonnement locales, exécuté en interne dans chaque pair pour chaque ontologie distincte. Un algorithme de tableau distribué est adopté pour vérifier la conformité du concept dans un ensemble d'ontologies inter-connectées en combinant des procédures de tableaux locaux qui vérifient la conformité à l'intérieur de l'ontologie unique.

1.6.2.3 P2P-DR

P2P-DR [Bikakis and Antoniou, 2008] propose une solution distribuée pour raisonner sur un contexte adapté aux spécificités des environnements d'intelligence ambiante. Cette approche modélise les entités d'un environnement ambiant en tant que nœuds dans un système pair-à-pair, dans lequel chaque pair collecte et traite indépendamment les informations de contexte disponibles. Plus précisément, il considère les pairs qui ont une connaissance exclusive et qui interagissent avec les pairs voisins pour échanger des informations de contexte. La connaissance de chaque pair est exprimée en termes de règles et les connaissances sont importées à partir d'autres pairs via des règles de correspondance. Comme chaque pair peut ne pas avoir un accès direct à toutes les sources d'information, il partage ses connaissances via des messages avec ses pairs voisins. En outre, l'algorithme de raisonnement *P2P-DR* modélise et résout les conflits potentiels pouvant survenir lors de l'intégration de la connaissance distribuée.

Dans un système *P2P-DR*, chaque pair dispose d'une capacité de calcul et de raisonnement pour résoudre une requête portant sur un littéral local, en fonction de ses connaissances locales et reçues. Un pair peut ne pas être en mesure de résoudre la requête localement,

il tente alors de la résoudre avec d'autres pairs voisins via des règles de correspondance. Chaque pair dispose d'une théorie locale défaisable (ensemble de règles strictes et défaisables). La logique défaisable est utilisée pour gérer la consistance et la contradiction des connaissances échangées. Chaque pair étant prêt à divulguer et à partager ses connaissances locales, ces derniers communiquent avec un sous-ensemble des autres pairs disponibles pour importer les connaissances nécessaires à la résolution de la requête.

1.6.2.4 P2PIS

Un système d'inférence pair-à-pair P2PIS [Adjiman et al., 2006] est un réseau de théories de pairs. Chaque pair comporte un ensemble fini de formules propositionnelles et peut être sémantiquement lié en partageant des variables avec d'autres pairs. Une variable partagée entre deux pairs se trouve à l'intersection des vocabulaires des deux pairs. Dans un *P2PIS*, aucun pair n'a la connaissance de la théorie *P2PIS* globale. Chaque pair ne connaît que sa propre théorie locale et les variables qu'il partage avec d'autres pairs du *P2PIS* (ses connaissances). Il ne connaît pas nécessairement toutes les variables qu'il a en commun avec d'autres pairs (y compris avec ses connaissances). Lorsqu'un nouveau pair rejoint un *P2PIS*, il déclare simplement ses connaissances dans le *P2PIS*, c'est-à-dire les pairs avec lesquels il partage des variables et déclare les variables partagées correspondantes.

1.6.2.5 DRS

Le système de raisonnement décentralisé (*Decentralized reasoning system DRS*) [Viterbo and Endler, 2009] est proposé pour effectuer un raisonnement via des règles sur les données de contexte. Dans ce système, il existe deux parties principales : le côté utilisateur et le côté ambiant. Les deux parties, *un raisonneur côté utilisateur* et *un raisonneur côté ambiant*, interagissent pour déduire des situations décrites par des règles impliquant des variables contextuelles sources et stockées des deux côtés effectuant un raisonnement coopératif.

Ainsi, DRS adopte une approche de raisonnement coopératif basé sur le modèle pair-à-pair, dans laquelle deux grandes parties coopèrent pour effectuer une déduction partagée de faits impliquant des données réparties entre les deux. DRS utilise une ontologie de contexte globale pour représenter les différents composants de l'environnement. Chaque pair raisonneur comporte une base de règles décrite avec le langage de règles SWRL [1].

1. https ://www.w3.org/Submission/SWRL/ date de consultation Janvier 2019

1.6.2.6 DDSR

[Lam et al., 2012] proposent une approche totalement distribuée pour l'intelligence ambiante à travers la plateforme *DDSR (Distributed Defeasible Speculative Reasoning)*. Les auteurs modélisent l'environnement ambiant comme un système multi-contextes et les agents ambiants en tant que entités logiques autonomes. Les connaissances possédées par un agent sont formalisées sous forme de théorie du contexte local et les associations entre les connaissances possédées par les autres agents ambiants utilisent des littéraux sous forme de sous buts. Les incohérences et les ambiguïtés de la théorie du contexte local sont traitées par la sémantique de la logique défaisable, tandis que les informations de contexte manquantes seront tout d'abord remplacées par les valeurs par défaut utilisées dans le processus de calcul spéculatif et seront remplacées par les informations réelles lorsqu'elles seront disponibles.

1.6.2.7 DESIA

[Maciel et al., 2015] décrivent les principales caractéristiques et les exigences de l'architecture d'intégration de dispositifs, les environnements et les réseaux sociaux à travers *DESIA*, une architecture destinée à soutenir le développement d'applications ubiquitaires. Cette architecture est basée sur deux travaux, le Campus Framework [Seghrouchni et al., 2008] et le projet Ao Dai [El Fallah Seghrouchni et al., 2012]. Sa principale motivation est d'actualiser et d'étendre les principales fonctionnalités de ces deux travaux afin d'agréger les technologies émergentes telles que les villes intelligentes, les réseaux de capteurs et les réseaux sociaux, en mettant l'accent sur la sécurité et la confidentialité, des aspects qui ne sont pas toujours couverts par d'autres architectures.

DESIA est une architecture multi-agents divisée en trois couches différentes, la couche personnelle, la couche ambiante et la couche de nuages. *La couche personnelle* doit être mise en place chez le périphérique personnel de chaque utilisateur. Elle a pour but de fournir des services nécessaires pour prendre en charge les applications clientes. *La couche ambiante* doit être implémentée dans chaque environnement physique. Elle est chargée de fournir les fonctionnalités permettant de prendre en charge et d'intégrer les entités matérielles et logicielles dans ces espaces. *La couche cloud* est le noyau de l'architecture proposée. Elle est responsable de l'intégration des données de grands groupes d'utilisateurs et d'organisations, de la collecte d'informations sur le Web et de la fourniture de services d'inférence de haut niveau.

Dans cette thèse, nous avons considéré plusieurs défis comme des objectifs à atteindre et

TABLE 1.3. Les systèmes de raisonnement distribué dans les environnements ambiants

Travaux	Système distribué	Modèle de raisonnement	Technique de raisonnement	Problématique	Proposition
Système Gaia [Roman et al., 2002]	orienté composants	Un composant synthétiseur contient une base de règles	Logique de premier ordre	Il n'existe aucune infrastructure logicielle appropriée permettant de développer des applications pour des habitats informatiques ou des espaces de vie ubiquitaires	Une infrastructure intergiciel distribuée qui coordonne les entités logicielles et les périphériques hétérogènes mis en réseau existants dans un espace physique
Système DRAGO [Serafini and Tamilin, 2005]	Pair à pair	Un pair contient un ensemble d'ontologies interconnectées	Logique de description distribuée	Le problème du raisonnement avec de multiples ontologies reliées entre elles par des correspondances sémantiques	Un algorithme de tableau distribué, capable de vérifier la conformité des concepts dans un ensemble d'ontologies inter-connectées
Système P2P-DR [Bikakis and Antoniou, 2008]	Pair à pair	Un pair contient une théorie locale défaisable	Logique défaisable	Les communications sans fil sont peu fiables et limitées	Un algorithme P2P-DR dans chaque pair pour résoudre une requête localement ou d'une manière distribuée
Système P2PIS [Adjiman et al., 2006]	Pair à pair	La théorie locale de chaque pair est composée d'un ensemble de clauses propositionnelles définies sur un ensemble de variables propositionnelles	Logique propositionnelle	Le premier algorithme de recherche de conséquences dans un contexte pair-à-pair	Un algorithme P2PIS exécutée d'une manière distribuée
Système DRS [Viterbo and Endler, 2009]	Pair à pair	Deux tiers : un raisonneur utilisateur et un raisonneur ambiant	Langage SWRL	Le calcul du raisonnement peut être trop lourd pour être effectué par une entité centrale	Une approche décentralisée basée sur les règles : deux parties, chacune ayant accès à des informations de contexte différentes, interagissent dans le processus de raisonnement pour trouver un résultat.
Système DDSR [Lam et al., 2012]	Multi-agent	Un agent comporte une théorie défaisable locale	logique défaisable + calcul spéculatif	Les travaux antérieurs sur le raisonnement spéculatif supposaient que les agents étaient structurés de manière hiérarchique	Une approche totalement distribuée où les agents possèdent les mêmes fonctionnalités et collaborent entre eux pour atteindre leurs objectifs communs.
Système DESIA [Maciel et al., 2015]	Multi-agent	CLAIM (langage de programmation orienté agent) : connaissance, buts, messages et capacités	Logique des prédicats du premier ordre	La mise en œuvre de l'intelligence ambiante nécessite le support d'outils et de technologies capables d'interpréter de grandes quantités de données collectées à partir de différentes sources	Une architecture multi-agent incluant les technologies émergentes telles que les réseaux sociaux, l'informatique en nuage et les écosystèmes numériques, en insistant sur la sécurité et la confidentialité.

avons envisagé de proposer une architecture totalement distribué pour la reconnaissance de situations/d'activités. Parmi ces défis, nous citons :

- **La gestion des données de capteurs** : Le système de raisonnement distribué doit gérer le grand volume des données entrantes issues des capteurs.
- **La fraîcheur des données** : Le système de raisonnement distribué doit éviter d'avoir des données capteurs obsolètes.
- **L'identification** : Le système de raisonnement distribué doit étudier et apprendre le comportement de l'habitant pour identifier les situations/d'activités.
- **L'incertitude** : Le système de raisonnement distribué doit faire face à la gestion de l'incertitude lors de la communication entre les sources de données.
- **La gestion de conflits** : Le système de raisonnement distribué doit mettre en place des stratégies de résolution de conflits.

Le Tableau 1.4 (✓ : défi géré ; ✗ : défi non géré) positionne notre solution [Jarraya et al., 2016b, Jarraya et al., 2016a, Jarraya et al., 2017, Jarraya et al., 2018] par rapport aux travaux étudiés précédemment selon les défis cités.

1.7 Conclusion

Dans ce chapitre, nous avons étudié l'état de l'art de l'identification de situations/ d'activités dans un environnement pervasif. Cette étude révèle plusieurs défis de recherche à savoir la gestion de grands volumes de données de capteurs, l'analyse de ces données pour identifier et détecter les situations/les activités. Après une introduction des concepts manipulés et une définition des termes liés au domaine d'études, nous avons introduit le processus d'identification de situations/d'activités. Ce processus consiste en cinq étapes successives qui sont le pré-traitement des données capteurs, la segmentation, l'extraction des attributs, la sélection des attributs et reconnaissance de situations/d'activités. La principale tâche dans cette thèse est la reconnaissance de situations/d'activités. C'est pourquoi nous avons effectué une analyse sur les techniques d'identification existantes qui sont classées en deux grandes familles : orientées connaissances et orientées données. Finalement, nous avons discuté les principales limites de raisonnement dans la reconnaissance de situations/d'activités et qui sont le raisonnement dans un contexte imparfait et distribué. Nous considérons ces deux limites comme étant les objectifs à atteindre de cette thèse.

Dans les chapitres qui suivent, nous présentons l'ensemble de nos contributions concernant les trois étapes du processus de reconnaissance de situations/d'activité à savoir la col-

TABLE 1.4. Positionnement par rapport aux systèmes de raisonnement distribué étudiés

Travaux	Type	Raisonnement	Gestion des données capteurs	Fraîcheur des données	Identification	Incertitude	Gestion de conflits
Système Gaia [Roman et al., 2002]	Basé sur les règles/ Classification	Locale	✓	✓	✓	✗	✗
Système DRAGO [Serafini and Tamilin, 2005]	Basé sur les ontologie	Distribuée	✗	✗	✗	✗	✗
Système P2P-DR [Bikakis and Antoniou, 2008]	Basé sur les règles	Distribuée	✗	✗	✓	✓	✓
Système P2PIS [Adjiman et al., 2006]	Basé sur les règles	Distribuée	✗	✗	✗	✗	✗
Système DRS [Viterbo and Endler, 2009]	Basé sur les règles	Deux-Tiers	✗	✓	✗	✗	✗
Système DDSR [Lam et al., 2012]	Basé sur les règles	Distribuée	✗	✓	✗	✓	✓
Système DESIA [Maciel et al., 2015]	Basé sur les règles	Distribuée	✓	✓	✓	✗	✗
Notre solution [Jarraya et al., 2016a, Jarraya et al., 2016b, Jarraya et al., 2017, Jarraya et al., 2018]	Classification	Distribuée	✓	✓	✓	✓	✓

lecte des données de capteurs, la sélection des attributs et l'identification. Nous commençons tout d'abord par la première contribution pour la phase de collecte des données de capteurs.

Chapitre 2

Perception : un modèle de traitement des évènements complexes sémantiques flous (l'approche FSCEP)

Sommaire

2.1 Introduction

Avec l'émergence de l'Internet des Objets, la notion des maisons intelligentes devient de plus en plus populaire. L'objectif principal de ce chapitre est de mettre en place un système orienté évènements dans une maison intelligente. Ce système détecte les changements d'environnement et renvoie des signaux/des données brutes issues de différents capteurs déployés en utilisant la technique *traitement des évènements complexes* (Complex Event Processing CEP) [Luckham, 2008]. Ainsi, cette technique permet de transformer ces signaux/données brutes en des évènements simples bas niveau puis en des évènements plus complexes de haut niveau d'abstraction. Ces évènements peuvent ensuite être utilisés par plusieurs applications dédiées aux maisons intelligentes ayant pour but l'identification de situation ou d'activité particulière. Cependant, dans la vie réelle, les évènements de bas niveau sont généralement incertains. Nous présentons dans ce chapitre un nouveau modèle de perception nommé *modèle de traitement d'évènements complexes sémantiques flous (FSCEP)* permettant la gestion des évènements incertains. Le chapitre commence par introduire la nécessité d'avoir des systèmes de perception orientés évènements en présentant les défis à relever puis évoque les travaux connexes relatifs. Nous détaillons notre principale contribution *FSCEP* pour la perception ainsi que son évaluation [Jarraya et al., 2016b, Jarraya et al., 2016a]. Nous terminons ce chapitre par une discussion et une synthèse pour positionner notre approche par rapport aux défis définis au préalable.

2.2 Les systèmes orientés évènements et les défis relevés

L'évolution de l'Internet des Objets (Internet Of Things IoT), répondant aux besoins des utilisateurs, offrant des services n'importe quand et n'importe où, a donné naissance au domaine des maisons intelligentes (Smart Homes) ou des systèmes domestiques automatisés (Automated Home Systems) [Kramp et al., 2013]. Une maison intelligente est maison équipée par des technologies sophistiquées qui permettent aux utilisateurs d'avoir un contrôle ou de surveiller les différents appareils électroniques par le biais d'une seule commande ou en appuyant sur un bouton. Le développement des maisons intelligentes est basé sur l'exploitation des signaux/des données brutes issues des différents types de capteurs déployés. Ces signaux/données brutes sont à analyser pour réagir face aux situations/activités identifiées et avoir les actions désirées. Le stockage et la gestion de cette grande quantité de données est un grand défi pour les maisons intelligentes. Pour répondre aux besoins de traitement continu

des données en temps réel, il est nécessaire de concevoir une architecture de traitement de flux de données permettant de détecter les changements contextuels. Le traitement d'évènements complexes (CEP) traite le flux de données continu et fournit une réponse en temps réel aux évènements détectés. Le CEP est basé sur le constat que dans de nombreux cas, les actions ne sont pas activées par un seul évènement mais par une composition d'évènements qui se produisent à différents moments et dans différents contextes. Une vue d'ensemble de haut niveau du CEP dans les maisons intelligentes est illustrée dans la Figure 2.1. Le moteur de traitement des évènements complexes CEP est le module de traitement central qui détecte les changements des flux de données à partir des capteurs déployés et génère une liste d'évènements complexes pour une utilisation ultérieure dans différentes applications IoT. Nous pouvons citer comme exemple la localisation d'un habitant qui peut être déduite grâce au CEP et ensuite utilisée pour l'identification de son activité courante.

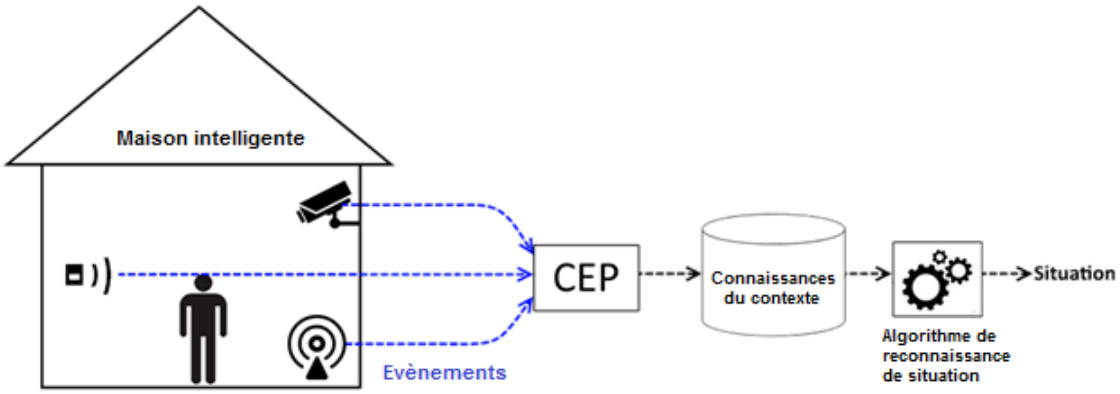


FIGURE 2.1. CEP dans les maisons intelligentes

Toutefois, comme déjà expliqué précédemment, l'incertitude représente une grande problématique dans les maisons intelligentes. En effet, les données de capteurs peuvent être incertaines (erronées, manquantes, contradictoires ou obsolètes). Par exemple, un thermomètre peut ne pas être précis ou un capteur de mouvement peut être déclenché de manière intempestive. Dans de tels cas, l'incertitude peut conduire à la production d'évènements complexes eux-mêmes incertains qui peuvent causer des erreurs de traitement. Par exemple, si le CEP renvoie que l'utilisateur est dans la pièce A alors qu'il se trouve dans la pièce B, une erreur de reconnaissance est alors générée.

Il existe plusieurs définitions de l'incertitude dans la littérature (discuté dans la section 1.6.1.1). Dans cette thèse, nous avons opté pour la définition de Ye et ses co-auteurs proposée dans [Ye et al., 2012]. Ces derniers décomposent l'incertitude en cinq dimensions : la fraîcheur, l'imprécision, l'inconsistance, la contradiction et l'incomplétude des données

capteurs. Nous nous sommes limités aux quatre premières dimensions de l'incertitude : **(1) Fraîcheur :** si une donnée est trop ancienne, elle peut être périmée ; **(2) Imprécision :** les données imprécises sont correctes, mais inexactes ; **(3) Inconsistance :** une donnée inexacte est complètement ou partiellement erronée ; **(4) Contradiction :** une contradiction se produit lorsque deux données provenant de deux sources différentes, sont opposées. La dimension *incomplétude* ne fait pas partie des objectifs de ce travail.

L'incertitude n'est généralement pas abordée dans les études CEP existantes. Peu de travaux s'attaquent à ce défi se limitant souvent à la gestion d'un sous-ensemble des dimensions discutées ci-dessus. Dans ce travail, nous avons considéré les quatre dimensions d'incertitude comme des exigences à atteindre. L'objectif principal de ce travail est de proposer un nouveau CEP appelé *FSCEP* (Fuzzy Semantic Complex Event Processing) qui assure chacune de ces exigences. Il s'appuie sur un fond de connaissances sémantiques combiné avec l'aspect flou. Notre contribution peut être résumée selon les quatre exigences définies comme suit :

1. **Exigence de fraîcheur** : Cette exigence est satisfaite en temps réel. En effet, le CEP ne peut prendre en compte que les évènements déclenchés dans une fenêtre temporelle définie par une requête, assurant ainsi la fraîcheur des évènements.

2. **Exigence de précision** : grâce aux processus de l'enrichissement sémantique et de la fuzzification, les évènements sont respectivement enrichis par des connaissances sémantiques fournies par une ontologie et des valeurs floues. Cet enrichissement nous permet d'affiner la précision des évènements.

3. **Exigence de consistance** : en attribuant à chaque capteur une valeur de confiance fournie par un expert, les évènements sont donc pondérés avec un degré de confiance. Ainsi, les évènements fournis par un capteur, ayant une valeur de confiance au dessous d'un seuil défini, sont considérés comme des évènements inconsistants.

4. **Exigence de contradiction** : le processus de fuzzification d'évènements nous permet de fournir des données avec plusieurs interprétations valides, qui peuvent être contradictoires au début.

Avant d'entamer la description détaillée de notre approche, la section suivante évoque les travaux récents proposés dans la littérature pour la gestion de l'incertitude dans les systèmes orientés évènements.

2.3 Préliminaires

Cette section introduit les notions basiques du CEP et des définitions extraites depuis [Luckham, 2008] qui seront utilisées tout au long de ce chapitre.

L'objectif principal du CEP est d'assurer un traitement en ligne des données. En effet, c'est une méthode de gestion de flux de données provenant de plusieurs sources. Il vise à analyser au fur et à mesure des données brutes, au moment où elles sont détectées ou mesurées, par des requêtes pré-établies dont le but est de filtrer, de combiner et de transformer les évènements bas niveau en des évènements de haut niveau. Par exemple, s'il y a un évènement provenant d'un capteur de mouvement dans la salle A suivi d'un évènement provenant d'un capteur de mouvement dans une salle B, le CEP peut générer un évènement signalant que l'utilisateur se déplace de la salle A vers la salle B. Dans le CEP, ce traitement est défini par un expert via des requêtes. Ces requêtes permettent de collecter les évènements entrants suivant une fenêtre glissante et de les filtrer en appliquant des conditions. Les évènements sont issus d'un contexte d'environnement, dans notre cas, c'est la maison intelligente.

Définition 2.3.1. Le contexte (*context*) est l'ensemble des données décrivant un environnement (tel qu'une maison intelligente dans notre cas) fourni par des capteurs. Par exemple, cela peut inclure la température ou la localisation de l'utilisateur.

Définition 2.3.2. Un évènement (*event*) est un changement d'état d'une donnée contextuelle se produisant à un moment précis (e.g. lorsqu'une mesure dépasse un seuil prédéfini de temps ou de température).

Le CEP considère deux types d'évènements : simples et complexes. Les évènements simples peuvent être vus comme des évènements bruts n'ayant aucun traitement alors que les évènements complexes sont le résultat de l'analyse des évènements simples.

Définition 2.3.3. Un évènement simple (*simple event*) est fourni par une source externe du CEP. Il peut être vu comme un évènement brut fourni par un capteur. Les données d'un évènement simple sont généralement basiques (e.g. un évènement simple peut être un signal issu d'un capteur de mouvement).

Définition 2.3.4. Un évènement complexe (*complex event*) résulte de l'abstraction d'autres évènements. Dans le CEP, les évènements complexes sont souvent la sortie des requêtes définies par un expert et sont généralement de haut niveau par rapport aux évènements simples (e.g. un évènement complexe peut être la localisation d'un utilisateur déterminée via une requête appliquée sur plusieurs évènements de capteurs de mouvement).

Définition 2.3.5. Un flux d'évènements (*event stream*) est une séquence temporelle et linéaire d'évènements ordonnés. Cette séquence peut être bornée et contient des évènements de différents types.

Définition 2.3.6. Un collecteur d'évènements (*Event Sink*) : est un système qui reçoit des évènements.

Définition 2.3.7. Une source d'évènements (*Event source*) : est un système qui envoie des évènements.

Le CEP fonctionne en observant un ensemble d'évènements simples ou primitifs se produisant dans l'environnement externe suite aux changements contextuels. Ces évènements sont interprétés et combinés pour identifier les évènements complexes de haut niveau d'abstraction. Ainsi, ces systèmes envoient des notifications aux autres systèmes chargés de réagir.

Les domaines d'application sont divers et variés : les réseaux de capteurs pour la surveillance de l'environnement [Broda et al., 2009], les applications financières nécessitant une analyse continue des stocks pour détecter les mouvements [Demers et al., 2006] et les applications de détection de fraude, qui observent le flux de transactions des cartes de crédit pour prévenir les fraudes [Schultz-Møller et al., 2009].

Dans la section suivante, nous présentons certains travaux récents.

2.4 L'incertitude dans les systèmes orientés évènements

Les systèmes de traitement d'évènements complexes acceptent en entrée un flux de données brutes (e.g des signaux de capteurs), les transforment en des évènements simples horodatés, appliquent un traitement sur ces évènements et génèrent un ensemble d'évènements complexes sous certaines conditions. En raison de la nature complexe des sources de données et de l'hétérogénéité des capteurs, les évènements d'entrée arrivant à un système CEP sont incertains. Les réseaux de capteurs introduisent une incertitude dans le système pour des raisons allant de mesures inexactes à des défaillances locales. Toutefois, la principale préoccupation des travaux proposés autour du CEP [Zhang et al., 2014, Wu et al., 2006, Brenna et al., 2007, Wang et al., 2006] est la performance en termes de gestion de flux de données entrant comme souligné par [Cugola et al., 2015]. Peu de travaux abordent le problème de l'incertitude des données capteurs. La gestion de l'incertitude dans le CEP est indispensable dans la vie réelle. Ainsi, nous présentons quelques travaux récents traitant une partie des dimensions de l'incertitude.

[Wasserkrug et al., 2012] proposent un modèle pour raisonner avec des règles incertaines lors de la génération des évènements complexes. Ils abordent deux types d'incertitudes : l'imprécision des évènements et les relations incertaines entre les évènements. Pour ce faire, en se basant sur la théorie probabiliste, les auteurs ont défini un espace de probabilités et se sont appuyés sur l'utilisation des réseaux bayésiens générés dynamiquement pour déterminer les probabilités pertinentes. C'est l'un des premiers travaux traitant l'incertitude dans les CEP. Cependant, par rapport à nos exigences définies au début, ils ne s'attaquent qu'à la dimension *imprécision*.

[Artikis et al., 2012] étudient les différents types d'incertitude rencontrés dans les applications de CEP et discutent des implications sur la représentation et le raisonnement des évènements. Ils identifient les dimensions suivantes de l'incertitude :

1. Flux d'évènements incomplets : certains évènements ne sont pas déclenchés lorsqu'ils le devraient. Considérons, par exemple, le cas où une caméra ne parvient pas à détecter une activité humaine à un moment donné (par exemple, en raison d'une occultation. Cette dernière se produit si la personne est masquée par un autre objet. Comme deux personnes qui se croisent).
2. Dictionnaire des évènements insuffisants : certains types d'évènements peuvent ne pas être pris en compte.
3. Reconnaissance erronée des évènements : cela correspond à notre définition de l'imprécision.
4. Annotation d'évènements incohérents : l'annotation des évènements complexes dans le jeu de données d'apprentissage est inconsistant.
5. Modèles d'évènements imprécis : imprécision de la requête

Les auteurs traitent deux dimensions d'incertitude à savoir l'imprécision et l'inconsistance. Cependant, la contradiction et la fraîcheur des évènements n'ont pas été gérées.

[Cugola et al., 2015] proposent un nouveau modèle pour traiter l'incertitude dans le traitement des évènements complexes *CEP2U (Complex Event Processing under Uncertainty)*. CEP2U considère deux types d'incertitude, à savoir l'incertitude dans les données provenant des sources, et l'incertitude dans l'étape d'induction qui dérive des évènements composites des évènements primitifs. La première forme d'incertitude modélise l'imprécision inhérente des données collectées auprès des sources. Un exemple de ce type d'incertitude est l'erreur introduite par un ensemble de capteurs répartis qui mesurent la température et l'humidité dans une grande zone pour les prévisions météorologiques. CEP2U permet de représenter

TABLE 2.1. Les approches CEP avec l'incertitude

Travaux	Dimensions de l'incertitude			
	Fraîcheur	Imprécision	Inconsistance	Contradiction
[Wasserkrug et al., 2012]	✗	✓	✗	✗
[Artikis et al., 2012]	✗	✓	✓	✗
[Cugola et al., 2015]	✗	✗	✓	✗
[Lee and Jung, 2017]	✓	✗	✗	✗
Notre approche *FSCEP*	✓	✓	✓	✓

une telle incertitude et de la propager dans les évènements composites générés, par exemple la prévision de la météo. La seconde forme d'incertitude modélise l'imprécision des règles, c'est-à-dire la possibilité que les règles ne reflètent pas complètement le comportement de l'environnement surveillé. Pour modéliser l'incertitude, CEP2U utilise la théorie de probabilités pour les évènements et les réseaux bayésiens (BN) pour les règles. Chaque évènement est associé à une fonction de distribution de probabilité. D'un autre côté, les règles sont liées à un BN.

[Lee and Jung, 2017] vise à aborder le problème de changement dynamique de l'environnement. Cela peut être vu comme un problème de fraîcheur des données, mais c'est appliqué sur les règles. Pour surmonter cela, ils ont proposé un outil connu sous le nom de génération de règles automatisées basées sur le clustering de séquences (SCARG Sequence Clustering-based Automated Rule Generation) qui génère des règles en analysant l'historique de la prise de décision. SCARG repose sur quatre étapes. Premièrement, il collecte des échantillons de séquences d'évènements à partir d'un expert. Ensuite, il calcule un modèle de clustering grâce au clustering de séquence. Une fois les clusters définis, chaque cluster est modélisé graphiquement à l'aide du modèle MPTM (Markov Probability Transition Model). De ce fait, SCARG est capable de prendre en charge les changements dynamiques de l'environnement. Bien qu'ils ne traitent pas l'incertitude des données provenant des capteurs, leur approche peut être appliquée dans les maisons intelligentes qui sont également soumises aux changements environnementaux.

Le tableau 2.1 (✓ : dimension gérée, ✗ : dimension non gérée) résume ces approches CEP et met en relief notre proposition par rapport à elles selon les défis fixés dans la section 2.2.

La gestion de l'incertitude dans le CEP est abordée de différentes manières dans les travaux discutés ci-dessus. Cependant, ces travaux n'ont pas géré toutes les dimensions de l'incertitude à la fois. Il serait intéressant de gérer toutes les dimensions de l'incertitude des évènements à savoir la fraîcheur, l'imprécision, l'inconsistance et la contradiction des évènements. Ceci est réalisé en combinant le moteur CEP (pour garantir la fraîcheur), l'aspect

sémantique (pour garantir la consistance) et la logique floue (pour gérer l'imprécision et la contradiction). *FSCEP* représente notre nouveau modèle CEP. Avant d'aborder *FSCEP*, nous présentons le scénario suivant qui illustre les quatre exigences dans la gestion des évènements.

2.5 Scénario d'application

Monika vit toute seule dans une maison intelligente équipée de plusieurs types de capteurs connectés. L'environnement déploie des caméras, des capteurs de pression, des capteurs de mouvement et des estimotes (des capteurs de proximité). En particulier, le salon et le bureau, possède chacun des estimotes, des capteurs de pression, un capteur de mouvement et une caméra. Monika travaille dans son bureau et est détectée par la caméra. Cependant, elle a laissé son téléphone dans le salon. Le téléphone est détecté à l'aide d'un estimote en envoyant un signal. La fenêtre du salon est ouverte et le vent fait bouger le rideau. Ainsi, le capteur de mouvement détecte un mouvement et envoie un signal.

Plusieurs évènements sont reçus à partir des capteurs :

— Dans le salon : un évènement provient du téléphone et trois évènements provient du capteur de mouvement. La caméra n'envoie aucun évènement.

— Dans le bureau : un évènement provient du capteur de pression (lié à la chaise), un évènement provient du capteur de mouvement et de la caméra.

Dans ce cas, il est difficile de détecter la localisation de Monika. Deux interprétations de sa localisation sont possibles à partir des évènements déclenchés : elle peut être soit dans le salon ou bien dans son bureau.

Avec un CEP classique, une seule interprétation est générée dans la sortie de l'évènement complexe. Le choix de celui-ci est aléatoire et peut conduire à une mauvaise identification de l'activité en cours de Monika.

En utilisant des évènements complexes sémantiques flous (FSCE Fuzzy Semantic Complex Event), aucune interprétation possible n'est éliminée. Dans ce cas, Monika est localisée dans le salon **ET** dans le bureau, avec un degré de confiance pour chacun. Le *FSCEP* renvoie un évènement FSCE portant la localisation de Monika avec deux interprétations possibles accompagnées par deux degrés de confiance, respectivement. Comme les données de capteurs sont non précises et parfois erronées, nous proposons d'affecter une valeur de confiance pour chaque type de capteur (e.g. 0.8 pour les caméras). Ce poids révèle le niveau de confiance qu'on peut donner au type de capteur. Nous prenons le cas d'un capteur de mouvement qui

peut envoyer un signal alors qu'il n'y avait pas de mouvement, par exemple en raison d'un changement de lumière intempestif. Ce poids est défini par un expert et détermine à quel point un capteur est fiable. Il est ensuite utilisé par le *FSCEP* pour déterminer les degrés de confiance des évènements complexes sémantiques flous (FSCE). Par exemple, la caméra est la plus fiable des autres capteurs.

Dans ce qui suit, nous entamons la présentation détaillée de l'approche *FSCEP*.

2.6 Le modèle de perception FSCEP

Le modèle *FSCEP* est un modèle de perception permettant la collecte des évènements bas niveau issus des capteurs déployés dans une maison intelligente, le traitement de ces évènements et la génération des évènements haut niveau complexes sémantiques flous FSCEs. L'architecture globale du modèle est illustrée dans la Figure 2.2, elle est composée de trois principaux modules :

— Détection : elle est dédiée à la collecte des données brutes à partir des capteurs déployés dans l'environnement.

— Perception : elle correspond au traitement des données capteurs brutes renvoyées par la phase détection, fournissant une opération d'abstraction supplémentaire visant à transformer les données capteurs bas niveau en des données de haut niveau d'abstraction.

— Application : c'est l'exécution d'un ensemble d'actions en fonction des résultats de la perception. Les applications pervasives doivent être constamment adaptées à des changements de contexte hautement dynamiques et sont conçues pour aider les utilisateurs dans leur vie quotidienne. Les phases de détection et de perception constituent des phases primordiales et leurs sorties servent d'intrants à la couche application, puisqu'elles fournissent des informations de contexte de haut niveau d'abstraction.

Ces trois modules sont détaillés dans les sections suivantes.

2.6.1 Le module de détection

La phase de détection vise à détecter les signaux ou les données brutes fournis par différents types de capteurs déployés dans l'environnement. Ces capteurs sont des dispositifs capables de détecter les changements contextuels. Chaque capteur est responsable de la collecte de diverses données contextuelles, telles que la localisation ou la température de l'uti-

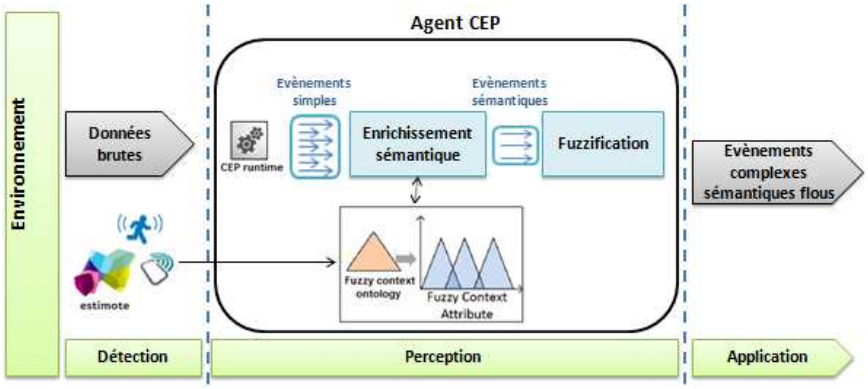


FIGURE 2.2. Approche CEP sémantique floue (*FSCEP*) pour la perception de l'environnement

lisateur. Les caméras, les thermomètres ou les capteurs de mouvement sont des exemples typiques.

Nous distinguons deux types de capteurs de l'environnement :

— *Capteurs actifs (push mode)* : envoient une donnée ou un signal au système. C'est généralement le cas des capteurs orientés évènements, tels que les capteurs de mouvement.
— *Capteurs passifs (pull mode)* : qui sont invités à fournir une donnée. En d'autres termes, les données sont fournies à la demande. C'est typiquement le cas des capteurs qui analysent en continu le contexte tels que les thermomètres.

Dans notre scénario, nous avons utilisé des capteurs actifs, à savoir des capteurs de mouvement, des capteurs de pression et des estimotes. Nous avons aussi utilisé un capteur de vision qui est la caméra. Chaque évènement de capteur indique un changement d'état contextuel qui doit être identifié. Il est modélisé par un *horodatage* (représenté en millisecondes) utilisé pour établir des relations temporelles entre les différents évènements. De plus, un évènement porte les propriétés suivantes : un *idSensor* qui est l'identifiant du capteur ; un *typeOfSensor* décrivant le type du capteur (e.g. mouvement, caméra ou téléphone) ; un *status* indiquant l'état du capteur (ON/OFF) et une valeur représentant la valeur demandée (cette propriété concerne les capteurs passifs). Un exemple d'écoute d'évènements capteurs est illustré dans le Tableau 2.2. Un évènement est déclenché lorsque son état ou sa valeur est modifiée.

TABLE 2.2. Un exemple d'évènements capteurs horodatés

Timestamp	IdSensor	typeOfSensor	Status	Value
2016-04-08 14 :46 :05	Motion1	Motion	ON	-
2016-04-08 15 :10 :17	MonikaNokia	Phone	ON	-
2016-04-08 20 :37 :20	Therm1	Thermometer	ON	25

2.6.2 Le module de perception

Ce module constitue notre principale contribution *FSCEP*. L'architecture globale de *FSCEP* consiste en un ensemble de nœuds collecteurs assignés aux différents capteurs déployés dans la maison intelligente. Le paradigme des systèmes multi-agents est choisi pour modéliser l'ensemble de ces nœuds comme étant des *agents collecteurs*. Les systèmes multi-agents sont bien adaptés pour la modélisation des maisons intelligentes [Maciel et al., 2015]. Les agents collecteurs sont chargés de détecter les signaux et collecter les données brutes à partir des différents types de capteurs (actifs ou passifs), de les transformer en des évènements simples, de les traiter pour générer en sortie des évènements complexes sémantiques flous (FSCEs). Des agents collecteurs multiples sont déployés pour observer différentes données contextuelles (e.g. la localisation, la température, la pression, etc.). Chaque agent est associé à un ou plusieurs capteurs qui observent la même donnée contextuelle. Considérons l'exemple d'une donnée contextuelle telle que la localisation d'un utilisateur, l'agent collecteur peut être relié à trois capteurs pour observer sa position (comme décrit dans le scénario d'application Section 2.5) à savoir le capteur de mouvement, l'estimote du téléphone et la caméra (voir Figure 2.3).

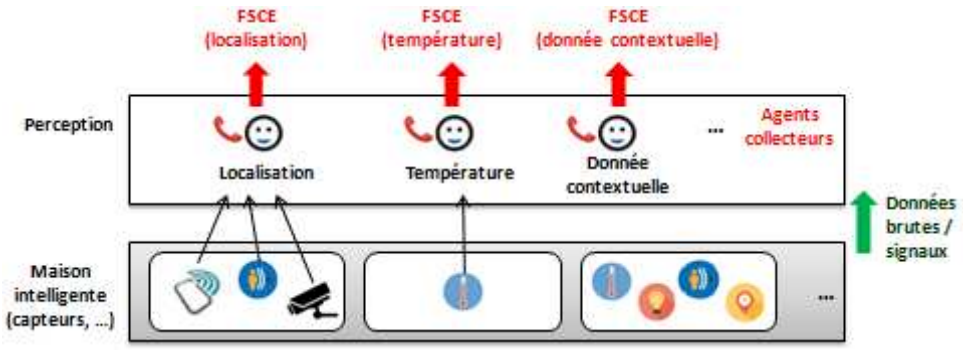


FIGURE 2.3. Un exemple illustré pour la perception de la localisation de l'utilisateur

Un agent collecteur intègre un moteur CEP qui identifie, traite et répond aux évènements discrets provenant de plusieurs capteurs. Afin de traiter une séquence continue de données capteurs, le CEP utilise généralement deux types de fenêtrage coulissant les plus populaires :

les fenêtres coulissantes temporelles et les fenêtres coulissantes d'évènements [Krishnan and Cook, 2014].

Une fenêtre coulissante temporelle a pour but de partitionner une séquence de données capteurs en des intervalles de temps fixe, tandis qu'une fenêtre coulissante d'évènements a pour but de partitionner une séquence de données capteurs en des fenêtres ayant le même nombre d'évènements de capteur. Dans ce travail, nous avons choisi la technique du fenêtrage coulissant temporel.

CEP détecte les signaux émis par les capteurs (e.g changement d'état), les transforment en des évènements simples et les filtrent selon des critères (e.g, type de capteur, type de donnée observée) en utilisant un système de fenêtrage coulissant. Nous pouvons donner à titre d'exemple la requête suivante définie par un expert :

Select events of type == "location" in time_batch (1min) where sourceType == "motionSensor"or sourceType == "camera"

Cette requête permet de sélectionner les évènements liés à la donnée contextuelle "localisation" provenant des capteurs de type capteur de mouvement ou de type caméra pendant une durée d'une minute. Ainsi, CEP génère un ensemble d'évènements simples et filtrés si ces types de capteurs se manifestent dans une fenêtre de 1 min.

Suite à cette détection et génération d'évènements simples, notre contribution commence à enrichir ces évènements par des connaissances sémantiques avec l'étape *enrichissement sémantique des évènements* puis à les fuzzifier avec l'étape *fuzzification* pour générer à la fin un seul évènement complexe sémantique flou FSCE. Les détails de ces deux étapes sont fournis ci-dessous.

2.6.2.1 Enrichissement sémantique d'évènements

L'enrichissement sémantique d'évènements vise à améliorer la sortie du CEP en incorporant une ontologie dans le processus de génération des évènements complexes afin de garantir leurs consistances. Le choix des ontologies n'est pas arbitraire puisque les ontologies représentent un outil puissant pour représenter, stocker les connaissances et raisonner sur celle-ci. Dans notre cas, les évènements CEP vont être enrichis par des connaissances supplémentaires provenant des ontologies. Ces connaissances sémantiques sont rajoutées suite à une mise en correspondance des attributs de l'évènement simple CEP avec des entités sémantiques. Pour résumer, l'enrichissement sémantique d'évènements prend en entrée

les évènements simples CEP et les transforme en des évènements sémantiques CEP. Nous proposons dans la suite des définitions afin de formaliser cette étape.

Définition 2.6.1. *Evènement simple (simple event)*. Nous définissons un évènement simple e comme un vecteur de données brutes de capteurs défini comme suit : $e = \{t, id, type, st, v\}$ où t est l'horodatage, id est l'identifiant du capteur, t est le type du capteur (e.g. capteur de mouvement, caméra ou téléphone), st est l'état du capteur (ON/OFF) et v est la valeur demandée (e.g. une valeur de retour d'un thermomètre).

Définition 2.6.2. *Evènement sémantique (semantic event)*. Un évènement sémantique se est modélisé comme un ensemble de triplets fini (s, p, o) d'un graphe RDF [1] où le sujet, le prédicat, l'objet sont respectivement $s, p, o \in U$ où U est l'ensemble infini des URI [2].

Comme exemple, nous pouvons enrichir l'évènement simple portant l'identifiant du capteur *Motion1* en rajoutant son emplacement $(MotionSensor : Motion1, IsLocatedIn, Location : office1)$ et sa valeur de confiance $(MotionSensor : Motion1, hasTrust, "0.4")$ (voir Figure 2.4).

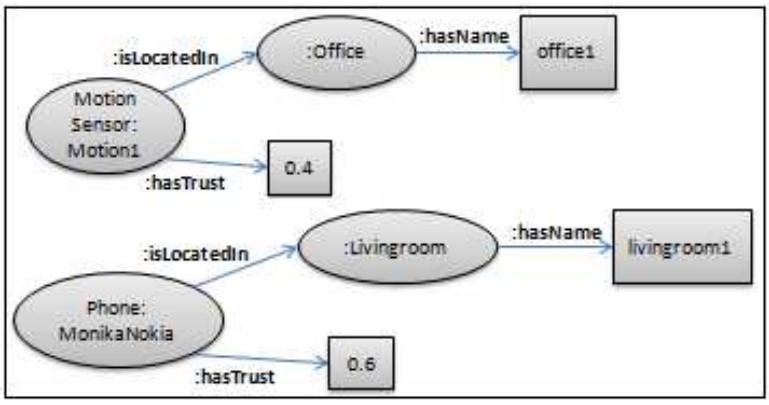


FIGURE 2.4. Enrichissement sémantique des évènements via le graphe RDF du scénario

Définition 2.6.3. Processus d'enrichissement sémantiques des évènements (*Event semantization process*). Le processus d'enrichissement sémantique des évènements (voir Figure 2.5) consiste à enrichir et transformer un évènement simple e_i en un évènement sémantique se_i et ceci s'effectue via l'accès à une ontologie existante.

1. RDF (Resource Description Framework) https ://www.w3.org/RDF/ date de consultation Janvier 2019
2. URI (Uniform Ressource Identifier) pour identifier les ressources : https ://www.w3.org/TR/2004/REC-rdf-concepts-20040210/ date de consultation Janvier 2019

$$EventSemantization(\{e_1, e_2, \ldots, e_i, \ldots e_n\}) \longrightarrow \{se_1, se_2, \ldots, se_i, \ldots se_n\}$$

où $0 < i <= n$ et n est le nombre d'évènements simples dans la fenêtre coulissante temporelle.

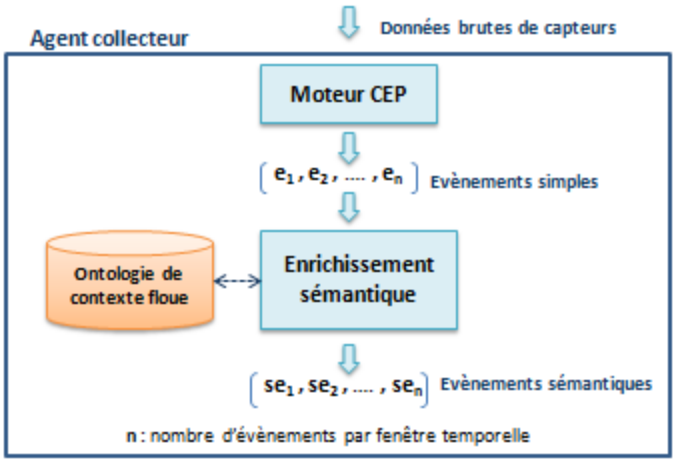


FIGURE 2.5. Processus d'enrichissement sémantique des évènements

Exemple 1

Considérons l'exemple du scénario d'application, un agent collecteur est responsable de la collecte des données contextuelles relatives à la localisation de Monika dans la maison intelligente. Cet agent collecteur intègre un moteur CEP dont la taille de la fenêtre coulissante temporelle est de 5 min. Cette fenêtre comporte deux évènements : celui du capteur de mouvement et celui du téléphone. Suite au processus d'enrichissement sémantique, ces deux évènements seront enrichis par des connaissances supplémentaires récupérées à partir d'une ontologie de contexte : l'emplacement du capteur et la valeur de confiance de ce capteur (voir Figure 2.6).

Dans cette contribution, nous avons utilisé une ontologie de contexte existante pour la représentation de l'activité humaine [3] [Rodriguez et al., 2014]. Cette ontologie a été adaptée selon nos besoins et nos exigences de *FSCEP* comme suit :

— Rajouter plusieurs sous-classes du concept *Sensor* telle que *MotionSensor* (voir Figure 2.7)

— Rajouter la propriété de données *hasTrust* associée aux sous-classes *Sensor*, qui exprime la valeur de confiance d'un type de capteur donné. Par conséquent, un évènement sémantique contient nécessairement une valeur de confiance liée au capteur

3. http ://users.abo.fi/ndiaz/public/FuzzyHumanBehaviourOntology/FuzzyHumanBehaviourV11.owl date de consultation Janvier 2019

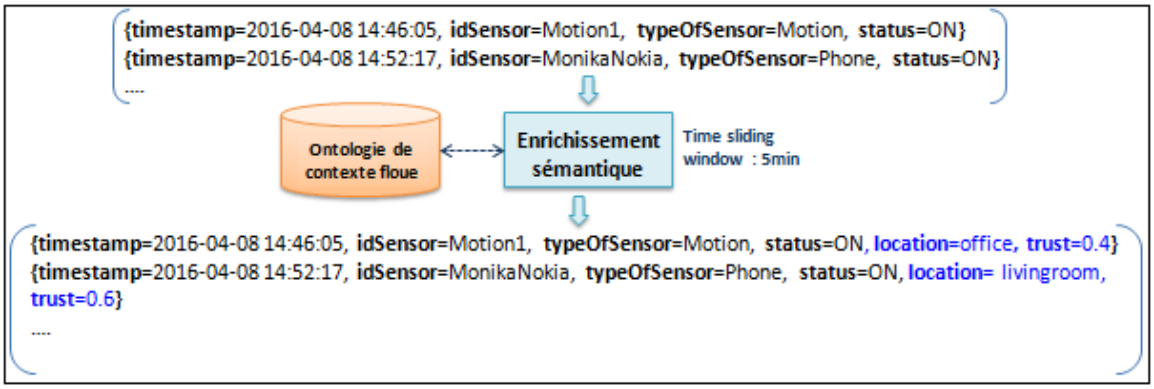


FIGURE 2.6. Un exemple d'enrichissement sémantique des évènements

correspondant. Les valeurs de poids sont statiques dans l'ontologie et sont censées être fournies par un expert du domaine.

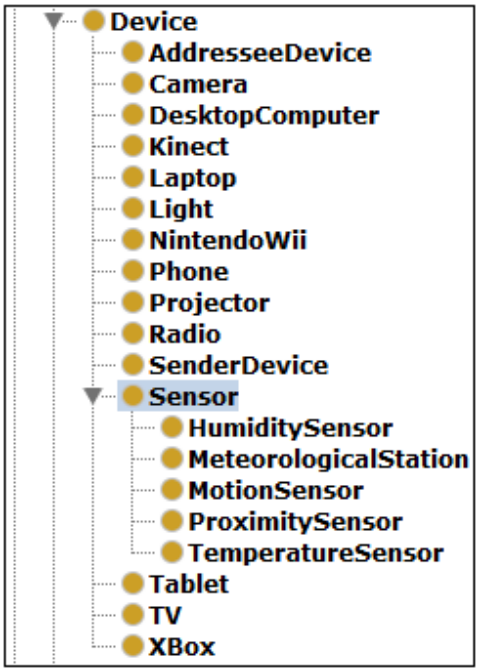


FIGURE 2.7. Extrait des sous-classes *Sensor* dans l'ontologie

Des extraits de la structure de cette ontologie sont disponibles dans l'Annexe C.

Revenons à notre scénario d'application et l'ontologie contextuelle montrée dans la Figure 2.7, un enrichissement sémantique est réalisé en ajoutant des triplets RDF donnant des informations de haut niveau telles que la localisation des capteurs et leurs valeurs de confiance. Par exemple, les évènements reçus à partir du téléphone sont enrichis comme suit :

$(phone : MonikaNokia, islocatedIn, Livingroom)$ et $(phone : MonikaNokia, hasTrust, 0.4)$. Un nouveau graphe RDF est alors construit comme il est illustré dans la Figure 2.4.

2.6.2.2 Fuzzification des évènements sémantiques

La logique floue est largement utilisée pour gérer des connaissances imparfaites, en particulier des connaissances imprécises. Dans l'informatique pervasive, la logique floue est utilisée pour mettre en correspondance les données brutes capteurs et les attributs flous. L'objectif est de palier l'absence de fiabilité des capteurs et l'imprécision des données capteurs. Par conséquent, cette étape vise à déterminer un évènement sémantique flou à partir d'un ensemble d'évènements sémantiques observant la même donnée contextuelle. Ces évènements générés portent alors des valeurs floues pour ces données contextuelles.

Pour appliquer la fuzzification à des évènements sémantiques, nous utilisons le formalisme Fuzzy OWL2. Dans ce formalisme, le flou est modélisé par des annotations. Avec une telle connaissance, il est possible d'utiliser des outils dédiés pour gérer des données imprécises et contradictoires. Dans l'ontologie de [Rodriguez et al., 2014], les auteurs considèrent que les types de données, les concepts, les propriétés et les relations sont flous. Pour les types de données floues, ils ont défini les fonctions d'appartenance : (a) fonction trapézoïdale ; (b) fonction triangulaire ; (c) fonction de l'épaule gauche ; (d) fonction de l'épaule droite ; et (e) fonction linéaire (voir Figure 2.8).

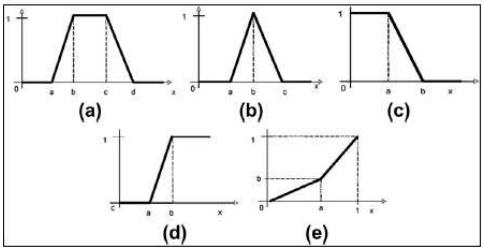


FIGURE 2.8. Fonctions d'appartenance pour le type de données flou

Définition 2.6.4. *Evènement complexe sémantique flou (Fuzzy semantic complexe event FSCE).* Pour une donnée contextuelle observée *cd*, un évènement complexe sémantique flou $FSCE_{cd}$ est une extension du graphe RDF de l'évènement sémantique *se* avec une donnée sémantique additionnelle qui est la valeur floue.

Un $FSCE_{cd}$ est un évènement ayant plusieurs interprétations avec une valeur floue pour une donnée contextuelle observée. En particulier, un $FSCE_{cd}$ présente une ou plusieurs valeurs détectées pour des données contextuelles observées avec des poids de confiance. Les

données contextuelles observées peuvent être la pression, la présence de l'utilisateur, la température, etc.

Définition 2.6.5. *Processus de fuzzification (Fuzzification process).* Le processus de fuzzification (cf. Figure 2.9) peut être vu comme une fonction qui prend en entrée une liste d'évènements sémantiques $\{se_1, se_2, \ldots, se_n\}$ et génère un $FSCE_{cd}$ pour une donnée contextuelle observée.

$$Fuzzification(\{se_1, se_2, \ldots, se_n\}) \rightarrow FSCE_{cd}$$

Pour générer un $FSCE_{cd}$, nous utilisons une fonction d'appartenance dédiée MF_{cd}. Cette fonction utilise comme entrées les valeurs détectées pour la donnée contextuelle observée et l'ensemble des évènements sémantiques correspondant à cette valeur. Si nous considérons que la donnée contextuelle est la présence de l'utilisateur et donc les valeurs détectées peuvent être *bedroom*, *kitchen*, etc. Par conséquent, un poids de confiance est attribué pour chaque valeur détectée, ce qui signifie que la donnée contextuelle a été observée pour différentes valeurs avec un certain poids. Par exemple, un $FSCE_{presence}$ peut être de la forme $(user, isLocatedIn, \{kitchen\ w1; office\ w2\})$ ce qui signifie que l'utilisateur est présent dans *kitchen* avec un poids $w1$ et présent dans *office* avec un poids $w2$. Le processus de fuzzification appelle la fonction d'appartenance pour toutes les valeurs détectées.

Soit MF_{cd} la fonction d'appartenance :

$$WT_{val} = MF_{cd}(SE_{cd}, val)$$

où SE_{val} est un ensemble d'évènements sémantiques, *val* est une valeur détectée commune pour ces évènements et WT_{val} est la confiance pondérée déterminée pour *val*. MF_{cd} est la somme pondérée des poids de confiance de ces évènements sémantiques correspondant à la valeur *val* pour une donnée contextuelle observée. En effet, cette fonction utilise les poids de confiance des capteurs définis dans l'ontologie. La fonction d'appartenance MF_{cd} est indiquée ci-dessous :

$$MF_{cd}(SE_{cd}, val) = [\sum_{i=1}^{n}(\alpha_i)]/[\sum_{j=1}^{N}(\alpha_j)]$$

où N est le nombre total d'évènements pour une donnée contextuelle observée *cd* dans la fenêtre coulissante temporelle ; n est le nombre total d'évènements par *val* pour une donnée contextuelle observée *cd* et α est la valeur de confiance d'un capteur appartenant à SE_{cd}. Le $FSCE_{cd}$ déterminé par la fonction de fuzzification peut alors être défini comme suit :

$$FSCE_{cd} = \cup\{val, MF_{cd}(SE_{cd}, val)\}\ \forall val \in VAL$$

où VAL est l'ensemble de toutes les différentes valeurs possibles pour une donnée contextuelle observée cd.

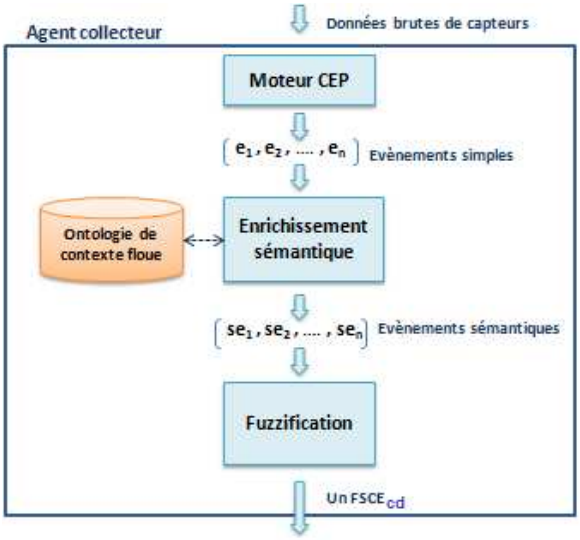


FIGURE 2.9. Processus d'enrichissement sémantique des évènements

Exemple 2

Considérons l'exemple de la Figure 2.10, nous nous intéressons à l'observation de la présence de l'utilisateur. Étant donnée une fenêtre coulissante temporelle (intervalle de 5 min) contenant trois évènements capteurs, chacun de ces évènements est enrichi avec des connaissances sémantiques qui sont la position du capteur en question et sa valeur de confiance, suite à l'étape *enrichissement sémantique*. Ensuite, ces évènements sémantiques vont subir la phase de fuzzification qui génère en sortie un seul évènement complexe sémantique flou FSCE par donnée contextuelle observée comme suit :

1)- $MF_{(presence)}(SE_{(presence)}, office) \Rightarrow WT_{(office)} = 0.6 + 0.4/(0.6 + 0.4 + 0.8) = 0.55$

2)- $MF_{(presence)}(SE_{(presence)}, livingroom) \Rightarrow WT_{(livingroom)} = 0.8/(0.6 + 0.4 + 0.8) = 0.45$

$=> FSCE_{(presence)} = MF_{(presence)}(SE_{(presence)}, livingroom) \cup MF_{(presence)}(SE_{(presence)}, office)$
$= \{[livingroom; 0.55], [office; 0.45]\}$

Exemple 3

Considérons le scénario d'application, nous supposons que nous avons plusieurs évènements sémantiques déclenchés :

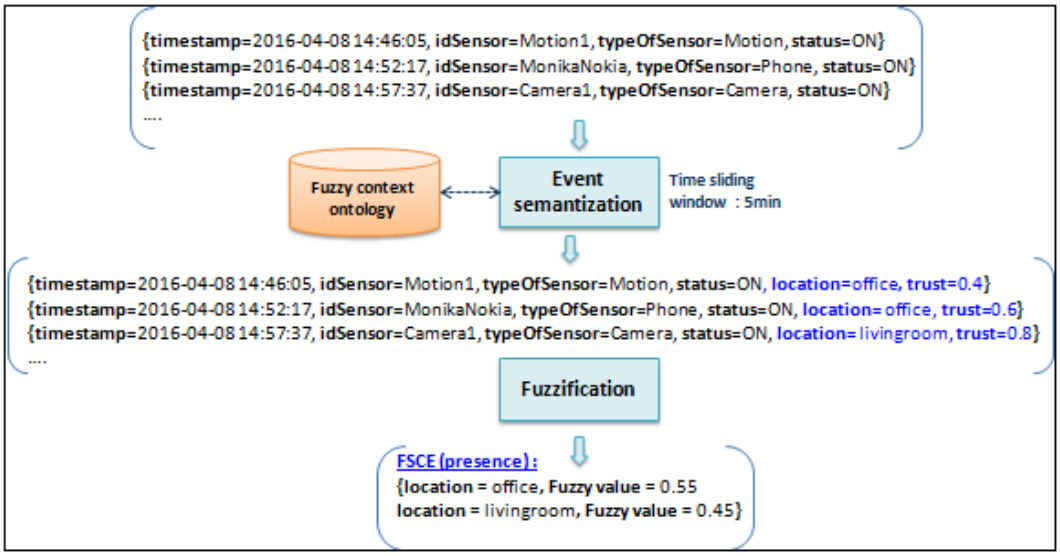


FIGURE 2.10. Un exemple de fuzzification des évènements sémantiques

— Dans le salon : un évènement sémantique provenant de l'estimote du téléphone (degré de confiance : 0,6), deux évènements sémantiques provenant du capteur de mouvement (degré de confiance : 0,4).
— Dans le bureau : un évènement sémantique provenant du capteur de pression (degré de confiance : 0,6), un évènement sémantique provenant du capteur de mouvement et un évènement sémantique provenant de la caméra (degré de confiance : 0,8).

Ensuite, nous appliquons le processus de fuzzification et nous générons un seul FSCE la donnée contextuelle observée qui est la présence de l'utilisateur comme suit :

$$FSCE_{presence} = \{[livingroom; 0.43], [office; 0.57]\}$$

Les poids de confiance $WT_{(livingroom)}$) et $WT_{(office)}$ sont déterminés en appliquant la fonction $MF_{presence}$. Pour l'exemple de la présence de l'utilisateur :

$$WT_{livingroom} = MF_{presence}(SE_{presence}, livingroom) =$$
$$[0.6 + (2 \times 0.4)]/[0.6 + (2 \times 0.4) + 0.6 + 0.4 + 0.8] = 0.43$$
$$WT_{office} = MF_{presence}(SE_{presence}, office) = [0.6+0.4+0.8]/[0.6+(2\times 0.4)+0.6+0.4+0.8] = 0.57$$

Dans cet exemple, nous concluons que l'utilisateur est présent dans le salon avec 0.43 de confiance **ET** dans le bureau avec 0.57 de confiance. Dans ce travail, nous avons utilisé MF_{cd} comme une fonction de somme pondérée mais nous pouvons appliquer d'autres fonctions proposées dans la littérature pour calculer les poids de confiance. La fuzzification des valeurs

des données contextuelles observées à travers la fonction d'appartenance MF_{cd} est rajoutée dans le graphe RDF (voir Figure 2.11).

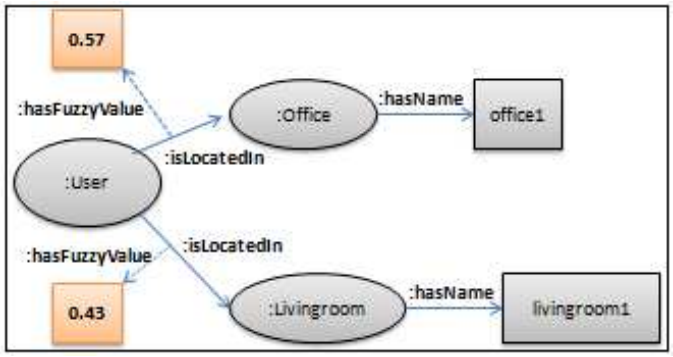


FIGURE 2.11. Un exemple de fuzzification des données contextuelles de la présence de l'utilisateur dans le graphe RDF

2.6.3 Le module d'application

Les phases de détection et de perception constituent des phases primordiales pour collecter et transformer les données brutes capteurs en des données haut niveau d'abstraction. Les applications de l'informatique pervasive utilisent ces données de haut niveau pour fournir aux utilisateurs un ensemble de fonctionnalités appropriées à savoir des recommandations, des alertes ou bien des notifications.

Les applications dans le domaine de l'identification de la situation/de l'activité visent à comprendre ce qui se passe dans l'environnement et à réagir selon la situation/l'activité détectée. Par exemple, si l'utilisateur dort pendant qu'une fuite de gaz se produit, une situation de "danger" peut être détectée et donc une alarme peut se déclencher. L'identification de la situation peut être vue comme une généralisation de la reconnaissance d'activité : elle analyse tous les composants de l'environnement et pas seulement l'activité d'un utilisateur. Détecter des situations/des activité est un défi, c'est encore plus compliqué quand l'incertitude est prise en considération. Fournir des données enrichies, en particulier avec un degré d'incertitude, à une telle application est un atout.

2.6.4 L'algorithme FSCEP

Dans cette section, nous présentons l'algorithme *FSCEP* qui implémente l'approche proposée. L'algorithme *FSCEP* (Algorithme 1) est appliqué à tous les agents collecteurs durant

la phase de perception. Il prend comme paramètres d'entrée : i) un ensemble d'évènements SEQ pour une donnée contextuelle observée existant dans la fenêtre coulissante temporelle, ii) une requête q pour filtrer les évènements simples en évènements batchs, iii) une ontologie floue O pour enrichir les évènements avec des connaissances sémantiques.

L'algorithme *FSCEP* vise à fournir, en sortie, un évènement complexe sémantique flou FSCE. L'algorithme *FSCEP* comprend trois étapes principales :

— Première étape (ligne 6) : chaque agent collecteur intègre un moteur CEP qui reçoit les données brutes/les signaux provenant de l'environnement, les transforme en des évènements simples et applique la requête q définie par l'expert pour générer une liste d'évènements filtrés et batchés L_{BE}.

— Deuxième étape (lignes 7-9) : les évènements simples sont enrichis avec des connaissances sémantiques à savoir la valeur de confiance des capteurs et leurs localisations dans la maison (e.g. chambre, bureau, etc.) et se transforment en graphe RDF. L_{se} contient l'ensemble des évènements sémantiques.

— Troisième étape (lignes 10-13) : pour chaque valeur de données contextuelles observées, nous mesurons la valeur floue en appliquant notre fonction d'appartenance et en annotant le graphe avec cette valeur pour obtenir comme résultat un évènement complexe sémantique flou FSCE.

La complexité en termes de calcul de l'algorithme *FSCEP* est $\mathcal{O}(n)$ où n est le nombre des évènements simples entrants pour une donnée contextuelle observée.

2.7 Implémentation et évaluation

Nous avons implémenté le modèle *FSCEP* et l'avons évalué par simulation. Notre prototype combine un CEP bien connu, ESPER [event stream intelligence, 2010], et une ontologie à travers la plateforme Jena [4]. Afin de mettre en place l'architecture multi-agent, la plateforme JADE [5] a été choisie. Nous avons utilisé une ontologie de contexte flou présentée dans [Rodriguez et al., 2014]. La logique floue est gérée suivant des annotations et l'ontologie est compatible avec un raisonneur flou comme FuzzyDL [6]. Notre prototype utilise l'ontologie comme une boîte en T-box [F. et al., 2010] pour son processus. Le résultat flou de notre *FSCEP* est sauvegardé dans la case A-box [F. et al., 2010] ; l'ontologie floue est ensuite instanciée et peut être utilisée pour d'autres besoins, tels que la reconnaissance d'activité.

4. https ://jena.apache.org/
5. http ://jade.tilab.com/
6. http ://www.umbertostraccia.it/cs/software/fuzzyDL/fuzzyDL.html

Algorithme 1 . L'algorithme *FSCEP* pour une donnée contextuelle observée

Input :
cd : une donnée contextuelle observée $SEQ = \{e_1, e_2, \ldots, e_n\}$: un ensemble d'évènements pour une donnée contextuelle observée cd
q : la requête CEP définie par un expert
O : l'ontologie de contexte floue
Output :
$FSCE$: un évènement complexe sémantique flou (graphe RDF)

begin
  $initialise(L_{BE})$ `// Liste des évènements par fenêtre`
  $initialise(L_{se})$`// Liste des évènements sémantiques`
  $initialise(L_v)$`// Liste des valeurs de la donnée contextuelle observée`

  `// Première étape: détecter et filtrer les évènements avec CEP`
  $L_{BE} = CEPengine(q, SEQ)$
  `// Deuxième étape: enrichissement sémantique`
  for $e_i \in L_{BE}$ **do**
    $< v_{cd_{e_i}}, trust_{e_i} >= queryOntology(O, idSensor_{e_i}, typeOfSensor_{e_i})$
    $graph_{e_i} = semantizeEvent(e_i, v_{cd_{e_i}}, trust_{e_i})$
    $L_{se}.add(graph_{e_i})$
  `// Troisième étape: fuzzification des évènements sémantiques`
  $L_v = loadValuesFrom(L_{se})$
  for $v_i \in L_v$ **do**
    $fv_i = MF_{v_i}(L_{se}, v_i)$
    $FSCE.addAnnotation(v_i, fv_i)$
  $return(FSCE)$

Afin de simuler notre environnement, nous avons utilisé Freedomotic [7], une plateforme de développement utilisée pour la gestion des espaces intelligents. Freedomotic nous permet de contrôler les appareils virtuels et physiques. Il comporte une unité de traitement des évènements basiques, une connaissance des salles et quelques plugins. L'un des objectifs de Freedomotic est de simuler l'environnement avant de le configurer. En effet, les valeurs des capteurs peuvent être facilement simulées. Pour notre expérimentation, nous avons implémenté notre scénario et avons ajouté les capteurs présentés en plugins, à savoir les capteurs de mouvement, les capteurs de pression, les estimotes et les caméras (voir Figure 2.12). Ces appareils nouvellement ajoutés génèrent des évènements et peuvent communiquer avec notre *FSCEP*.

Pour évaluer notre modèle, nous avons mis en place un simple scénario de reconnais-

7. http ://www.freedomotic.com/

FIGURE 2.12. Initialisation de l'interface de Freedomotic

sance d'activité proche de celui décrit dans la section 2.5. Dans ce scénario, l'utilisateur, Monika, travaille dans le bureau et nous cherchons à identifier son activité courante. Pour ce faire, nous devons acquérir des données contextuelles sur sa position actuelle. Le bureau est équipé d'un capteur de pression, d'un capteur de mouvement et d'un estimote lié au téléphone, tandis que le salon dispose d'une caméra et d'un capteur de mouvement. Les valeurs de confiance des capteurs ainsi que leurs localisations sont configurées dans l'ontologie. Dans cette expérience, une activité est reconnue en appliquant des règles simples dans notre ontologie contextuelle (A-Box) alimentée par *FSCEP* ou par le CEP classique : l'objectif de cette simulation est d'évaluer le gain d'utilisation du *FSCEP* par rapport à CEP. Ainsi, nous mesurons le taux de reconnaissance d'activité avec et sans *FSCEP*. Nous présentons quelques exemples de règles utilisées pour identifier l'activité de Monika :

Règle 1 : [ruleWorking : (?u :isLocatedIn :office) (?u :hasStance 'sitting') -> (?u :performsActivity :Work)]

Règle 2 : [ruleEnjoying : (?u :isLocatedIn :livingroom) (?u :hasStance 'standing') -> (?u :performsActivity :WorkBreak)]

Ces deux règles concernent l'identification de deux activités *Work* (règle 1) ou bien *WorkBreak* (règle 2). La variable *u* représente la personne en question.

Pour chaque cas, 500 instances ont été simulées. Une exécution consiste en : détection et

TABLE 2.3. Évaluation de FSCEP par rapport à CEP selon des simulations

[tél, c.mvt1, cam, c.mvt2]	FSCEP		CEP	
	Office	Livingroom	Office	Livingroom
[1,0,1,1]	0.44	0.56	0	1
[1,0,1,0]	0.57	0.43	1	0
[0,1,1,1]	0.28	0.72	0	1
[1,1,1,1]	0.54	0.46	1	0
[0,1,1,0]	0.4	0.6	0	1
[1,1,0,0]	1	0	1	0

génération d'une liste d'évènements, le traitement de ces évènements (avec *FSCEP* ou avec CEP) et la reconnaissance d'activité. Pour modéliser l'incertitude (la consistance), chaque capteur a une probabilité de défaillance. Un exemple de sortie de notre *FSCEP* peut être trouvé dans le Tableau 2.3. Ce dernier montre les valeurs floues déterminées en fonction du déclenchement des capteurs. La première colonne indique l'ensemble des évènements déclenchés (valeur 1 : un évènement généré suite au déclenchement du capteur une seule fois ; valeur 0 : capteur non déclenché) via les capteurs : l'estimote du téléphone *tél* et le capteur de mouvement *c.mvt1* dans le bureau, et la caméra *cam* et le capteur de mouvement *c.mvt2* dans le salon. Prenant le premier exemple, le prototype *FSCEP* génère une valeur floue pour la présence de l'utilisateur. En effet, l'utilisateur est situé dans le bureau avec 0.44 de confiance **ET** dans le salon avec 0.56 de confiance. Selon le CEP classique, l'utilisateur est situé **SEULEMENT** dans le salon. Nous pouvons voir que le *FSCEP* empêche la disqualification de certains évènements, ce qui peut être vrai. Ces résultats sont ensuite utilisés pour reconnaître l'activité de l'utilisateur en cours.

Le taux de reconnaissance des deux activités (cette métrique est définie dans l'annexe A.2) est mesuré avec *FSCEP* et sans *FSCEP* (avec seulement CEP) (voir Figure 2.13). Bien que ce soit un simple scénario, le *FSCEP* montre un taux de reconnaissance meilleur que le CEP avec une différence remarquable de 10%. Cela prouve l'apport de notre modèle *FSCEP* par rapport à CEP.

2.8 Discussion et synthèse

Dans cette section, nous discutons des quatre exigences abordées dans la section Introduction à savoir l'exigence de fraîcheur, de précision, de consistance et de contradiction. Notre approche *FSCEP* gère différentes dimensions de l'incertitude en proposant différentes techniques.

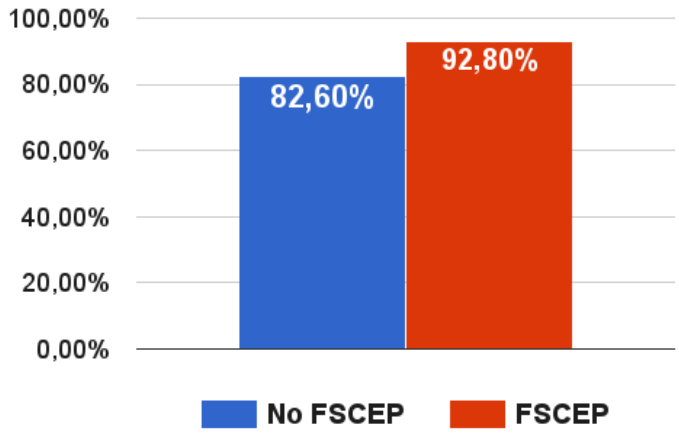


FIGURE 2.13. Taux de reconnaissance d'activité

En effet, nos contributions répondent aux exigences comme suit :

— Exigence de fraîcheur : Pour conserver les données de contexte à jour, nous avons utilisé une technique basée sur la détection d'évènements *Complex Event Processing*. Il est conçu pour traiter les notifications d'évènements de bas niveau afin d'identifier les évènements complexes de haut niveau en fonction d'un ensemble de requêtes définies par l'utilisateur et dans une fenêtre coulissante temporelle. Avec l'exemple de notre scénario, la localisation de l'utilisateur est déterminée immédiatement par la position des capteurs de mouvement, l'estimote lié au téléphone et la caméra lorsqu'ils sont déclenchés.

— Exigence de précision : Les notifications d'évènements pour les données contextuelles observées sont enrichies par des connaissances sémantiques selon le processus d'enrichissement sémantique d'évènements. Par exemple, les notifications d'évènements provenant des capteurs de mouvement, de l'estimote lié au téléphone et de la caméra sont enrichies par les valeurs de confiance et de la position des capteurs correspondants. Ensuite, ils sont fuzzifiés en appliquant notre fonction d'appartenance pour générer une valeur floue pour les données contextuelles observées. Selon le scénario, chaque évènement sémantique aura une valeur floue pour la présence de l'utilisateur observé.

— Exigence de consistance : Nous attribuons une valeur de confiance pour chaque capteur dans l'ontologie de contexte flou. Elle définit le niveau de confiance auquel une donnée de contexte est correcte. Dans le cas de notre exemple, les capteurs de mouvement, l'estimote lié au téléphone et la caméra ont respectivement des valeurs de confiance de 0,2, 0,4 et 0,8. Les évènements sémantiques contiennent des connais-

sances sémantiques incluant la valeur de confiance du capteur correspondant. Par conséquent, les évènements sémantiques avec une valeur de confiance sous un seuil défini sont considérés comme faux.

— Exigence de contradiction : Grâce au processus de fuzzification des évènements sémantiques, la valeur floue générée par la fonction d'appartenance définie, nous permet de fournir des interprétations multiples pour les données contextuelles observées qui permettent de gérer les ambiguïtés et les contradictions. Dans l'exemple 3 (section 2.6.2.2), l'utilisateur est situé dans le salon avec la valeur floue 0.33 et dans le bureau avec la valeur floue 0.67.

2.9 Conclusion

Dans ce travail, nous avons présenté une extension du CEP pour gérer plusieurs dimensions de l'incertitude, à savoir la fraîcheur, la précision, la consistance et la contradiction des évènements. Nous avons proposé un modèle de traitement d'évènements complexes sémantiques flous (*FSCEP*) qui fonctionne en trois étapes. La première étape vise à collecter les données capteurs, les transformer en des évènements simples, les filtrer et les regrouper en utilisant un CEP classique. Ces évènements simples sont ensuite enrichis par des connaissances sémantiques via une ontologie de contexte. Cette étape (enrichissement d'évènements) inclut l'ajout d'une valeur de confiance dans les évènements. Enfin, les évènements enrichis sont analysés pour générer un seul évènement complexe sémantique flou. La sortie de notre *FSCEP* peut ensuite être utilisée par d'autres applications, y compris la reconnaissance d'activité ou l'identification de situation.

Suite à la détection des signaux/des données brutes issues des capteurs et à la génération d'un ensemble d'évènements, une analyse de ces évènements est indispensable pour la phase d'identification de situations/d'activités. Le chapitre suivant entame cette phase d'analyse, il se focalise en particulier sur le niveau *observation* décrit dans l'architecture distribuée globale (voir Figure 3).

Chapitre 3

Observation : sélection des attributs basée sur l'intégrale de Choquet (l'approche FSCI)

Sommaire

3.1 Introduction

Après la phase de perception (l'approche FSCEP), un ensemble d'évènements flous sont générés à partir des données capteurs. Une observation et une analyse de ces évènements sont nécessaires pour la phase d'identification des activités humaines. Cette observation consiste en trois étapes successives : la segmentation (*segmentation*), l'extraction des attributs (*feature extraction*) et la sélection des attributs (*feature selection*) pour passer par la suite à la phase d'identification des activités et en particulier à la reconnaissance d'activités humaines. En effet, les systèmes HAR (Human Activity Recognition) segmentent les évènements capteurs puis extraient les attributs pour une utilisation ultérieure dans la reconnaissance. La segmentation et l'extraction des attributs n'entrent pas dans le cadre de cette thèse. Nous nous sommes intéressés à l'étape **Sélection des attributs**. Cette étape encore appelée **réduction de dimension** est une étape cruciale dans le processus HAR visant à réduire la dimension potentiellement importante des attributs et à fournir des paramètres pertinents pour améliorer la classification des activités. Les méthodes de sélection des attributs classiques sont généralement linéaires et ne tiennent pas compte des dépendances et des interactions existantes entre les activités. Pour surmonter cela, une nouvelle méthode de sélection des attributs basée sur l'intégrale de Choquet pour les systèmes HAR est proposée dans ce chapitre.

Le chapitre commence par introduire la nécessité d'avoir une méthode de sélection des attributs en présentant nos défis puis présente les travaux connexes. Ce chapitre introduit dans un premier temps, une nouvelle méthode de sélection avec en entrée des attributs non flous qui est l'approche (*FSCI Feature Selection based on Choquet Integral*) [Jarraya et al., 2017]. Ensuite, une étude théorique sur l'extension de *FSCI* à des données floues en entrée est proposée, nous introduisons la version floue de *FSCI* qui est l'approche *Fuzzy-FSCI*. Puis, nous détaillons l'étape d'implémentation et d'évaluation de l'approche *FSCI*. Nous terminons ce chapitre par une discussion et une synthèse de l'approche *FSCI* pour la positionner par rapport aux défis définis au début.

3.2 La sélection des attributs et les défis relevés

La reconnaissance d'activités humaines est devenue récemment un domaine de recherche émergent et l'un des défis de l'informatique pervasive. Elle vise à déterminer les activités humaines en se basant sur l'observation des données capteurs et/ou de vidéos [Ranasinghe et al., 2016]. Les principaux objectifs des systèmes HAR sont d'observer et d'analyser les activités

humaines et d'interpréter avec succès les événements qui arrivent au cours du temps. A partir des données brutes capteurs, les systèmes HAR extraient et traitent des données de haut niveau (environnementales, spatiales, temporelles, etc.) pour comprendre le comportement humain.

En général, le processus HAR (voir Figure 3.1) implique plusieurs étapes en commençant par la collecte d'informations sur le comportement humain à partir des données brutes capteurs jusqu'à l'identification finale de l'activité courante. Ces étapes sont les suivantes : (1) *pré-traitement* des données brutes à partir des flux de capteurs tels que traiter l'incomplétude, éliminer le bruit et la redondance, effectuer l'agrégation et la normalisation des données ; (2) *la segmentation* qui consiste à identifier les segments de données les plus significatifs ; (3) *l'extraction des attributs* qui a pour objectif d'extraire des données de plus haut niveau d'abstraction (e.g. des informations temporelles et spatiales) à partir des segments (4) *la réduction de la dimension* pour diminuer le nombre d'attributs extraits, améliorer leurs qualités et réduire l'effort de calcul nécessaire pour la classification ; (5) *la classification* qui est la reconnaissance de l'activité donnée [Alzahrani and Kammoun, 2016].

FIGURE 3.1. Le processus de la reconnaissance d'activité [Ranasinghe et al., 2016]

Les attributs sont le résultat d'une abstraction des données brutes des capteurs, généralement déterminés à partir d'un segment ou d'une fenêtre (e.g. une fenêtre coulissante temporelle contenant une liste d'évènements déclenchés pendant une durée précise). Alors que de nombreux attributs peuvent être extraits à partir d'un signal issu d'un capteur, l'augmentation du nombre d'attributs n'augmente pas nécessairement la précision et la qualité du classifieur puisque les attributs peuvent être redondants ou parfois non significatifs pour la classe en question (i.e. l'activité à classer). Par conséquent, le choix d'un ensemble approprié d'attributs est important pour une reconnaissance d'activité précise. En effet, la sélection des attributs joue un rôle essentiel dans l'augmentation du taux de la reconnaissance en réduisant le nombre d'attributs pour ne garder que les plus pertinents et utiles pour le classifieur [Chu et al., 2012]. Un attribut est considéré comme pertinent si son retrait diminue la qualité de prédiction, alors qu'un attribut est considéré comme redondant si son retrait n'impacte pas la performance de la prédiction. Dans ce travail, nous nous sommes concentrés sur l'étape de réduction de dimension dans le processus HAR afin d'améliorer les performances de classification des activités humaines.

Différentes méthodes de sélection des attributs existent dans la littérature [Chandrashekar and Sahin, 2014] et peuvent être regroupées en trois grandes catégories selon la manière dont elles interagissent avec le classifieur : les méthodes filtre (*filter methods*), les méthodes d'enveloppes (*wrapper methods*) et les méthodes embarquées (*embedded methods*). Les méthodes filtre [Chandrashekar and Sahin, 2014] opèrent directement sur le jeu de données et sont indépendantes de tout algorithme d'apprentissage automatique. Elles sont basées sur des mesures statistiques pour classer les attributs par un score. Ces méthodes ont l'avantage d'être rapides et indépendantes du modèle de classification, mais au prix de résultats inférieurs (performance de classification). Les méthodes d'enveloppes [Chandrashekar and Sahin, 2014] effectuent une recherche dans l'espace des sous-ensembles des attributs et sélectionnent un sous-ensemble d'attributs pertinents fournissant une précision maximale à partir d'un classifieur spécifique. Elles présentent souvent de meilleurs résultats que les méthodes filtre mais au prix d'un temps de calcul plus important. Les méthodes embarquées [Chandrashekar and Sahin, 2014] effectuent la sélection des attributs dans le processus d'apprentissage. Elles utilisent l'information interne du modèle de classification (e.g. le vecteur de poids dans le cas des machines à vecteurs de support). Ainsi, les méthodes d'enveloppes et embarquées dépendent des résultats du classifieur et sont plus intensives en termes de calcul que les méthodes filtre qui sont des méthodes simples et rapides. Toutefois, les méthodes filtre sont généralement des méthodes linéaires [Jovic et al., 2015] telles que l'algorithme *Linear forward selection*. En outre, elles attribuent un score ou un rang aux attributs sans regarder et analyser les liens de dépendances entre les classes et supposent qu'il n'y a ni conflit ni synergie entre elles.

Dans les systèmes HAR, les activités humaines peuvent être corrélées et/ou fortement liées e.g. préparer le dîner et laver la vaisselle sont deux activités qui sont fortement corrélées en termes de contexte, de temps et de lieu [Ye et al., 2015]. Dans ce cadre, nous nous sommes investis dans la cherche de moyens et de méthodes nous permettant d'étudier les interactions et les dépendances qui existent entre les activités. L'intégrale floue non additive telle que l'intégrale de Choquet peut étudier les dépendances entre les activités par rapport aux autres intégrales floues [Wang et al., 2005].

3.3 Travaux connexes dans la sélection des attributs

La réduction de la dimension des attributs est un problème commun en apprentissage. La sélection des attributs permet de déterminer les attributs les plus pertinents et constitue ainsi

une technique efficiente pour la réduction de la dimension pour le pré-traitement de données afin de supprimer les attributs bruitées et/ou inutiles. Un attribut pertinent pour une tâche d'apprentissage peut être défini comme un attribut dont l'élimination dégrade de manière significative la qualité et la précision de l'apprentissage réalisé. La suppression des attributs non pertinents permet donc la réduction de la dimension et implique un accroissement de la précision et de la compréhensibilité des modèles construits. Dans ce travail, nous nous sommes intéressés aux méthodes de sélection des attributs dans le cas de l'apprentissage supervisé et elles sont regroupées en trois grandes catégories : les méthodes filtre, les méthodes d'enveloppe et les méthodes embarquées (voir Figure 3.2).

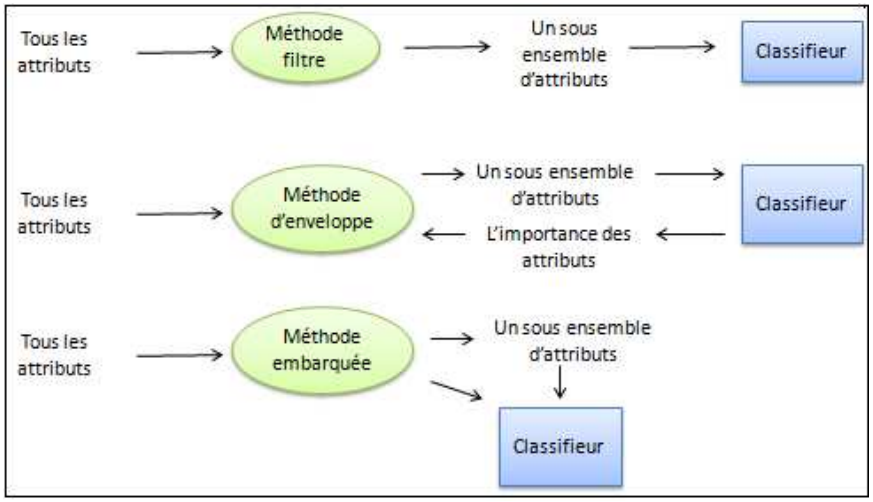


FIGURE 3.2. Les méthodes de sélection des attributs

3.3.1 Les méthodes filtre

Les méthodes filtre sélectionnent les attributs en fonction d'une mesure de pertinence qui évalue la pertinence d'un attribut. Ainsi, la classification peut être réalisée une fois l'ensemble des attributs les plus pertinents déterminé. Les méthodes filtre peuvent classer les attributs individuels ou évaluer des sous-ensembles complets d'attributs. Les mesures de pertinence peuvent être regroupées pour le filtrage des attributs en : mesure d'information, de distance, de cohérence et de similarité [Alzahrani and Kammoun, 2016]. Il existe de nombreuses méthodes filtre décrites dans la littérature, une liste de méthodes courantes est donnée dans le Tableau 3.1, avec les références appropriées qui fournissent des détails.

Les filtres d'attributs univariés évaluent (et classent généralement) un seul attribut, tandis que les filtres multivariés évaluent un sous-ensemble complet d'attributs. La génération de

TABLE 3.1. Étude des méthodes filtre existantes

Méthode filtre	caractéristiques	Application	Travail
Gain d'information	Univarié, information	classification	[Hoque et al., 2014]
Corrélation	Univarié, statistique	régression	[Yu and Liu, 2003]
Chi-square	Univarié, statistique	classification	[Witten et al., 2011a]
Correlation-based feature selection (CFS)	Multivarié, statistique	classification, régression	[Witten et al., 2011a]
Fast correlation-based filter (FCBF)	Multivarié, information	classification	[Yu and Liu, 2003]
Relief and ReliefF	Univarié, distance	classification, régression	[Robnik-Šikonja and Kononenko, 2003]

sous-ensembles d'attributs pour les filtres multivariés dépend de la stratégie de recherche. En général, les méthodes filtre sont des méthodes de pré-traitement et sont indépendantes de la classification (voir Figure 3.2). Chaque attribut est classé par un score calculé sur la base d'une mesure de pertinence.

Dans la reconnaissance d'activité, les auteurs dans [Capela et al., 2015] utilisent trois méthodes filtre qui sont *ReliefF*, *CFS* et *FCBF*. Les résultats d'expérimentation sur le jeu de données USC-HAD [Zhang and Sawchuk, 2012] avec les classifieurs *Naive Bayes*, *J48* et *SVM* montrent que la précision reste stable ou légèrement améliorée en appliquant la méthode *CFS*. Ceci confirme que les attributs redondants sont éliminés sans dégrader la précision de la classification. L'avantage majeur des méthodes filtre est leur rapidité de calcul. Cependant, elles ignorent les résultats du classifieur.

3.3.2 Les méthodes d'enveloppe

Au lieu de classer chaque attribut, ces méthodes classent les sous-ensembles d'attributs en s'appuyant sur les performances de prédiction d'un classifieur d'un sous-ensemble donné (voir Figure 3.2). Contrairement aux filtres, les méthodes d'enveloppe peuvent être utilisées pour rechercher tous les sous-ensembles possibles d'attributs et explorer les informations mutuelles entre ces derniers. Après avoir choisi un classifieur, les méthodes d'enveloppe évaluent les performances de prédiction par exemple les performances en validation croisée sur les données d'apprentissage. Outre le choix des classifieurs, les enveloppes diffèrent dans les stratégies de recherche. En effet, l'évaluation est répétée pour chaque sous-ensemble et la génération du sous-ensemble dépend de la stratégie de recherche. Les enveloppes sont beaucoup plus lentes que les filtres pour trouver des sous-ensembles suffisamment bons car ils dépendent de la qualité du modèle de classification.

Dans la reconnaissance d'activité, les auteurs dans [Zhang and Sawchuk, 2011, Gupta and Dallas, 2014] utilisent respectivement les deux méthodes *Sequential Forward Floating Search (SFFS)* et *sequential forward selection (SFS)* pour sélectionner les attributs les plus pertinents. Ces deux méthodes réalisent de meilleures performances par rapport à la méthode filtre *ReliefF*. Cependant, les méthodes d'enveloppe sont généralement coûteuses en termes de calcul, en particulier si le classifieur a un coût de calcul élevé. En outre, elles dépendent des résultats de la classification.

3.3.3 Les méthodes embarquées

Contrairement aux deux types de méthodes présentées ci-dessus, la recherche d'un sous-ensemble optimal d'attributs est réalisée pendant la construction du classifieur (voir Figure 3.2). Elles utilisent l'information interne du modèle de classification, par exemple, la valeur d'importance de chaque attribut dans le cas des arbres aléatoires.

Dans la reconnaissance d'activité, les auteurs dans [Ronao and Cho, 2014] déterminent les sous-ensembles d'attributs optimaux en utilisant la mesure d'importance des arbres aléatoirs (Random Forest RF) sur le jeu de données *HAR using Smartphones* [Anguita et al., 2013]. En effet, le taux d'erreur sur les données de test est significativement plus faible par rapport aux autres méthodes présentées. Les méthodes embarquées ont l'avantage d'incorporer l'interaction avec le classifieur, alors qu'elles sont généralement plus intensives en termes de calcul que les méthodes d'enveloppe.

Lors de l'analyse des méthodes de sélection d'attributs étudiées ci-dessus, nous remarquons qu'elles déterminent les attributs les plus pertinents sans tenir compte des dépendances qui peuvent exister entre les activités humaines. Elles supposent que toutes les activités sont indépendantes. Alors que ce n'est pas réaliste, car même si les activités sont dans le même contexte, elles peuvent avoir une forte corrélation lorsqu'elles sont combinées.

Notre objectif est de proposer une nouvelle méthode de sélection non coûteuse en terme de temps de traitement comme les méthodes filtre et performante en terme de taux de reconnaissance comme les méthodes d'enveloppes et embarquées. À cet égard, notre approche de sélection d'attributs proposée basée sur l'intégrale de Choquet vise à satisfaire ces deux contraintes décrites au dessus. En effet, elle est capable de déterminer l'importance d'un attribut à travers une densité floue et d'analyser les dépendances entre les activités à travers une mesure floue. De plus, notre méthode peut faire partie soit des méthodes filtre, soit des méthodes d'enveloppe (expliquées dans la section suivante).

3.4 La sélection des attributs basée sur l'intégrale de Choquet

Dans cette section, nous présentons notre nouvelle méthode de sélection des attributs nommée *FSCI* [Jarraya et al., 2017]. Elle est basée sur l'utilisation de l'intégrale non additive, l'intégrale de Choquet pour sélectionner les attributs les plus pertinents. Elle consiste à attribuer un score pour chaque attribut. Ce score est calculé par le biais de l'intégrale de Choquet.

Considérons l'architecture globale du raisonnement distribué, notre approche de sélection *FSCI* se positionne dans le niveau *Observation* (voir Figure 3.3). Un ensemble d'agents observateurs reçoit une liste d'évènements générés non flous par le niveau *Perception*, procède par une observation et une analyse (incluant la segmentation, l'extraction des attributs puis la réduction de dimension) et génère un ensemble de vecteurs d'attributs réduits. Pour l'étape *réduction de dimension*, chaque agent observateur utilise l'approche *FSCI* pour générer une liste d'attributs les plus pertinents par rapport à un seuil fixé par l'expert.

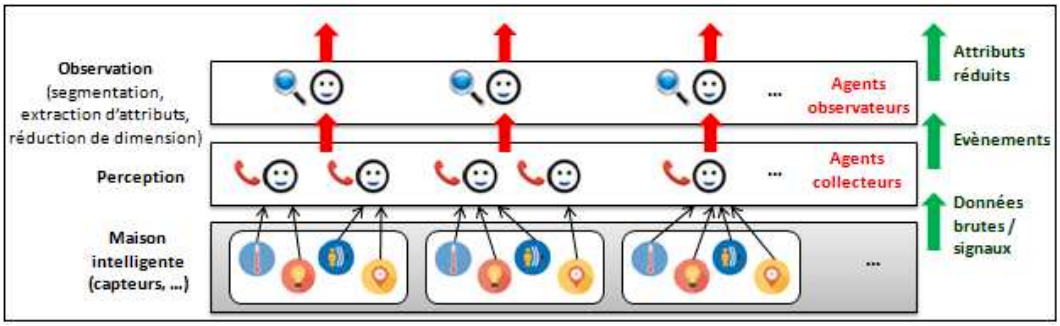


FIGURE 3.3. Position de *FSCI* par rapport à l'architecture globale du raisonnement distribué

Avant d'entamer la présentation de l'approche *FSCI*, nous présentons dans ce qui suit les concepts de base utilisés (qui sont propres au domaine HAR) pour déterminer le score des attributs dans l'ordre suivant : la densité floue, la mesure floue et l'intégrale de Choquet.

3.4.1 Spécification formelle des concepts de base

Nous supposons que le jeu de données en question comporte des lignes qui sont l'ensemble des observations de cardinalité n_o, les colonnes qui sont l'ensemble des attributs de cardinalité n_f. La cardinalité de l'ensemble des activités est de n_a.

Définition 3.4.1. *La densité floue (fuzzy density)*. La densité floue est le degré de liaison entre un attribut et une activité, elle est interprétée comme la valeur d'importance d'un attribut par rapport à une activité. La densité floue $FD_{(a_j,f_k)}$ d'un attribut f_k ($k = 1,2,...n_f$) concernant une activité a_j ($j = 1,2,...n_a$), est déterminée selon deux cas :

1. **Si les attributs sont continus**

   La densité floue $FD_{(a_j,f_k)}$ est déterminée sur la base du calcul de la fonction de densité probabiliste [Witten et al., 2011b].

   Ainsi, la densité floue $FD_{(a_j,f_k)}$ (avec $FD_{(a_j,f_k)} \in \mathbb{R}^+$) d'un attribut f_k pour une activité a_j est définie comme suit :

   $$FD_{(a_j,f_k)}=(1/n_a)*P(a_j)*(1/2\sqrt{\pi})*[1/\sigma_{(f_k,a_j)}-1/\sigma_{(f_k)}] \quad (1)$$

   Avec :

   — n_a : le nombre d'activités.
   — $P(a_j)$: la probabilité d'apparition de l'activité a_j dans le jeu de données en question.
   — σ : l'écart-type des données, une fois en prenant en compte l'activité a_j ($\sigma_{(f_k,a_j)}$) et une fois indépendamment de l'activité ($\sigma_{(f_k)}$).

   Nous rappelons que la fonction de densité probabiliste pour la distribution normale est définie par deux paramètres qui sont la moyenne λ et l'écart-type σ comme suit :
   — $\lambda = (1/n_f)\sum_{i=1}^{n_f} f_i$
   — $\sigma = [(1/n_f)\sum_{i=1}^{n_f}(f_i-\lambda)^2]^{0.5}$

2. **Si les attributs sont discrets**

   La densité floue $FD_{(a_j,f_k)}$ est déterminée sur la base du calcul de la catégorie utilitaire [Witten et al., 2011b].

   Ainsi, la densité floue $FD_{(a_j,f_k)}$ (avec $FD_{(a_j,f_k)} \in \mathbb{R}^+$) d'un attribut f_k pour une activité a_j est définie comme suit :

   $$FD_{(a_j,f_k=v_{kl})} = (1/n_a)*P(a_j)*[P(f_k=v_{kl}|a_j)^2 - P(f_k=v_{kl})^2] \quad (2)$$

   Avec :

— n_a : le nombre d'activités.
— $P(a_j)$: la probabilité d'apparition de l'activité a_j dans le jeu de données en question.
— $P(f_k = v_{kl}|a_j)$: la probabilité d'apparition de l'attribut f_k ayant la valeur v_{kl} pour l'activité a_j.
— $P(f_k = v_{kl})$: la probabilité d'apparition de l'attribut f_k ayant la valeur v_{kl} pour n'importe quelle activité.

Par conséquent, nous obtenons une matrice incluant le degré d'importance des attributs par rapport à chaque activité. Cette matrice est également connue sous le nom de matrice des utilitaires [Grabisch et al., 2008].

Définition 3.4.2. *La mesure floue ou capacité (fuzzy measure)* [Grabisch et al., 2008]. La mesure floue μ sur $A = (a_1, a_2, ..., a_{n_a})$ est une fonction d'ensemble $\mu : 2^A \rightarrow [0,1]$ satisfaisant les trois propriétés suivantes :

1. $\mu(\emptyset) = 0$
2. $\mu(A) = 1$ (normalité)
3. $\forall B, C \subset A$, si $B \subseteq C$ alors $\mu(B) \leq \mu(C)$ (monotonie)

Différentes méthodes d'identification de mesures floues ont été proposées. Dans ce travail, nous avons choisi la méthode des moindres carrés car c'est la méthode d'optimisation la plus utilisée dans la littérature [Grabisch et al., 2008]. Cette méthode utilise deux paramètres en entrée qui sont : la matrice des utilitaires contenant le degré d'importance des attributs et un vecteur nommé "les utilités globales souhaitées" contenant en général l'ordre de préférence des attributs (ce dernier est détaillé dans la sous-section suivante).

Définition 3.4.3. *L'intégrale de Choquet (Choquet integral)* [Grabisch and Labreuche, 2010]. L'intégrale de Choquet de $F = (f_1, f_2, ..., f_j, ..., f_{n_f}) \in \mathbb{R}_+^{n_f}$ par rapport à la mesure floue μ est définie par :

$C_\mu(f_j) = \sum_{i=1}^{n_a} (f_{j\{a_{(i)}\}} - f_{j\{a_{(i-1)}\}} \mu(A_{(i)})$

Avec : $._{(i)}$ indique que les indices ont été interchangés telle que $f_{j(a_1)} \leq f_{j(a_2)} \leq ... \leq f_{j(a_{(n_a-1)})} \leq f_{j(a_{(n_a)})}$ et $A_{(i)} = \{a_{(i)}, ..., a_{(n_a)}\}$ et $f_{j(a_0)} = 0$

À cet égard, l'intégrale de Choquet est utilisée comme une méthode de scoring donnant un score pour chaque attribut à partir d'une mesure floue non additive.

Nous introduisons les indices d'importance et d'interaction permettant respectivement de mesurer l'impact d'une activité sur le calcul de score via l'intégrale de Choquet ainsi que les dépendances existantes entre les activités.

Définition 3.4.4. *L'indice d'importance (importance indice)* [Grabisch et al., 2008]

L'importance globale d'une activité $a_j \in A$ par rapport à une capacité μ peut être mesurée au moyen de sa valeur de Shapley [Shapley, 1953], qui est définie comme suit :

$$\phi_\mu(a_j) = \sum_{C \subseteq A \setminus a_j} \left[\frac{(n_a - c - 1)!c!}{n_a!}\right][(\mu(C \cup a_j) - \mu(C))]$$

c étant la cardinalité de C.

La valeur de Shapley peut être interprétée comme la valeur moyenne pondérée de la contribution $\mu(C \cup a_j) - \mu(C)$ de l'activité a_j parmi toutes les combinaisons. Une propriété intéressante est que la somme de toutes les valeurs d'indice d'importance est égale à 1. En d'autres termes : $\sum_{j=1}^{n_a} \phi_\mu(a_j) = 1$. Ainsi, une activité avec un degré d'importance plus petit que $1/n_a$, peut être interprétée comme une importance faible pour la décision finale.

Définition 3.4.5. *L'indice d'interaction (interaction indice)* [Grabisch et al., 2008]

L'indice d'interaction vise à mesurer l'interaction moyenne entre deux activités a_i et a_j et est défini comme suit :

$$I_\mu(a_i, a_j) = \sum_{C \subseteq A \setminus a_i a_j} \left[\frac{(n_a - c - 2)!c!}{(n_a - 1)!}\right][(\mu(C \cup a_i a_j) - \mu(C \cup a_i) - \mu(C \cup a_j) + \mu(C))]$$

La valeur de l'indice d'interaction est interprétée selon trois cas :

— $I_\mu(a_i, a_j) = 0$: les activités a_i et a_j sont indépendantes ou non interactives.

— $I_\mu(a_i, a_j) > 0$: il y a une synergie positive ou une complémentarité entre les activités a_i et a_j.

— $I_\mu(a_i, a_j) < 0$: il y a une redondance ou une synergie négative entre les activités a_i et a_j.

La section suivante détaille la nouvelle approche *FSCI* de sélection d'attributs proposée dans cette thèse.

3.4.2 La méthode de sélection des attributs FSCI

Notre nouvelle méthode de sélection d'attributs basée sur l'intégrale de Choquet (*FSCI*), vise à classer les attributs selon un score. La Figure 3.4 décrit les étapes pour déterminer ce score à partir d'un jeu de données. Le jeu de données est une matrice où les colonnes sont des attributs et les lignes des observations.

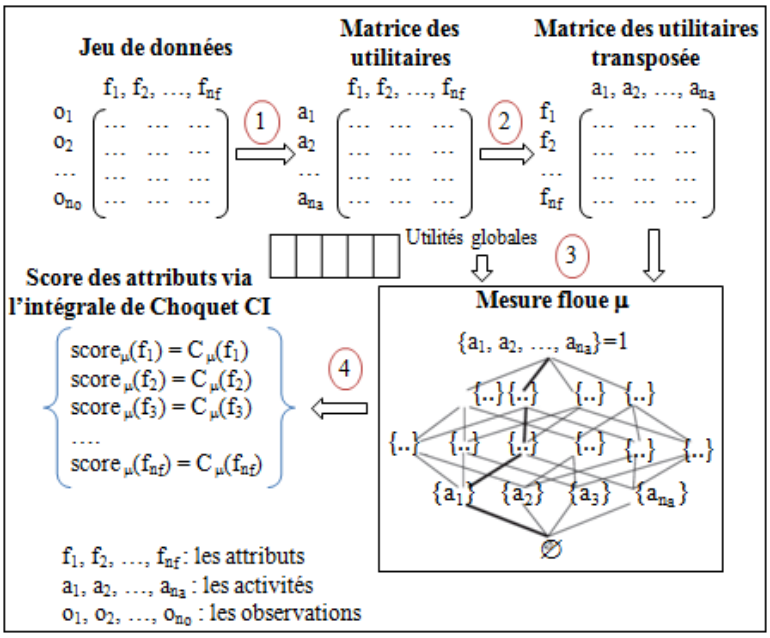


FIGURE 3.4. L'approche *FSCI*

Notre approche *FSCI* se décline en quatre étapes (voir Figure 3.4) comme suit :

1. **Construction de la matrice des utilitaires** : ayant en entrée un jeu de données, la matrice des utilitaires est construite sur la base de calcul du la densité floue de chaque attribut par rapport à chaque activité.
2. **Transposition de la matrice des utilitaires** : cette étape est nécessaire pour identifier la mesure floue μ. Dans [Grabisch and Labreuche, 2010], les critères (dans notre cas, les critères sont les activités humaines) doivent être les colonnes pour déterminer l'interaction des sous-ensembles et les membres pour lesquels un score sera calculé (dans notre cas, les membres sont les attributs) doivent être les lignes.
3. **Identification de la mesure floue μ** : dans ce travail, nous avons utilisé la méthode des moindres carrés [Grabisch et al., 2008] pour déterminer la mesure floue. Ainsi, nous avons besoin de la matrice transposée des utilitaires et les utilités globales sou-

haitées [Grabisch et al., 2008] qui comportent des contraintes exprimant l'ordre de préférence des attributs, l'ordre d'importance des attributs, etc.

4. **Calcul du score des attributs** : cette étape détermine le score de chaque attribut à travers l'intégrale de Choquet selon la mesure floue μ.

Dans la sous-section suivante, l'approche *FSCI* est détaillée à travers un algorithme.

3.4.3 L'algorithme FSCI

L'algorithme *FSCI* (Algorithme 2) sélectionne les attributs optimaux en utilisant l'intégrale de Choquet. Il prend comme paramètres d'entrée un jeu de données D contenant différents attributs en colonnes et différentes observations en lignes, l'ensemble des activités (classes) A à identifier, l'ensemble des attributs F et le seuil de score ε à partir duquel les attributs pertinents peuvent être sélectionnés. L'algorithme *FSCI* vise à fournir, en sortie, la liste SF des attributs les plus pertinents en fonction du seuil fixé. L'algorithme *FSCI* se déroule en quatre étapes principales :

— *Première étape (lignes 6-7-8)* : la matrice des utilitaires est construite en calculant la fonction de densité probabiliste Eq (1) de chaque attribut pour une activité spécifique par le biais de la fonction $computeFD$.

— *Deuxième étape (ligne 10)* : la matrice tFD est la matrice transposée de FD où les colonnes sont les activités et les lignes sont les attributs.

— *Troisième étape (lignes 11-12)* : le vecteur de mesure floue FM contient des valeurs des sous-ensembles représentant le degré d'interaction.
La fonction $computeFuzzyMeasure$ utilise comme paramètres la matrice transposée tFD et le vecteur *overall* incluant les utilités globales désirées. Ce vecteur est calculé à l'aide de la fonction $computeOverall$. Dans notre cas, nous n'avons aucune idée sur l'ordre de préférence des attributs. Dans ce cas, deux choix peuvent être présentés pour déterminer le vecteur *overall* :

1. Le vecteur *overall* est initialisé à zéro. Ainsi, la méthode *FSCI* peut être considérée comme une méthode filtre puisqu'elle est indépendante de n'importe quelle méthode d'apprentissage. Cette approche *FSCI* version filtre est nommée l'approche **FSCI-F**.
2. Le vecteur *overall* prend le degré d'importance des attributs dans l'ordre. Pour le déterminer, des méthodes de mesure d'importance peuvent être utilisées telles que la mesure d'importance des arbres aléatoires [Strobl et al., 2007]. Ainsi, la

méthode *FSCI* est considérée comme une méthode d'enveloppe car elle dépend des résultats de la méthode d'apprentissage. Cette approche *FSCI* version enveloppe est nommée l'approche **FSCI-W**.

Dans ce travail, nous nous sommes concentrés sur l'approche **FSCI-F** où le vecteur *overall* prend des zéros.

— *Quatrième étape (lignes 13-14-15-16)* : l'intégrale de Choquet détermine le score $score_{(FM,f_i)}$ de chaque attribut grâce à la fonction *computeChoquetIntegral*. Cette fonction prend comme paramètres la mesure floue FM et le vecteur f_i de la matrice des utilitaires tFD. Ce score est ajouté à la liste SF avec l'attribut en question si son score est supérieur au seuil ε.

Analyse de complexité

Nous étudions la complexité de calcul de l'algorithme *FSCI* (voir Algorithme 2) dans le pire des cas. Tout d'abord, nous identifions les paramètres suivants :

— $|A|$: la cardinalité des activities

— $|F|$: la cardinalité des attributs

La complexité, dans le pire des cas, est la complexité des quatre étapes de l'algorithme *FSCI* :

— *Première étape (lignes 6-7-8)* : afin de construire la matrice des utilitaires FD, la densité floue est calculée $(|A| \times |F|)$ fois pour chaque attribut par rapport à l'activité en question. Ainsi, la complexité de calcul de cette étape est $\mathcal{O}(|A| \times |F|)$.

— *Deuxième étape (ligne 10)* : la complexité de transposer la matrice FD est $\mathcal{O}(|F| \times |A|)$.

— *Troisième étape (lignes 11-12)* : la complexité de la fonction *computeOverall* dépend de la complexité de la technique de mesure de l'importance utilisée. Nous nous sommes intéressés dans ce travail à l'approche *FSCI-F* dont la variable *overall* est nulle. Pour la complexité de mesure floue, la dimension de FM est $2^{|A|}$. En outre, nous avons toujours $|A| << |F|$. Par conséquent, $2^{|A|}$ tend vers une constante. Ainsi, la complexité de *computeFuzzyMeasure* est $\mathcal{O}(1)$.

— *Quatrième étape (lignes 13-14-15-16)* : la fonction *computeChoquetIntegral* détermine un score pour chaque attribut. Ainsi, la complexité de cette fonction est $\mathcal{O}(|F|)$.

Au final,le coût global (GC) est :

$$GC = \mathcal{O}(|A| \times |F|) + \mathcal{O}(|F| \times |A|) + \mathcal{O}(1) + \mathcal{O}(|F|) \simeq \mathcal{O}(|F| \times |A|)$$

Algorithme 2. L'algorithme de sélection des attributs basé sur l'intégrale de Choquet (*FSCI*)

Input :
D : le jeu de données
A : l'ensemble d'activités
F : l'ensemble d'attributs
ε : le seuil de score
Output :
SF : liste des attributs pertinents sélectionnés

```
1  begin
2      initialise(FD) // initialisation de la matrice de densité floue
3      initialise(FM) // initialisation de la mesure floue μ
4      initialise(SF)
5
       // Première étape: construire la matrice de densité floue
6      for a_i ∈ A do
7          for f_j ∈ F do
8              FD[i, j] = computeFD(D, A, F)
9
       // Deuxième étape: transposer la matrice FD
10     tFD = t(FD)
       // Troisième étape: déterminer la mesure floue FM
11     overall = computeOverall(D, A, F)
12     FM = computeFuzzyMeasure(tFD, overall)
       // Quatrième étape: déterminer le score des attributs via
          l'intégrale de Choquet
13     for f_i ∈ F do
14         score_(FM,f_i) = computeChoquetIntegral(FM, tFD[f_i, ])
15         if score_(FM,f_i) ≥ ε then
16             SF.add(f_i, score_(FM,f_i))
17     return(SF)
```

Par conséquent, nous avons une complexité temporelle linéaire pour l'algorithme *FSCI* en fonction du nombre d'attributs et d'activités de l'ordre de $(|F| \times |A|)$.

3.5 Extension de FSCI à des données d'entrée floues

Nous rappelons que l'approche *FSCI* est une approche de sélection des meilleurs attributs non flous. Étant donné que l'approche de perception *FSCEP* génère en sortie une liste d'évènements flous, il serait intéressant de pouvoir appliquer les phases de segmentation et d'extraction d'attributs pour générer un ensemble de vecteurs d'attributs flous. Or, l'approche *FSCI* s'applique sur des données non floues et ne peut donc pas être utilisée telle quelle.

L'objectif de cette section est de présenter une étude théorique sur la nouvelle approche **Fuzzy-FSCI**. Cette dernière représente l'approche *FSCI* avec des observations floues en entrée. Avec cette nouvelle approche, la nature floue des données est conservée entre l'approche de perception *FSCEP*, la segmentation, l'extraction d'attributs flous et la sélection d'attributs flous.

3.5.1 Liaison avec l'approche FSCEP

L'approche *FSCEP* permet de détecter les signaux/collecter les données brutes depuis l'environnement intelligent et de générer en sortie une liste d'évènements flous pour les données contextuelles observées pendant un intervalle de temps. Ces données contextuelles observées peuvent être la localisation de l'habitant, la température, la pression, etc.

Après les phases de segmentation et d'extraction des attributs flous, on aura une liste d'observations ou encore une liste de vecteurs d'attributs flous (voir Figure 3.5).

	Horodatage	Localisation	Température	Pression	....	Attribut (n)	Activité
Observation 1	19-07-2018 22:13:56	Livingroom 0.4 Bedroom 0.6	High 0.3 Medium 0.5 Low 0.2	High 0.7 Medium 0.2 Low 0.1	....	.....	sleeping
Observation 2	20-07-2018 08:26:40	Kitchen 0.9 Corridor 0.1	High 0.1 Medium 0.7 Low 0.2	High 0.1 Medium 0.1 Low 0.8	....	.....	preparing_breakfast
....	.....	.....	.....	.....	....	.....	.....

FIGURE 3.5. Les vecteurs d'attributs flous générés suite à l'étape *extraction d'attributs*

Un attribut flou est un ensemble flou défini avec sa fonction d'appartenance comme défini ci-après.

Définition 3.5.1. *Ensemble flou et fonction d'appartenance* [Zadeh, 1988].

Un ensemble flou est défini par ses variables linguistiques et sa fonction d'appartenance :

— *Variable linguistique* : Dans une partition floue, chaque ensemble correspond à un concept linguistique, par exemple Très faible, Faible, Moyen, Elevé, Très élevé.

— *Une fonction d'appartenance* : une fonction d'appartenance μ_A d'un ensemble flou A est une fonction qui associe à chaque élément x de l'univers de discours son degré d'appartenance $\mu_A(x)$ appartenant a l'intervalle $[0,1]$.

Un ensemble flou est alors présenté comme suit : $A = \{(x, \mu_A(x)), x \in X, \mu_A : X \rightarrow [0,1]\}$. Plusieurs types de fonctions d'appartenance sont proposés dans la littérature. Les fonctions d'appartenance les plus communément utilisées sont : la fonction triangulaire, la fonction trapézoïdale, la fonction monotone croissante, la fonction monotone décroissante et la fonction Gaussienne.

Dans ce cas, l'approche *FSCI* ne peut pas s'appliquer car elle prend en entrée des vecteurs d'attributs discrets ou continus et non flous. La sous section suivante présente la version floue de l'approche *FSCI* qui pallie cette limite.

3.5.2 L'approche Fuzzy-FSCI

L'approche *FSCI* prend en entrée des vecteurs d'attributs continus ou discrets alors que l'approche *Fuzzy-FSCI* prend en entrée des vecteurs d'attributs flous. Les deux approches *FSCI* et *Fuzzy-FSCI* visent le même objectif qui est la sélection des attributs les plus pertinents.

De même que l'approche *FSCI*, l'approche *Fuzzy-FSCI* vise à déterminer un score pour chaque attribut flou et sélectionne les attributs flous ayant un score supérieur à un seuil fixé par l'expert. La démarche de *FSCI* décrite dans la figure 3.4 reste la même avec les quatre étapes. Cependant, ce qui différencie l'approche *Fuzzy-FSCI* par rapport à l'approche *FSCI* en terme de traitement c'est la première étape. Cette dernière consiste à construire la *matrice des utilitaires* depuis le jeu de données ayant des attributs continus ou discrets. Elle s'appuie sur le calcul de la densité floue de chaque attribut (continu ou discret) par rapport à l'activité en question.

Par conséquent, la première étape est à modifier pour l'approche *Fuzzy-FSCI*. En effet, l'approche *Fuzzy-FSCI* prend en entrée un jeu de données flou ayant des attributs flous. Pour déterminer la *matrice des utilitaires*, deux cas se présentent :

1. Appliquer la defuzzification pour l'ensemble des attributs flous. Cette étape génère

un jeu de données non flou. L'enchainement des étapes de l'approche *FSCI* peut s'appliquer dans ce cas.

2. Garder les attributs flous et modifier l'étape de construction de la *matrice des utilitaires*.

Considérons le deuxième cas. Afin de déterminer la *matrice des utilitaires*, il faut calculer *la densité floue* de chaque attribut flou par rapport à l'activité en question. Toutefois, *la densité floue* est déterminée sur la base de calcul de *la fonction de densité probabiliste* si les attributs sont continus ou bien sur la base de calcul de *la catégorie utilitaire* si les attributs sont discrets.

Nous rappelons la définition de *la densité floue* $FD_{(a_j,f_k)}$ sur la base du calcul de *la fonction de densité probabiliste* dans le cas où nous avons des attributs continus :

$$FD_{(a_j,f_k)}=(1/n_a)*P(a_j)*(1/2\sqrt{\pi})*[1/\sigma_{(f_k,a_j)}-1/\sigma_{(f_k)}]$$

Cette même définition peut s'appliquer dans le cas où nous avons des attributs flous ou encore appelés des nombres flous [Zhu and Xu, 2011]. Plus précisément, la définition de la moyenne et l'écart type d'un nombre flou est proposée dans plusieurs travaux tel que [Zhu and Xu, 2011]. Les auteurs dans [Zhu and Xu, 2011], modélisent les nombres flous en utilisant les fonctions d'appartenance triangulaire et trapézoïdale. Ils présentent la définition de calcul de la moyenne floue et l'écart type flou dans ces cas. Ainsi, nous pouvons déterminer la densité floue puisque nous pouvons calculer la moyenne floue et l'écart type flou d'un attribut flou.

Pour simuler l'approche *Fuzzy-FSCI*, un jeu de données publique avec un nombre important d'attributs est à utiliser. Il n'existe pas de jeu de données comportant des attributs flous. Les jeux de données existants sont non flous. La fuzzification [Zadeh, 1988] devient alors indispensable pour transformer un jeu de données non flou en un jeu de données flou.

La mise en place et l'évaluation de cette extension floue de *FSCI* sont considérées comme des perspectives pour cette thèse.

3.6 Simulation et évaluation de FSCI

Dans cette section nous évaluons l'approche *FSCI*. La mise en place et l'évaluation de l'approche *Fuzzy-FSCI* reste un travail futur pour cette thèse.

L'algorithme *FSCI-F* proposé est évalué et comparé à d'autres méthodes de sélection d'attributs dans un contexte HAR. Nous avons utilisé le cas d'un jeu de données ayant des

attributs continus. Dans cette section, nous présentons le jeu de données HAR utilisé, l'environnement de simulation et les évaluations expérimentales.

3.6.1 Le jeu de données HAR

Nous avons besoin d'un jeu de données publique pour HAR. Il doit présenter un nombre important d'attributs. C'est pourquoi nous avons choisi le jeu de données UCI HAR (Human Activity Recognition Using Smartphones Data Set) [1] du domaine public [Anguita et al., 2013] qui présente 561 attributs. Il est utilisé dans toutes nos expériences dans ce travail. Ce jeu de données contient des attributs extraits des données provenant d'accéléromètres et de gyroscopes xyz recueillies à partir d'un téléphone intelligent Samsung Galaxy SII porté par 30 personnes volontaires. Chaque personne exécute six activités : WALKING (W), WALKING UPSTAIRS (WU), WALKING DOWNSTAIRS (WD), SITTING (SI), STANDING (ST), LAYING (L). Le jeu de données est partitionné en deux ensembles, où 70% des volontaires ont été sélectionnés pour générer les données d'apprentissage et 30% pour les données de test. L'étape de pré-traitement du signal (accéléromètre et gyroscope) est déjà effectuée et les attributs sont déjà générés en calculant des variables à partir du domaine temporel et fréquentiel.

3.6.2 Environnement de simulation

Pour évaluer notre approche *FSCI-F*, nous avons choisi l'environnement **Anaconda** [2] V3, la principale plateforme de l'analyse des données open-source. Il inclut les paquets **Python** et **R** les plus populaires pour l'analyse des données. Nous avons utilisé la bibliothèque Python open source **Scikit-learn** [3] qui implémente une gamme d'algorithmes d'apprentissage automatique, de pré-traitement, de validation croisée et de visualisation. Cependant, les intégrales non additives ne sont pas implémentées dans la bibliothèque Python. C'est pourquoi, nous avons utilisé le package **Kappalab** [4] dans la programmation **R** qui intègre des méthodes d'identification de mesure floue et applique l'intégrale de Choquet. L'interfaçage entre Python et le langage R est réalisé grâce à la bibliothèque **rpy24** [5].

1. https ://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones
2. https ://www.continuum.io/anaconda-overview
3. http ://scikit-learn.org/
4. https ://cran.r-project.org/web/packages/kappalab/index.html
5. http ://rpy2.bitbucket.org/

TABLE 3.2. Le taux de reconnaissance des différents classifieurs pour le jeu de données HAR

RF		DT		AdaB	
Testset	10CV	Testset	10CV	Testset	10CV
92.3	93.5	86.73	86.87	53.1	54.08

SVM (LinearSVC)		NB	
Testset	10CV	Testset	10CV
96.47	96.25	77.03	72.61

3.6.3 Évaluation expérimentale

Avant de réduire la dimension des attributs, nous devons examiner le jeu de données en appliquant des algorithmes d'apprentissage automatique avec tous les attributs (561 attributs) afin d'avoir une idée sur la performance de ces algorithmes. Ainsi, nous appliquons cinq classifieurs (existant dans la bibliothèque Scikit-learn) qui sont Random Forest (RF) avec 100 arbres, Decision Tree (DT), AdaBoost (AdaB) avec 100 arbres, Support Vector Machine (SVM) et Naive Bayes (NB). La classification a été réalisée en utilisant deux méthodes de validation : une validation croisée de 10 fois (10 CV) et une approche de validation d'apprentissage qui signifie 70% des données pour la base d'apprentissage et 30% des données pour la base de test (Testset).

Le tableau 3.2 indique le taux de bonne classification des différents classifieurs qui représente le taux de reconnaissance des bonnes activités. Les classifieurs les plus précis sont le classifieur SVM suivi du classifieur RF pour le jeu de données HAR.

La réduction de la dimension des attributs peut augmenter le taux de reconnaissance en éliminant les attributs redondants et non pertinents. *FSCI-F* est évalué avec les deux classifieurs SVM et RF les plus précis et il est comparé à d'autres méthodes de sélection des attributs qui sont :

— *Méthodes filtre* : reliefF [Robnik-Šikonja and Kononenko, 2003] et selectKBest proposée par le paquet Scikit-learn.
— *méthode d'enveloppe* : SVM-LinearSVC
— *méthodes embarquées* : RF mean decrease impurity (RF MDI) et RF mean decrease accuracy (RF MDA). Ces deux méthodes sont décrites dans [Breiman, 2001].

Afin d'évaluer notre approche *FSCI-F*, trois métriques sont choisies : le taux de reconnaissance (voir Annexe A.2), la matrice de confusion (voir Annexe A.1) et le temps d'ap-

prentissage.

3.6.3.1 Évaluation avec le taux de reconnaissance

Nous avons appliqué différentes méthodes de sélection d'attributs décrites ci-dessus, y compris notre approche *FSCI-F*. Nous avons fait varier le nombre d'attributs les plus pertinents de 50 attributs à 400 attributs afin de calculer le taux de reconnaissance à chaque fois avec une validation croisée d'ordre 10 (10CV). La Figure 3.6 indique les résultats obtenus.

La Figure 3.6a montre l'évolution du taux de reconnaissance lors de l'augmentation du nombre d'attributs en utilisant le classifieur RF. Le taux, lors de la sélection de 280 attributs en utilisant la méthode *FSCI-F*, atteint la ligne de référence (la ligne de référence est la ligne horizontale en orange qui représente le taux de reconnaissance sans appliquer une méthode de sélection d'attributs). Ainsi, la moitié des attributs restants affectent faiblement le taux de reconnaissance du classifieur. Cependant, trois méthodes de sélection d'attributs dépassent la ligne de référence lors de la sélection de 280 attributs, à savoir RF MDI, RF MDA et LinearSVC. Ceci souligne l'apport principal de ces méthodes : le choix des attributs les plus pertinents est basé sur les résultats du classifieur.

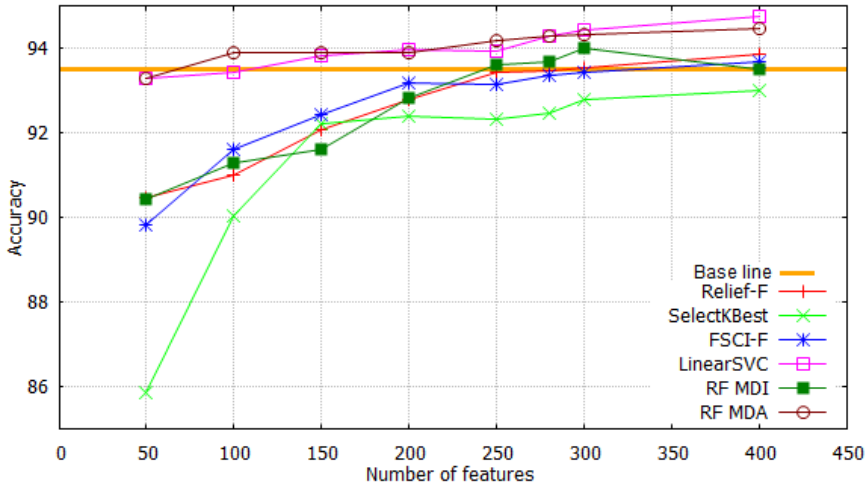
Pour le classifieur SVM (voir Figure 3.6b), les résultats du taux de reconnaissance ne sont pas satisfaisants. Le taux, lors de la sélection de 280 attributs, est proche de la ligne de référence de 1% pour toutes les méthodes sauf la méthode LinearSVC.

Discussion : l'approche proposée *FSCI-F* présente des résultats satisfaisants avec le classifieur RF. Cela nous a permis de conserver le taux de reconnaissance avec la moitié des attributs même si les méthodes d'enveloppe et embarquées présentent de meilleurs résultats. Cependant ces dernières nécessitent un traitement coûteux en utilisant les résultats du classifieur. Pour cette raison, nous mettons en évidence l'évaluation de la méthode *FSCI-F* avec les mesures suivantes en utilisant uniquement le classifieur RF.

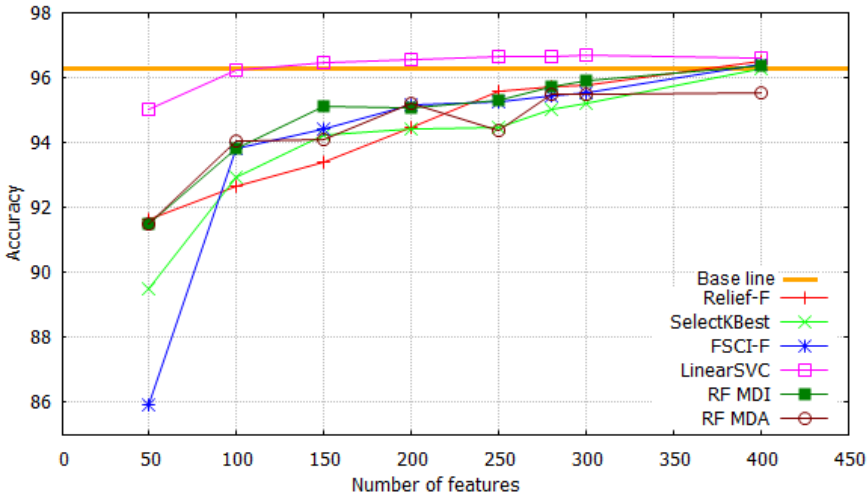
3.6.3.2 Évaluation avec la matrice de confusion

Pour améliorer notre analyse, nous avons utilisé la matrice de confusion (décrite dans l'annexe) pour évaluer le taux de la classification avec le classifieur RF (voir Figure 3.7). La Figure 3.7a correspond à la matrice de confusion considérant tous les attributs. Par exemple, l'activité W possède réellement 496 instances. Le classifieur RF n'a prédit que 477 instances pour l'activité W, 12 activités WU et 7 activités WD.

La Figure 3.7b correspond à la matrice de confusion en appliquant notre approche *FSCI-F*. En comparant avec la Figure 3.7a, la méthode *FSCI-F* améliore légèrement le nombre des

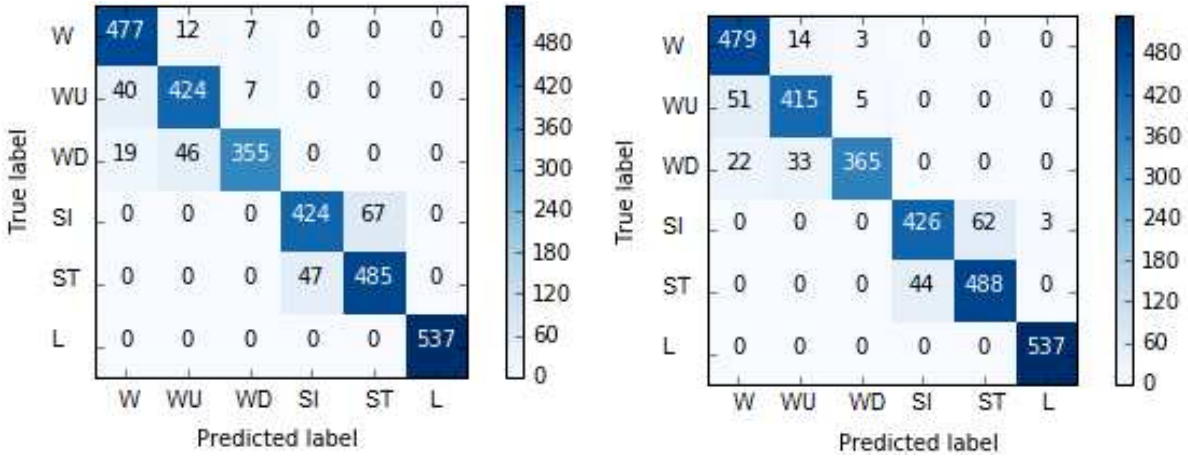


(a) RF avec les méthodes de sélection des attributs



(b) SVM avec les méthodes de sélection des attributs

FIGURE 3.6. FSCI-F vs les autres méthodes de sélection des attributs en terme de taux de reconnaissance



(a) Sans une méthode de sélection des attributs (b) Avec la méthode de sélection FSCI-F

FIGURE 3.7. Matrice de confusion avec le classifieur RF pour le jeu de données HAR

bonnes activités sauf l'activité WU lors de la classification avec 280 attributs.

3.6.3.3 Évaluation avec le temps d'apprentissage

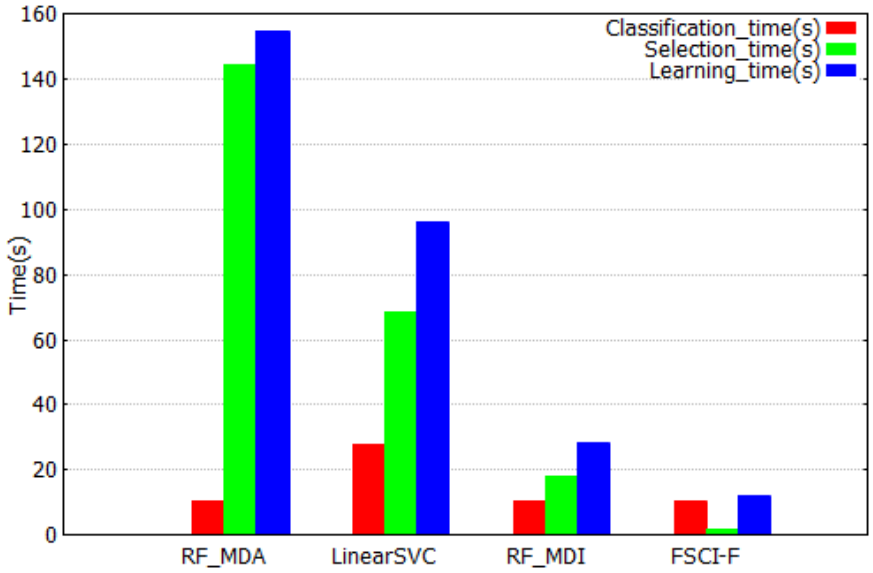
Outre l'analyse du taux de reconnaissance et de la matrice de confusion, nous avons analysé la performance du *FSCI-F* par rapport aux méthodes d'enveloppe et embarquées *LinearSVC*, *RF MDI* et *RF MDA* (qui sont plus performants en termes de taux de reconnaissance) en terme de temps d'apprentissage (en secondes (s)) d'un côté mais d'un autre côté par rapport aux méthodes filtre (*ReliefF* et *selectKBest*).

Nous divisons le **temps d'apprentissage** en temps de classification et en temps de sélection comme suit :

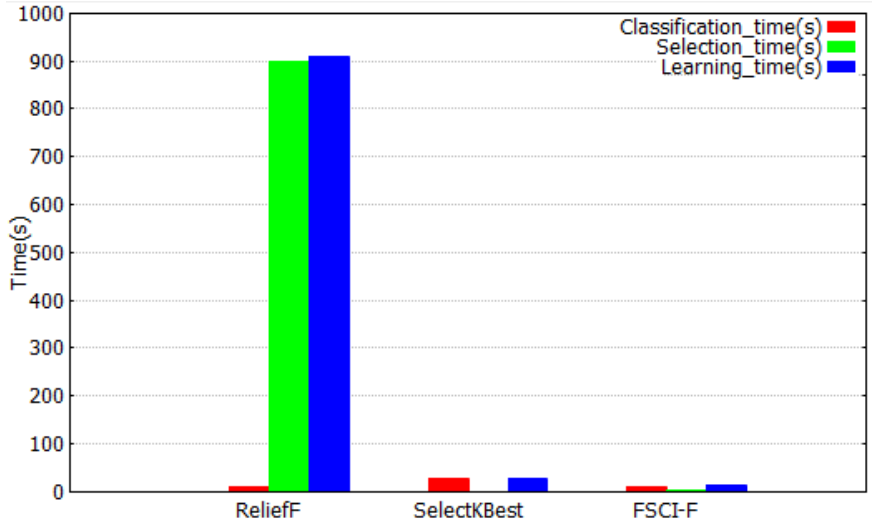
— **Temps de classification** : c'est le temps nécessaire pour classer les activités d'un classifieur spécifique.

— **Temps de sélection** : c'est le temps nécessaire pour faire la sélection. En fait, le temps de sélection pour les méthodes filtre est le temps nécessaire pour classer les attributs selon un score. Le temps de sélection pour les méthodes d'enveloppe et pour les méthodes embarquées est le temps de classification avec le temps de détermination des attributs les plus pertinents, car les méthodes d'enveloppe et embarquées sont dépendantes des résultats de la classification.

La Figure 3.8 souligne le temps de classification, le temps de sélection et le temps d'apprentissage des différentes méthodes de sélection d'attributs lors de la sélection des 280 attributs les plus pertinents et lors de la classification avec un classifieur RF.



(a) FSCI-F vs les méthodes d'enveloppe et embarquées



(b) FSCI-F vs les méthodes filtre

FIGURE 3.8. FSCI-F vs les méthodes de sélection en terme de temps d'apprentissage

FSCI-F est la méthode la plus efficace par rapport à toutes les méthodes traitées. Le temps d'apprentissage est évalué à 12 s pour la sélection et la classification avec RF. La figure 3.8a montre que les autres méthodes de sélection des attributs d'enveloppes et embarquées sont moins rapides notamment pour la méthode *RF MDA* où le temps d'apprentissage est d'environ 2 min. En général, les méthodes d'enveloppe et les méthodes embarquées sont plus coûteuses en terme de temps de traitement que les méthodes filtre. De plus, la figure 3.8b montre que *FSCI-F* est la méthode filtre la plus rapide.

3.6.3.4 Analyse des indices d'importance et d'interaction

Dans cette sous section, nous analysons les deux indices d'importance et d'interaction des six activités (W, WU, WD, SI, ST et L).

1. **Indice d'importance**

   Le Tableau 3.3 montre les valeurs d'importance de chaque activité. Nous constatons que toutes les activités ont la même valeur d'importance et la somme de ces valeurs vaut 1. Par conséquent, aucune activité n'est considérée plus importante qu'une autre.

TABLE 3.3. Indice de l'importance des six activités

W	WU	WD	SI	ST	L
0.1666667	0.1666667	0.1666667	0.1666667	0.1666667	0.1666667

2. **Indice d'interaction**

   L'indice d'interaction obtenu suite au calcul de la mesure floue de l'ensemble des activités est illustré dans le Tableau 3.4. Chaque activité interagit positivement et d'une manière équitable avec les autres. Ainsi, les activités sont indépendantes et complémentaires.

TABLE 3.4. Indice de l'interaction des six activités

Id	W 1	WU 2	WD 3	SI 4	ST 5	L 6
1	NA	0.1999988	0.1999988	0.1999988	0.1999988	0.1999988
2	0.1999988	NA	0.1999988	0.1999988	0.1999988	0.1999988
3	0.1999988	0.1999988	NA	0.1999988	0.1999988	0.1999988
4	0.1999988	0.1999988	0.1999988	NA	0.1999988	0.1999988
5	0.1999988	0.1999988	0.1999988	0.1999988	NA	0.1999988
6	0.1999988	0.1999988	0.1999988	0.1999988	0.1999988	NA

3.7 Discussion et synthèse

Dans cette section, nous discutons de notre nouvelle méthode de sélection d'attributs *FSCI* et en particulier *FSCI-F*. Cette dernière est une méthode de sélection filtre. Nous la positionnons par rapport aux méthodes décrites dans la section des travaux connexes dans le Tableau 3.5.

TABLE 3.5. Étude des méthodes filtre existantes

Méthode filtre	caractéristiques	Application	Travail
Gain d'information	Univarié, information	classification	[Hoque et al., 2014]
Corrélation	Univarié, statistique	régression	[Yu and Liu, 2003]
Chi-square	Univarié, statistique	classification	[Witten et al., 2011a]
Correlation-based feature selection (CFS)	Multivarié, statistique	classification, régression	[Witten et al., 2011a]
Fast correlation-based filter (FCBF)	Multivarié, information	classification	[Yu and Liu, 2003]
Relief and ReliefF	Univarié, distance	classification, régression	[Robnik-Šikonja and Kononenko, 2003]
FSCI-F	Univarié, intégrale de Choquet	classification	[Jarraya et al., 2017]

La méthode *FSCI-F* se présente comme la méthode la plus performante en termes de taux de reconnaissance et de temps d'apprentissage :

— Par rapport aux méthodes filtre : *FSCI-F* est plus rapide et plus performante que les deux méthodes reliefF et selectKBest.

— Par rapport aux méthodes d'enveloppe et embarquées : *FSCI-F* est légèrement moins performante pour une classification avec la moitié des attributs dans la base. Toutefois, elle est plus rapide que les autres.

Finalement, notre méthode proposée *FSCI-F* conserve le même taux de reconnaissance avec les 280 attributs les plus pertinents lors d'une classification avec RF sur le jeu de données HAR. En outre, elle améliore le temps d'apprentissage par rapport aux autres méthodes de sélection puisque *FSCI-F* est indépendant des résultats de la classification.

3.8 Conclusion

Dans ce travail, nous avons présenté un aperçu des travaux récents sur les méthodes de sélection des attributs appliquées au domaine HAR. Nous avons proposé deux nouvelles méthodes de sélection des attributs les plus pertinents en utilisant l'intégrale de Choquet.

Ces deux nouvelles méthodes diffèrent en termes de données d'entrée. L'approche *FSCI* est proposée pour gérer les données d'entrée non floues et l'approche *Fuzzy-FSCI* est proposée pour gérer les données d'entrée floues. Ces deux approches modélisent l'importance des attributs par rapport aux activités à travers la matrice des utilitaires et l'interaction entre les activités à travers la théorie de la mesure floue. Nous avons utilisé la méthode des moindres carrés pour identifier cette mesure floue.

La méthode *FSCI* peut être considérée comme une méthode filtre (méthode *FSCI-F*) ou comme une méthode d'enveloppe (méthode *FSCI-W*). Dans cette thèse, nous nous sommes intéressés à la méthode *FSCI-F*. Les résultats expérimentaux réalisés sur le jeu de données HAR démontrent la performance de la méthode proposée par rapport à différentes mesures. En effet, l'approche *FSCI-F* élimine les attributs non pertinents sans impacter le taux de reconnaissance des activités pour le jeu de données HAR avec seulement la moitié des attributs. En outre, elle améliore le temps de réponse par rapport aux méthodes de sélection étudiées.

Côté implémentation et évaluation, la simulation et l'évaluation de l'approche *Fuzzy-FSCI* reste un travail de recherche futur. Seule l'approche *FSCI* a été testée et évaluée sur un jeu de données contenant des attributs extraits à partir des valeurs d'accéléromètres et de gyroscope (axes x y et z) collectées depuis un téléphone portable. Nous notons que notre approche est générique et peut s'appliquer sur n'importe quel type de jeu de données qui présente des attributs redondants ou inutiles. La méthode *FSCI-W* reste un travail futur pour cette thèse.

Chapitre 4

Identification : un modèle de raisonnement distribué pour la reconnaissance d'activités humaines

Sommaire

4.1 Introduction

Après la phase d'observation et en particulier l'étape de sélection des meilleurs attributs, un ensemble de vecteurs d'attributs discrets et/ou continus ou bien flous sont générés. Une phase de traitement et de raisonnement sur ces vecteurs est obligatoire pour aboutir à l'identification et la reconnaissance des activités courantes de l'habitant dans la maison intelligente.

Ce chapitre présente une étude théorique des deux dernières contributions de cette thèse concernant le niveau *Identification*. Précisément, ces deux contributions sont des modèles de raisonnement distribué pour la reconnaissance d'activité humaine dans les maisons intelligentes. La première contribution, nommée *DCR (Distributed Collaborative Reasoning)* prend en entrée un ensemble de vecteurs d'attributs discrets et/ou continus et génère en sortie l'ensemble des activités correspondantes, sa version améliorée d'apprentissage en ligne est nommée *DCR-OL (Distributed Collaborative Reasoning with an Online Learning*. La deuxième contribution est nommée *Fuzzy-DCR (Fuzzy-Distributed Collaborative Reasoning)*, elle prend en entrée un ensemble de vecteurs d'attributs flous et génère en sortie l'ensemble des activités (non) floues.

Le chapitre commence par introduire la nécessité d'avoir des systèmes de reconnaissance d'activité distribués en présentant les défis scientifiques associés puis relate les travaux connexes étudiés afin d'en extraire des exigences à vérifier. Nous détaillons ensuite notre principale contribution *DCR* pour la reconnaissance d'activité [Jarraya et al., 2018] ainsi que sa version améliorée *DCR-OL*. Nous entamons par la suite l'étude théorique de l'approche *Fuzzy-DCR*. Nous terminons ce chapitre par une discussion et une synthèse.

4.2 La reconnaissance d'activité et les défis à relever

La reconnaissance d'activité humaine (*Human Activity Recognition HAR*) est devenue un sujet de recherche important dans différents domaines tels que l'informatique pervasive et mobile [Choudhury et al., 2008], la robotique sociale [Fong et al., 2003], la sécurité basée sur la surveillance [Noorit and Suvonvorn, 2014], l'informatique sensible au contexte [Van Laerhoven and Aidoo, 2001] et l'assistance à l'autonomie à domicile (Ambient Assisted Living AAL) [Philipose et al., 2004]. Ce dernier domaine utilise le paradigme des maisons intelligentes. En effet, la reconnaissance d'activité est une tâche fondamentale dans les maisons intelligentes grâce à laquelle les activités humaines telles que *laver_ mains*, *préparer_repas*, *manger*, *dormir*, peuvent être identifiées et suivies.

Pour effectuer la reconnaissance de ces activités, différents types de capteurs sont déployés dans les maisons intelligentes pour surveiller les comportements des habitants et capturer les changements contextuels générés par les actions humaines. Les données fournies par ces capteurs doivent être traitées à l'aide des techniques d'analyse de données pour créer des modèles d'activité appropriés et ensuite les utiliser pour la reconnaissance d'activité.

Différentes approches proposées dans la littérature utilisent les techniques de raisonnement pour la modélisation et la reconnaissance de l'activité humaine telles que le raisonnement ontologique [Chen et al., 2012, Mohd Noor et al., 2016], le raisonnement probabiliste [Riboni et al., 2016, Liu et al., 2017], le raisonnement évidentiel [McKeever et al., 2010, Sebbak et al., 2014] ou encore le raisonnement flou [Abdelhedi et al., 2016, Rodriguez et al., 2014]. Nous détaillons ces approches comme suit :

— *Raisonnement ontologique* : les auteurs, dans [Chen et al., 2012], utilisent l'ontologie pour la modélisation des capteurs, la fusion des contextes, la modélisation des activités et leurs reconnaissances. Étant donné que les données de capteurs sont incertaines (incomplètes, imprécises, obsolètes ou contradictoires), les auteurs dans [Mohd Noor et al., 2016] proposent un nouvel algorithme de raisonnement qui combine le raisonnement ontologique basé sur la logique de description (Description Logic DL) avec la théorie de Dempster–Shafer (DS) afin de gérer l'incertitude et en particulier gérer les données de capteurs manquantes pour HAR.

— *Raisonnement probabiliste* : les auteurs, dans [Liu et al., 2017], utilisent les réseaux bayésiens simples (Bayesian Network BN) pour la reconnaissance des activités complexes. Les auteurs, dans [Riboni et al., 2016], proposent d'utiliser les réseaux logiques de Markov (MLN) pour identifier les activités chevauchées.

— *Raisonnement évidentiel* : les auteurs, dans [McKeever et al., 2010], étendent la théorie DS pour inclure l'aspect temporel et proposent une plateforme de raisonnement évidentiel pour inférer des activités à partir des données de capteurs selon un graphe orienté acyclique (Directed Acyclic Graphs DAG). De même, les auteurs de [Sebbak et al., 2014] proposent une méthode de correspondance permettant la conversion et l'agrégation efficace des données de capteurs en des données de plus niveau (les activités). En outre, ils présentent une technique de résolution de conflits afin d'optimiser la prise de décision en présence de conflits entre des activités.

— *Raisonnement flou* : les auteurs, dans [Abdelhedi et al., 2016], construisent un système d'inférence flou pour la détection de chutes des personnes à travers l'analyse des séquences vidéo en temps réel. Dans [Rodriguez et al., 2014], les auteurs proposent

une ontologie floue pour la modélisation et le raisonnement sur des données vagues et incomplètes. Cette ontologie floue est capable de modéliser les connaissances incertaines et de représenter les relations temporelles entre les activités en utilisant une représentation de machines d'états.

Le Tableau 4.1 résume les différentes approches discutées ci-dessus selon les critères suivants : la catégorie à laquelle l'approche appartient, la technique de raisonnement adoptée pour HAR et la technique de modélisation des données de bas niveau (e.g. données de capteurs).

TABLE 4.1. Les approches de raisonnement pour HAR

Approche	Méthode de raisonnement	Technique de raisonnement	Technique de modélisation
[Chen et al., 2012]	Orienté connaissance	OWL-DL	Ontologie
[Mohd Noor et al., 2016]	Orienté connaissance	OWL-DL + Théorie DS	Ontologie
[Liu et al., 2017]	Orienté donnée	BN	BN
[Riboni et al., 2016]	Hybride	MLN	Ontologie
[Abdelhedi et al., 2016]	Orienté connaissance	Règles floue	Attributs flous
[Rodriguez et al., 2014]	Orienté connaissance	Règles floue	Ontologie + attributs flous
[McKeever et al., 2010]	Orienté connaissance	DS	DAG
[Sebbak et al., 2014]	Orienté connaissance	DS	DAG

Ces différentes approches partagent des caractéristiques communes :

— Les données brutes de bas niveau sont transformées en des informations de haut niveau porteuses de sens (e.g. actions, activités)

— Le traitement et le raisonnement sur les données de capteurs s'effectuent d'une manière centralisée. En effet, l'approche centralisée intègre un modèle de reconnaissance déjà construit et identifie les activités au fur et à mesure que l'environnement change de contexte.

— Différentes techniques de raisonnement et modélisation sont utilisées pour gérer une problématique dans le domaine de la reconnaissance d'activités telles que la gestion de l'incertitude et la gestion temporelle.

Néanmoins, ces approches font face aux contraintes suivantes :

— Les capteurs déployés dans les maisons intelligentes sont hétérogènes (e.g capteurs de mouvement, thermomètre, capteurs de présence, etc.) et répartis dans les différentes pièces de la maison.

— La gestion du flux de données de capteurs entrant peut diminuer les performances du système en terme de taux de traitement lors de la reconnaissance d'activité.

— La communication avec une entité centrale n'est pas toujours garantie et les commu-

nications sans fil sont généralement peu fiables et limitées sous certaines conditions.

Pour remédier à cela, une approche totalement décentralisée est nécessaire permettant de répartir à la fois les sources de données et le raisonnement. Étant donné que les maisons intelligentes sont des environnements ouverts, dynamiques et imparfaits, les principaux défis auxquels nous devons faire face dans la reconnaissance d'activité distribuée sont :

— **D1 : La gestion des données de capteurs** : comment gérer l'importance et la rapidité du flux de données de capteurs entrants ? Le système HAR distribué doit avoir la capacité à s'adapter à une croissance non bornée de la taille des données entrantes issues des capteurs.
— **D2 : La fraîcheur des données** : comment éviter d'avoir des données de capteurs obsolètes ? Avec le changement continu des données de capteurs, le modèle d'activité risque d'être ancien et obsolète.
— **D3 : l'identification** : comment identifier les activités humaines courantes à partir d'un historique des comportements de l'habitant dans la maison ? Le système doit étudier et apprendre le comportement des habitants.
— **D4 : La précision** : comment augmenter le taux de la reconnaissance d'activité ? la précision du système HAR est importante afin d'évaluer sa performance.
— **D5 : L'hétérogénéité** : comment gérer la diversité des données de capteurs ? Le système HAR doit prendre en compte la variété des données en termes de forme, de fréquence, etc.
— **D6 : L'incertitude** : comment faire confiance aux données provenant d'autres agents ? Le système HAR doit faire face à la gestion de l'incertitude lors de la communication entre les agents.
— **D7 : La gestion de conflits** : comment faire face à des données qui s'opposent provenant de différents agents ? Le système HAR doit mettre en place des stratégies de résolution de conflits.

Le besoin d'avoir une approche décentralisée pour la reconnaissance d'activité dans les maisons intelligentes a récemment fait l'objet de quelques travaux. Cependant, ces travaux ne traitent qu'une partie des défis décrits ci-dessus. Dans ce travail, nous considérons ces défis mentionnés ci-dessus comme des buts à atteindre.

Le but de cette contribution est de proposer une approche de raisonnement collaboratif totalement distribuée pour HAR. Avant d'entamer la description détaillée de notre approche, la section suivante évoque les travaux récents proposés dans la littérature pour la reconnaissance d'activité distribuée.

4.3 Travaux connexes sur la reconnaissance d'activité humaine basée sur une architecture distribuée

La reconnaissance d'activité humaine distribuée est un champ de recherche qui a pris de l'ampleur au cours des dernières années. Des travaux récents se sont investis dans la reconnaissance d'activité distribuée. Nous pouvons regrouper ces travaux selon leur architecture :

— Approches client-serveur : les auteurs, dans [Fortino et al., 2015, Cicirelli et al., 2016], proposent une plateforme qui s'appuie principalement sur l'architecture CASE (Cloud-assisted Agent-based Smart home Environment) et qui est composée de trois couches : la couche IoT, la couche Virtualization et la couche Analytics. Cette dernière est composée de plusieurs nœuds. Chaque nœud contient un agent serveur qui héberge un nombre limité d'agents et leur permet d'exécuter et d'interagir les uns avec les autres de manière transparente. La tâche de reconnaissance d'activité utilise des attributs et produit en sortie de nouveaux agents classifieurs (lorsqu'une nouvelle activité est découverte) qui sont déployés de manière dynamique sur les agents serveurs.

— Approches hiérarchiquement distribuées : les auteurs, dans [Amft and Lombriser, 2011], introduisent un détecteur événement-activité (activity-event-detector AED) pour les systèmes de reconnaissance d'activité distribués basés sur des graphes acycliques directs. C'est un ensemble de nœuds de collecte et de détection (les détecteurs) distribué où chaque détecteur procède à l'acquisition des données locales, à un repérage des activités atomiques et communique des informations de type évènement à d'autres nœuds dans le réseau. Par ailleurs, les auteurs de [Marin-Perianu et al., 2008] proposent une architecture de reconnaissance d'activité basée sur la logique floue, à travers laquelle différent nœuds collaborent pour produire un résultat de reconnaissance fiable à partir des données de capteurs non fiables. Les trois étapes principales sont : la détection d'événements simples, la combinaison d'événements avec des opérations basiques et la classification finale des activités.

— Approches totalement distribuées : les auteurs, dans [Wang and Ji, 2014], proposent une approche de détection d'activité anormale distribuée (DetectingAct) qui emploie un ensemble de nœuds capteurs pour détecter les activités anormales. Dans DetectingAct, un nœud détecte les activités normales en utilisant une technique de fouille de données : les motifs fréquents via l'algorithme FP-tree. Ensuite, l'activité est étiquetée par chaque nœud qu'elle traverse. Son statut devient anormal dès qu'elle est

marquée comme une activité anormale par au moins un nœud. De plus, les auteurs, dans [Mosabbeb et al., 2013], proposent un modèle de machines à vecteurs de support (Suport Vector Machine SVM) distribué multi-vues pour les réseaux de capteurs types caméra où différents nœuds intègrent des classifieurs de type SVM. Afin de rassembler les résultats d'apprentissage des différentes vues, une régularisation est utilisée pour concaténer toutes ces vues. Enfin, les auteurs de [Ramakrishnan et al., 2013a, Ramakrishnan et al., 2013b, Ramakrishnan et al., 2014] proposent une plateforme bayésienne modulaire et distribuée qui est une collection de réseaux bayésiens (Bayesian Networks BN) où chaque BN modélise un contexte de haut niveau unique. Le nœud parent de chaque BN est le nœud de contexte de haut niveau à inférer et leurs nœuds enfants respectifs correspondent aux capteurs utilisés pour inférer le contexte haut niveau de leurs parents.

Le Tableau 4.2 compare les différentes approches pour la reconnaissance d'activité distribuée discutées ci-dessus selon six paramètres (Modèle d'activité utilisé, l'architecture, Type de communication, les entrées, l'aspect collaboratif).

Malgré leurs différences majeures par rapport à leurs architectures, toutes ces approches (voir Tableau 4.2) proposent une approche HAR distribuée où les données de capteurs sont traitées d'une manière ascendante (bottom-up) pour identifier les activités. De plus, dans certaines approches, les entités utilisent des modèles d'activité orientés donnée (les classifieurs KNN et SVM, FP-tree) pour le raisonnement sur les données de capteurs. Les modèles orientés donnée sont utilisés dans le cas où on a un volume important de données de capteurs. Néanmoins, ces approches présentent des limites :

— *Système distribué* : les approches utilisent soit des réseaux bayésiens soit des réseaux de capteurs sans fil (Wireless Sensor Network WSN). Cependant, les réseaux de capteurs sans fil ont des contraintes de ressources et sont limités en termes de bande passante [Mahmood et al., 2013]. Les réseaux bayésiens ne sont pas flexibles. Si un fragment de réseau d'origine est modifié, il faut changer tout le modèle utilisant ce fragment. C'est un processus manuel couteux [Wei, 2001]. Ces lacunes peuvent être résolues en introduisant le paradigme orienté agents. Ce dernier est particulièrement approprié dans de tel scénario car les agents offrent des fonctionnalités importantes telles que le raisonnement proactif et réactif, l'autonomie, les aptitudes sociales et l'apprentissage [Maciel et al., 2015]. En effet, les systèmes multi-agents représentent un paradigme bien adapté pour la modélisation des maisons intelligentes, facilitant le traitement des données locales et la décision finale est prise suite à une collaboration

TABLE 4.2. COMPARAISON DES APPROCHES HAR DISTRIBUÉES

Système	*Système distribué*	*Modèle d'activité*	*Architecture*	*Communication*	*Input*	*Collaboration*
Système CASE [Cicirelli et al., 2016, Fortino et al., 2015]	Système multi-agent	Classifieurs KNN	Client-Serveur	Agent serveur - agents	Ensemble d'évènements	Non (pas de collab. entre agents serveurs)
Système DetectingAct [Wang and Ji, 2014]	Réseau de capteurs sans fil	L'algorithme FP-tree	Totalement distribuée	Un noeud-un sous ensemble de noeuds	Données de trajectoire et durée	Oui (partage avec les voisins pour marquer une activité normale ou anormale)
Système SVM distribué multi-vues [Mosabbeb et al., 2013]	Un ensemble de noeuds capteurs (cameras)	Classifieurs SVM	Totalement distribuée	Un noeud-tous les nœuds	Données de caméra (dataset IXMAS)	Oui (régularisation des différentes vues)
Plateforme bayésienne distribuée [Ramakrishnan et al., 2013a, Ramakrishnan et al., 2014, Ramakrishnan et al., 2013b]	Un ensemble de noeuds	Réseaux bayésiens	Totalement distribuée	Un noeud-tous les nœuds	Capteurs binaires, données d'accéléromètre, etc.	Oui (Génération d'une vue globale)
Système de reconnaissance distribué [Amft and Lombriser, 2011]	Un ensemble de noeuds	Graphe AED	Hiérarchiquement distribuée	Parent-enfants	Actions composant l'activité	Oui (Évènements détectés sont communiqués)
Système HAR flou distribué [Marin-Perianu et al., 2008]	Réseau de capteurs sans fil (détecteurs)	Inférence floue	Hiérarchiquement distribuée	Parent-enfants	Données d'accéléromètres	Non (pas de collab. entre noeuds dans la même couche)
Notre approche	Système multi-agent	Différent types de classifieurs construits sur différentes bases d'apprentissage	Totalement distribuée	Un agent-Un sous ensemble d'agents	Ensemble d'évènements (Dataset Aruba)	Oui (Partage des données avec les agents d'accointances pour identifier l'activité finale)

de manière distribuée.

— *Modèle d'activité* : toutes les entités adoptent la même méthode de raisonnement (modèle d'activité). Le choix de celui-ci dépend de la nature des données des capteurs. Cependant, les données de capteurs sont différentes d'un endroit à un autre dans la maison. Par conséquent, la méthode de raisonnement doit être différente d'une entité à une autre.

— *Communication* : pour les approches totalement distribuées, la troisième et la quatrième approche présentent une communication avec toutes les entités. D'un autre côté, dans la deuxième approche, la communication s'effectue avec certaines entités prédéfinies à l'avance. Afin de surmonter, respectivement, le coût élevé de la communication et de répondre aux besoins de traitement des données en ligne, la

TABLE 4.3. POSITION DES APPROCHES PAR RAPPORT AU DÉFIS FIXÉS

Système	*D1 : Gestion de données de capteurs*	*D2 : fraîcheur des données*	*D4 : précision*	*D5 : hétérogénéité*	*D6 : incertitude*	*D7 : gestion de conflits*
Système CASE [Cicirelli et al., 2016, Fortino et al., 2015]	✓	✗	✓	✗	✗	✗
Système DetectingAct [Wang and Ji, 2014]	✓	✓	✓	✗	✗	✓
Système SVM distribué multi-vues [Mosabbeb et al., 2013]	✓	✗	✓	✗	✗	✓
Plateforme bayésienne distribuée [Ramakrishnan et al., 2013a, Ramakrishnan et al., 2014, Ramakrishnan et al., 2013b]	✓	✗	✓	✗	✗	✓
Système de reconnaissance distribué [Amft and Lombriser, 2011]	✗	✓	✗	✗	✗	✓
Système HAR flou distribué [Marin-Perianu et al., 2008]	✗	✗	✗	✗	✗	✗
Notre approche	✓	✓	✓	✓	✓	✓

communication avec un ensemble d'entités devrait être définie dynamiquement.

— *Collaboration* : la plupart des approches utilisent la collaboration entre les différentes entités par le partage d'informations pour avoir une vision globale du système. Ceci permet d'améliorer le taux de reconnaissance des activités.

Toutefois, aucune de ces approches a abordé le problème de l'incertitude des données communiquées. Les entités font confiance en toutes les données fournies par les autres lors de la collaboration.

Le Tableau 4.3 positionne ces différentes approches par rapport aux défis fixés dans la section 4.2 (✓ : défi géré ; ✗ : défi non géré). Certes, ces approches traitent une partie des défis mais il n'existe aucune approche ayant répondu à tous ces défis.

À cet égard, nous proposons une approche de raisonnement collaboratif distribuée fournissant les propositions suivantes :

— **P1** : une approche multi-agent totalement distribuée qui assure une distribution complète du processus de raisonnement et des données via les entités.

— **P2** : une approche ascendante devrait garantir la fraîcheur des données.
— **P3** : enrichir les agents par des modèles d'activités d'apprentissage automatique (classifieurs basés sur des techniques d'apprentissage supervisé).
— **P4** : assurer la communication et la collaboration avec un sous ensemble dynamique d'agents afin d'augmenter le taux de la reconnaissance d'activité.
— **P5** : adopter des agents avec différents types de modèles d'activité suivant la nature des données de capteurs.
— **P6** : attribuer un degré de confiance pour les activités reconnues.
— **P7** : trois stratégies de résolution de conflits sont proposées.

La section suivante présente en détail notre approche de raisonnement collaboratif distribuée pour la reconnaissance d'activités humaines.

4.4 L'approche de raisonnement distribuée DCR

Dans cette section, nous introduisons notre contribution nommée Raisonnement Collaboratif Distribué (Distributed Collaborative Reasoning *DCR*) en présentant sa position par rapport à l'architecture globale décrite dans le chapitre *Introduction*, le comportement de ses agents, ses algorithmes et leurs propriétés, à savoir, la terminaison, la complexité et le nombre de messages et enfin les différentes stratégies de résolution de conflits adoptés par ses agents.

Nous rappelons l'architecture globale de HAR distribuée adaptée à notre scénario d'une maison intelligente illustrée dans la figure 4.1.

Nous nous intéressons dans ce chapitre à la phase d'identification des activités et nous proposons à cet égard notre approche *DCR*. Les données d'entrée de cette phase d'identification sont les données de sortie générées depuis la phase *Observation* c'est à dire un ensemble de vecteurs d'attributs réduits. Puisque les étapes *segmentation* et *extraction d'attributs* ne font pas l'objet de cette thèse, nous nous sommes basés sur des résultats fournis par les auteurs de [Yala et al., 2017] qui sont les vecteurs d'attributs générés depuis un jeu de données publique (détaillé dans le chapitre suivant).

La phase d'identification utilise la liste des attributs générée par le module observation. Un ensemble d'agents identificateurs sont placés dans les pièces de la maison et sont rattachés aux agents observateurs.

DCR modélise une maison intelligente comme étant un système multi-agent $MAS = \{A_{l_1}, A_{l_2}, ..., A_{l_n}\}$ où chaque agent A_{l_i} est assigné à une pièce ou à une localisation l_i (e.g.

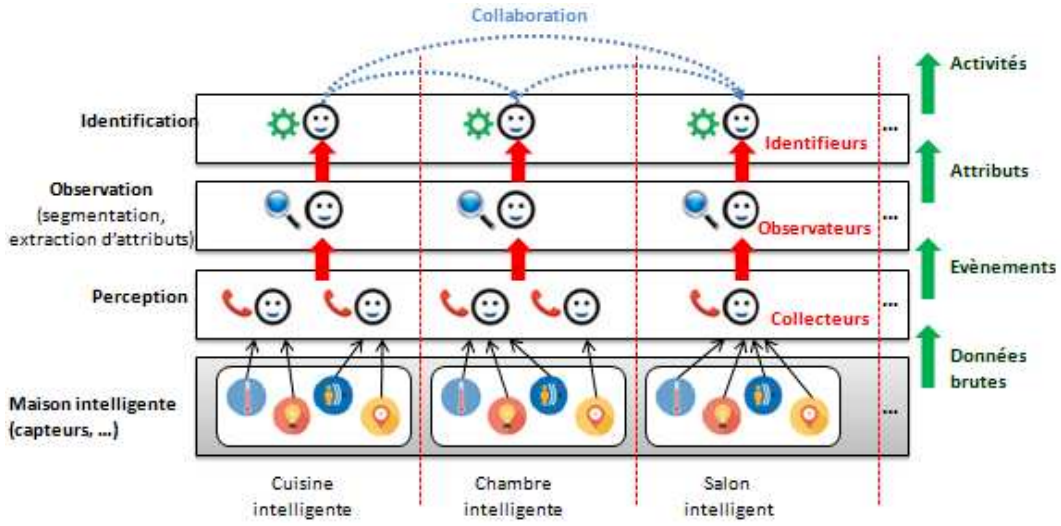


FIGURE 4.1. Architecture globale de HAR distribué

chambre, cuisine, salon, etc.) dans la maison. Les agents sont des agents autonomes et possèdent les mêmes fonctionnalités. Chaque agent $A_{l_i} \in MAS$ est défini par un tuple $A_{l_i} = (id_{A_{l_i}}, clf_{A_{l_i}}, ACT_{A_{l_i}}, ACQ_{A_{l_i}})$ où :

- $id_{A_{l_i}}$: représente l'identifiant de l'agent A_{l_i} dans la localisation l_i.
- $clf_{A_{l_i}}$: est un classifieur associé à un agent identificateur A_{l_i}. Il existe dans la littérature plusieurs types de classifieurs. Nous nous sommes particulièrement intéressés à ceux basés sur un raisonnement inductif tels que les forêts aléatoires (Random Forest RF), les arbres de décision (Decision Tree DT), les extra-arbres (Extra-Trees ExT) (basés sur le mécanisme de chaînage avant) et le réseau bayésien naïf (Bayes Naive BN) (basé sur le raisonnement probabiliste). Le choix de ces classifieurs est motivé par : i)- ils partent depuis des observations pour aboutir à la classe ; ii)- nous pouvons générer un ensemble de règles depuis ces classifieurs. Nous citons ces travaux qui détaillent la génération des règles à partir de ces classifieurs : le naif bayes [Śnieżyński, 2006], l'arbre de décision [Quinlan, 1987], la forêt aléatoire [Mashayekhi and Gras, 2015]. Le but de générer des règles à partir de ces modèles est d'explorer ces boites noires et d'améliorer leurs interprétations. La génération des règles n'entre pas dans le cadre de cette thèse mais ça reste un travail futur.
- $ACT_{A_{l_i}} = \{< a1_{(l_i)}, d1_{(l_i)} >, < a2_{(l_i)}, d2_{(l_i)} >, ..., < am_{(l_i)}, dm_{(l_i)} >\}$: est une liste de $m_{(l_i)}$ activités déjà reconnues par l'agent A_{l_i} via son classifieur $clf_{A_{l_i}}$ relative à la localisation l_i. Ces $m_{(l_i)}$ activités sont pondérées par des degrés de confiance $d1_{(l_i)}, d2_{(l_i)}, ..., dm_{(l_i)}$. Ces derniers expriment le degré de vérité d'une activité. La liste $ACT_{A_{l_i}}$ est prédéfi-

nie à l'avance lors de la construction du classifieur $clf_{A_{l_i}}$ et est fixe. Le degré de confiance $d_{k_{(l_i)}}$ est calculé en utilisant la métrique de F1-score (définie dans [Ting, 2010] et décrite dans l'Annexe A.2) pour chaque activité $a_{k_{(l_i)}}$ ($k \in \{1,..,m\}$) dans la liste $ACT_{A_{l_i}}$.

— $ACQ_{A_{l_i}}$: est la liste des accointances de l'agent A_{l_i}. Elle est dynamiquement réajustée et contient l'ensemble des agents voisins avec lesquels l'agent A_{l_i} peut collaborer. L'agent A_{l_i} détermine cette liste lorsqu'il a du mal à reconnaître localement l'activité courante (c'est à dire dont le taux de reconnaissance est inférieur à un seuil δ fixé par un expert). Ainsi, il choisit certains agents qui reconnaissent mieux cette activité. Dans le pire des cas, cette liste peut contenir tous les autres agents. Pour éviter ce cas, nous définissons un seuil ω représentant un nombre restreint d'agents avec lesquels l'agent A_{l_i} va collaborer, où $|\omega| < |MAS - 1|$.

Tous les agents dans *MAS* possèdent les caractéristiques décrites ci-dessus et résumées dans la Figure 4.2. Chaque agent prend en entrée un vecteur d'attributs (Feature Vector *FV*) qui est le résultat des deux modules *Perception* puis *Observation* et génère une seule activité finale. Pour ce faire, ces agents exhibent un comportement autonome présenté dans la section suivante.

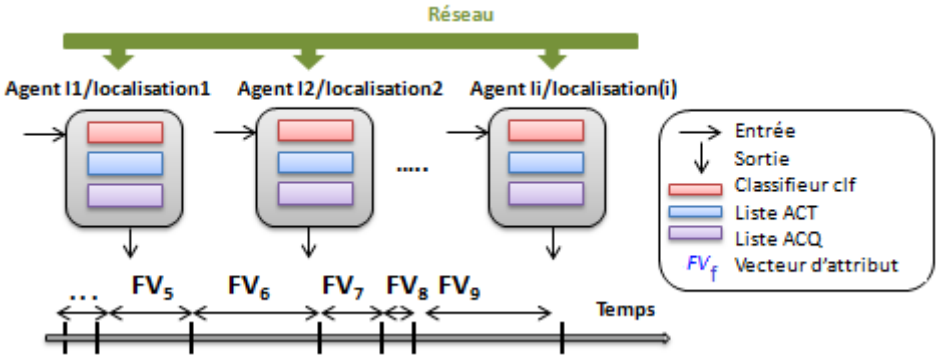


FIGURE 4.2. Composition des agents dans *DCR*

4.5 Comportement des agents dans DCR

Les agents dans *MAS* possèdent un cycle de vie suite à l'arrivée d'un vecteur d'attributs *FV* à partir du module observation. Le cycle de vie de ces agents est illustré dans la Figure 4.3 qui représente le diagramme d'états. Ce diagramme décrit les étapes du processus permettant d'aboutir à la reconnaissance d'une activité relative au vecteur d'attributs *FV*. Les étapes sont

enchaînées comme suit :

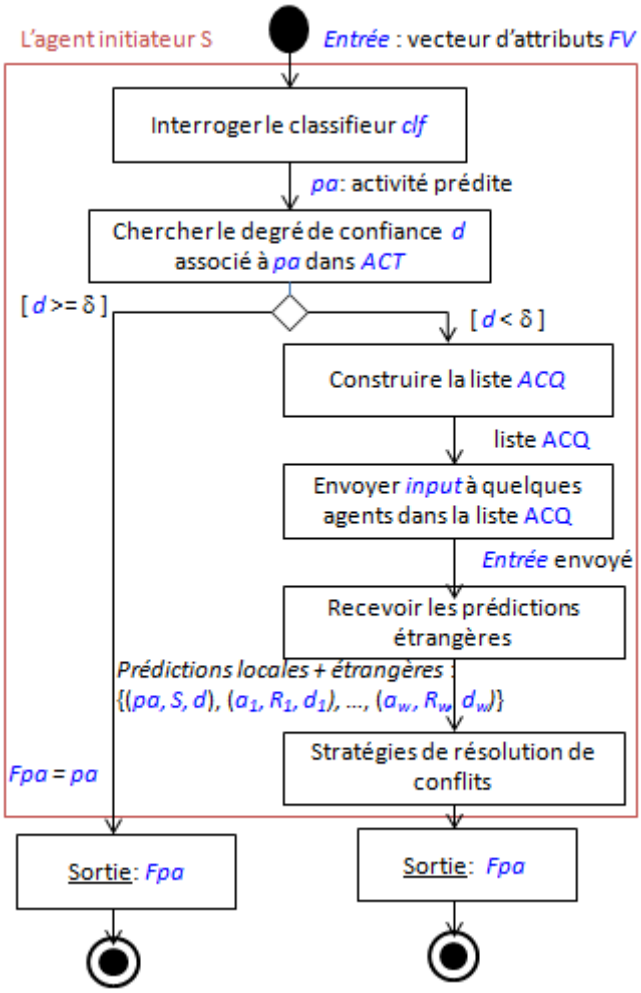


FIGURE 4.3. Diagramme d'état d'un agent initiateur

1. Ayant comme entrée un vecteur d'attributs, l'agent A_{l_i} interroge son classifieur local $clf_{A_{l_i}}$ pour obtenir l'activité prédite $pa_{A_{l_i}}$.
2. L'agent A_{l_i} détermine le degré de confiance d_{pa} relatif à l'activité prédite $pa_{A_{l_i}}$ à partir de la liste $ACT_{A_{l_i}}$.
3. Selon la valeur du degré d_{pa}, nous distinguons deux cas :
   (a) Soit l'agent A_{l_i} reconnaît l'activité prédite $pa_{A_{l_i}}$ dans sa liste $ACT_{A_{l_i}}$ avec un degré de confiance supérieur au seuil δ ($d_{pa} >= \delta$ avec δ un seuil fixé par un expert), alors il ne va pas collaborer avec les autres agents et se contente uniquement de sa prédiction locale et la génère en sortie comme une activité finale($Fpa_{A_{l_i}} = pa_{A_{l_i}}$).

(b) Soit l'agent A_{l_i} reconnaît l'activité prédite $pa_{A_{l_i}}$ dans sa liste $ACT_{A_{l_i}}$ avec un degré de confiance inférieur au seuil δ ($d_{pa} < \delta$), dans ce cas, il a besoin de consulter les autres agents pour confirmer ou infirmer sa prédiction locale. L'agent A_{l_i} procède par les sous étapes suivantes :

i. Construite sa liste d'accointance $ACQ_{A_{l_i}}$. Cette dernière contient l'ensemble des agents qui reconnaissent l'activité $pa_{A_{l_i}}$.

ii. Envoyer son vecteur d'attribut FV (son input) à certains agents de sa liste $ACQ_{A_{l_i}}$. Ces derniers sont sélectionnés si le degré de confiance de l'activité est supérieur au degré de confiance locale d_{pa}.

iii. Recevoir les activités prédites avec leurs degrés de confiance $\{< a_1, d_1 >, < a_2, d_2 >, ..., < a_{|ACQ_{A_{l_i}}|}, d_{|ACQ_{A_{l_i}}|} >\}$ à partir des autres agents sélectionnés.

iv. Appliquer des stratégies de résolution de conflits si un conflit survient i.e lorsque la prédiction locale est différente des prédictions reçues.

v. Générer en sortie la prédiction ou l'activité finale $Fpa_{A_{l_i}}$ qui peut être différente de l'activité prédite en local $pa_{A_{l_i}}$.

La sous section suivante fournit plus de détails sur le système *DCR* à travers la description des algorithmes *DCR*.

4.6 Les algorithmes du système DCR

Nous présentons les différents algorithmes du système *DCR*. Un agent est déclenché dans les deux situations suivantes :

— Lorsqu'un nouveau vecteur d'attribut FV arrive. Dans ce cas, cet agent est nommé *l'agent initiateur*.

— Lorsqu'une demande d'un autre agent arrive. Dans ce cas, cet agent est nommé *l'agent récepteur*. Cette demande est lancée pour deux raisons :

1. Afin de déterminer sa liste d'accointances $ACQ_{A_{l_i}}$, l'agent initiateur A_{l_i} demande à tous les autres agents récepteurs de vérifier si l'activité locale prédite appartient à leurs listes d'activité $ACT_{A_{l_j}}$ ($j \neq i$).
2. Afin de déterminer l'activité finale, l'agent initiateur A_{l_i} collabore avec certains agents sélectionnés à partir de sa liste d'accointance $ACQ_{A_{l_i}}$ en leur envoyant son entrée (son vecteur d'attributs FV) pour obtenir leurs prédictions correspondantes.

L'enchaînement des algorithmes *DCR* effectués par l'agent initiateur et l'agent récepteur sont décrits par le diagramme de séquence Figure 4.4.

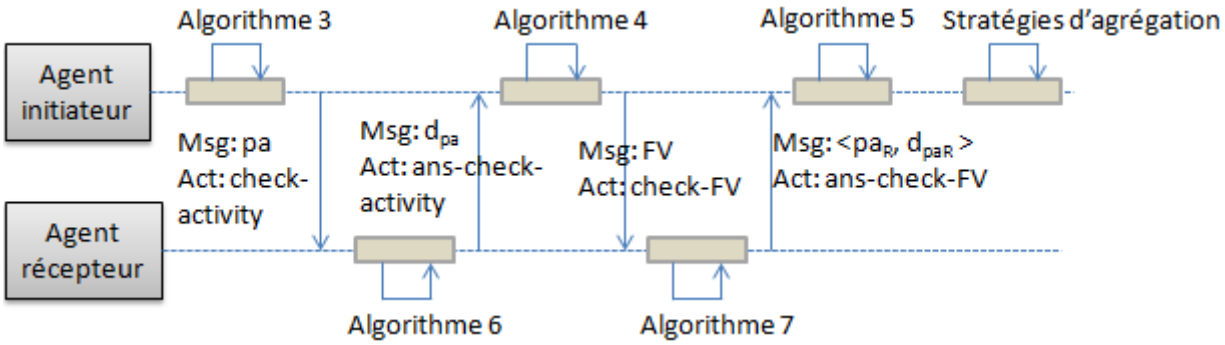


FIGURE 4.4. Diagramme de séquence de l'approche *DCR*

Nous commençons par présenter les algorithmes de l'agent initiateur puis nous poursuivons avec ceux de l'agent récepteur.

4.6.1 Les algorithmes de l'agent initiateur

L'agent initiateur possède trois algorithmes qui se déclenchent dans l'ordre suivant : l'Algorithme 3, l'Algorithme 4 et l'Algorithme 5.

— **Traitement d'un vecteur d'attributs (voir Algorithme 3)** cet algorithme prend en entrée le vecteur d'attributs FV et génère en sortie une liste de triplets $pAct\_list_{(S)} = \{(pa_{(S)}, S, d_{pa_{(S)}}), (pa_{(R_1)}, R_1, d_{pa_{(R_1)}}), (pa_{(R_2)}, R_2, d_{pa_{(R_2)}}), ..., (pa_{(R\omega)}, R\omega, d_{pa_{(R\omega)}})\}$. Ce triplet est constitué par l'activité prédite pa de l'agent S ou R_f ($f \in \{1, ..., \omega\}$) avec son degré de confiance d_{pa}. Cet agent initiateur S commence à charger son classifieur clf (ligne 3), à prévoir l'activité $pa_{(S)}$ (ligne 4) et à déterminer le degré de confiance associé $d_{pa_{(S)}}$ (ligne 5). Si ce dernier est supérieur au seuil δ, alors l'agent initiateur S reconnaît bien l'activité pa_S. Ainsi, il est inutile de communiquer et de collaborer avec les autres agents car nous risquons de dégrader le taux de reconnaissance de $pa_{(S)}$. Si le degré de confiance $d_{pa_{(S)}}$ est inférieur au seuil δ, alors l'agent initiateur S envoie un message contenant l'activité locale prédite $pa_{(S)}$ à tous les agents afin de trouver les agents qui sont capables de la reconnaître (lignes 9-10).

— **Traitement de la réponse pour la demande "check-activity" (voir Algorithme 4)** cet algorithme a pour but de déterminer la liste d'accointance $ACQ_{(S)}$ de l'agent initiateur S. Cette liste contient les agents qui peuvent identifier l'activité $pa_{(S)}$. L'algorithme prend en entrée le message retour de l'agent récepteur R pendant une durée

Δt. Si l'agent récepteur R comporte dans sa liste d'activité $ACT_{(R)}$ l'activité $pa_{(S)}$, alors le message contient le degré de confiance correspondant $d_{pa_{(R)}}$. Après une durée de Δt, aucun message ne sera traité. Si le degré de confiance $d_{pa_{(R)}}$ reçu est supérieur au degré de confiance local $d_{pa_{(S)}}$, alors l'agent initiateur S envoie son vecteur d'attributs FV à l'agent récepteur correspondant (lignes 6 et 7).

— **Traitement de la réponse pour la demande "check-FV" (voir Algorithme 5)** l'agent initiateur S reçoit toutes les activités prédites avec leurs degrés de confiance à partir des agents sélectionnés dans la liste $ACQ_{(S)}$. En effet, chaque message msg contient la réponse de l'agent sélectionné qui identifie correctement l'activité $pa_{(S)}$ (ligne 3). L'agent initiateur S rajoute la réponse dans sa liste $pAct\_list_{(S)}$ et remplace la dernière ligne du fichier résultat $rFile$ en concaténant l'ensemble des activités prédites par les autres avec la nouvelle activité en question. Le fichier résultat $rFile$ comporte des lignes où chaque ligne est une liste des activités prédites y compris l'activité locale prédite et les activités prédites reçues, l'agent source et les degrés de confiance associés.

ALGORITHME 3. Traitement d'un vecteur d'attributs

Input :
FV : un vecteur d'attributs ; S : un agent initiateur ; O : liste des autres agents ; δ : un seuil
$ACT_S = \{< a_1, d_1 >, < a_2, d_2 >, ..., < a_m, d_m >\}$: une liste d'activités reconnues avec leurs degrés de confiance
Output :
$pAct\_list_{(S)} =$
$\{(pa_{(S)}, S, d_{pa_{(S)}}), (pa_{(R_1)}, R_1, d_{pa_{(R_1)}}), (pa_{(R_2)}, R_2, d_{pa_{(R_2)}}), ..., (pa_{(R_\omega)}, R_\omega, d_{pa_{(R_\omega)}})\}$: liste des activités prédites, l'agent source et les degrés de confiance

```
1  begin
2      initialise(pAct_list)
3      clf = loadClassifier()
4      pa_S = predictActivity(clf, FV)
5      d_pa(S) = getDegreeFrom(pa_S, ACT_S)
6      if d_pa(S) ≥ δ then
7          pAct_list.add(< pa_S, S, d_pa(S) >)
8      else
9          for o_i ∈ O do
10             Send message(pa_S, S, o_i, act : "check − activity")
11     return(pAct_list)
```

Algorithme 4 . Traitement de la réponse pour la demande *check-activity*

Input :
msg : message reçu ; R : agent récepteur ; S : agent initiateur ; act : action ; FV : vecteur d'attributs
pa_S : activité prédite ; $d_{pa_{(S)}}$: degré de confiance of pa_S
Output :
$ACQ_{(S)}$: liste d'accointances

```
1 begin
2     if (act = "ans − check − activity") then
3         d_pa(R) = msg.getContent()
4         if d_pa(R) <> 0 then
5             ACQ(S).add(R)
6             if d_pa(R) > d_pa(S) then
7                 Send message(FV, S, R, act : "check − FV")
8     return(ACQ(S))
```

4.6.2 Les algorithmes de l'agent récepteur

L'agent récepteur possède deux algorithmes qui se déclenchent dans l'ordre suivant : l'Algorithme 6 puis l'Algorithme 7.

— **Traitement de la demande "check-activity" (voir Algorithme 6)** : cet algorithme détermine si l'agent récepteur R peut reconnaître l'activité prédite $pa_{(S)}$ de l'agent initiateur S. En effet, l'agent récepteur vérifie si $pa_{(S)}$ est dans sa liste d'activité $ACT_{(R)}$. Si c'est le cas, il répond à l'agent initiateur S en envoyant le degré de confiance associé $d_{pa_{(R)}}$ (lignes 6 et 7). Sinon, il n'envoie rien.

— **Traitement de la demande "check-FV" (voir Algorithme 7)** : L'agent récepteur R qui reconnaît l'activité $pa_{(S)}$ mieux que l'agent initiateur S, sera sélectionné et recevra un message par ce dernier. Ce message contient une demande pour traiter le vecteur d'attribut $FV_{(S)}$ afin d'identifier l'activité en question. Par conséquent, l'agent récepteur R charge son classifieur $CLF_{(R)}$ (ligne 4), prédit l'activité $pa_{(R)}$ relative à $FV_{(S)}$ (ligne 5), détermine son degré de confiance $d_{(R)}$ (ligne 6) et répond à l'agent initiateur en envoyant un message de retour contenant l'activité prédite et son degré de confiance $< pa_{(R)}, R, d_{(R)} >$ (ligne 7).

ALGORITHME 5 . Traitement de la réponse pour la demande *check-FV*

Input :
msg : message reçu ; S : agent initiateur ; R : agent récepteur
$pAct\_list_S$: liste des activités prédites avec leurs degrés de confiance
Output :
$rFILE$: fichier résultat

begin
  if $act =$ "$ans-check-FV$" **then**
    $< pa_{(R)}, R, d_{(R)} >= msg.getContent()$
    $pAct\_list_{(S)}.add(< pa_{(R)}, R, d_{(R)} >)$
    $last\_line = read\_last\_line(rFILE)$
    $concat(last\_line, < pa_{(R)}, R, d_{(R)} >)$
    $replaceLastLine(rFILE, last\_line)$
  $return(rFILE)$

ALGORITHME 6 . Traitement de la demande *check-activity*

Input :
msg : message reçu ; S : agent initiateur ; R : agent récepteur ; act : action
Output :
$d_{pa_{(R)}}$: degré de confiance

begin
  if $(act =$ "$check-activity$") **then**
    $pa_{(S)} = msg.getContent()$
    $ACT_{(R)} = R.getACTlist()$
    if $pa_{(S)} \in ACT_{(R)}$ **then**
      $d_{pa_{(R)}} = R.getDegreeFrom\ (pa_{(S)}, ACT_{(R)})$
      Send $message(d_{pa_{(R)}}, R, S, act :$ "$ans-check-activity$")
  $return(d_{pa_{(R)}})$

4.6.3 Propriétés des algorithmes DCR

Dans cette section, nous discutons certaines propriétés des algorithmes *DCR* à savoir la terminaison, le nombre de messages et la complexité. Nous les présentons comme suit :

— **Terminaison** : nous supposons que le système multi-agent $MAS = \{A_{l_1}, A_{l_2}, ..., A_{l_n}\}$ comporte un nombre fini d'agents. De plus, la variable Δt définie par l'agent initiateur, représente la durée maximale pour avoir les réponses des agents récepteurs. L'agent initiateur ne sera jamais en attente. Ainsi, le système *MAS* se termine toujours.

— **Nombre de messages** : Considérons n le nombre total d'agents, l'échange de mes-

ALGORITHME 7 . Traitement de la demande *check-FV*

Input :
msg : message reçu ; S : agent initiateur ; R : agent récepteur ; act : action ; $clf_{(R)}$: classifieur de l'agent récepteur
Output :
$< pa_{(R)}, R, d_{(R)} >$: triplet contenant l'activité prédite de l'agent récepteur R et son degré de confiance

```
1 begin
2     if (act = "check − FV") then
3         FV_(S) = msg.getContent()
4         clf_(R) = loadClassifier()
5         pa_(R) = predictActivity(clf_(R), FV_(S))
6         d_(R) = getDegreeFrom(ACT_(R), pa(R))
7         Send message(< pa_(R), R, d_(R) >, S, act : "ans − check − FV")
8     return(< pa_(R), R, d_(R) >)
```

sages entre l'agent initiateur S et l'agent récepteur R s'effectue dans ces deux situations :

1. Lors de la construction de la liste d'accointance $ACQ_{(S)}$: les messages s'échangent entre l'agent initiateur S et tous les autres agents. Dans ce cas, le nombre de messages échangés vaut : $2 \times (n-1)$.

2. Durant la collaboration avec certains agents sélectionnés dans la liste d'accointance $ACQ_{(S)}$ de l'agent initiateur S : dans le pire des cas, tous les agents dans la liste $ACQ_{(S)}$ reconnaissent l'activité prédite $pa_{(S)}$ mieux que l'agent initiateur S. Sauf que nous avons limité le nombre maximal d'agents avec lesquels l'agent S collabore avec la variable ω. Ainsi, le nombre de messages échangés vaut : $2 \times \omega$.

Par conséquent, le nombre de messages échangés entre l'agent initiateur S et tous les agents récepteurs R pour un vecteur d'attribut FV est égal à : $NbrMsg = 2 \times (n-1+\omega)$. Supposons que nous avons m_{FV} vecteurs d'attributs, le nombre total de messages échangés est :

$$ToT = m_{FV} \times NbrMsg = \mathcal{O}(\omega \times n \times m_{FV})$$

Toutefois, le nombre de vecteurs d'attributs m_{FV} est très important par rapport au nombre d'agents $\omega < n$. A ce moment là, le nombre total de messages échangés devient linéaire par rapport au nombre de vecteur d'attributs

$$ToT = \mathcal{O}(m_{FV})$$
$$\text{où } |\omega| < |n| << |m_{FV}|$$

— **Complexité** : la complexité des différents algorithmes est illustrée dans le Tableau 4.4 pour le traitement d'un seul vecteur d'attributs FV.

TABLE 4.4. Complexité des algorithmes *DCR*

Algorithme 3	Algorithme 4	Algorithme 5	Algorithme 6	Algorithme 7
$\mathcal{O}(O) = \mathcal{O}(n)$ où $\|O\| < n$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(ACT_{(R)}) = \mathcal{O}(1)$	$\mathcal{O}(1)$

Supposons que nous avons m_{FV} vecteurs d'attributs, la complexité de calcul global des algorithmes *DCR* pour un seul agent est proportionnelle au nombre des vecteurs d'attributs m_{FV} :

$$C_{(DCR)} = \mathcal{O}(n \times m_{FV}) = \mathcal{O}(m_{FV}) \text{ où } |n| << |m_{FV}|$$

4.7 Stratégies de résolution de conflits dans DCR

Un système multi-agent est un ensemble d'agents communiquant et interagissant entre eux pour atteindre des objectifs individuels ou collectifs [Wooldridge, 2009] . Cependant, les agents peuvent parfois présenter des opinions différentes ou contradictoires ce qui entraîne une inconsistance du système. Du coup, la résolution des conflits est essentielle pour le comportement coordonné des agents. Ces dernières années, les chercheurs ont développé diverses stratégies de résolution de conflits pour les systèmes multi-agents [Tessier et al., 2002]. Les stratégies de résolution de conflits les plus populaires sont regroupés selon les catégories suivantes la négociation, l'arbitrage, le vote et l'auto-modification [Barber et al., 2000] que nous expliquons très brièvement comme suit :

— *La négociation* : est la stratégie la plus utilisée dans les systèmes multi-agents. Elle intervient lorsque des agents interagissent pour arriver à un accord et prendre des décisions communes, alors qu'ils présentent des réponses différentes.

— *L'arbitrage* : l'arbitrage est un processus dans lequel les conflits sont arbitrés ou négociés par un tiers. La décision de ce tiers (arbitre) doit être acceptée par les agents en conflit.

— *Le vote* : chaque agent exprime ses préférences et ses votes par rapport aux statuts des candidats générés en fonction du but de chaque agent pour maximiser la satisfaction totale du système.

— *L'auto-modification* : est la stratégie de résolution de conflits utilisée par un agent qui a détecté des conflits avec d'autres agents mais ne souhaite pas interagir avec ces derniers pour résoudre ces conflits. Cette stratégie est également appelée "Indépendance".

Dans cette thèse, nous nous sommes intéressés à l'auto-modification pour résoudre les conflits entre agents. La raison d'utiliser ce type de stratégie vient de sa simplicité et son efficacité, aucun effort/coût supplémentaire n'est requis [Barber et al., 2000].

Dans notre cas, le conflit peut se manifester lorsque l'agent initiateur S reçoit des réponses (des activités) reçues différentes de la réponse (l'activité prédite) locale à partir d'autres agents sélectionnés. En effet, l'agent initiateur S reçoit toutes les activités prédites lors de sa demande "check-FV" auprès des agents récepteurs sélectionnés dans sa liste $ACQ_{(S)}$. Ces activités reçues sont accompagnées par leurs degrés de confiance et se retrouvent dans la liste $pAct\_list_{(S)}$ y compris l'activité locale prédite $pa_{(S)}$.

$$pAct\_list_{(S)} = \{(pa_{(S)}, S, d_{pa_{(S)}}), (pa_{(R_1)}, R_1, d_{pa_{(R_1)}}), (pa_{(R_2)}, R_2, d_{pa_{(R_2)}}), ..., (pa_{(R_\omega)}, R_\omega, d_{pa_{(R_\omega)}})\}$$

Ainsi, nous entamons la présentation des trois stratégies de résolution de conflits utilisées dans l'approche *DCR* à savoir la stratégie *max-trust*, la stratégie *max-freq.* et la stratégie *stacking*. Ces trois stratégies font partie de la catégorie de l'auto-modification.

4.7.1 La stratégie de résolution max-trust

Cette stratégie favorise l'activité qui a le degré de confiance maximum dans la liste $pAct\_list_{(S)}$ de l'agent initiateur S. Elle procède comme suit :

— Faire la moyenne des degrés de confiance pour les activités récurrentes dans la liste $pAct\_list_{(S)}$

— Choisir l'activité qui a le degré de confiance maximal et l'assigner à l'activité prédite finale Fpa

— Générer en sortie l'activité finale Fpa

4.7.2 La stratégie de résolution max-freq.

Cette stratégie favorise l'activité qui a le maximum d'occurrence dans la liste $pAct\_list_{(S)}$ de l'agent initiateur S. Il s'agit de la méthode *vote majoritaire* [Kuncheva, 2004, Oh, 2003] qui est souvent utilisée comme technique d'agrégation dans les méthodes d'ensemble de classification en *apprentissage automatique* ("machine learning"). Elle consiste à compter le nombre de votes pour chaque classe et sélectionne la classe majoritaire. Dans notre cas, elle procède comme suit :

— Compter le nombre d'occurrence de chaque activité dans la liste $pAct\_list_{(S)}$

— Choisir l'activité qui a le nombre maximal d'occurrences et l'assigner à Fpa
— Générer en sortie l'activité finale Fpa
— Si plus d'une activité possède le nombre maximal d'occurrences, alors l'activité la plus confiante (celle qui a la moyenne de degré de confiance maximale) est choisie.

4.7.3 La stratégie de résolution "stacking"

En apprentissage automatique, le méta-apprentissage est une technique générale permettant d'améliorer les performances de plusieurs classifieurs en utilisant les méta-informations qu'ils fournissent [Schaul and Schmidhuber, 2010]. Une approche commune du méta-apprentissage est la généralisation empilée "stacked generalization (stacking)" [Wolpert, 1992] qui construit un classifieur de niveau supérieur (niveau 1) à partir des sorties générées des classifieurs de base (niveau 0). En effet, ce méta-classifieur est construit à partir d'une méta base de test où les méta-attributs ("meta-features") sont les prédictions générées des classifieurs bas niveau et la classe est la classe correcte. Le mécanisme de la méthode "stacking" est décrit dans la Figure 4.5.

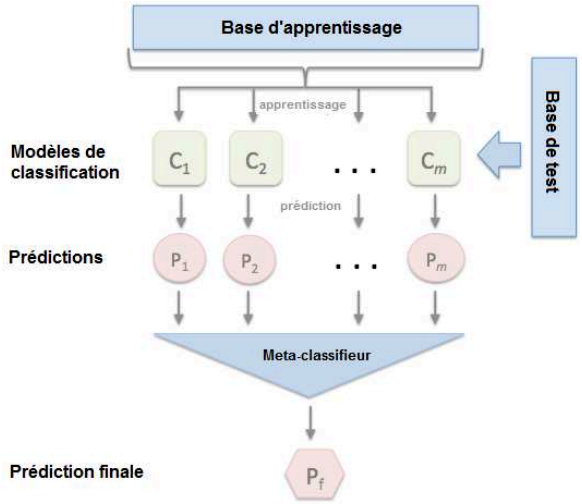


FIGURE 4.5. Mécanisme du méta-apprentissage (source [1])

Dans notre cas, la métabase de test contient les méta-attributs qui sont les activités prédites des agents récepteurs sélectionnés y compris aussi l'activité locale prédite de l'agent initiateur S. Ainsi, chaque agent initiateur comporte un méta-classifieur qu'il va utiliser pour aboutir à l'activité finale. Ce méta-classifieur est créé en utilisant la régression logistique [Ting and Witten, 1999].

1. https ://rasbt.github.io/mlxtend/user_guide/classifier/StacingClassifier/

4.8 Vers des agents apprenants

Dans cette section, nous discutons des limites de l'approche *DCR* et nos motivations par rapport à ces limites. Le but est de proposer une nouvelle approche *DCR* en appliquant un apprentissage en ligne en utilisant des agents apprenants. Cette nouvelle approche est nommée *DCR-OL* (*DCR with Online Learning*).

4.8.1 Limites du système DCR et motivations

Le système *DCR* consiste en un ensemble d'agents ayant des fonctionnalités similaires qui communiquent et interagissent entre eux pour identifier l'activité courante de l'habitant d'une maison intelligente. Ces agents comprennent des classifieurs hétérogènes, des listes d'activités prédéfinies à l'avance et des listes d'accointances. Les composants des agents de *DCR* sont initialisés une seule fois et ne seront pas mis à jour au fil du temps. Toutefois, le système *DCR* doit faire face aux changements suivants qui peuvent avoir lieu au cours du temps :

1. Les changements de l'environnement intelligent (e.g, un nouveau capteur installé, une nouvelle activité à reconnaître, etc.)
2. Le changement de comportement de l'habitant. Pour *DCR*, nous avons supposé que la fréquence de changement de comportement de l'habitant est très faible.

Dans cette thèse, nous nous sommes intéressés au deuxième point décrit ci-dessus. En effet, *DCR* doit prendre en considération les variations du comportement de l'habitant en appliquant un apprentissage en ligne. En effet, un agent doit apprendre en ligne et s'adapter suite aux retours des autres agents. En d'autres termes, il doit évaluer sa prédiction locale et les prédictions reçues des autres agents par rapport à l'activité réelle.

4.8.2 Travaux connexes dans l'apprentissage en ligne

Les approches de reconnaissance d'activités orientées donnée exploitent la grande quantité de données collectées à partir des capteurs distribués et hétérogènes. L'utilisation efficace de ce flux de données élevé implique également un défis important : une approche de reconnaissance orientée donnée doit prendre en compte le fait que la nature des données change dans le temps et que la relation entre les données et les activités peut changer. L'évolution des données, sous forme de séquences spatio-temporelles, fait apparaître la notion de *dérive*

de concept [Zliobaite, 2010]. La *dérive de concept* caractérise l'évolution dans le temps des données observées, de leurs étiquettes, et de leurs relations.

Plusieurs travaux ont été proposés pour répondre à ce défi. Nous étudions deux travaux existants qui sont [van Rijn et al., 2018] et [Canzian et al., 2015]. Les auteurs de [van Rijn et al., 2018] se sont intéressés aux méthodes d'ensemble d'apprentissage d'une manière centralisée. Les méthodes d'ensemble d'apprentissage [Dietterich, 2000] consistent en un ensemble de classifieurs dont les décisions individuelles sont combinées d'une manière ou d'une autre, généralement par vote pondéré ou non pondéré, pour classifier de nouveaux exemples. En revanche, les auteurs de [Canzian et al., 2015] ont proposé une approche distribuée. Nous présentons ces deux travaux comme suit :

1. Ensemble d'apprenants distribués pour la classification en ligne de flux de données dynamiques [Canzian et al., 2015] :

   Ils proposent la majorité pondérée de Perceptron (*PWM Perceptron Weighted Majority*), une approche d'apprentissage d'ensemble en ligne distribuée pour classer les données capturées à partir des sources de données distribuées. Leur système consiste en un ensemble d'apprenants distribués, qui analysent différents flux de données afin de les classer. Chaque apprenant utilise un classifieur local pour générer une prédiction locale. Les prédictions locales sont ensuite collectées par chaque apprenant et combinées à l'aide d'une règle d'agrégation qui est la règle de majorité pondérée pour déterminer la prédiction finale. Ils présentent un nouvel algorithme d'apprentissage d'ensemble en ligne pour mettre à jour la règle d'agrégation afin de l'adapter aux données dynamiques. En effet, après avoir effectué la prédiction finale, l'apprenant consulte la prédiction réelle, c'est-à-dire l'étiquette, associée à l'évènement à classer. En comparant la prédiction finale et la prédiction réelle, l'apprenant met à jour les poids d'agrégation de la règle d'apprentissage du perceptron comme suit :
   - Si la prédiction finale correspond à la prédiction réelle, alors les poids associés aux apprenents ne sont pas modifiés.
   - Si la prédiction finale est différente de la prédiction réelle, alors les poids des apprenants ayant signalé une prédiction erronée sont diminués d'une unité. Les poids des apprenants ayant signalé une prédiction correcte sont augmentés d'une unité.

   [Canzian et al., 2015] apprennent seulement une règle d'agrégation pour gérer le flux de données dynamique. Les classifieurs maintiennent une règle d'agrégation à jour et sont en mesure de suivre la dérive de concept. Ils traitent chaque observation à

l'arrivée une seule fois, sans qu'il soit nécessaire de stocker et de retraiter des blocs de données. De plus, ils n'exigent pas que les classifieurs locaux soient regénérés de manière centralisée (par exemple, dans le cas distribué, il est peut être coûteux de regénérer les classifieurs locaux ou irréalisable si les classifieurs sont gérés par des entités différentes). Cependant, leur approche se focalise seulement sur les problèmes de classification binaire.

2. Une estimation de performance en ligne : apprentissage d'ensemble hétérogène pour les flux de données [van Rijn et al., 2018]

   Les auteurs étudient l'utilisation des ensembles de classifieurs hétérogènes pour les flux de données d'une manière centralisée et proposent une plateforme en ligne d'évaluation de la performance, qui pondère de manière dynamique les votes des classifieurs individuels dans l'ensemble. Cette approche a pour appellation *BLAST (short for **best last**)*. En effet, chaque classifieur maintient une valeur de performance qui est mise à jour par une fonction d'estimation de performance. Pendant la phase d'apprentissage, la fonction d'estimation de performance estime la valeur de performance de chaque classifieur en s'appuyant sur la comparaison de la prédiction locale avec la prédiction réelle. Ensuite, l'algorithme *BLAST* met à jour le classifieur avec l'instance en cours. Pour chaque exemple de test, l'ensemble des classifieurs désigne un classifieur qui a la valeur de performance la plus élevée pour effectuer sa prédiction. Ce classifieur est considéré comme le classifieur actif. Lorsque plusieurs classifieurs obtiennent la même valeur de performance estimée, un classifieur arbitraire peut être sélectionné comme le classifieur actif.

Le tableau 4.5 résume les points de différence des deux approches décrites ci-dessus. A cet égard, nous présentons notre approche *DCR-OL* qui combine les apports de [Canzian et al., 2015] et de [van Rijn et al., 2018] et qui aborde les propositions suivantes :

— S'intéresser à la reconnaissance d'activités (identification multi-classe)
— Ensemble d'agents apprenants ayant chacun une valeur de performance
— L'activité finale est l'agrégation de plusieurs prédictions (locale et reçues) selon une nouvelle stratégie d'agrégation nommée *stratégie de performance pondérée maximale*
— Les valeurs de performance des agents sont mises à jour si la prédiction finale est différente de la prédiction réelle.
— Les modèles d'activité ne seront pas modifiés. Seules les valeurs de performance se modifient lors de l'arrivée de nouveaux vecteurs d'attributs (la régénération des

classifieurs locaux est coûteuse).

TABLE 4.5. Étude comparative entre les deux approches [Canzian et al., 2015] et [van Rijn et al., 2018]

Critères	L'approche [Canzian et al., 2015]	L'approche [van Rijn et al., 2018]
Axe	Problème de classification binaire	Problème de classification multi-classe
Identification de la prédiction finale	Combinaison des prédictions locales des apprenants et agrégation d'une prédiction finale	Sélectionner un classifieur actif ayant une performance maximale pour générer la prédiction finale
Apprentissage en ligne	Si la prédiction finale est différente de la prédiction réelle, les poids des apprenants sont modifiés	Les valeurs de performance des classifieurs sont instantanément à jour pendant la phase d'apprentissage
Re-génération des classifieurs	Pas de mise à jour des classifieurs	Mise à jour instantanée des classifieurs avec l'arrivée des observations pendant la phase d'apprentissage

4.8.3 DCR-OL : l'approche DCR avec l'apprentissage en ligne

Dans cette sections, nous présentons notre nouvelle approche DCR en appliquant un apprentissage en ligne nommée *DCR-OL (DCR with Online Learning)*. Tout d'abord, nous commençons par introduire la notion d'agents apprenants.

4.8.3.1 Définition des agents apprenants

Un agent apprenant est capable d'apprendre de ses expériences. Il commence par quelques connaissances de base et est ensuite capable d'agir et de s'adapter de manière autonome, par le biais de l'apprentissage, pour améliorer ses propres performances. Contrairement aux agents intelligents qui agissent sur les informations fournies par un expert, les agents apprenants sont en mesure d'effectuer des tâches, d'analyser les performances et de rechercher de nouveaux moyens d'améliorer ces tâches [Russell, 1996].

Un agent apprenant [Russell, 1996] est composé de quatre modules de base (voir Figure 4.6) :

1. Élément de performance : l'élément de performance choisit l'action à prendre. Il passe ensuite à une nouvelle action basée sur les retours et les suggestions d'amélioration.
2. Élément critique : l'élément critique détermine le résultat de l'action et donne un retour

3. Élément d'apprentissage : il prend en compte les retours de l'élément critique et explique comment améliorer l'action la prochaine fois.

4. Générateur de problèmes : il s'agit du composant chargé de développer de nouvelles expériences que l'agent apprenant doit essayer. C'est le module qui aide l'agent à continuer à apprendre.

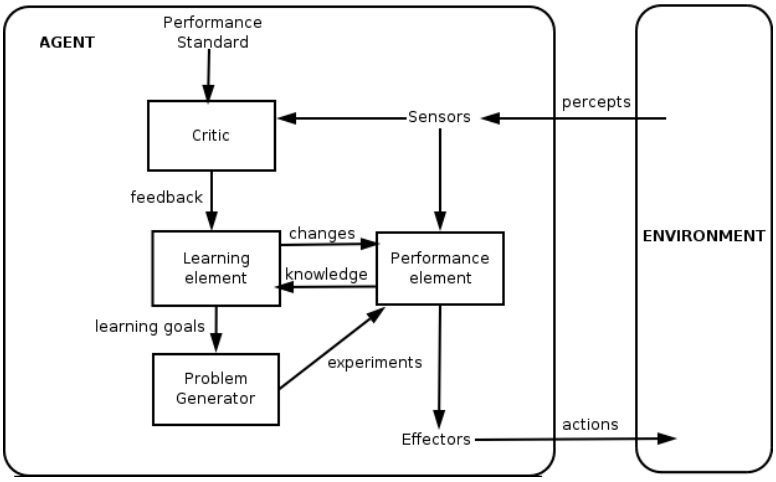


FIGURE 4.6. Composition d'un agent apprenant [Russell, 1996]

4.8.3.2 Le modèle DCR-OL

Nous rappelons que le système *DCR* est un système multi-agent $MAS = \{A_{l_1}, A_{l_2}, ..., A_{l_n}\}$ où chaque agent A_{l_i} est défini par un tuple $A_{l_i} = (id_{A_{l_i}}, clf_{A_{l_i}}, ACT_{A_{l_i}}, ACQ_{A_{l_i}})$. Le système *DCR-OL* adopte la même représentation que le système *DCR*, Sauf, qu'il remplace les agents intelligents de *DCR* par des agents apprenants et modifie la définition d'un agent A_{l_i} en rajoutant une cinquième composante qui est une liste des valeurs de performance de tous les agents dans *MAS*. Le but de rajouter cette liste est d'évaluer la performance des agents en termes de prédictions par rapport à la prédiction réelle.

Ainsi, *DCR-OL* est un ensemble d'agents apprenants LA_i ($i \in n; n$ est le nombre d'agents apprenants dans *MAS*) répartis dans la maison intelligente. Chaque agent apprenant LA_i est défini par un tuple $LA_i = (id_{LA_i}, clf_{LA_i}, ACT_{LA_i}, ACQ_{LA_i}, PERF_{LA_i})$ où $PERF_{LA_i} = \{p_{LA_1}, p_{LA_2}, ..., p_{LA_n}\}$ est une liste des valeurs de performance de chaque agent LA_i dans le système *MAS*.

$p_{LA_i} \in [0,1]$ signifie que lorsque p_{LA_i} vaut 0 alors l'agent apprenant en question n'est pas performant. Lorsque p_{LA_i} vaut 1 alors l'agent apprenant en question est très performant.

La liste $PERF_{LA_i}$ est propre à chaque agent LA_i et dépend seulement de son contexte. Elle est différente d'un agent à un autre. Lors de l'initialisation du *DCR-OL*, cette liste est initialisée avec la valeur 1 c'est à dire toutes les valeurs de performance des agents dans *MAS* sont égales à 1. Tous les agents apprenants dans *MAS* sont supposés être performants à l'initialisation.

L'agent apprenant LA_i raisonne de la même manière que l'agent A_{l_i} dans *DCR*. Lors de l'arrivée d'un vecteur d'attributs FV, il est nommé agent initiateur S interroge son classifieur pour avoir l'activité locale prédite avec son degré de confiance. Si ce dernier est inférieur au seuil δ, l'agent S collabore avec certains agents dans sa liste d'accointances pour avoir leurs prédictions. Si l'activité locale prédite est différente des prédictions reçues, l'agent S appliquera une stratégie de résolution de conflits pour décider une seule prédiction finale.

Après la génération de la prédiction finale, l'agent S doit apprendre suite aux retours des autres agents et adapter sa liste de valeurs de performance $PERF_S$. Ainsi, il compare sa prédiction finale Fpa_S avec la prédiction réelle RP. Nous supposons que cette dernière est récupérée lors de l'interaction avec le participant qui confirme ou contredit l'activité identifiée finale. A cet effet, nous distinguons deux cas :

1. Si la prédiction finale Fpa_S est différente de la prédiction réelle RP, la liste $PERF_S$ se met à jour.
2. Si la prédiction finale Fpa_S est identique à la prédiction réelle RP, la liste $PERF_S$ n'est pas modifiée.

Mise à jour de la liste $PERF_S$:

Les valeurs de performance de $PERF_S = \{p_{LA_1}, p_{LA_2}, ..., p_{LA_n}\}$ vont être mises à jour suivant la mesure de performance $Perf-measure$ utilisée par l'approche *BLAST* [van Rijn et al., 2018] comme suit :

$$p_{LA_i} = (p_{LA_i} \times \alpha) + \{(1 - Loss(pa_{LA_i}(FV_{(S)}), RP(FV_{(S)}))) \times (1 - \alpha)\}$$

Avec :

— α : est le facteur d'atténuation. Ce paramètre est choisi par l'expert et est initialisé avec la valeur 0.999. Cette valeur a été choisie selon une étude effectuée par [van Rijn et al., 2018].

— $Loss()$: est une fonction de perte 0/1, donnant une pénalité de 1 à tous les vecteurs d'attributs FV mal classés.

— $pa_{LA_i}(FV_{(S)})$: la prédiction de l'agent LA_i pour le vecteur d'attributs FV.

— $RP(FV_{(S)})$: la prédiction réelle pour le vecteur d'attributs FV.

Ainsi, l'agent apprenant S prendra en compte les retours des autres agents apprenants en les évaluant avec la mesure de performance $Perf - measure$.

La stratégie de performance pondérée maximale

Les valeurs de performance des agents apprenants existantes dans la liste $PERF_{LA_i}$ seront utilisées par une nouvelle stratégie de résolution de conflits nommée *stratégie de performance pondérée maximale*. Cette dernière génère la prédiction finale en calculant la performance pondérée maximale de chaque activité prédite dans la liste $pAct\_list_{(S)}$.

En effet, la performance pondérée $PP(pa_j)$ de chaque activité pa_j dans la liste $pAct\_list_{(S)}$ est déterminée comme suit :

$$PP(pa_j) = \frac{\sum_{k=1}^{K} p_{LA_k}(pa_j)}{\frac{|pAct\_list_{(S)}|}{\sum_{f=1} p_{LA_f}}}$$

Avec :

— j : indice de l'activité prédite dans la liste $pAct\_list_{(S)}$.

— K : nombre d'agents apprenants qui ont prédit l'activité pa_j.

L'activité prédite finale Fpa_S est l'activité prédite pa_j qui a la valeur maximale de $PP(pa_j)$ dans la liste $pAct\_list_{(S)}$.

La figure 4.7 décrit deux phases : *la phase d'apprentissage* dans laquelle l'agent apprenant compare sa prédiction finale par rapport à la prédiction réelle. Si ces deux prédictions sont différentes, l'agent S met à jour sa liste $PERF_S$ en calculant la valeur de performance p_{LA_i} de chaque agent apprenant LA_i existant dans la liste $pAct\_list_{(S)}$. *La phase de test* consiste à appliquer la méthode de résolution de conflits *la performance pondérée maximale* en prenant en compte la nouvelle liste $PERF_S$ mise à jour.

La figure 4.8 décrit le comportement d'un agent apprenant initiateur S dans l'approche *DCR-OL* et montre son apport par rapport à l'approche *DCR* en rajoutant la boucle *feedback* en vert.

Concernant la définition d'un agent apprenant (énoncée précédemment), le modèle *DCR-OL* caractérise les différents éléments d'un agent apprenant LA_i comme suit (voir Figure 4.9) :

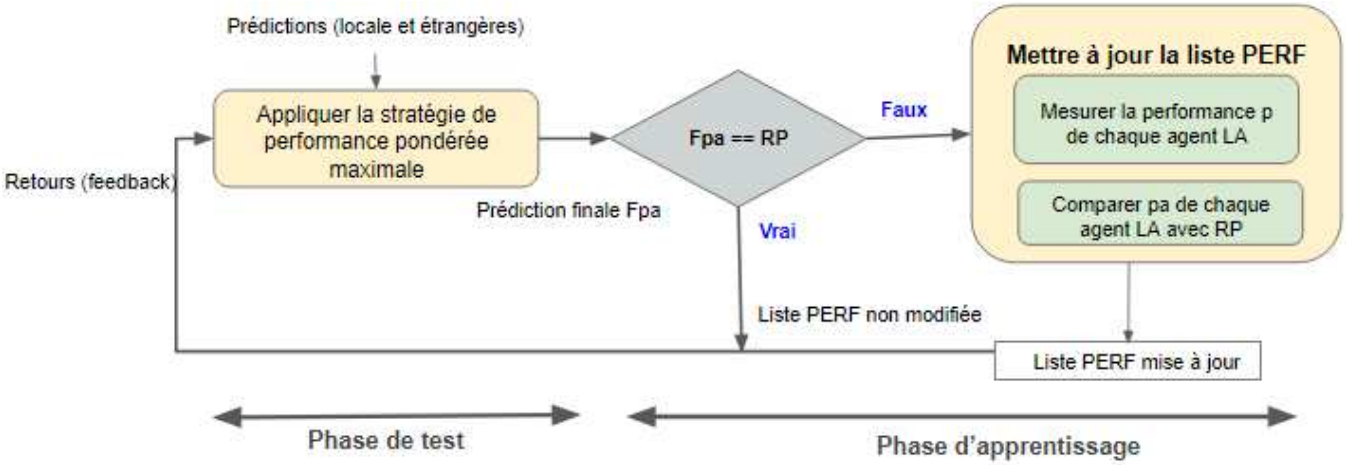


FIGURE 4.7. L'approche DCR avec l'apprentissage en ligne

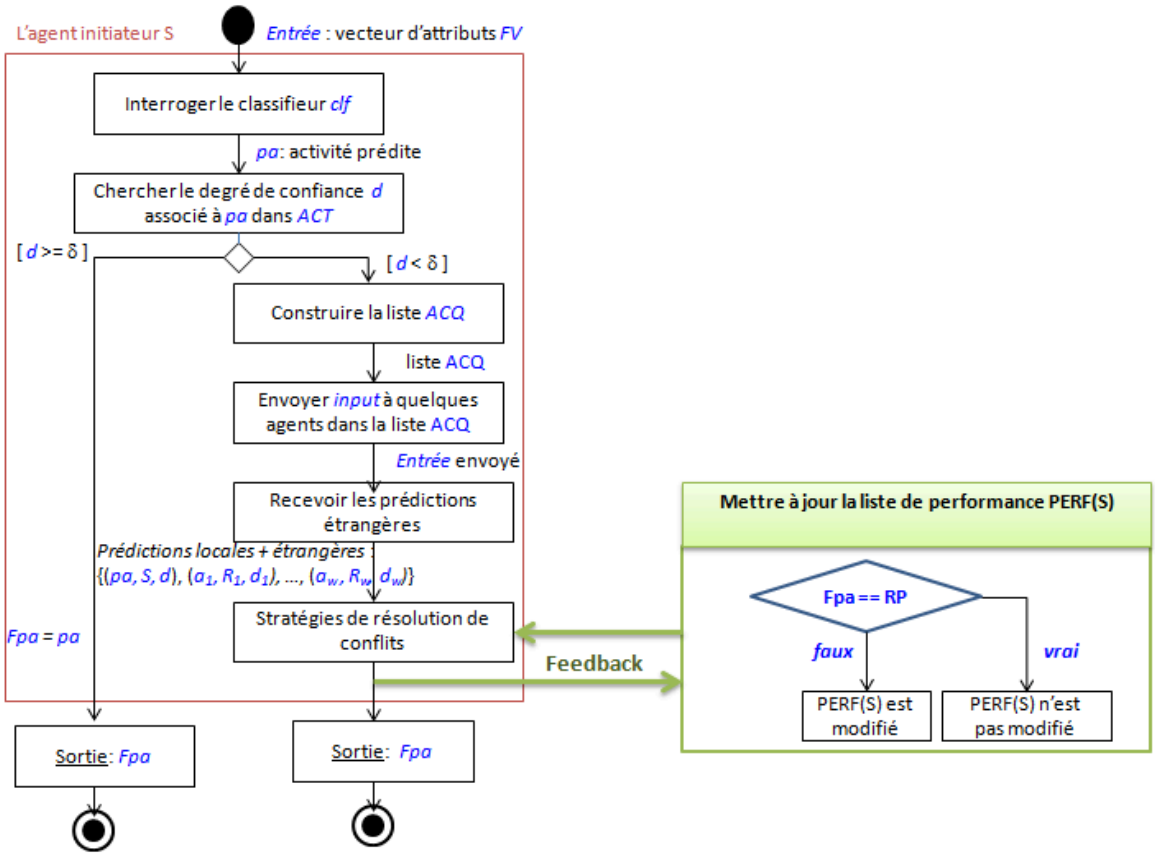


FIGURE 4.8. Diagramme d'état d'un agent apprenant initiateur

— Élément critique : retourne le résultat de comparaison entre la prédiction finale et la prédiction réelle.

— Élément d'apprentissage : reçoit les retours de l'élément critique et détermine l'action à appliquer. Dans notre cas, le vecteur $PERF_{LA_i}$ sera modifié si l'élément critique retourne faux.
— Élément de performance : exerce la décision prise par l'élément d'apprentissage. Dans notre cas, il met à jour le vecteur $PERF_{LA_i}$ si l'élément d'apprentissage décide de le modifier.

Pour l'élément *générateur de problèmes*, le système *DCR-OL* ne l'utilise pas puisque les vecteurs d'attributs arrivent au cours du temps depuis le niveau *Observation*.

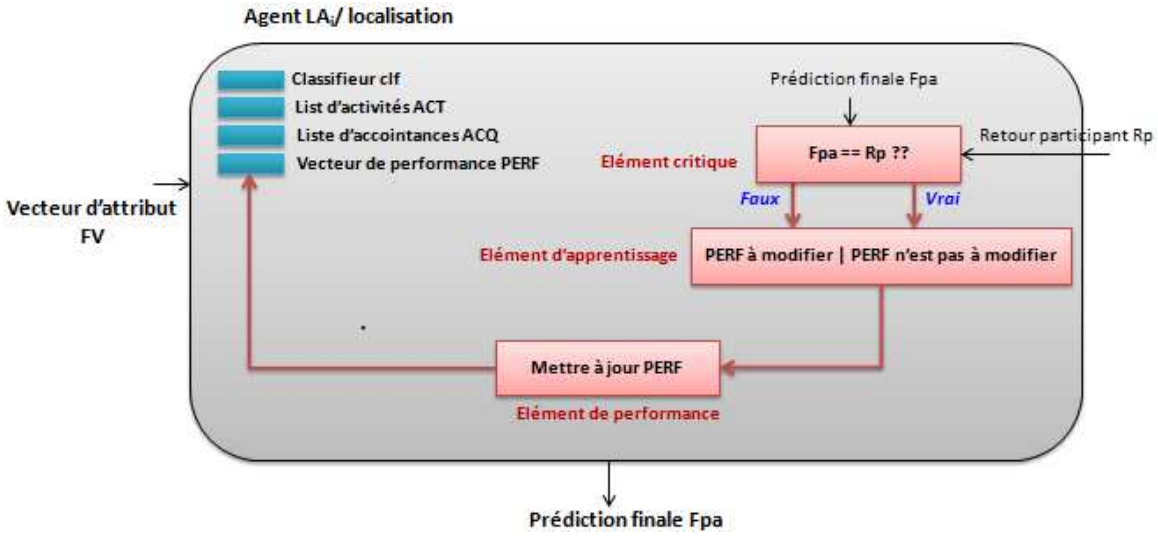


FIGURE 4.9. Composition d'un agent apprenant initiateur

4.8.3.3 L'algorithme DCR-OL

L'algorithme *DCR-OL* (Algorithme 8) décrit la nouvelle méthode de résolution de conflits en ligne qui est *la stratégie de performance pondérée maximale*. L'agent apprenant initiateur S applique cet algorithme en ligne pour générer la prédiction finale depuis la liste $pAct\_list_{(S)}$ qui contient la prédiction locale et toutes les prédictions reçues. Cet algorithme consiste en deux grandes étapes après l'initialisation du vecteur de performance $PERF_S$ à la valeur 1 de l'agent S (ligne 2) :

1. Générer la prédiction finale Fpa_S en appliquant *la stratégie de performance pondérée maximale* (ligne 4). Cette étape est réalisée via l'appel de la méthode *maxWeighted-Performance* (algorithme 9).

2. Mettre à jour le vecteur de performance $PERF_S$ en évaluant les agents avec lesquels l'agent S a collaboré (ligne 5). Cette étape est réalisée via l'appel de la méthode *updatePerformanceValues* (algorithme 10).

ALGORITHME 8 . DCR-OL : méthode de résolution de conflit en ligne

Input :
S : un agent initiateur ; FV_S : un vecteur d'attributs ; n : nombre d'agents dans MAS ; $pAct\_list_{(S)}$: liste de triplets (activité prédite, agent source, degré de confiance) ; $RP_{(FV_S)}$: l'activité réelle ; α : facteur d'atténuation ;

```
1  begin
       // Initialisation du vecteur PERF_S à 1
2      set PERF_S = {p_S, p_LA1, ..., p_LAw} ← 1
       // Parcourir tous les vecteurs d'attributs
3      for all observations o = (FV_S, RP(FV_S)) do
4          Fpa_S = maxWeightedPerformance(pAct_list_(S), PERF_S)
5          if Fpa_S <> RP_(FV_S) then
6              PERF_S = updatePerformanceValues(pAct_list_(S), PERF_S, RP(FV_S), α)
```

Nous décrivons ces deux étapes via les deux algorithmes comme suit :

— La méthode *maxWeightedPerformance* : permet de calculer la performance pondérée de chaque activité dans la liste $pAct\_list_{(S)}$ (ligne 4) et de générer l'activité finale ayant une performance pondérée maximale (ligne 6).

— La méthode *updatePerformanceValues* : permet de mettre à jour le vecteur de performance $PERF_S$ de l'agent initiateur. En effet, elle commence par parcourir tous les agents concernés dans la liste $pAct\_list_{(S)}$ (ligne 2), puis récupére leurs prédictions correspondantes (ligne 3), détermine leurs valeurs de performance selon la mesure *P-measure* (ligne 4) et enfin met à jour le vecteur $PERF_S$ pour les agents concernés (ligne 5).

Ces algorithmes sont implémentés et testés dans le chapitre suivant.

4.9 Extension de DCR à des données d'entrée floues

Nous rappelons que l'approche *DCR* est une approche de raisonnement distribuée pour la reconnaissance d'activités humaines avec en entrée un ensemble de vecteurs d'attributs réduits non flous (le résultat de l'approche *FSCI*) (voir Figure 4.10).

L'approche de sélection des attributs *Fuzzy-FSCI* génère quant à elle, en sortie, une liste

ALGORITHME 9 . La méthode *maxWeightedPerformance*

Input :
$pAct\_list_{(S)}$: liste de triplets (activité prédite, agent source, degré de confiance) ;
$PERF_S = \{p_S, p_{LA_1}, ..., p_{LA_w}\}$: vecteur des valeurs de performance des agents dans *MAS*.
Output :
Fpa_S : l'activité prédite finale

```
1  begin
       // Liste des activités prédites PaList_S
2      PaList_S = getDistinctPredictedActivities(pAct_list_(S))
3      for each a_k in PaList_S do
           // Déterminer la performance pondérée pour chaque activité
4          PP_{a_k} = computePP(pAct_list_(S), PERF_S)
5          PPlist.add(< a_k, PP_{a_k} >)
       // Fpa_S est l'activité ayant une performance pondérée maximale
6      Fpa_S = maxPP(PPlist)
7      return(Fpa_S)
```

de vecteurs d'attributs flous réduits, une approche d'identification floue doit alors être proposée pour étendre *DCR* au cas flou.

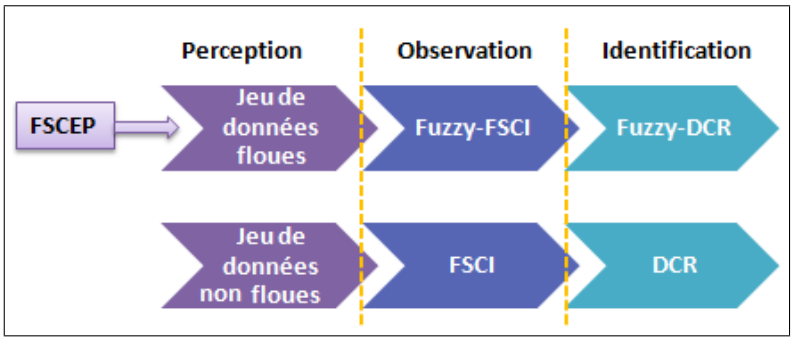


FIGURE 4.10. Enchaînement des approches proposées pour tous les niveaux de l'architecture

L'objectif de cette section est de présenter une étude théorique sur la nouvelle approche **Fuzzy-DCR**. Cette dernière généralise l'approche *DCR* à des entrées des vecteurs d'attributs flous. Avec cette nouvelle approche, la liaison en terme de nature de données entre l'approche de *sélection des attributs Fuzzy-FSCI* et l'approche d'*identification Fuzzy-DCR* devient possible (voir Figure 4.10).

ALGORITHME 10 . La méthode $updatePerformanceValues$

Input :
$pAct\_list_{(S)}$: liste de triplets (activité prédite, agent source, degré de confiance) ;
$PERF_S = \{p_S, p_{LA_1}, ..., p_{LA_w}\}$: vecteur des valeurs de performance des agents dans MAS.
$RP_{(FV_S)}$: l'activité réelle correspondante à FV_S ; α : facteur d'atténuation ;
Output :
$PERF_S$: vecteur de performance mis à jour pour l'agent S

```
1 begin
2     for all agents LA_i in pAct_list_(S) do
          // Récupérer la prédiction locale de l'agent LA_i depuis pAct_list_(S)
3         LP_LA_i(FV_S) = getLocalPrediction(pAct_list_(S), LA_i)
          // Calculer la nouvelle valeur de performance de l'agent LA_i
4         p_LA_i = (p_LA_i × α) + {(1 − Loss(LP_LA_i(FV_S), RP(FV_S))) × (1 − α)}
          // Mettre à jour le vecteur PERF_S de l'agent initiateur S
5         PERF_S = updatePerformanceVector(PERF_S, p_LA_i)
```

4.9.1 Liaison avec l'approche Fuzzy-FSCI

L'approche *Fuzzy-FSCI* est une méthode de sélection d'attributs flous les plus pertinents et non redondants. En sortie de cette étape, un ensemble de vecteurs d'attributs flous réduits est généré (voir Figure 4.11). Ces vecteurs représentent les entrées de l'étape *Identification*.

	Horodatage	Localisation	Température	Pression	....	Attribut (n)	Activité
Observation 1	19-07-2018 22:13:56	Livingroom 0.4 Bedroom 0.6	High 0.3 Medium 0.5 Low 0.2	High 0.7 Medium 0.2 Low 0.1	....	.....	sleeping
Observation 2	20-07-2018 08:26:40	Kitchen 0.9 Corridor 0.1	High 0.1 Medium 0.7 Low 0.2	High 0.1 Medium 0.1 Low 0.8	....	.....	preparing_breakfast
.....	.....	.....	.....	.....	....	.....	.....

FIGURE 4.11. Les vecteurs d'attributs floues réduits générés depuis l'approche *Fuzzy-FSCI*

Pour la phase d'identification, l'approche *DCR* prend en entrée un ensemble de vecteurs d'attributs continus et/ou discrets et non flous. Elle ne peut pas donc s'appliquer avec des vecteurs d'attributs flous. La sous section suivante présente la version floue de l'approche *DCR* qui pallie cette limite.

4.9.2 L'approche Fuzzy-DCR

Nous rappelons que le système *DCR* est un système multi-agent $MAS = \{A_{l_1}, A_{l_2}, ..., A_{l_n}\}$ où chaque agent A_{l_i} est défini par un tuple $A_{l_i} = (id_{A_{l_i}}, clf_{A_{l_i}}, ACT_{A_{l_i}}, ACQ_{A_{l_i}})$. Le modèle *Fuzzy-DCR* adopte la même représentation que le système *DCR*. Sauf que les agents A_{l_i} prennent en entrée les vecteurs d'attributs flous et donc ils incorporent un classifieur flou $clf_{A_{l_i}}$. L'agent initiateur *S* se déclenche lorsqu'il reçoit un vecteur d'attribut flou et génère en sortie selon la nature de son classifieur utilisé :

1. Si l'agent *S* utilise un classifieur flou qui génère une sortie non floue c'est à dire une activité locale prédite, alors la démarche pour déterminer l'activité finale est la même que l'approche *DCR*, il est possible de reprendre les étapes depuis l'étape 2 dans la section 4.5.
2. Si l'agent *S* utilise un classifieur flou qui génère une sortie floue c'est à dire l'ensemble des activités pondérées de degrés d'appartenance. Dans ce cas, il existe deux modalités de traitement de cette sortie floue comme suit :

   (a) L'étape de defuzzification est possible pour obtenir une seule activité locale prédite. La démarche pour déterminer l'activité finale est la même que l'approche *DCR*, il est possible de reprendre les étapes depuis l'étape 2 dans la section 4.5.

   (b) Nous gardons la sortie floue c'est à dire l'ensemble des activités pondérées de degrés d'appartenance. Le traitement de *DCR* est alors différent. Nous détaillons la démarche dans ce qui suit.

Si la sortie du classifieur est floue :

Les agents dans *MAS* suivent un enchaînement suite à l'arrivée d'un vecteur d'attributs flous *FV* à partir du module observation. Le cycle de vie de ces agents décrit les étapes pour aboutir à la reconnaissance floue c'est à dire un ensemble d'activités pondérées de degrés de confiance. Les étapes sont enchaînées comme suit :

1. Ayant comme entrée un vecteur d'attributs flous, l'agent A_{l_i} interroge son classifieur flou local $clf_{A_{l_i}}$ pour obtenir l'ensemble des activités prédites pondérées de degrés d'appartenance. Une sélection d'un sous ensemble d'activités est effectuée selon le degré d'appartenance par rapport à l'α. Seules les activités ayant un degré d'appartenance supérieur à l'α seront considérées. Ainsi, une liste d'activités réduite pondérées

de degrés d'appartenance est générée : $\{(pa_1, md_1), (pa_2, md_2), ..., (pa_k, md_k)\}$ (avec $k < m$).

2. L'agent A_{l_i} détermine le degré de confiance d_{pa_j} relatif à chaque activité locale prédite $pa_j \in \{(pa_1, md_1), (pa_2, md_2), ..., (pa_k, md_k)\}$ (avec $j : 1..k$) depuis la liste $ACT_{A_{l_i}}$.
3. Lorsque les activités ont un degré d_{pa_j} inférieur au seuil δ dans la liste $ACT_{A_{l_i}}$, l'agent A_{l_i} a du mal à reconnaître ces activités et il a besoin de communiquer avec les autres agents pour confirmer ou infirmer ses prédictions locales. Une stratégie de collaboration et d'agrégation doivent donc être définies et mises en place.

   Nous notons que la complexité de ces stratégies à définir doivent être étudiées en termes de taille et de nombre de messages échangés entre les agents.

La mise en place de l'approche *Fuzzy-DCR* nécessite tout d'abord la mise en place d'un classifieur flou. Dans la littérature, plusieurs approches ont été proposées pour construire des classifieurs flous tels que l'arbre de décision flou [Jin et al., 2014], la forêt aléatoire floue [Bonissone et al., 2010]. En outre, il n'existe pas de jeu de données comportant des observations floues. Les jeux de données existants sont non flous. La fuzzification [Zadeh, 1988] est alors nécessaire pour transformer un jeu de données non flou en un jeu de données flou. La mise en place de l'approche *Fuzzy-DCR* reste un travail futur pour cette thèse.

4.10 Discussion et synthèse

Dans cette section, nous discutons l'approche *DCR* et la version améliorée de l'approche *DCR* en rajoutant l'apprentissage en ligne.

Discussion de l'approche DCR

Nous analysons nos propositions P_i par rapport aux défis fixés D_i au début de ce chapitre comme suit :

— **D1 (la gestion des données de capteurs) -> P1 (une approche multi-agent totalement distribuée)** : l'architecture du système *DCR* peut traiter un énorme flux de données grâce à son architecture multi-agent totalement distribuée où les agents sont autonomes et possèdent les mêmes fonctionnalités. En outre, la technique de segmentation choisie permet de limiter le nombre d'évènements existant dans la fenêtre.

— **D2 (la fraîcheur des données) -> P2 (une approche ascendante "bottom-up")** : L'architecture globale du système (voir Figure 4.1) assure une stratégie ascendante

pour la remontée, le traitement et la transformation des données brutes émises par les capteurs en des activités.

— **D3 (l'identification) -> P3 (des agents incorporant des classifieurs)** : nos agents incorporent des classifieurs construits sur un historique décrivant les expériences et les comportements de l'habitant. Ainsi, les activités courantes peuvent être identifiées grâce à ces classifieurs.

— **D4 (la précision) -> P4 (la collaboration)** : nous avons défini une stratégie de collaboration qui consiste à sélectionner des agents qui identifient l'activité en question avec plus de précision que l'agent initiateur. Cette stratégie de collaboration a pour but de renforcer le taux de reconnaissance d'une activité.

— **D5 (l'hétérogénéité) -> P5 (différents types de classifieurs)** : les agents détiennent différents types de classifieurs. Ces derniers dépendent principalement de la nature des données émises par les capteurs dans chaque localisation de la maison intelligente.

— **D6 (l'incertitude) -> P6 (un degré de confiance assigné)** : les agents attribuent un degré de confiance à chaque activité reconnue dans leurs localisations ce qui permet à l'agent initiateur de prendre une décision finale.

— **D7 (la gestion de conflits) -> P7 (trois stratégies de résolution de conflits proposées)** : la stratégie stacking est considérée comme la meilleure stratégie de résolution de conflits en termes de taux de reconnaissance. Cependant, elle est coûteuse par rapport aux deux autres max-freq. et max-trust en terme de temps de traitement car elle nécessite la construction des méta-classifieurs.

Discussion de l'approche DCR-OL

Nous positionnons notre approche *DCR-OL* par rapport aux deux travaux étudiés qui sont [Azkune et al., 2015] et [van Rijn et al., 2018]. Le tableau 4.6 compare l'approche *DCR-OL* par rapport à l'approche de [Azkune et al., 2015]. Le tableau 4.7 compare l'approche *DCR-OL* par rapport à l'approche de [van Rijn et al., 2018].

4.11 Conclusion

Dans ce chapitre, nous avons étudié les approches distribuées HAR existantes dans la littérature afin de proposer l'approche principale *DCR* comme une nouvelle approche de

TABLE 4.6. Étude comparative entre l'approche *DCR-OL* et l'approche [Canzian et al., 2015]

L'approche *DCR-OL*	L'approche [Canzian et al., 2015]
Une approche distribuée : un ensemble d'agents apprenants avec des classifieurs hétérogènes	Une approche distribuée : un ensemble d'apprenants avec des classifieurs homogènes
Identification multi-classe	Identification binaire
Un agent apprenant possède un vecteur de performance des autres agents	Un apprenant possède un vecteur de poids des autres agents

TABLE 4.7. Étude comparative entre l'approche *DCR-OL* et l'approche [van Rijn et al., 2018]

L'approche *DCR-OL*	L'approche [van Rijn et al., 2018]
Une approche distribuée : un ensemble d'agents apprenants avec des classifieurs hétérogènes	Une approche centralisée : un ensemble de classifieurs hétérogènes
Traitement de chaque vecteur d'attributs *à l'arrivée*, sans avoir recours à une phase d'apprentissage (le vecteur de performance est construit et mis à jour en ligne)	Une phase d'apprentissage est nécessaire pour déterminer les valeurs de performance des classifieurs ainsi que pour la re-génération des classifieurs
La prédiction finale est le résultat de la stratégie de performance pondérée maximale	La prédiction finale est la prédiction du classifieur actif (le classifieur ayant une mesure de performance maximale)
Pas de re-génération des classifieurs (seulement le vecteur de performance se met à jour)	Les classifieurs sont reconstruits avec les nouvelles intances pendant la phase d'apprentissage

raisonnement collaboratif distribuée pour reconnaître les activités humaines à partir d'une séquence de capteurs continue dans les maisons intelligentes.

L'approche *DCR* consiste en plusieurs agents, avec divers classifieurs basés sur un mécanisme de raisonnement, capables d'analyser le flux de données entrant (les vecteurs d'attributs non flous) provenant des deux modules perception et observation. Cet ensemble d'agents collaborent en échangeant une partie de leurs connaissances pour identifier correctement les activités. En effet, ils analysent les vecteurs d'attributs en fonction de leurs localisations dans la maison intelligente, prédisent l'activité locale en cours et collaborent entre eux pour aboutir à l'activité finale en appliquant trois stratégies de décision (max-trust, max-freq. et stacking).

En outre, nous avons présenté une version améliorée de l'approche *DRC* qui est l'approche *DCR-OL*. Cette dernière emploie un apprentissage en ligne des valeurs de performance des différents agents dans *MAS*. Cet apprentissage s'appuie sur une évaluation de la prédiction finale de l'agent initiateur par rapport à la prédiction réelle. Nous avons supposé que la prédiction réelle est récupérée depuis l'interaction avec le participant. Toutefois, il existe des méthodes pour estimer cette prédiction réelle telles que l'apprentissage par renfor-

cement ou l'apprentissage semi-supervisé.

D'autre part, nous avons introduit une étude théorique sur l'approche *Fuzzy-DCR* qui représente la version floue de l'approche *DCR*. Cette dernière utilise des agents intelligents incorporant des classifieurs flous qui prennent en entrée un vecteur d'attributs flous et génèrent une sortie floue c'est à dire un ensemble d'activités identifiées avec leurs degrés de confiance.

Le chapitre suivant présente l'implémentation et l'évaluation expérimentale de l'approche *DCR* et de l'approche *DCR-OL* avec un scénario de reconnaissance d'activités dans une maison intelligente. Pour l'approche *Fuzzy-DCR*, sa mise en place ainsi que son évaluation reste un travail futur pour cette thèse.

Chapitre 5

Etude expérimentale des approches DCR et DCR-OL

Sommaire

5.1 Introduction

Nous présentons dans ce chapitre l'étude expérimentale de l'approche *DCR* ainsi que sa version en ligne *DCR-OL* pour la reconnaissance d'activités humaines dans les maisons intelligentes. Cette étude repose sur un seul jeu de données réelles *Aruba*. La mise en place et l'évaluation de l'approche *Fuzzy-DCR* reste un travail futur pour cette thèse.

Le chapitre commence par introduire le jeu de données *Aruba*. Nous détaillons ensuite la phase de pré-traitement du jeu de données en question. Nous décrivons l'environnement de simulation puis nous évoquons l'initialisation du système *DCR* et du système *DCR-OL*. Ensuite, nous présentons un exemple de simulation de l'approche *DCR* et de l'approche *DCR-OL*. Nous terminons ce chapitre par une évaluation expérimentale à travers des métriques, une discussion et une conclusion.

5.2 Le jeu de donnée utilisé : Aruba

Afin de simuler notre approche *DCR*, le choix du jeu de donnée doit s'aligner avec les caractéristiques suivantes :

— Il doit contenir des attributs qui représentent les résultats des deux modules *Perception* puis *Observation* discutés dans la section 1.4.1.

— Il doit concerner un seul participant. La gestion de multi-participants ne rentre pas dans les objectifs de cette thèse.

Le projet CASAS Smart Home [Cook and Schmitter-Edgecombe, 2009] est un projet de recherche multidisciplinaire à la Washington State University (Pullman, WA) dédié à la création d'un environnement domestique intelligent. Il propose divers jeux de données tels que le jeu de données **Aruba** qui répond à nos besoins. Le descriptif de ce jeu de données est noté dans le Tableau 5.1. Ce jeu sauvegarde la vie quotidienne d'une personne âgée. Les données collectées sont obtenues à l'aide de 31 capteurs de mouvement, trois capteurs de porte, cinq capteurs de température et trois capteurs de luminosité. 11 activités ont été réalisées pendant 220 jours (7 mois). Dans notre cas, seuls les capteurs de mouvement et de porte sont considérés.

TABLE 5.1. Détails du jeu de données Aruba

Jeu de données	Genre et âge	nombre de capteurs	Intervalle de temps
Aruba	Une femme âgée	34	7 mois

TABLE 5.2. Statistiques des activités dans le jeu de données Aruba

Id	Activité	Nombre d'évènements	Id	Activité	Nombre d'évènements
1	Bed_to_Toilet	844	7	Relax	173 337
2	Eating	10 421	8	Resperate	211
3	Enter_Home	1 097	9	Sleeping	21 247
4	Housekeeping	10 583	10	Wash_Dishes	8 021
5	Leave_Home	1 161	11	Work	9 927
6	Meal_Preparation	149 875	12	Other	493 274

Ces données sont toutes représentées sous la forme d'une séquence d'événements horodatés avec des activités annotées, comme illustré dans la Figure 5.1. Dans cette dernière, chaque ligne correspond à un évènement de la forme : *{horodatage, id du capteur, statut, étiquette de l'activité}* . Nous notons que l'étiquette de l'activité n'est pas toujours présente.

```
2010-11-04 05:40:46.310862 M003 OFF
2010-11-04 05:40:51.303739 M004 ON Bed_to_Toilet begin
2010-11-04 05:40:52.342105 M005 OFF
2010-11-04 05:40:57.176409 M007 OFF
2010-11-04 05:40:57.941486 M004 OFF
2010-11-04 05:43:24.021475 M004 ON
2010-11-04 05:43:26.273181 M004 OFF
2010-11-04 05:43:26.345503 M007 ON
2010-11-04 05:43:26.793102 M004 ON
2010-11-04 05:43:27.195347 M007 OFF
2010-11-04 05:43:27.787437 M007 ON
2010-11-04 05:43:29.711796 M005 ON
2010-11-04 05:43:30.279021 M004 OFF Bed_to_Toilet end
2010-11-04 05:43:34.261135 M005 OFF
2010-11-04 05:43:35.941892 M007 OFF
2010-11-04 05:43:40.821615 M007 ON
2010-11-04 05:43:45.619681 M007 OFF
2010-11-04 05:43:45.7324 M003 ON   Sleeping begin
2010-11-04 05:43:52.044085 M003 OFF
2010-11-04 05:43:53.185335 M002 ON
2010-11-04 05:43:53.253809 M003 ON
```

FIGURE 5.1. Un exemple d'événements capteurs horodatés avec des activités annotées dans le jeu de données Aruba

Le jeu de données est déséquilibré, car certaines activités sont plus fréquentes que d'autres. Le Tableau 5.2 présente les statistiques des activités en indiquant l'identifiant de l'activité, le nom de l'activité et le nombre d'évènements capteurs par activité dans le jeu de données Aruba. L'activité *Other* contient des événements avec des étiquettes manquantes. Il couvre 54% de l'ensemble des événements capteurs.

Comme décrit ci-dessus, le jeu de données *Aruba* contient un ensemble d'évènements capteur. Or, pour simuler notre approche *DCR*, il faut avoir comme entrée, une base contenant une liste d'attributs. Pour cela, un pré-traitement de ce jeu de données est nécessaire. Nous le présentons dans ce qui suit.

5.3 Pré-traitement du jeu de données Aruba

Le pré-traitement du jeu de données *Aruba* a pour but de préparer nos données d'entrée pour simuler l'approche *DCR*. Ce processus implique quatre étapes principales : la segmentation, l'extraction des attributs, le rajout de l'attribut *localisation* et la création des sous jeux de données selon l'attribut *localisation*. Dans ce travail, nous avons utilisé les résultats des étapes de segmentation et d'extraction des attributs fournis par les auteurs dans [Yala et al., 2017]. La figure 5.2 résume ces étapes présentées ci-dessous :

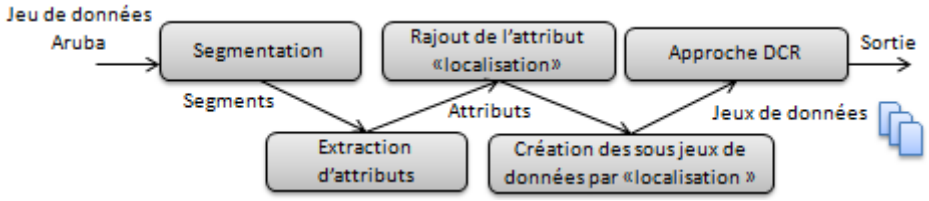


FIGURE 5.2. Pré-traitement du jeu de données Aruba pour simuler l'approche *DCR*

1. **La segmentation** : Segmenter une séquence continue d'évènements capteurs est généralement un processus préalable à la reconnaissance d'activité. Il vise à diviser la liste d'évènements en des segments ou en des fenêtres de taille fixe (intervalle de temps fixe ou nombre d'évènements fixe) contenant des informations sur les activités. La récupération d'informations importantes et utiles à partir d'un flux continu d'évènements capteurs est un problème difficile pour des activités continues [Alzahrani and Kammoun, 2016]. Dans [Yala et al., 2017], les auteurs ont utilisé la technique de fenêtrage à base de capteurs (*the sensor-based windowing technique*) pour traiter les évènements capteurs. Chaque fenêtre contient un nombre égal d'événements qui est fixé à 10 événements. Il est influencé par le nombre moyen d'événements capteurs qui couvrent la durée des différentes activités.
2. **L'extraction des attributs** : Elle vise à extraire les attributs depuis un segment d'évènements capteurs. En d'autres termes, c'est la transformation d'une grande quantité de données en entrée en une représentation réduite avec un ensemble d'attributs, que l'on peut également désigner comme un vecteur d'attributs [Alzahrani and Kammoun, 2016]. Une fois que la technique de segmentation a été effectuée, chaque segment peut être transformé en un vecteur d'attributs. Dans [Yala et al., 2017], les auteurs ont utilisé la méthode basique comme méthode d'extraction des attributs (cette méthode est détaillée dans [Krishnan and Cook, 2014]). Ainsi, à partir

TABLE 5.3. Liste de capteurs dans chaque localisation

Localisation	Id capteur	Localisation	Id capteur
Living	M009,M010,M012,M013,M020	Bedroom 1	M023,M024
Dining	M014	Bedroom 2	M001,M002,M003,M005,M006,M007
Kitchen	M015,M017,M018,M019	Bathroom 1	M031
Office	M025,M026,M027,M028	Bathroom 2	M004
Corridor	M021,M022, M008, M029	Exit	D001,D002, D004, M011, M016, M030

d'un segment, un vecteur d'attributs FV_i est construit avec une dimension fixe contenant : la date du début du premier événement, la date de fin du dernier évènement, la durée du segment et le nombre d'occurrence N de chaque capteur apparu dans le segment $(N_{(D001)}, N_{(D002)}, ..., N_{(M031)}$ où $Dxxx$ sont les capteurs de porte et $Mxxx$ sont les capteurs de mouvement). Étant donné le jeu de données Aruba, si par exemple 34 capteurs sont installés dans la maison intelligente, la dimension du vecteur FV_i doit être égale à $34+3$. FV_i est marqué avec l'étiquette y_i du dernier évènement capteur dans le segment correspondant. Chaque étiquette y_i correspond à une activité (l'ensemble des activités et leurs identifiants sont décrits dans le tableau 5.2). La Figure 5.3 résume les caractéristiques d'un vecteur d'attributs FV. Ainsi, une collection de FV_i et leurs y_i correspondants deviennent les données d'apprentissage sur lesquelles les classifieurs de chaque agent vont être construits.

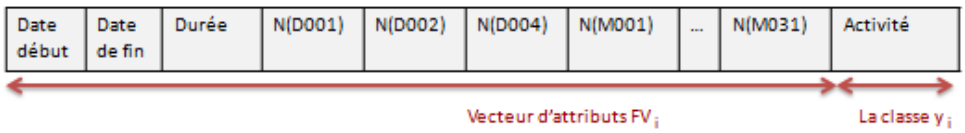


FIGURE 5.3. Vecteur d'attributs et sa classe correspondante

3. **Rajout de l'attribut *localisation*** : Notre approche *DCR* contient un ensemble d'agents affectés à différentes pièces de la maison intelligente. Par conséquent, afin de simuler notre approche, nous devons déterminer l'emplacement de chaque vecteur d'attributs dans la maison intelligente (voir Figure 5.4). Nous distinguons 10 emplacements : *living*, *kitchen*, *dining*, *bedroom1*, *bedroom2*, *bathroom2*, *bathroom2*, *exit*, *corridor* et *office*.

   Le Tableau 5.3 résume l'ensemble des capteurs déployés dans chaque emplacement.

   La localisation de chaque vecteur d'attributs est définie selon la stratégie suivante :

   (a) Déterminer l'emplacement de chaque capteur apparu dans le vecteur d'attributs FV_i

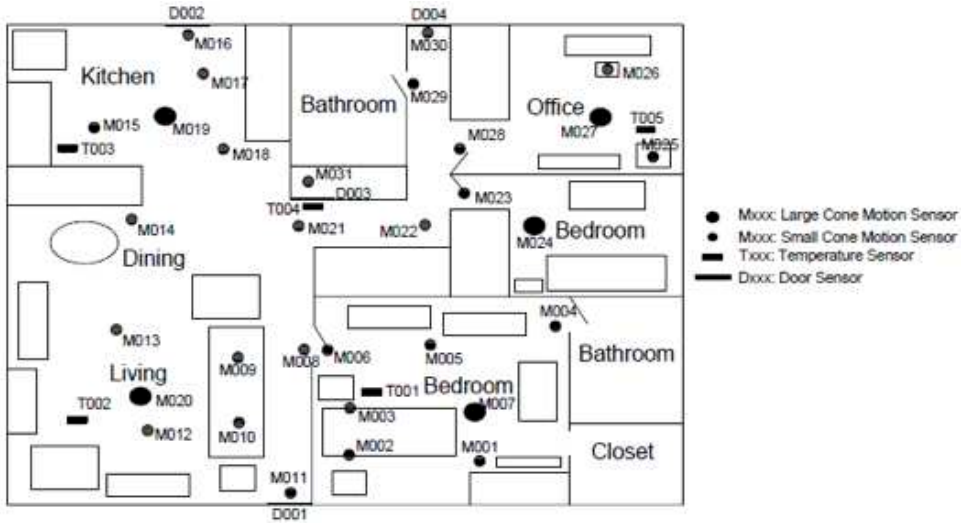


FIGURE 5.4. La carte de la maison intelligente du jeu de données Aruba (source : http ://casas.wsu.edu/datasets/)

(b) Ajouter le nombre d'occurrences de capteurs par emplacement

(c) La localisation choisie est l'emplacement qui a la somme maximale d'occurrences

L'attribut *localisation* est rajouté à chaque vecteur d'attributs (voir Figure 5.5).

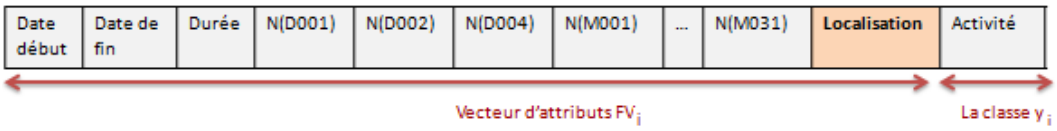


FIGURE 5.5. Rajout de l'attribut *localisation* dans FV_i

4. **Création des sous jeux de données par localisation** : L'ajout de l'attribut *localisation* dans le vecteur FV_i vise à diviser le jeu de données Aruba en différents sous jeux de données selon l'attribut *localisation*. Ainsi, chaque sous jeu de données correspond à une localisation spécifique, contient tous les vecteurs d'attributs associés et peut ensuite être affecté à l'agent identificateur dans cette localisation. Le Tableau 5.4 présente les 10 sous jeux de données par localisation ainsi que l'ensemble des activités qui s'y produisent. Les vecteurs d'attributs FV_i dans ces sous jeux de données n'incluent pas l'attribut *localisation*.

Suite à ces 4 étapes décrites ci-dessus, les sous jeux de données Aruba sont prêts à être utilisés par les agents identificateurs pour simuler l'approche *DCR*. Nous abordons dans la sous-section suivante le descriptif de l'environnement de simulation.

TABLE 5.4. Liste des activités exercées dans chaque localisation de la maison intelligente

Localisation l_i	Jeu de données D_{l_i}	Nombre d'instances	Activités exercées
Living	D_l	322530	{'7' : 158179, '12' : 150616, '6' : 8791, '4' : 3354, '2' : 879, '9' : 312, '10' : 276, '5' : 75, '11' : 48}
Dining	D_d	33789	{'12' : 20715, '2' : 7952, '6' : 3142, '7' : 1020, '4' : 656, '10' : 264, '5' : 32, '11' : 7, '9' : 1}
Kitchen	D_k	241218	{'6' : 128501, '12' : 98821, '10' : 7197, '7' : 3649, '4' : 1810, '2' : 1067, '9' : 118, '5' : 34, '11' : 15, '3' : 6}
Office	D_o	30485	{'12' : 19589, '11' : 9005, '7' : 980, '6' : 322, '4' : 226, '8' : 195, '10' : 73, '5' : 41, '9' : 34, '2' : 20}
Corridor	D_c	52600	{'12' : 45111, '6' : 3114, '7' : 1472, '4' : 1235, '5' : 870, '9' : 249, '11' : 225, '2' : 180, '10' : 99, '3' : 36, '8' : 9}
Bedroom 1	D_{bed1}	36127	{'12' : 23545, '7' : 7120, '6' : 2367, '9' : 1743, '11' : 583, '4' : 478, '2' : 242, '10' : 30, '5' : 18, '3' : 1}
Bedroom 2	D_{bed2}	146002	{'12' : 122432, '9' : 18684, '4' : 2555, '6' : 777, '1' : 771, '7' : 663, '2' : 69, '10' : 27, '11' : 24}
Bathroom 1	D_{bath1}	990	{'12' : 835, '4' : 132, '6' : 21, '7' : 2}
Bathroom 2	D_{bath2}	2667	{'12' : 2507, '1' : 73, '4' : 65, '9' : 8, '7' : 8, '6' : 7}
Exit	D_e	13582	{'12' : 9103, '6' : 2833, '3' : 1054, '7' : 244, '9' : 98, '5' : 91, '4' : 72, '10' : 55, '11' : 20, '2' : 12}

Rappel - '1' : Bed_to_toilet ; '2' : Eating ; '3' : Enter_Home ; '4' : Housekeeping ; '5' : Leave_Home ; '6' : Meal_Preparation ; '7' : Relax ; '8' : Resperate ; '9' : Sleeping ; '10' : Wash_Dishes ; '11' : Work ; '12' : Other.

5.4 Environnement de simulation

Afin de tester et d'évaluer notre approche *DCR* sur l'ensemble des sous jeux de données Aruba, l'environnement de simulation a été utilisé avec deux types de configuration :

— **Configuration matérielle** : les tests ont été effectués sur une machine virtuelle *Windows Server 2012*. Celle-ci possède les caractéristiques suivantes : une CPU Intel(R) Xeon(R) E5-2673 v3 @2.40GHz 2.39GHz (dotée de 8 cœurs hyper threading activés), 28.0Go de RAM et 126Go de disque dur.

— **Configuration logicielle** : nous avons choisi **Anaconda** [1] V3, la plateforme dédiée à la science des données ouvertes. Il comprend le langage le plus populaire **Python**. Pour l'analyse des données, nous avons utilisé la bibliothèque Python open source **Scikit-learn** [2] qui implémente une série de techniques d'apprentissage automatique à savoir le pré-traitement, la validation croisée et la visualisation des algorithmes. Pour la mise en place d'un système multi-agents, nous avons utilisé la plateforme **SPADE** [3] V2.3 (Smart Python multi-Agent Development Environment) qui est une

1. https ://www.continuum.io/anaconda-overview
2. http ://scikit-learn.org/
3. https ://pypi.python.org/pypi/SPADE

plateforme multi-agents basée sur la technologie XMPP/Jabber et écrite en Python.

Avec la mise en place de cet environnement de simulation, nous pouvons entamer l'analyse des sous jeux de données Aruba dans la sous section suivante. L'objectif est de construire les éléments qui composent nos différents agents identificateurs à savoir leurs ensembles de classifieurs $Clf_{A_{l_i}}$ et leurs listes d'activité $ACT_{A_{l_i}}$.

5.5 Initialisation des systèmes DCR et DCR-OL

Dans cette section, nous décrivons l'étape initialisation deux modèles DCR et DCR-OL.

5.5.1 Initialisation du système DCR

DCR modélise la maison intelligente (voir Figure 5.4) comme étant un système multi-agents $MAS = \{A_l, A_d, A_k, A_o, A_c, A_{bed1}, A_{bed2}, A_{bath1}, A_{bath2}, A_c\}$ où les agents sont assignés respectivement aux localisations livingroom, dining, kitchen, office, corridor, bedroom1, bedroom2, bathroom1, bathroom2 et corridor . Le seuil δ est choisi à 80%. Les éléments composants des différents agents identificateurs dans *MAS* à savoir leurs classifieurs $Clf_{A_{l_i}}$ et leurs listes d'activité $ACT_{A_{l_i}}$, doivent être construits au préalable.

— **Construction des classifieurs des agents** : une analyse des sous jeux de données est requise pour choisir les classifieurs adéquats par rapport à la nature des données de chaque localisation des agents identificateurs. Nous avons choisi d'utiliser des algorithmes d'apprentissage basés sur un mécanisme de raisonnement tels que les forêts aléatoires ("Random Forests RF"), les arbres de décision ("Decision trees DT"), les extra-arbres ("extra-trees ExT") et les réseaux bayésiens naïves ("Naive Bayes NB"). Le but d'utiliser ce type de classifieurs est de pouvoir générer des règles logiques (Si-Alors) pour une utilisation future. La classification a été réalisée en utilisant une validation croisée d'ordre 10. Cette analyse vise à choisir le meilleur classifieur pour chaque sous jeux de données. Le Tableau 5.5 indique la valeur de la métrique *taux de reconnaissance* (voir Annexe A.2) pour tous les sous jeux de données selon la localisation de l'agent identificateur en employant les quatre classifieurs RF, DT, Ext et NB.

  À la lecture du Tableau 5.5, nous remarquons que le classifieur RF présente la meilleure valeur du taux de reconnaissance pour tous les sous jeux de données. Par conséquent, tous les agents adopteront un modèle d'activité basé sur le classifieur forêt aléatoire

TABLE 5.5. La *taux de reconnaissance* des différents classifieurs pour les sous jeux de données Aruba

Jeu de données	Localisation	Nombre d'instances	RF (%)	DT (%)	ExT (%)	NB (%)
D_l	Living	322530	75.96	72.36	60.3	59.73
D_d	Dining	33789	66.04	61.46	64.72	58.71
D_k	Kitchen	241218	58.96	55.67	56.43	51.38
D_o	Office	30485	66.04	61.54	65.55	65.55
D_c	Corridor	52600	84.46	77.82	83.15	26.66
D_{bed1}	Bedroom 1	36127	66.98	63.31	66.28	42.12
D_{bed2}	Bedroom 2	146002	91.84	89.89	90.99	48.61
D_{bath1}	Bathroom 1	990	81.23	70.01	79.91	64.92
D_{bath2}	Bathroom 2	2667	92.7	90.94	92.44	58.32
D_e	Exit	13582	74.05	68.64	73.06	35.68

RF. Il convient de noter que le choix d'un classifieur pour construire un modèle d'activité dépend principalement de l'ensemble des données. Ainsi, avec notre jeu de données, nous avons obtenu le même type de classifieur pour tous les sous jeux de données, mais ce n'est pas toujours le cas en fonction des jeux de données utilisés. Les modèles d'activités peuvent en effet être construits avec des classifieurs différents.

— **Construction des listes d'activité des agents** : Après avoir construit les classifieurs RF pour tous les agents, nous passons à la détermination de la liste $ACT_{A_{l_i}}$. Chaque agent incorpore cette liste qui est un ensemble d'activités reconnues par le classifieur avec leurs degrés de confiance. Ces activités se déterminent après la construction du classifieur en question. Le degré de confiance correspond à la mesure du F1-score (voir Annexe A.2) de chaque activité pour un classifieur bien spécifique. Le contenu de la liste $ACT_{A_{l_i}}$ de chaque agent est décrit dans le Tableau 5.6.

5.5.2 Initialisation du système DCR-OL

L'approche *DCR-OL* est une version améliorée de l'approche *DCR*. Ainsi, *DCR-OL* adopte la même représentation et les mêmes données d'initialisation que celles de *DCR*. Pour la nouvelle composante des agents apprenants LA_i qui est la liste des valeurs de performance $PERF_{LA_i} = \{p_{LA_1}, p_{LA_2}, ..., p_{LA_n}\}$, est initialisée avec la valeur 1 c-à-d que toutes les valeurs p_{LA_i} sont égales à 1.

Afin d'illustrer l'approche *DCR* et l'approche *DCR-OL*, nous présentons dans la sous section suivante un exemple d'application pour chacune.

TABLE 5.6. Liste des activités reconnues $ACT_{A_{(l_i)}}$ pour chaque agent

Agent A_{l_i}	Jeu de données D_{l_i}	Liste $ACT_{A_{(l_i)}}$
A_l	D_l	{(2, 42.19%), (4, 43.64%), (5, 21.62%), (6, 61.34%), (7, 86.21%), (9, 89.03%), (10, 24.14%), (11, 30.0%), (12, 83.92%)}
A_d	D_d	{(2, 73.23%), (4, 31.02%), (5, 44.44%), (6, 51.91%), (7, 75.71%), (9, 0.0%), (10, 28.57%), (11, 0.0%), (12, 84.62%)}
A_k	D_k	{(2, 19.62%), (3, 100.0%), (4, 25.2%), (5, 11.11%), (6, 74.6%), (7, 64.37%), (9, 81.58%), (10, 23.56%), (11, 0.0%), (12, 68.78%)}
A_o	D_o	{(2, 0.0%), (4, 64.29%), (5, 42.86%), (6 , 68.93%), (7, 82.84%), (8, 77.86%), (9, 90.91%), (10, 64.52%), (11, 69.71%), (12, 85.38%)}
A_c	D_c	{(2, 37.93%), (3, 23.53%), (4, 33.95%), (5, 48.97%), (6, 59.0%), (7, 63.13%), (8, 0.0%), (9, 83.93%), (10, 22.86%), (11, 5.56%), (12, 94.62%)}
A_{bed1}	D_{bed1}	{(2, 66.67%), (4, 39.51%), (5, 54.55%), (6, 70.44%), (7, 69.77%), (9, 70.09%), (10, 50.0%), (11, 47.2%), (12, 84.63%)}
A_{bed2}	D_{bed2}	{(1, 51.58%), (2, 0.0%), (4, 31.2%), (6, 61.08%), (7, 52.0%), (9, 89.84%), (10, 44.44%) (11, 46.15%), (12, 96.87%)}
A_{bath1}	D_{bath1}	{(4, 56.67%), (12, 22.22%), (6, 93.71%)}
A_{bath2}	D_{bath2}	{(1, 57.78%), (4, 56.25%), (6, 0.0%), (7, 66.67%), (9, 0.0%), (12, 97.5%)}
A_e	D_e	{(2, 50.0%), (3, 61.13%), (4, 55.56%), (5, 21.62%), (6, 80.1%), (7, 75.0%), (9, 98.25%), (10, 30.77%), (11, 75.0%), (12, 89.16%)}

5.6 Exemples d'application de DCR et de DCR-OL

5.6.1 Exemple d'application de DCR

Étant donné l'exemple de l'agent office A_o, il comprend les éléments suivants dans son état initial :

— son sous jeu de données D_o.

— son classifieur clf_{A_o}

— sa liste d'activités ACT_{A_o}

— sa liste d'accointances ACQ_{A_o} qui est vide

L'agent A_o procède par les étapes suivantes :

Première étape : il prend en entrée le premier vecteur d'attributs dans D_o, interroge son classificateur clf_{A_o} et obtient l'activité prédite locale pa_o (e.g. pa_o est l'activité ayant un id 5 qui correspond à l'activité *leave_home*). Ensuite, il détermine son degré de confiance correspond d_{pa_o} depuis sa liste d'activité ACT_{A_o} (voir Tableau 5.6) qui vaut 42,86%. Celui-ci est inférieur à 80%, ce qui signifie que l'agent A_o n'est pas capable de bien reconnaître l'activité *leave_home* et doit collaborer avec certains agents qui sont capables de la reconnaître avec une précision meilleure que 42,86%.

Deuxième étape : il commence à chercher ces agents d'accointances et détermine sa

liste ACQ_{A_o} qui inclut les agents qui peuvent reconnaître d_{pa_o}. Dans notre exemple, la liste ACQ_{A_o} contient les agents A_l , A_d , A_k , A_c , A_{bed1} et A_e.

Troisième étape : l'agent A_o envoie son entrée (son vecteur d'attributs) à certains agents sélectionnés dans sa liste ACQ_{A_o}. Les agents sélectionnés sont les agents qui sont capables d'identifier pa_o mieux que A_o, ce qui signifie que leurs degrés de confiance pour pa_o est supérieur à d_{pa_o} (42.86%). Les agents sélectionnés sont A_d, A_{bed1} et A_c (voir Tableau 5.6).

Quatrième étape : l'agent A_o reçoit les prédictions étrangères des agents récepteurs sélectionnés et les insère dans la liste $pAct\_list_o$ y compris sa prédiction locale pa_o. Ensuite, il applique une stratégie d'agrégation parmi celles décrites dans la section 1.4.5. Dans cet exemple, nous avons choisi d'appliquer la stratégie *max-trust* qui permet de choisir l'activité la plus confiante dans la liste $pAct\_list_o$. L'agent Ao calcule la moyenne pondérée des activités récurrentes dans la liste. Dans notre cas, il s'agit de l'activité 5. Ainsi, il génère deux activités : l'activité *leave_home* (id 5) avec un degré de confiance égal à 47,28% et l'activité *Other* (id 12) avec un degré de confiance égal à 94,62%. Avec la stratégie *max-trust*, l'activité *Other* (id 12) est générée comme l'activité finale.

Cet exemple est illustré dans la Figure 5.6.

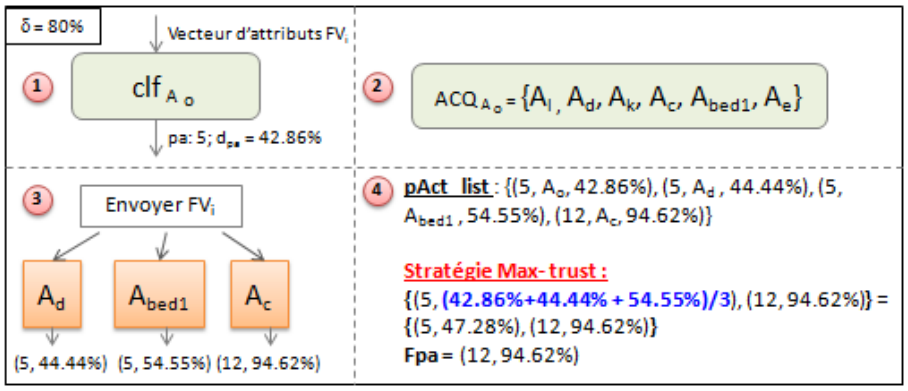


FIGURE 5.6. Un exemple d'application de l'approche *DCR* appliqué à l'agent *office* A_o

Ce processus est appliqué à tous les vecteurs d'attributs existants dans le sous jeu de données D_o.

5.6.2 Exemple d'application de DCR-OL

Nous gardons le même exemple de l'agent office A_o décrit précédemment (les mêmes données d'exemple) c-à-d nous gardons le même traitement des étapes 1, 2 et 3.

Quatrième étape : nous appliquons l'approche *DCR-OL* qui consiste à faire apprendre les agents apprenants du système *MAS* leurs valeurs de performance et à s'adapter aux nouveaux vecteurs d'attributs. Plus précisément, nous appliquons la stratégie de performance pondérée maximale.

Nous avons supposé qu'au début tous les agents du système *MAS* sont performants et donc la liste $PERF_{A_o}$ de l'agent A_o est initialisée à 1 (voir Figure 5.7). L'agent A_o reçoit les prédictions étrangères des agents récepteurs sélectionnés et les insère dans la liste $pAct\_list_o$ y compris sa prédiction locale pa_o. Il calcule la performance pondérée pour chaque activité distincte dans $pAct\_list_o$: $PP(5)$ et $PP(12)$. Avec la stratégie *Max-wPerf*, l'agent A_o choisit l'activité ayant la valeur de performance pondérée maximale qui est l'activité $id5$ avec une valeur de performance de 75%. Ainsi, l'activité prédite finale Fpa_o est l'activité *leave_home* $id5$.

Cinquième étape : L'agent A_o passe à la phase d'apprentissage du vecteur $PERF_{A_o}$. En effet, il compare l'activité prédite finale Fpa_o avec l'activité réelle *RP* (voir Figure 5.7). Dans notre cas, le test de comparaison retourne $faux$ ce qui signifie que le vecteur de $PERF_{A_o}$ doit être modifié.

Sixième étape : elle concerne la mise à jour du vecteur de $PERF_{A_o}$. Nous appliquons la mesure $Perf-measure$ pour chaque agent dans la liste $pAct\_list_o$. Par conséquent, les valeurs de performance des agents qui ont mal prédit l'activité *RP*, diminuent. Ces agents sont A_o, A_d et A_{bed1} dont leurs valeurs de performance p_{A_o}, p_{A_d} et $p_{A_{bed1}}$ valent respectivement 0.999. La valeur de performance de l'agent qui a correctement prédit l'activité *RP*, reste la même. Cet agent est l'agent A_c dont sa valeur de performance p_{A_c} vaut respectivement 1. Le vecteur de performance $PERF_{A_o}$ est mis à jour avec les nouvelles valeurs de performance des agents concernés qui sont A_o, A_d et A_{bed1}.

Au prochain arrivé d'un vecteur d'attribut, l'application de la stratégie *Max-wPerf* tiendra compte du vecteur de performance $PERF_{A_o}$ mis à jour.

Dans la section suivante, nous entamons l'évaluation expérimentale de l'approche *DCR*.

5.7 Évaluation expérimentale de l'approche DCR

Après l'analyse des sous jeux de données Aruba (construction des classifieurs $clf_{A_{l_i}}$ et des listes d'activité $ACT_{A_{l_i}}$ des agents), l'approche *DCR* peut être lancée. En effet, chaque agent initiateur S possède son sous jeu de données et applique l'approche *DCR* avec une validation

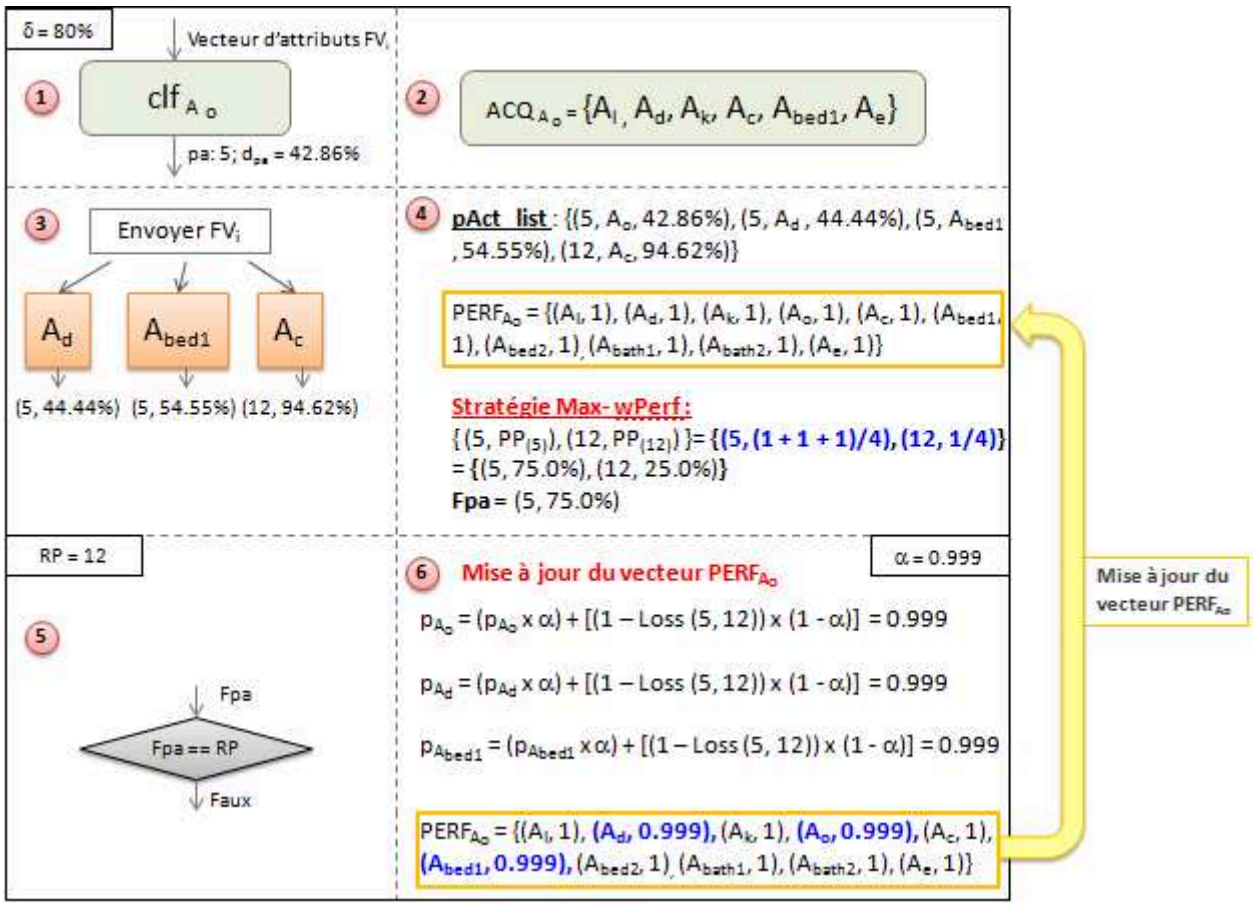


FIGURE 5.7. Un exemple d'application de l'approche *DCR-OL* appliqué à l'agent *office* A_o

croisée d'ordre 10. Ainsi, chaque sous jeu de données est divisé en 10 dossiers. Chaque dossier représente l'ensemble des données de test (*testing dataset*) et la fusion des autres dossiers représente l'ensemble des données d'apprentissage (*training dataset*). Chaque agent initiateur *S* possède respectivement 10 jeux de données d'apprentissage et de test et puis 10 classifieurs RF. Un agent initiateur *S* prend en entrée chaque vecteur d'attributs de l'ensemble de données de test (*testing dataset*), lance le système *DCR* et génère en sortie l'ensemble des activités prédites avec leur degré de confiance (la liste $pAct_{list_{(S)}}$). Cette tâche est répétée 10 fois pour tous les jeux de données de test et pour chaque agent initiateur dans *MAS*.

Pour évaluer la performance de notre approche *DCR*, les métriques d'évaluation utilisées dans cette contribution sont :

— **le taux de reconnaissance ("accuracy")** (voir Annexe A.2) - indique le pourcentage d'instances correctement classées

— **la F-mesure ("F-measure")** (voir Annexe A.2) - est la moyenne harmonique des

métriques la précision et le rappel (*precision and recall*) [Ting, 2010]

— **le temps de traitement** - le temps nécessaire pour exécuter le programme *DCR*

— **le nombre de messages** - le nombre de messages échangés entre les agents

5.7.1 Évaluation avec les métriques : taux de reconnaissance et F-mesure

Le taux de reconnaissance (*accuracy*) et la F-mesure sont utilisés pour évaluer les approches suivantes pour chaque sous jeu de données de Aruba en utilisant la validation croisée (k=10) :

— *W-DCR :* une version dégradée de l'approche *DCR* sans tenir compte de l'aspect collaboratif entre les agents. En effet, chaque agent initiateur *S* interroge son classifieur suite à l'arrivée de ses vecteurs d'attributs dans son jeu de données et génère directement en sortie l'activité prédite.

— *DCR avec la stratégie max-trust* : c'est *DCR* en utilisant la stratégie du plus confiant lors de la prise de décision finale.

— *DCR avec la stratégie max-freq.* : c'est *DCR* en utilisant la stratégie du plus fréquent lors de la prise de décision finale.

— *DCR avec la stratégie stacking* : c'est *DCR* en utilisant la stratégie du méta-apprentissage lors de la prise de décision finale.

Considérons l'exemple du sous jeu de données *office*, le Tableau 5.7 détaille le calcul des métriques pour les différentes approches décrites ci-dessus. En outre, nous rajoutons une colonne "upper bound" qui représente la borne supérieure des valeurs des métriques qu'une approche *DCR* peut atteindre. Le but est d'évaluer les trois stratégies de résolution de conflits. Concernant le reste des sous jeux de données de Aruba (l'évaluation des autres agents à part l'agent A_o), le détail des tableaux est décrit dans l'Annexe B.

Selon le Tableau 5.7, nous remarquons que *DCR* avec ses différentes stratégies dégrade, garde ou améliore les résultats de l'approche *W-DCR* pour les différents dossiers de la validation croisée. En particulier, les collaborations peuvent augmenter ou diminuer le taux de reconnaissance de l'agent initiateur *office*.

En outre, l'approche *DCR* présente des résultats remarquables en termes de taux de reconnaissance global et de F-mesure global par rapport à l'approche *W-DCR*. Nous notons que *DCR* avec application de la stratégie *stacking* est la meilleur méthode de résolution de conflits suivie par la méthode *max-freq*. Ceci s'explique par la collaboration entre l'agent initiateur *office* et ces agents récepteurs sélectionnés qui consolide le taux de reconnaissance de l'activité correcte.

TABLE 5.7. Agent A_o : évaluation du jeu de données *office* D_o

D_o		W-DCR	DCR			
			max-trust	max-freq.	stacking	upper bound
1-fold	Acc.	59.19%	53.02%	59.25%	**59.48%**	63.58 %
	F-meas.	52.93%	36.94%	53.02%	**53.18%**	56.52%
2-fold	Acc.	54.66%	26.54%	57.55%	**67.06%**	57.58%
	F-meas.	55.12%	11.38%	**57.54%**	54.20%	57.58%
3-fold	Acc.	66.31%	51.35%	66.7%	**67.58%**	71.62%
	F-meas.	62.17%	36.02%	62.53%	**62.63%**	67.17%
4-fold	Acc.	75.0%	76.67%	75.3%	**79.46%**	84.78%
	F-meas.	75.96%	69.90%	**75.95%**	70.57%	84.19%
5-fold	Acc.	60.63%	97.47%	61.15%	**98.13%**	98.43%
	F-meas.	73.35%	96.87%	74.29%	**97.20%**	98.16%
6-fold	Acc.	61.91%	90.06%	62.07%	**90.26%**	96.88%
	F-meas.	69.57%	**86.02%**	69.95%	85.82%	96.81%
7-fold	Acc.	54.66%	62.43%	54.59%	**64.11%**	76.41%
	F-meas.	52.50%	49.81%	**52.18%**	50.36%	71.97%
8-fold	Acc.	68.5%	65.65%	68.64%	**70.05%**	75.49%
	F-meas.	66.57%	54.02%	66.22%	**66.35%**	72.05%
9-fold	Acc.	61.91%	50.62%	61.91%	**62.98%**	68.96%
	F-meas.	58.01%	35.02%	57.55%	**58.12%**	64.96%
10-fold	Acc.	64.82%	53.0%	64.79%	**70.28%**	71.47%
	F-meas.	65.27%	40.87%	65.07%	**68.10%**	71.46%
Global	Acc.	62.76%	62.68%	63.19%	**72.93%**	76.52%
	F-meas.	63.16%	54.38%	63.43%	**66.65%**	74.08%

TABLE 5.8. Comparaison entre l'approche centralisée et les différentes approches distribuées en termes de taux de reconnaissance et de F-mesure

Agents		Central.	Système CASE	W-DCR	DCR			
					max-trust	max-freq.	stacking	upper bound
A_l	Acc.	-	63.52%	76.04%	76.38%	**76.4%**	76.36%	76.97%
	F-meas.	-	63.46%	75.43%	75.38%	**75.61%**	75.1%	76.37%
A_d	Acc.	-	51.54%	65.81%	67.25%	67.44%	**68.01%**	70.60%
	F-meas.	-	51.32%	64.24%	62.62%	**64.13%**	61.9%	68.25%
A_k	Acc.	-	52.17%	58.81%	50.41%	51.45%	**61.31%**	83.17%
	F-meas.	-	52.5%	58.97%	49.77%	50.88%	**52.24%**	83.16%
A_o	Acc.	-	53.51%	62.76%	62.68%	63.19%	**72.93%**	76.52%
	F-meas.	-	54.61%	63.16%	54.38%	63.43%	**66.65%**	74.08%
A_c	Acc.	-	75.67%	84.13%	85.21%	85.24%	**85.65%**	86.89%
	F-meas.	-	75.29%	80.76%	79.71%	**80.3%**	79.48%	82.67%
A_{bed1}	Acc.	-	49.82	65.81%	65.16%	67.01%	**68.40%**	77.97%
	F-meas.	-	51.22%	65.75%	58.37%	**64.7%**	60.48%	76.04%
A_{bed2}	Acc.	-	85.72%	91.99%	**92.57%**	92.52%	92.48%	92.77%
	F-meas.	-	85.51%	91.18%	91.25%	**91.35%**	91.0%	91.60%
A_{bath1}	Acc.	-	78.18%	80.10%	77.78%	**83.53%**	81.78%	86.36%
	F-meas.	-	76.79%	76.48%	75.69%	78.36%	**78.56%**	83.03%
A_{bath2}	Acc.	-	90.92%	92.34%	93.39%	**93.99%**	93.69%	94.07%
	F-meas.	-	91.02%	91.71%	92.31%	**92.55%**	91.38%	92.66%
A_e	Acc.	-	57.4%	73.86%	73.17%	74.24%	**74.26%**	76.97%
	F-meas.	-	57.25%	72.34%	68.52%	**72.35%**	70.37%	74.98%
Global	Acc.	72.94%	65.84	75.16%	74.4%	75.5%	**77.48%**	82.23%
	F-meas.	71.80%	65.89%	74.0%	70.8%	**73.36%**	72.71%	80.28%

Considérons le système *MAS*, le Tableau 5.8 résume les valeurs globales des métriques taux de reconnaissance et F-mesure de tous les agents. L'apport de notre approche *DCR* est mis en relief selon les comparaisons suivantes :

— **L'approche DCR vs l'approche centralisée** : vise à montrer l'intérêt d'une approche distribuée par rapport à une approche centralisée. Celle-ci consiste à construire le modèle d'activité (le classifieur RF) sur l'ensemble de jeu de données Aruba. D'après le Tableau 5.8, notre approche *DCR* avec ses différentes méthodes d'agrégation surpasse l'approche centralisée en termes des métriques taux de reconnaissance et F-mesure et en particulier avec *DCR (stacking) et DCR (max-freq.)* respectivement. Ces derniers surpassent l'approche centralisée jusqu'à 5% et 2%.

— **L'approche DCR vs l'approche CASE** : met en relief notre approche *DCR* par rapport à une approche existante et discutée dans la section des travaux connexes et que nous avons implémenté. Le Tableau 5.8 illustre les valeurs des métriques obtenues par le système CASE [Cicirelli et al., 2016]. Ce dernier est une approche client-serveur distribuée, dont l'ensemble des agents serveurs ne collaborent pas. Les valeurs des métriques obtenues par notre approche sont meilleures que celles obtenues par le système CASE et atteignent une différence approximative de 10%. Ceci met en évidence l'intérêt de l'aspect collaboratif entre les agents.

— **L'approche DCR vs l'approche W-DCR** : valorise l'aspect collaboratif de l'approche *DCR* par rapport à l'approche *W-DCR*. Le Tableau 5.8 indique que la valeur du taux de reconnaissance est légèrement meilleure avec l'approche *DCR (max-freq.)* comparé à l'approche *W-DCR*. De plus, l'approche *DCR (stacking)* améliore l'approche *W-DCR* jusqu'à 2.3%. Cependant, l'approche *DCR (max-trust)* dégrade l'approche *W-DCR* ce qui souligne que la méthode *max-trust* n'est pas appropriée pour la résolution de conflits dans le système *DCR*. Pour la métrique F-mesure, l'approche *DCR* dégrade sa valeur par rapport à l'approche *W-DCR* et en particulier par l'approche *DCR (max-trust)*.

— **L'approche DCR (max-trust) vs l'approche DCR (max-freq.) vs l'approche DCR (stacking) vs l'approche DCR (upper bound)** : Le but est de guider le choix de la méthode de résolution de conflits. L'approche *DCR (upper bound)* est l'approche *DCR* avec une méthode de résolution de conflits parfaite avec un risque d'erreur de 0%. Elle est représentée par la colonne *upper bound*. Cette dernière comporte les valeurs maximales des métriques qu'une méthode de résolution peut atteindre. Afin de déterminer les valeurs de la colonne *upper bound* pour le taux de reconnaissance par

exemple, il suffit de chercher l'activité correcte qui peut figurer dans la liste des activités prédites (locale + étrangères). Le Tableau 5.8 souligne que la méthode *stacking* est la meilleure méthode d'agrégation comparée aux deux autres en termes de taux de reconnaissance. Cependant, la méthode *max-freq.* est considérée meilleur que les deux autres en termes de F-mesure. Les deux méthodes choisies *DCR (max-freq.)* et *DCR(stacking)* se rapprochent considérablement de la borne supérieur *upper bound* (82.23%). Ainsi, une nouvelle méthode de résolution peut être mise en place pour se rapprocher plus du taux maximal (82.23%).

Dans la suite, nous continuons l'évaluation de l'approche *DCR* par rapport au temps de traitement.

5.7.2 Évaluation du temps de traitement

La figure 5.8 indique le temps de traitement moyen en minutes des différents sous jeux de données Aruba pour chaque agent dans *DCR* (sans tenir compte du temps d'application des méthodes de résolution de conflits). L'agent *kitchen* A_k présente le temps de traitement le plus élevé qui dépasse 300 mn et atteint les 30 heures. Ceci est dû à la collaboration importante effectuée avec les autres agents sélectionnés lors de sa prédiction locale de l'activité *other* (id 12) et de l'activité *meal_preparation* (id 6). Selon le Tableau 5.4, ces activités représentent respectivement 41% et 53% du total des activités dans le jeu de données D_k puisque leurs degrés de confiance sont inférieurs à 80%. Les autres agents présentent un temps de traitement n'excédant pas les 3 heures. Cela est dû à l'activité *other* qui présente plus de 50% du total des activités dans les jeux de données et dont le degré de confiance est supérieur à 80%. Ainsi, la collaboration avec les autres agents n'est pas requise.

5.7.3 Évaluation avec le nombre de messages échangés

Les différents agents échangent des messages lorsqu'ils collaborent ensemble. Les messages peuvent être des messages de type demande ou de type réponse. Il convient de noter que l'échange de messages commence lorsque l'activité prédite locale de l'agent initiateur S possède un degré de confiance inférieur au seuil δ (80%). La figure 11 décrit la moyenne du nombre de messages échangés entre l'agent initiateur et ses agents récepteurs sélectionnés. L'agent *kitchen* A_k présente le nombre de messages le plus élevé qui dépasse les 50000 messages et atteint les 652726 messages. Ce résultat est cohérent avec le temps de traitement qui atteint les 30 heures. La plupart des prédictions locales concernent les activités *other* (id 12)

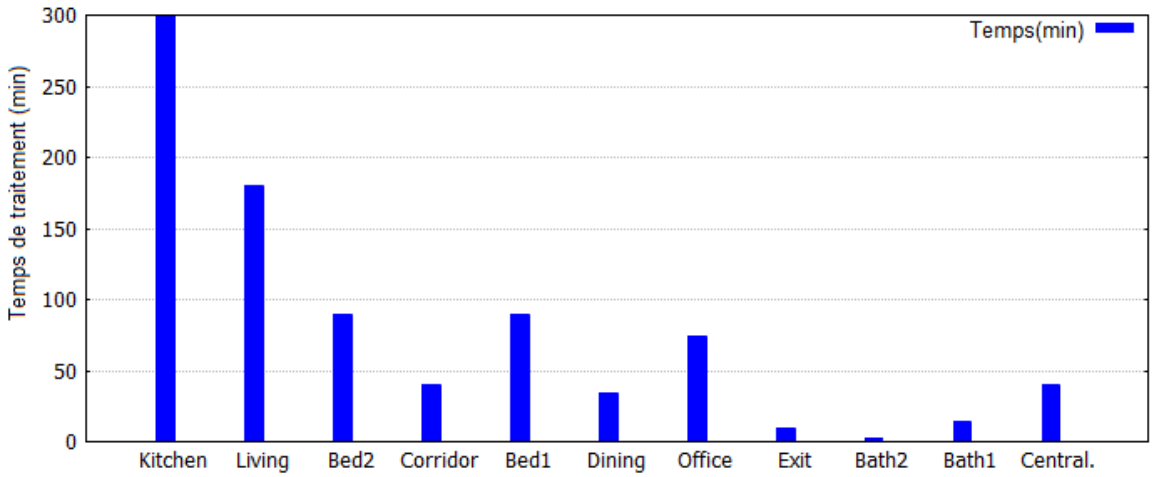


FIGURE 5.8. Temps de traitement des différents agents dans *DCR* et de l'approche centralisée

et *meal_preparation* (id 6), qui représentent respectivement 41% et 53% du total des activités dans le jeu de données D_k, avec un degré de confiance inférieur à 80%.

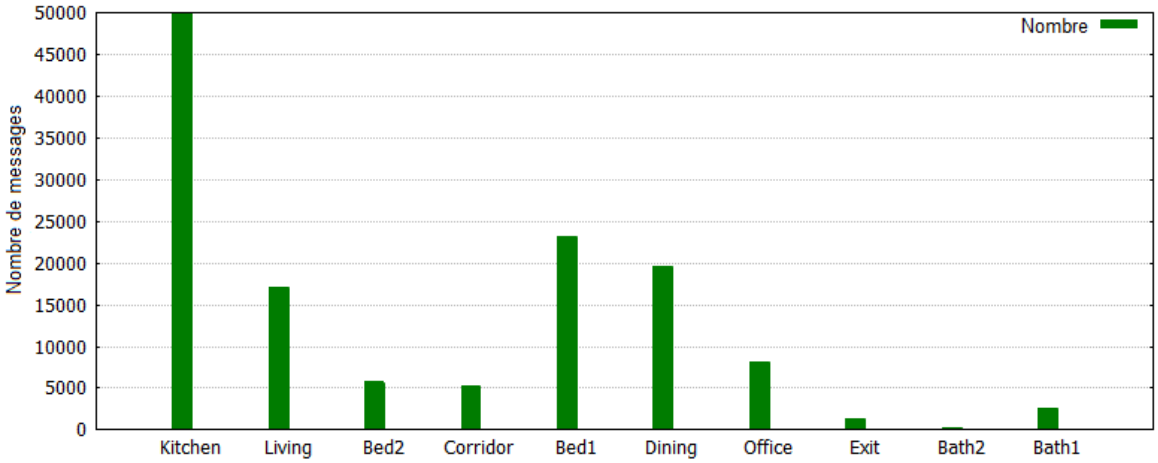


FIGURE 5.9. Nombre de messages échangés entre l'agent initiateur et les autres agents

5.8 Évaluation expérimentale de l'approche DCR-OL

L'évaluation expérimentale est réalisée avec le mode de validation croisée d'ordre 10. La simulation du système *DCR-OL* est la même que le système *DCR*. En effet, un agent initiateur *S* prend en entrée chaque vecteur d'attributs de l'ensemble de données de test (*testing dataset*), lance le système *DCR-OL* et génère en sortie l'activité prédite finale avec son degré de performance pondérée (ce degré joue le rôle d'un degré de confiance). Cette tâche est

répétée 10 fois pour tous les jeux de données de test et pour chaque agent initiateur dans *MAS*.

5.8.1 Évaluation avec les métriques : taux de reconnaissance et F-mesure

Pour évaluer la performance de notre approche *DCR-OL*, les métriques d'évaluation utilisées dans cette contribution sont : le taux de reconnaissance et la F-mesure.

Considérons le même exemple du sous jeu de données *office*, le Tableau 5.9 rajoute une colonne pour l'approche *DCR-OL* et en particulier la nouvelle stratégie de résolution de conflits *Max-wPerf*. Le but est d'évaluer cette nouvelle stratégie en ligne par rapport aux trois stratégies de résolution de conflits pour l'approche *DCR*. Concernant le reste des sous jeux de données de Aruba (l'évaluation des autres agents à part l'agent LA_o), le détail des tableaux est décrit dans l'Annexe B.

Nous constatons que la méthode *max-wPerf* améliore le taux de reconnaissance (66.51%) par rapport aux deux méthodes *max-freq* (taux de reconnaissance 63.19%) et *max-trust* (taux de reconnaissance 62.68%). Cependant, la méthode *stacking* reste toujours la meilleure en terme de taux de reconnaissance.

Concernant la métrique F-mesure, la méthode *max-wPerf* dépasse la méthode *max-trust* mais elle reste toujours moins performante que les deux méthodes *max-freq* et *stacking*.

Considérons le système *MAS*, le Tableau 5.10 résume les valeurs globales des métriques taux de reconnaissance et F-mesure de tous les agents. L'apport de notre approche *DCR-OL* est mis en relief par rapport à l'approche *DCR* :

L'approche DCR-OL (max-wPerf) vs l'approche DCR (max-trust) vs l'approche DCR (max-freq.) vs l'approche DCR (stacking) vs l'approche DCR (upper bound) : la méthode *max-wPerf* améliore le taux de reconnaissance par rapport aux deux méthodes *max-freq* et *max-trust*. Cependant, la méthode *stacking* reste toujours la meilleure en terme de taux de reconnaissance. D'autre part, la méthode *max-wPerf* dépasse la méthode *max-trust* et la méthode *stacking* mais elle reste toujours moins performante que la méthode *max-freq* en terme de F-mesure.

Pour conclure, la méthode *stacking* est la meilleure méthode de résolution de conflits suivie de la méthode en ligne *max-wPerf* en terme de taux de reconnaissance et la méthode *max-freq.* est la meilleure méthode de résolution de conflits suivie de la méthode en ligne *max-wPerf* en terme de F-mesure.

TABLE 5.9. Agent LA_o : évaluation du jeu de données *office* D_o avec l'approche *DCR-OL*

D_o		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	59.19%	53.02%	59.25%	**59.48%**	54.27%	63.58%
	F-meas.	52.93%	36.94%	53.02%	**53.18%**	39.72%	56.52%
2-fold	Acc.	54.66%	26.54%	57.55%	**57.58%**	37.57%	67.06%%
	F-meas.	55.12%	11.38%	**57.54%**	54.20%	33.26%	57.58%
3-fold	Acc.	66.31%	51.35%	66.7%	**67.58%**	57.28%	71.62%
	F-meas.	62.17%	36.02%	**62.53%**	62.63%	48.75%	67.17%
4-fold	Acc.	75.0%	76.67%	75.3%	**79.46%**	77.0%	84.78%
	F-meas.	75.96%	69.90%	**75.95%**	70.57%	70.28%	84.19%
5-fold	Acc.	60.63%	97.47%	61.15%	**98.13%**	97.57%	98.43%
	F-meas.	73.35%	96.87%	74.29%	**97.20%**	96.92 %	98.16%
6-fold	Acc.	61.91%	90.06%	62.07%	**90.26%**	89.86%	96.88%
	F-meas.	69.57%	**86.02%**	69.95%	85.82%	85.72%	96.81%
7-fold	Acc.	54.66%	62.43%	54.59%	**64.11%**	62.57%	76.41%
	F-meas.	52.50%	49.81%	**52.18%**	50.36%	50.08%	71.97%
8-fold	Acc.	68.5%	65.65%	68.64%	**70.05%**	65.94%	75.49%
	F-meas.	66.57%	54.02%	66.22%	**66.35%**	54.66%	72.05%
9-fold	Acc.	61.91%	50.62%	61.91%	**62.98%**	55.48%	68.96%
	F-meas.	58.01%	35.02%	57.55%	**58.12%**	44.70%	64.96%
10-fold	Acc.	64.82%	53.0%	64.79%	**70.28%**	67.18%	71.47%
	F-meas.	65.27%	40.87%	65.07%	**68.10%**	67.15%	71.46%
Global	Acc.	62.76%	62.68%	63.19%	**72.93%**	66.51%	76.52%
	F-meas.	63.16%	54.38%	63.43%	**66.65%**	59.12%	74.08%

TABLE 5.10. Comparaison entre l'approche centralisée et les différentes approches distribuées en termes de taux de reconnaissance et de F-mesure

Agents		Central.	CASE	W-DCR	DCR			DCR-OL	upper bound
					max-trust	max-freq.	stacking	max-wPerf	
LA_l	Acc.	-	63.52%	76.04%	76.38%	76.4%	76.36%	**76.51%**	76.97%
	F-meas.	-	63.46%	75.43%	75.38%	**75.61%**	75.1%	75.42%	76.37%
LA_d	Acc.	-	51.54%	65.81%	67.25%	67.44%	**68.01%**	67.99%	70.60%
	F-meas.	-	51.32%	64.24%	62.62%	64.13%	61.9%	**64.21%**	68.25%
LA_k	Acc.	-	52.17%	58.81%	50.41%	51.45%	**61.31%**	58.82%	83.17%
	F-meas.	-	52.5%	58.97%	49.77%	50.88%	52.24%	**57.22%**	83.16%
LA_o	Acc.	-	53.51%	62.76%	62.68%	63.19%	**72.93%**	66.51%	76.52%
	F-meas.	-	54.61%	63.16%	54.38%	63.43%	**66.65%**	59.12%	74.08%
LA_c	Acc.	-	75.67%	84.13%	85.21%	85.24%	**85.65%**	85.46%	86.89%
	F-meas.	-	75.29%	80.76%	79.71%	**80.3%**	79.48%	80.29%	82.67%
LA_{bed1}	Acc.	-	49.82	65.81%	65.16%	67.01%	**68.40%**	66.96%	77.97%
	F-meas.	-	51.22%	65.75%	58.37%	**64.7%**	60.48%	61.98%	76.04%
LA_{bed2}	Acc.	-	85.72%	91.99%	**92.57%**	92.52%	92.48%	92.6%	92.77%
	F-meas.	-	85.51%	91.18%	91.25%	**91.35%**	91.0%	91.3%	91.60%
LA_{bath1}	Acc.	-	78.18%	80.10%	77.78%	83.53%	81.78%	**84.43%**	86.36%
	F-meas.	-	76.79%	76.48%	75.69%	78.36%	78.56%	**79.36%**	83.03%
LA_{bath2}	Acc.	-	90.92%	92.34%	93.39%	**93.99%**	93.69%	**93.99%**	94.07%
	F-meas.	-	91.02%	91.71%	92.31%	**92.55%**	91.38%	**92.55%**	92.66%
LA_e	Acc.	-	57.4%	73.86%	73.17%	74.24%	**74.26%**	73.31%	76.97%
	F-meas.	-	57.25%	72.34%	68.52%	**72.35%**	70.37%	68.71%	74.98%
Global	Acc.	72.94%	65.84	75.16%	74.4%	75.5%	**77.48%**	76.64%	82.23%
	F-meas.	71.80%	65.89%	74.0%	70.8%	**73.36%**	72.71%	73.01%	80.28%

TABLE 5.11. Évolution du taux de reconnaissance des agents au cours du temps

Agents LA_i	1-fold	(1+2)-folds	(1+2+3)-folds	(1+2+3+4)-folds	(1+2+3+4+5)-folds	(1+2+3+4+5+6)-folds	(1+2+3+4+5+6+7)-folds	(1+2+3+4+5+6+7+8)-folds	(1+2+3+4+5+6+7+8+9)-folds	(1+2+3+4+5+6+7+8+9+10)-folds
LA_o	54.27%	47.0%	53.61%	59.06%	63.22%	67.47%	66.77%	67.0%	66.44%	66.27%
LA_{bath1}	61.62%	65.15%	70.71%	70.2%	71.31%	74.58%	77.78%	79.92%	80.36%	81.82%
LA_{bath2}	79.7%	84.77%	87.72%	89.0%	90.75%	91.73%	92.8%	93.28%	93.73%	94.04%
LA_{bed1}	82.03%	72.88%	71.3%	72.37%	69.3%	70.21%	67.7%	67.12%	67.41%	67.31%
LA_{bed2}	86.73%	87.8%	89.16%	90.47%	91.01%	91.64%	91.75%	92.21%	92.21%	93.4%
LA_c	78.08%	82.43%	84.96%	85.69%	84.8%	85.22%	84.48%	85.08%	85.27%	84.76%
LA_e	72.46%	70.69%	69.59%	70.23%	71.94%	72.99%	73.17%	73.58%	74.05%	73.5%
LA_d	61.81%	66.39%	67.74%	68.58%	68.74%	68.37%	66.8%	67.34%	67.38%	67.21%
LA_k	51.63%	57.33%	59.86%	58.88%	59.76%	60.0%	59.92%	59.57%	58.19%	58.67%
LA_l	74.03%	76.42%	78.03%	78.0%	77.4%	76.88%	76.28%	76.29%	75.89%	76.09%

5.8.2 Étude de l'évolution du taux de reconnaissance

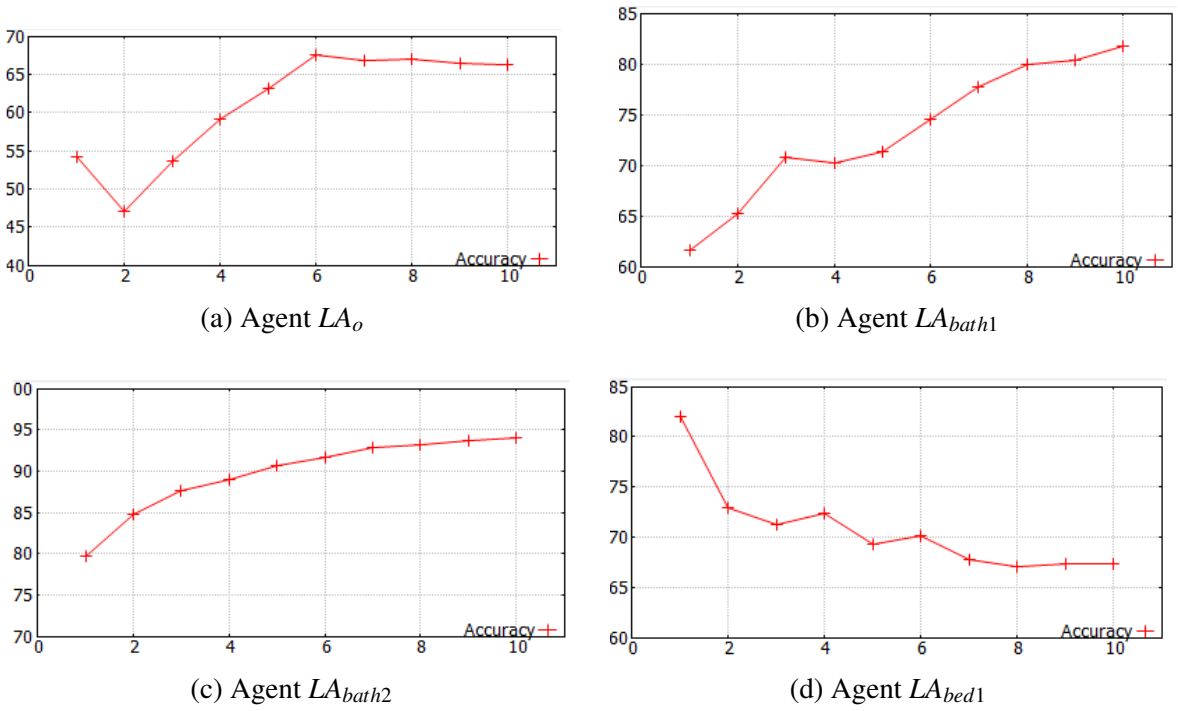
Cette sous section présente l'évolution du taux de reconnaissance au cours du temps. Cette étude reflète l'impact de l'évolution de la mesure de performance de chaque agent sur le taux de reconnaissance au cours du temps.

Afin de réaliser ceci, le jeu de données de chaque agent est divisé en 10 sous jeux de données. Nous commençons à mesurer le taux de reconnaissance du premier sous jeu de donnée dans l'ordre chronologique ascendant. Ensuite, nous rajoutons l'ensemble des vecteurs d'attributs du deuxième jeu de données dans le premier sous jeu de données en gardant l'ordre chronologique ascendant et nous mesurons le taux de reconnaissance. Puis, nous rajoutons l'ensemble des vecteurs d'attributs du troisième jeu de données dans le premier sous jeu de données en gardant toujours l'ordre chronologique ascendant et ainsi de suite.

Le tableau 5.11 montre l'évolution du taux de reconnaissance en fusionnant à chaque fois les *folds* un par un dans l'ordre chronologique ascendant.

Les figures 5.10 et 5.11 décrivent l'évolution du taux de reconnaissance en fusionnant à chaque fois les *folds* un par un dans l'ordre chronologique ascendant. Par exemple, l'axe des abscisses ayant la valeur 3 signifie la fusion des trois folds (1+2+3).

Synthèse : plus que les agents apprennent, s'adaptent et évoluent par rapport à leurs me-



(a) Agent LA_o

(b) Agent LA_{bath1}

(c) Agent LA_{bath2}

(d) Agent LA_{bed1}

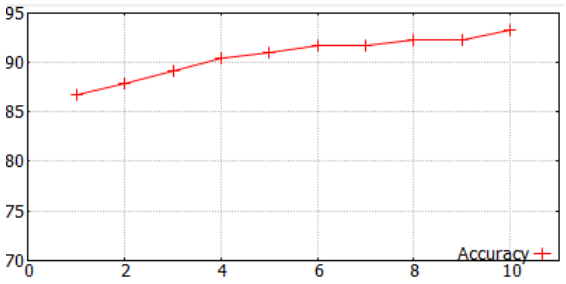
FIGURE 5.10. Évolution du taux de reconnaissance au cours du temps pour les agents LA_o, LA_{bath1}, LA_{bath2} et LA_{bed1}

sures de performance, plus qu'ils reconnaissent correctement les activités et donc le taux de reconnaissance s'améliore. Il y a seulement l'agent LA_{bed1} dont son taux de reconnaissance se dégrade avec le temps.

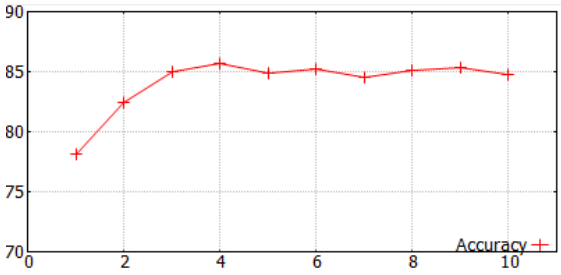
5.9 Conclusion

Dans ce chapitre, nous avons présenté l'étude expérimentale de l'approche *DCR* avec le jeu de données Aruba. Cette étude nous a permis de valider notre approche via les métriques en termes de taux de reconnaissance et F-mesure, de temps de traitement et de nombre de messages et de mettre en relief son apport par rapport à une approche centralisée et une approche existante dans la littérature.

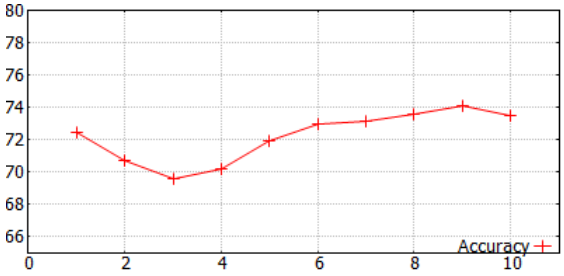
En outre, nous avons évalué la version améliorée de l'approche *DCR* qui est l'approche *DCR-OL*. Cette étude nous a permis de mettre en relief son apport par rapport à *DCR* en termes de temps de reconnaissance et F-mesure. En effet, la méthode *DCR-OL (max-wPerf)* se positionne comme la deuxième meilleure méthode de résolution de conflits après la méthode *DCR (stacking)* en terme de taux de reconnaissance et après la méthode *DCR (max-*



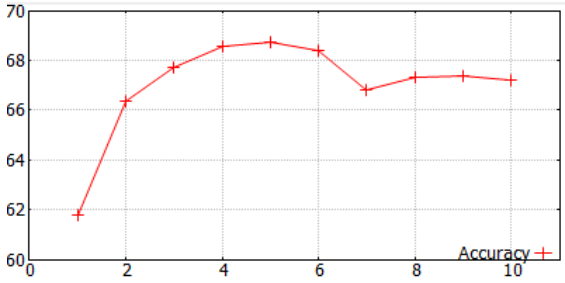
(a) Agent LA_{bed2}



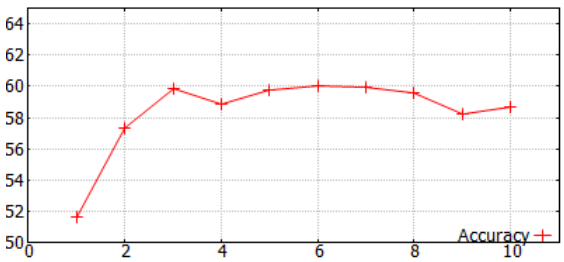
(b) Agent LA_c



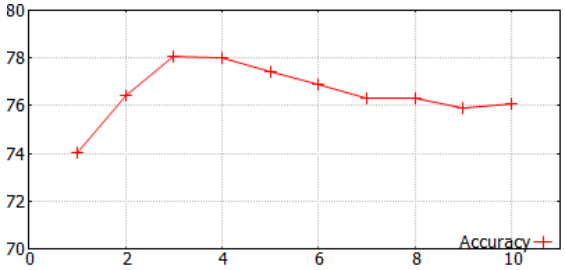
(c) Agent LA_e



(d) Agent LA_d



(e) Agent LA_k



(f) Agent LA_l

FIGURE 5.11. Évolution du taux de reconnaissance au cours du temps pour les agents LA_{bed2}, LA_c, LA_e, LA_d, LA_k et LA_l

freq.) en terme de taux de F-mesure.

Aussi, nous avons étudié le comportement des agents apprenants envers le taux de reconnaissance qui s'améliore avec le temps.

Conclusion générale et perspectives

Synthèse des contributions

L'informatique pervasive et l'intelligence ambiante constituent des domaines importants dans la recherche et le développement des prochaines générations des technologies de la communication et de l'information. Elles ont introduit beaucoup de défis de recherche dans le domaine de l'IA au cours de ces dernières années. Dans un environnement pervasif, les dispositifs intelligents sont censés se connecter d'une manière transparente et collaborer les uns avec les autres pour aider l'être humain, anticiper ses besoins et fournir l'information adéquate à la bonne personne, et ce quel que soient le lieu et le moment. Pour répondre à ces besoins, le système d'intelligence ambiante doit identifier tout d'abord la situation/l'activité en cours pour réagir et s'adapter au contexte courant. Cela passe initialement par la perception du contexte de l'utilisateur puis d'une analyse de ces informations contextuelles collectées pour l'identification de situations/d'activités.

Nous avons proposé une architecture générale hiérarchiquement distribuée pour la remontée des données brutes collectées. Cette architecture présente trois niveaux : la perception, l'observation et l'identification.

Dans cette thèse, nous avons contribué aux différents niveaux de l'architecture et en particulier :

— Niveau *Perception* : un nouveau modèle de perception **FSCEP (Fuzzy Semantic Complex Event Processing)** pour le traitement des évènements complexes sémantiques flous.
— Niveau *Observation (étape sélection des attributs)* : deux nouvelles méthodes de sélection des meilleurs attributs basées sur l'intégrale de Choquet : l'une prend en entrée des vecteurs d'attributs non flous qui est **FSCI (Feature Selection based on Choquet Integral)** ; et l'autre prend en entrée des vecteurs d'attributs flous **Fuzzy-FSCI**.
— Niveau *Identification* : une approche de raisonnement distribué principale basée sur le

TABLE 5.12. Les caractéristiques des différentes contributions

Niveau	Contributions	Type des dispositifs déployés	Données d'entrées	Données de sortie
Perception	FSCEP	Capteurs binaires (capteurs de présence, capteurs de mouvement, capteurs de pression, etc.) + caméras	Des signaux (changement d'état du capteur ou de la caméra ON/OFF)	Des évènements complexes sémantiques flous
Observation	FSCI	Jeu de données HAR (Accéléromètres et Gyroscopes : capteurs destinés aux objets portables)	Un ensemble de vecteurs d'attributs (les variables de domaine de temps et de fréquence)	Un ensemble de vecteurs d'attributs réduits
	Fuzzy-FSCI	étudié théoriquement	Un ensemble de vecteurs d'attributs flous	Un ensemble de vecteurs d'attributs flous réduits
Identification	DCR	Jeu de données Aruba (Capteurs de porte + capteurs de mouvements)	Un vecteur d'attributs	Une activité finale avec son degré d'incertitude
	DCR-OL	Jeu de données Aruba (Capteurs de porte + capteurs de mouvements)	Un vecteur d'attributs	Une activité finale avec son degré d'incertitude
	Fuzzy-DCR	étudié théoriquement	Un vecteur d'attributs flous	Les activités avec leurs degrés de confiance

modèle multi-agents pour la reconnaissance d'activité Humaine nommée l'approche **DCR (Distributed Cooperative Reasoning)**. Une version améliorée de *DCR* est proposée pour gérer l'apprentissage en ligne, c'est l'approche **DCR-OL**. Une étude théorique a été menée pour étendre l'approche *DCR* à des données d'entrée floues qui est l'approche **Fuzzy-DCR**.

Nous notons que les données d'entrées/de sorties de chaque contribution sont détaillées dans le Tableau 5.12.

Les perspectives pouvant faire suite à ces contributions sont multiples. Nous en citons quelques unes :

— Niveau *Perception* : le travail *FSCEP* n'a présenté qu'un premier pas et de multiples perspectives d'améliorations peuvent être envisagées. Tout d'abord, comme la performance et le passage à l'échelle sont deux problématiques sérieuses dans le domaine de la gestion d'évènements complexes (CEP), nous visons à les optimiser et à les mesurer avec notre *FSCEP* en utilisant des scénarios plus complexes. Deuxièmement, un défi est de rendre notre approche compatible avec les environnements intelligents ouverts où les capteurs peuvent être rajoutés ou supprimés aléatoirement n'importe quand et n'importe où dans l'environnement. Ainsi, nous envisageons d'améliorer notre approche *FSCEP* pour prendre en compte cette contrainte à savoir de gérer l'entrée/sortie des (nouveaux) capteurs sans avoir à re-paramétrer le système. Enfin, comme la confidentialité est un problème important pour les utilisateurs, une pers-

pective possible consisterait à ajouter des restrictions et des autorisations dans la définition des requêtes de l'expert afin de contrôler l'utilisation de notre évènement de sortie FSCE.

— Niveau *Observation* : Pour *FSCI*, nous envisageons de mener une évaluation plus approfondie avec d'autres jeux de données pour la reconnaissance des activités. De plus, nous avons effectué une identification de mesure floue avec seulement une technique qui est la technique des moindres carrés. Nous pouvons utiliser d'autres techniques d'identification de mesures floues. Enfin, l'approche *FSCI* version enveloppe (l'approche *FSCI-W*) fera aussi l'extension de ce travail. En effet, nous planifions d'étudier les techniques de mesure de l'importance des attributs pour déterminer l'ensemble des attributs souhaité afin d'améliorer le taux de reconnaissance par rapport à l'approche *FSCI-F*. Concernant la version floue *Fuzzy-FSCI*, sa mise en place et son évaluation restent un travail futur dans cette thèse.

— Niveau *Identification* : des perspectives sont envisagées pour les deux propositions :

1. L'approche *DCR* : elle présente encore des améliorations à poursuivre dans les travaux futurs que nous citons ici :

   (a) Pour généraliser notre approche *DCR*, nous envisageons de l'appliquer sur d'autres jeux de données pour HAR mais aussi sur des jeux de données d'un autre domaine tels que les séquences vidéo e.g, la reconnaissance de gestes.

   (b) Une autre direction de recherche future prometteuse est d'étendre la prise de décision en proposant une nouvelle stratégie pour la résolution de conflits pour aider l'agent initiateur à prendre la meilleure décision et donc d'augmenter le taux de reconnaissance et la F-mesure du système *DCR* et de se rapprocher de la borne maximale. La méthode *DCR (stacking)* est la meilleure en terme de taux de reconnaissance (77.48%) mais elle reste loin de la borne maximale (82.23%). De même, la méthode *DCR (max-freq.)* est la meilleure en terme de F-mesure (73.36%) mais elle reste loin de la borne maximale (80.28%).

   (c) Le nombre de messages échangés entre les agents et le temps de traitement sont importants. Ainsi, nous devons définir une stratégie de collaboration améliorant les performances du système *DCR*.

2. L'approche *DCR-OL* : tout d'abord, une première perspective est de faire évoluer les degrés de confiance dans la liste des activités ACT_{LA_i} au cours du temps.

D'autre part, nous pouvons utiliser les valeurs de performance des agents dans le système comme étant des facteurs de pondération pour les deux méthodes d'agrégation : méthode *max-trust* (le facteur de pondération de l'agent concerné sera appliqué au degré de confiance des ses activités reçues) et méthode *max-freq.* (le facteur de pondération de l'agent concerné sera appliqué à la fréquence des ses activités reçues).

3. L'approche *Fuzzy-DCR* : Une implémentation et une évaluation reste un travail futur pour cette thèse.

Bibliographie

[Abdelhedi et al., 2016] Abdelhedi, S., Wali, A., and Alimi, A. M. (2016). *Fuzzy Logic Based Human Activity Recognition in Video Surveillance Applications*, pages 227–235. Springer International Publishing, Cham.

[ABID, 2012] ABID, Z. (2012). *Management of the quality of context for ambient intelligence*. Theses, Institut National des Télécommunications, France.

[Adjiman et al., 2006] Adjiman, P., Chatalic, P., Goasdoué, F., Rousset, M.-C., and Simon, L. (2006). Distributed reasoning in a peer-to-peer setting : Application to the semantic web. *J. Artif. Int. Res.*, 25(1) :269–314.

[Alpaydin, 2010] Alpaydin, E. (2010). *Introduction to Machine Learning*. The MIT Press, 2nd edition.

[Alzahrani and Kammoun, 2016] Alzahrani, M. and Kammoun, S. (2016). *Human Activity Recognition : Challenges and Process Stages*.

[Amft and Lombriser, 2011] Amft, O. and Lombriser, C. (2011). Modelling of distributed activity recognition in the home environment. In *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1781–1784.

[Anguita et al., 2013] Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J. L. (2013). A public domain dataset for human activity recognition using smartphones. In *21th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2013*.

[Artikis et al., 2012] Artikis, A., Etzion, O., Feldman, Z., and Fournier, F. (2012). Event processing under uncertainty. In *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, pages 32–43. ACM.

[Atallah and Yang, 2009] Atallah, L. and Yang, G.-Z. (2009). The use of pervasive sensing for behaviour profiling - a survey. *Pervasive and Mobile Computing*, 5(5) :447 – 464.

[Attard et al., 2013] Attard, J., Scerri, S., Rivera, I., and Handschuh, S. (2013). Ontology-based situation recognition for context-aware systems. In *Proceedings of the 9th International Conference on Semantic Systems*, I-SEMANTICS '13, pages 113–120, New York, NY, USA. ACM.

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

[Azkune et al., 2015] Azkune, G., Almeida, A., López-de Ipiña, D., and Chen, L. (2015). Extending knowledge-driven activity models through data-driven learning techniques. *Expert Syst. Appl.*, 42(6) :3115–3128.

[Bao and Intille, 2004] Bao, L. and Intille, S. S. (2004). Activity recognition from user-annotated acceleration data. In Ferscha, A. and Mattern, F., editors, *Pervasive Computing*, pages 1–17, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Barber et al., 2000] Barber, K. S., Liu, T. H., and Han, D. C. (2000). *Strategic Decision-Making for Conflict Resolution in Dynamic Organized Multi-Agent Systems*.

[Bikakis and Antoniou, 2008] Bikakis, A. and Antoniou, G. (2008). Distributed reasoning with conflicts in an ambient peer-to-peer setting. In Mhlhuser, M., Ferscha, A., and Aitenbichler, E., editors, *Constructing Ambient Intelligence*, volume 11 of *Communications in Computer and Information Science*, pages 24–33. Springer Berlin Heidelberg.

[Bikakis and Antoniou, 2010] Bikakis, A. and Antoniou, G. (2010). Defeasible contextual reasoning with arguments in ambient intelligence. *Knowledge and Data Engineering, IEEE Transactions on*, 22(11) :1492–1506.

[Bikakis et al., 2008] Bikakis, A., Patkos, T., Antoniou, G., and Plexousakis, D. (2008). A survey of semantics-based approaches for context reasoning in ambient intelligence. In Mühlhäuser, M., Ferscha, A., and Aitenbichler, E., editors, *Constructing Ambient Intelligence*, pages 14–23, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Bonissone et al., 2010] Bonissone, P., Cadenas, J. M., Garrido, M. C., and Diaz-Valladares, R. A. (2010). A fuzzy random forest. *International Journal of Approximate Reasoning*, 51(7) :729 – 747.

[Bosc and Prade, 1997] Bosc, P. and Prade, H. (1997). *An Introduction to the Fuzzy Set and Possibility Theory-Based Treatment of Flexible Queries and Uncertain or Imprecise Databases*, pages 285–324. Springer US, Boston, MA.

[Bouquet et al., 2003] Bouquet, P., Giunchiglia, F., van Harmelen, F., Serafini, L., and Stuckenschmidt, H. (2003). C-owl : Contextualizing ontologies. In Fensel, D., Sycara, K., and Mylopoulos, J., editors, *The Semantic Web - ISWC 2003*, pages 164–179, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Brazdil and Clark, 1990] Brazdil, P. and Clark, P. (1990). Learning from imperfect data. In Brazdil, P. and Konolige, K., editors, *Machine Learning, Meta-Reasoning and Logics*, volume 82 of *The Kluwer International Series in Engineering and Computer Science*, pages 207–232. Springer US.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1) :5–32.

[Brenna et al., 2007] Brenna, L., Demers, A., Gehrke, J., Hong, M., Ossher, J., Panda, B., Riedewald, M., Thatte, M., and White, W. (2007). Cayuga : a high-performance event processing engine. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 1100–1102. ACM.

[Broda et al., 2009] Broda, K., Clark, K., Miller, R., and Russo, A. (2009). Sage : A logical agent-based environment monitoring and control system. In Tscheligi, M., de Ruyter, B., Markopoulus, P., Wichert, R., Mirlacher, T., Meschterjakov, A., and Reitberger, W., editors, *Ambient Intelligence*, pages 112–117, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Brézillon, 2002] Brézillon, P. (2002). Hors du contexte, point de salut. Number 26, Autrans, France.

[Bulling et al., 2014] Bulling, A., Blanke, U., and Schiele, B. (2014). A tutorial on human activity recognition using body-worn inertial sensors. *ACM Comput. Surv.*, 46(3) :33 :1–33 :33.

[Canzian et al., 2015] Canzian, L., Zhang, Y., and van der Schaar, M. (2015). Ensemble of distributed learners for online classification of dynamic data streams. *IEEE Transactions on Signal and Information Processing over Networks*, 1(3) :180–194.

[Capela et al., 2015] Capela, N. A., Lemaire, E. D., and Baddour, N. (2015). Feature selection for wearable smartphone-based human activity recognition with able bodied, elderly, and stroke patients. *PLOS ONE*, 10(4) :1–18.

[Cateni et al., 2013] Cateni, S., Vannucci, M., Vannocci, M., and Colla, V. (2013). Variable selection and feature extraction through artificial intelligence techniques. In Freitas, L. V. d. and Freitas, A. P. B. R. d., editors, *Multivariate Analysis in Management, Engineering and the Sciences*, chapter 06. InTech, Rijeka.

[Chandrashekar and Sahin, 2014] Chandrashekar, G. and Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*.

[Chen et al., 2012] Chen, L., Nugent, C. D., and Wang, H. (2012). A knowledge-driven approach to activity recognition in smart homes. *IEEE Transactions on Knowledge and Data Engineering*, 24(6) :961–974.

[Choudhury et al., 2008] Choudhury, T., Borriello, G., Consolvo, S., Haehnel, D., Harrison, B., Hemingway, B., Hightower, J., Klasnja, P. ., Koscher, K., LaMarca, A., Landay, J. A., LeGrand, L., Lester, J., Rahimi, A., Rea, A., and Wyatt, D. (2008). The mobile sensing platform : An embedded activity recognition system. *IEEE Pervasive Computing*, 7(2) :32–41.

[Chu et al., 2012] Chu, C., Hsu, A.-L., Chou, K.-H., Bandettini, P., and Lin, C. (2012). Does feature selection improve classification accuracy ? impact of sample size and feature selection on classification using anatomical magnetic resonance images. *NeuroImage*, 60(1) :59 – 70.

[Cicirelli et al., 2016] Cicirelli, F., Fortino, G., Giordano, A., Guerrieri, A., Spezzano, G., and Vinci, A. (2016). On the design of smart homes : A framework for activity recognition in home environment. *Journal of Medical Systems*, 40(9) :200.

[Cook and Das, 2007] Cook, D. J. and Das, S. K. (2007). How smart are our environments ? an updated look at the state of the art. *Pervasive and Mobile Computing*, 3(2) :53 – 73. Design and Use of Smart Environments.

[Cook and Schmitter-Edgecombe, 2009] Cook, D. J. and Schmitter-Edgecombe, M. (2009). Assessing the quality of activities in a smart environment. *Methods of information in medicine*, 48(5) :480.

[Costa et al., 2006] Costa, P. D., Guizzardi, G., Almeida, J. P. A., Pires, L. F., and Sinderen, M. V. (2006). Situations in conceptual modeling of context. In *2006 10th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW'06)*, pages 6–6.

[Coutaz and Crowley, 2008] Coutaz, J. and Crowley, J. (2008). Plan "intelligence ambiante" : Défis et opportunités. Document de réflexion conjoint du comité d'experts " Informatique Ambiante " du département ST2I du CNRS et du Groupe de Travail " Intelligence Ambiante " du Groupe de Concertation Sectoriel (GCS3) du Ministère de l'Enseignement Supérieur et de la Recherche, DGRI A3.

[Cugola et al., 2015] Cugola, G., Margara, A., Matteucci, M., and Tamburrelli, G. (2015). Introducing uncertainty in complex event processing : Model, implementation, and validation. *Computing*, 97(2) :103–144.

[Daniello et al., 2015] Daniello, G., Loia, V., and Orciuoli, F. (2015). A multi-agent fuzzy consensus model in a situation awareness framework. *Applied Soft Computing*, 30 :430 – 440.

[Díaz Rodríguez et al., 2014] Díaz Rodríguez, N., León Cadahía, O., Cuéllar, M. P., Lilius, J., and Delgado-Calvo-Flores, M. (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*, 14(10) :18131–18171.

[Delir Haghighi et al., 2008] Delir Haghighi, P., Krishnaswamy, S., Zaslavsky, A., and Gaber, M. M. (2008). Reasoning about context in uncertain pervasive computing environments. In Roggen, D., Lombriser, C., Tröster, G., Kortuem, G., and Havinga, P., editors, *Smart Sensing and Context*, pages 112–125, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Demers et al., 2006] Demers, A., Gehrke, J., Hong, M., Riedewald, M., and White, W. (2006). Towards expressive publish/subscribe systems. In Ioannidis, Y., Scholl, M. H., Schmidt, J. W., Matthes, F., Hatzopoulos, M., Boehm, K., Kemper, A., Grust, T., and Boehm, C., editors, *Advances in Database Technology - EDBT 2006*, pages 627–644, Berlin, Heidelberg. Springer Berlin Heidelberg.

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

[Dietterich, 2000] Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, MCS '00, pages 1–15, London, UK, UK. Springer-Verlag.

[Ding et al., 2011] Ding, D., Cooper, R. A., Pasquina, P. F., and Fici-Pasquina, L. (2011). Sensor technology for smart homes. *Maturitas*, 69(2) :131 – 136.

[El Fallah Seghrouchni et al., 2012] El Fallah Seghrouchni, A., Olaru, A., Nguyen, N. T. T., and Salomone, D. (2012). Ao dai : Agent oriented design for ambient intelligence. In Desai, N., Liu, A., and Winikoff, M., editors, *Principles and Practice of Multi-Agent Systems*, pages 259–269, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Endsley, 2011] Endsley, M. R. (2011). *Designing for Situation Awareness : An Approach to User-Centered Design, Second Edition*. CRC Press, Inc., Boca Raton, FL, USA, 2nd edition.

[Endsley, 1995] Endsley, M. R. (March 1995). Toward a theory of situation awareness in dynamic systems. *Human Factors : The Journal of the Human Factors and Ergonomics Society*, 37 :32–64(33).

[event stream intelligence, 2010] event stream intelligence, E. (2010). Esper–complex event processing.

[F. et al., 2010] F., B., D., C., D., M., NARDI, D., and P., P. S. (2010). The description logic handbook, theory, implementation, and applications (2nd edition). In *The Description Logic Handbook*, pages 555–555. Cambridge University Press, CAMBRIDGE.

[Fahad and Rajarajan, 2015] Fahad, L. G. and Rajarajan, M. (2015). Integration of discriminative and generative models for activity recognition in smart homes. *Applied Soft Computing*, 37 :992 – 1001.

[Fong et al., 2003] Fong, T., Nourbakhsh, I., and Dautenhahn, K. (2003). A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42 :143–166.

[Fortino et al., 2015] Fortino, G., Giordano, A., Guerrieri, A., Spezzano, G., and Vinci, A. (2015). *A Data Analytics Schema for Activity Recognition in Smart Home Environments*, pages 91–102. Springer International Publishing, Cham.

[Geurts et al., 2006] Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63(1) :3–42.

[Grabisch et al., 2008] Grabisch, M., Kojadinovic, I., and Meyer, P. (2008). A review of methods for capacity identification in choquet integral based multi-attribute utility theory. *European Journal of Operational Research*, 186(2) :766–785.

[Grabisch and Labreuche, 2010] Grabisch, M. and Labreuche, C. (2010). A decade of application of the choquet and sugeno integrals in multi-criteria decision aid. *Annals of Operations Research*, 175(1) :247–286.

[Gu et al., 2010] Gu, T., Chen, S., Tao, X., and Lu, J. (2010). An unsupervised approach to activity recognition and segmentation based on object-use fingerprints. *Data & Knowledge Engineering*, 69(6) :533 – 544.

[Gu et al., 2005] Gu, T., Pung, H. K., and Zhang, D. Q. (2005). A service-riented middleware for building context-aware services. *Journal of Network and Computer Applications*, 28(1) :1 – 18.

[Gu et al., 2004] Gu, T., Wang, X. H., Pung, H. K., and Zhang, D. Q. (2004). An ontology-based context model in intelligent environments. In *IN PROCEEDINGS OF COMMUNICATION NETWORKS AND DISTRIBUTED SYSTEMS MODELING AND SIMULATION CONFERENCE*, pages 270–275.

[Gupta and Dallas, 2014] Gupta, P. and Dallas, T. (2014). Feature selection and activity recognition system using a single triaxial accelerometer. *IEEE Transactions on Biomedical Engineering*, 61(6) :1780–1786.

[Henricksen and Indulska, 2004] Henricksen, K. and Indulska, J. (2004). Modelling and using imperfect context information. In *IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second*, pages 33–37.

[Henricksen and Indulska, 2006] Henricksen, K. and Indulska, J. (2006). Developing context-aware pervasive computing applications models and approach. *Pervasive Mob. Comput.*, 2(1) :37–64.

[Hoque et al., 2014] Hoque, N., Bhattacharyya, D., and Kalita, J. (2014). Mifs-nd : A mutual information-based feature selection method. *Expert Systems with Applications*, 41(14) :6371 – 6385.

[Huynh et al., 2007] Huynh, T., Blanke, U., and Schiele, B. (2007). Scalable recognition of daily activities with wearable sensors. In Hightower, J., Schiele, B., and Strang, T., editors, *Location- and Context-Awareness*, pages 50–67, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Jarraya et al., 2017] Jarraya, A., Arour, K., Bouzeghoub, A., and Borgi, A. (2017). Feature selection based on choquet integral for human activity recognition. In *2017 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2017, Naples, Italy, July 9-12, 2017*, pages 1–6.

[Jarraya et al., 2018] Jarraya, A., Bouzeghoub, A., Borgi, A., and Arour, K. (2018). Distributed collaborative reasoning for HAR in smart homes. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pages 1971–1973.

[Jarraya et al., 2016a] Jarraya, A., Ramoly, N., Bouzeghoub, A., Arour, K., Borgi, A., and Finance, B. (2016a). FSCEP : A new model for context perception in smart homes. In *On the Move to Meaningful Internet Systems : OTM 2016 Conferences - Confederated International Conferences : CoopIS, C&TC, and ODBASE 2016, Rhodes, Greece, October 24-28, 2016, Proceedings*, pages 465–484.

[Jarraya et al., 2016b] Jarraya, A., Ramoly, N., Bouzeghoub, A., Arour, K., Borgi, A., and Finance, B. (2016b). A fuzzy semantic CEP model for situation identification in smart homes. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 1678–1679.

[Jin et al., 2014] Jin, C., Li, F., and Li, Y. (2014). A generalized fuzzy id3 algorithm using generalized information entropy. *Knowledge-Based Systems*, 64 :13–21.

[Jovic et al., 2015] Jovic, A., Brkic, K., and Bogunovic, N. (2015). A review of feature selection methods with applications. In *38th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2015, Opatija, Croatia, May 25-29, 2015*, pages 1200–1205.

[Kanda et al., 2008] Kanda, T., Glas, D. F., Shiomi, M., Ishiguro, H., and Hagita, N. (2008). Who will be the customer ? : A social robot that anticipates people's behavior from their trajectories. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, pages 380–389, New York, NY, USA. ACM.

[Kim et al., 2010] Kim, E., Helal, S., and Cook, D. (2010). Human activity recognition and pattern discovery. *IEEE Pervasive Computing*, 9(1) :48–53.

[Kramp et al., 2013] Kramp, T., van Kranenburg, R., and Lange, S. (2013). *Introduction to the Internet of Things*, pages 1–10. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Krishnan and Cook, 2014] Krishnan, N. C. and Cook, D. J. (2014). Activity recognition on streaming sensor data. *Pervasive and Mobile Computing*, 10(Part B) :138 – 154.

[Kuncheva, 2004] Kuncheva, L. I. (2004). *Combining Pattern Classifiers : Methods and Algorithms*. Wiley-Interscience.

[Lam et al., 2012] Lam, H.-P., Governatori, G., Satoh, K., and Hosobe, H. (2012). Distributed defeasible speculative reasoning in ambient environment. In *Computational Logic in Multi-Agent Systems*, volume 7486, pages 43–60. Springer Berlin Heidelberg.

[Lee and Jung, 2017] Lee, O.-J. and Jung, J. E. (2017). Sequence clustering-based automated rule generation for adaptive complex event processing. *Future Generation Computer Systems*, 66 :100 – 109.

[Liu et al., 2017] Liu, L., Wang, S., Su, G., Huang, Z.-G., and Liu, M. (2017). Towards complex activity recognition using a bayesian network-based probabilistic generative framework. *Pattern Recognition*, 68(Supplement C) :295 – 309.

[Loke, 2010] Loke, S. W. (2010). Incremental awareness and compositionality : A design philosophy for context-aware pervasive systems. *Pervasive and Mobile Computing*, 6(2) :239 – 253. Context Modelling, Reasoning and Management.

[Luckham, 2008] Luckham, D. (2008). The power of events : An introduction to complex event processing in distributed enterprise systems. In Bassiliades, N., Governatori, G., and Paschke, A., editors, *Rule Representation, Interchange and Reasoning on the Web*, pages 3–3, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Maciel et al., 2015] Maciel, C., de Souza, P. C., Viterbo, J., Mendes, F. F., and El Fallah Seghrouchni, A. (2015). A multi-agent architecture to support ubiquitous applications in smart environments. In *Agent Technology for Intelligent Mobile Services and Smart Societies*, pages 106–116, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Mahmood et al., 2013] Mahmood, A., Shi, K., Khatoon, S., and Xiao, M. (2013). Data mining techniques for wireless sensor networks : A survey. *International Journal of Distributed Sensor Networks*, 9(7) :406316.

[Marin-Perianu et al., 2008] Marin-Perianu, M., Lombriser, C., Amft, O., Havinga, P., and Tröster, G. (2008). *Distributed Activity Recognition with Fuzzy-Enabled Wireless Sensor Networks*, pages 296–313. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Mashayekhi and Gras, 2015] Mashayekhi, M. and Gras, R. (2015). Rule extraction from random forest : the rf+hc methods. In Barbosa, D. and Milios, E., editors, *Advances in Artificial Intelligence*, pages 223–237, Cham. Springer International Publishing.

[McKeever et al., 2010] McKeever, S., Ye, J., Coyle, L., Bleakley, C. J., and Dobson, S. (2010). Activity recognition using temporal evidence theory. *JAISE*, 2(3) :253–269.

[Meditskos et al., 2015] Meditskos, G., Kontopoulos, E., and Kompatsiaris, I. (2015). Redef : Context-aware recognition of interleaved activities using owl 2 and defeasible reasoning. In Kyzirakos, K., Henson, C. A., Perry, M., Varanka, D., Grütter, R., Calbimonte, J.-P., Celino, I., Valle, E. D., Dell'Aglio, D., Krötzsch, M., and Schlobach, S., editors, *SSN-TC/OrdRing@ISWC*, volume 1488 of *CEUR Workshop Proceedings*, pages 31–42. CEUR-WS.org.

[Mohd Noor et al., 2016] Mohd Noor, M. H., Salcic, Z., and Wang, K. (2016). Enhancing ontological reasoning with uncertainty handling for activity recognition. 114 :–.

[Mosabbeb et al., 2013] Mosabbeb, E. A., Raahemifar, K., and Fathy, M. (2013). Multi-view support vector machines for distributed activity recognition. In *2013 Seventh International Conference on Distributed Smart Cameras (ICDSC)*, pages 1–2.

[Motro, 1995] Motro, A. (1995). Imprecision and uncertainty in database systems. In Bosc, P. and Kacprzyk, J., editors, *Fuzziness in Database Management Systems*, volume 5 of *Studies in Fuzziness*, pages 3–22. Physica-Verlag HD.

[Motro, 1997] Motro, A. (1997). Sources of uncertainty, imprecision, and inconsistency in information systems. In Motro, A. and Smets, P., editors, *Uncertainty Management in Information Systems*, pages 9–34. Springer US.

[Mühlenbrock et al., 2004] Mühlenbrock, M., Brdiczka, O., Snowdon, D., and Meunier, J.-L. (2004). Learning to detect user activity and availability from a variety of sensor data. In *Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications (PerCom'04)*, PERCOM '04, pages 13–, Washington, DC, USA. IEEE Computer Society.

[Noorit and Suvonvorn, 2014] Noorit, N. and Suvonvorn, N. (2014). *Human Activity Recognition from Basic Actions Using Finite State Machine*, pages 379–386. Springer Singapore, Singapore.

[Oh, 2003] Oh, S.-B. (2003). On the relationship between majority vote accuracy and dependency in multiple classifier systems. *Pattern Recognition Letters*, 24(1) :359 – 363.

[Okeyo et al., 2014] Okeyo, G., Chen, L., and Wang, H. (2014). Combining ontological and temporal formalisms for composite activity modelling and recognition in smart homes. *Future Generation Computer Systems*, 39 :29 – 43. Special Issue on Ubiquitous Computing and Future Communication Systems.

[Ortiz Laguna et al., 2011] Ortiz Laguna, J., Olaya, A. G., and Borrajo, D. (2011). A dynamic sliding window approach for activity recognition. In Konstan, J. A., Conejo, R., Marzo, J. L., and Oliver, N., editors, *User Modeling, Adaption and Personalization*, pages 219–230, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Patterson et al., 2003] Patterson, D. J., Liao, L., Fox, D., and Kautz, H. (2003). Inferring high-level behavior from low-level sensors. In Dey, A. K., Schmidt, A., and McCarthy, J. F., editors, *UbiComp 2003 : Ubiquitous Computing*, pages 73–89, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Philipose et al., 2004] Philipose, M., Fishkin, K. P., Perkowitz, M., Patterson, D. J., Fox, D., Kautz, H., and Hahnel, D. (2004). Inferring activities from interactions with objects. *IEEE Pervasive Computing*, 3(4) :50–57.

[Quinlan, 1987] Quinlan, J. R. (1987). Generating production rules from decision trees. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'87, pages 304–307, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Ramakrishnan et al., 2013a] Ramakrishnan, A. K., Preuveneers, D., and Berbers, Y. (2013a). A loosely coupled and distributed bayesian framework for multi-context recognition in dynamic ubiquitous environments. In *2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing, UIC/ATC 2013, Vietri sul Mare, Sorrento Peninsula, Italy, December 18-21, 2013*, pages 270–277.

[Ramakrishnan et al., 2013b] Ramakrishnan, A. K., Preuveneers, D., and Berbers, Y. (2013b). A modular and distributed bayesian framework for activity recognition in dynamic smart environments. In *Ambient Intelligence - 4th International Joint Conference, AmI 2013, Dublin, Ireland, December 3-5, 2013. Proceedings*, pages 293–298.

[Ramakrishnan et al., 2014] Ramakrishnan, A. K., Preuveneers, D., and Berbers, Y. (2014). A bayesian framework for life-long learning in context-aware mobile applications. In *Context in Computing - A Cross-Disciplinary Approach for Modeling the Real World*, pages 127–141.

[Ranasinghe et al., 2016] Ranasinghe, S., Machot, F. A., and Mayr, H. C. (2016). A review on applications of activity recognition systems with regard to performance and evaluation. *International Journal of Distributed Sensor Networks*, 12(8) :1550147716665520.

[Rashidi and Mihailidis, 2013] Rashidi, P. and Mihailidis, A. (2013). A survey on ambient-assisted living tools for older adults. *IEEE Journal of Biomedical and Health Informatics*, 17(3) :579–590.

[Riboni et al., 2016] Riboni, D., Sztyler, T., Civitarese, G., and Stuckenschmidt, H. (2016). Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, pages 1–12, New York, NY, USA. ACM.

[Robnik-Šikonja and Kononenko, 2003] Robnik-Šikonja, M. and Kononenko, I. (2003). Theoretical and empirical analysis of relieff and rrelieff. *Machine Learning*, 53(1) :23–69.

[Rodriguez et al., 2014] Rodriguez, N. D., Cuellar, M. P., Lilius, J., and Calvo-Flores, M. D. (2014). A fuzzy ontology for semantic modelling and recognition of human behaviour. *Knowledge-Based Systems*, 66(Supplement C) :46 – 60.

[Roman et al., 2002] Roman, M., Hess, C., Cerqueira, R., Ranganathan, A., Campbell, R. H., and Nahrstedt, K. (2002). A middleware infrastructure for active spaces. *IEEE Pervasive Computing*, 1(4) :74–83.

[Ronao and Cho, 2014] Ronao, C. A. and Cho, S. B. (2014). Human activity recognition using smartphone sensors with two-stage continuous hidden markov models. In *2014 10th International Conference on Natural Computation (ICNC)*, pages 681–686.

[Russell, 1996] Russell, S. (1996). Chapter 4 - machine learning. In Boden, M. A., editor, *Artificial Intelligence*, Handbook of Perception and Cognition, pages 89 – 133. Academic Press, San Diego.

[Schaul and Schmidhuber, 2010] Schaul, T. and Schmidhuber, J. (2010). Metalearning. *Scholarpedia*, 5(6) :4650. revision #91489.

[Schilit and Theimer, 1994] Schilit, B. and Theimer, M. (1994). Disseminating active map information to mobile hosts. *Network, IEEE*, 8(5) :22–32.

[Schultz-Møller et al., 2009] Schultz-Møller, N. P., Migliavacca, M., and Pietzuch, P. (2009). Distributed complex event processing with query rewriting. In *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, DEBS '09, pages 4 :1–4 :12, New York, NY, USA. ACM.

[Sebbak et al., 2014] Sebbak, F., Benhammadi, F., Chibani, A., Amirat, Y., and Mokhtari, A. (2014). Dempster–shafer theory-based human activity recognition in smart home environments. *annals of telecommunications - annales des télécommunications*, 69(3) :171–184.

[Seghrouchni et al., 2008] Seghrouchni, A. E. F., Breitman, K., Sabouret, N., Endler, M., Charif, Y., and Briot, J. (2008). Ambient intelligence applications : Introducing the campus framework. In *13th IEEE International Conference on Engineering of Complex Computer Systems (iceccs 2008)*, pages 165–174.

[Serafini and Tamilin, 2005] Serafini, L. and Tamilin, A. (2005). Drago : Distributed reasoning architecture for the semantic web. In Gómez-Pérez, A. and Euzenat, J., editors, *The Semantic Web : Research and Applications*, pages 361–376, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Shapley, 1953] Shapley, L. S. (1953). A value for n-person games. In Kuhn, H. W. and Tucker, A. W., editors, *Contributions to the Theory of Games II*, pages 307–317. Princeton University Press, Princeton.

[Shoaib et al., 2013] Shoaib, M., Scholten, H., and Havinga, P. J. M. (2013). Towards physical activity recognition using smartphone sensors. In *2013 IEEE 10th International*

Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing, pages 80–87.

[Smets, 1997] Smets, P. (1997). *Imperfect Information : Imprecision and Uncertainty*, pages 225–254. Springer US, Boston, MA.

[Śnieżyński, 2006] Śnieżyński, B. (2006). Converting a naive bayes model into a set of rules. In Kłopotek, M. A., Wierzchoń, S. T., and Trojanowski, K., editors, *Intelligent Information Processing and Web Mining*, pages 221–229, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Snoussi et al., 2012] Snoussi, M., Gensel, J., and Davoine, P.-A. (2012). Extending timeml and spatialml languages to handle imperfect spatio-temporal information in the context of natural hazards studies.

[Stevenson et al., 2009] Stevenson, G., Knox, S., Dobson, S., and Nixon, P. (2009). Ontonym : A collection of upper ontologies for developing pervasive systems. In *Proceedings of the 1st Workshop on Context, Information and Ontologies*, CIAO '09, pages 9 :1–9 :8, New York, NY, USA. ACM.

[Strobl et al., 2007] Strobl, C., Boulesteix, A.-L., Zeileis, A., and Hothorn, T. (2007). Bias in random forest variable importance measures : Illustrations, sources and a solution. bmc bioinformatics. 8 :25.

[Su et al., 2014] Su, X., Tong, H., and Ji, P. (2014). Activity recognition with smartphone sensors. *Tsinghua Science and Technology*, 19(3) :235–249.

[Tapia et al., 2006] Tapia, E. M., Choudhury, T., and Philipose, M. (2006). Building reliable activity models using hierarchical shrinkage and mined ontology. In Fishkin, K. P., Schiele, B., Nixon, P., and Quigley, A., editors, *Pervasive Computing*, pages 17–32, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Tessier et al., 2002] Tessier, C., Müller, H.-J., Fiorino, H., and Chaudron, L. (2002). *Agents' Conflicts : New Issues*, pages 1–30. Springer US, Boston, MA.

[Ting, 2010] Ting, K. M. (2010). *Precision and Recall*, pages 781–781. Springer US, Boston, MA.

[Ting and Witten, 1999] Ting, K. M. and Witten, I. H. (1999). Issues in stacked generalization. *J. Artif. Int. Res.*, 10(1) :271–289.

[van Kasteren et al., 2008] van Kasteren, T., Noulas, A., Englebienne, G., and Kröse, B. (2008). Accurate activity recognition in a home setting. In *Proceedings of the 10th In-*

ternational Conference on Ubiquitous Computing, UbiComp '08, pages 1–9, New York, NY, USA. ACM.

[Van Laerhoven and Aidoo, 2001] Van Laerhoven, K. and Aidoo, K. (2001). Teaching context to applications. *Personal and Ubiquitous Computing*, 5(1) :46–49.

[van Rijn et al., 2018] van Rijn, J. N., Holmes, G., Pfahringer, B., and Vanschoren, J. (2018). The online performance estimation framework : heterogeneous ensemble learning for data streams. *Machine Learning*, 107(1) :149–176.

[Viterbo and Endler, 2009] Viterbo, J. and Endler, M. (2009). Decentralized reasoning in ambient intelligence. In *2009 33rd Annual IEEE Software Engineering Workshop*, pages 115–124.

[Wang et al., 2006] Wang, F., Liu, S., Liu, P., and Bai, Y. (2006). Bridging physical and virtual worlds : complex event processing for rfid data streams. In *Advances in Database Technology-EDBT 2006*, pages 588–607. Springer.

[Wang and Ji, 2012] Wang, X. and Ji, Q. (2012). Learning dynamic bayesian network discriminatively for human activity recognition. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 3553–3556.

[Wang and Ji, 2014] Wang, X. and Ji, Q. (2014). Context augmented dynamic bayesian networks for event recognition. *Pattern Recognition Letters*, 43(Supplement C) :62 – 70. ICPR2012 Awarded Papers.

[Wang et al., 2005] Wang, Z., Leung, K.-S., and Klir, G. J. (2005). Applying fuzzy measures and nonlinear integrals in data mining. *Fuzzy Sets and Systems*, 156(3) :371 – 380.

[Wasserkrug et al., 2012] Wasserkrug, S., Gal, A., and Etzion, O. (2012). A model for reasoning with uncertain rules in event composition systems. *arXiv preprint arXiv :1207.1427*.

[Wei, 2001] Wei, X. (2001). Bayesian networks and intelligent agent models. *semantic-scholar*.

[Weiser, 1991] Weiser, M. (1991). The computer for the 21st century. *Scientific American*, 265(3) :66–75.

[Wen and Zhong, 2015] Wen, J. and Zhong, M. (2015). Activity discovering and modelling with labelled and unlabelled data in smart environments. *Expert Systems with Applications*, 42(14) :5800 – 5810.

[Wen et al., 2015] Wen, J., Zhong, M., and Wang, Z. (2015). Activity recognition with weighted frequent patterns mining in smart environments. *Expert Systems with Applications*, 42(17) :6423 – 6432.

[Witten et al., 2011a] Witten, I. H., Frank, E., and Hall, M. A. (2011a). *Data Mining : Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition.

[Witten et al., 2011b] Witten, I. H., Frank, E., and Hall, M. A. (2011b). *Data Mining : Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition.

[Wolpert, 1992] Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2) :241 – 259.

[Wooldridge, 2009] Wooldridge, M. (2009). *An Introduction to MultiAgent Systems*. Wiley Publishing, 2nd edition.

[Wu et al., 2006] Wu, E., Diao, Y., and Rizvi, S. (2006). High-performance complex event processing over streams. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 407–418. ACM.

[Yala et al., 2017] Yala, N., Fergani, B., and Fleury, A. (2017). Towards improving feature extraction and classification for activity recognition on streaming data. *Journal of Ambient Intelligence and Humanized Computing*, 8(2) :177–189.

[Ye et al., 2007] Ye, J., Coyle, L., Dobson, S., and Nixon, P. (2007). Ontology-based models in pervasive computing systems. *Knowl. Eng. Rev.*, 22(4) :315–347.

[Ye et al., 2012] Ye, J., Dobson, S., and McKeever, S. (2012). Situation identification techniques in pervasive computing : A review. *Pervasive and Mobile Computing*, 8(1) :36 – 66.

[Ye et al., 2015] Ye, J., Stevenson, G., and Dobson, S. (2015). Kcar : A knowledge-driven approach for concurrent activity recognition. *Pervasive and Mobile Computing*, 19 :47 – 70.

[Yilmaz et al., 2006] Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking : A survey. *ACM Comput. Surv.*, 38(4).

[Yu and Liu, 2003] Yu, L. and Liu, H. (2003). Feature selection for high-dimensional data : A fast correlation-based filter solution. pages 856–863.

[Zadeh, 1965] Zadeh, L. (1965). Fuzzy sets. *Information and Control*, 8(3) :338 – 353.

[Zadeh, 1988] Zadeh, L. A. (1988). Fuzzy logic. *Computer*, 21(4) :83–93.

[Zhang et al., 2014] Zhang, H., Diao, Y., and Immerman, N. (2014). On complexity and optimization of expensive queries in complex event processing. In *Proceedings of the*

2014 ACM SIGMOD international conference on Management of data, pages 217–228. ACM.

[Zhang and Sawchuk, 2012] Zhang, M. and Sawchuk, A. (2012). Usc-had : A daily activity dataset for ubiquitous activity recognition using wearable sensors. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 1036–1043.

[Zhang and Sawchuk, 2011] Zhang, M. and Sawchuk, A. A. (2011). A feature selection-based framework for human activity recognition using wearable multimodal sensors. In *Proceedings of the 6th International Conference on Body Area Networks*, BodyNets '11, pages 92–98.

[Zhu and Xu, 2011] Zhu, L. and Xu, R. (2011). Ranking fuzzy numbers based on fuzzy mean and standard deviation. In *2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, volume 2, pages 854–857.

[Zimmermann, 2008] Zimmermann, A. (2008). *Semantics of Networked Knowledge : Managing Heterogeneity by Mediation*. Theses, Université Joseph-Fourier - Grenoble I.

[Zliobaite, 2010] Zliobaite, I. (2010). Learning under concept drift : an overview. *CoRR*, abs/1010.4784.

Annexe A

Annexe : Analyse des critères d'évaluation

Dans de nombreux problèmes d'apprentissage automatique, la performance des algorithmes est évaluée à l'aide des indicateurs suivants :

A.1 Matrice de confusion

Prenons l'exemple d'un classifieur binaire, c'est-à-dire, qui prédit 2 classes notées classe 0 et classe 1.

Pour mesurer les performances de ce classifieur, il est d'usage de distinguer 4 types d'éléments classés pour la classe voulue :

1. Vrai positif VP. Elément de la classe 1 correctement prédit
2. Vrai négatif VN. Elément de la classe 0 correctement prédit
3. Faux positif FP. Elément de la classe 1 mal prédit
4. Faux négatif FN. Elément de la classe 0 mal prédit

Ces informations peuvent être rassemblées et visualisées sous forme de tableau dans une matrice de confusion. La figure illustre le cas d'un classifieur binaire.

En particulier, lorsque la matrice de confusion est diagonale, le classifieur est parfait.

Notons que la matrice de confusion est aussi généralisable lorsqu'il y a k > 2 classes à prédire.

		Classe prédite	
		Classe 0	Classe 1
Classe réelle	Classe 0	VN	FN
	Classe 1	FP	VP

FIGURE A.1. Matrice de confusion d'un classifieur binaire

A.2 Indicateurs de base

Il est possible de calculer plusieurs indicateurs résumant la matrice de confusion. Par exemple si nous souhaitons rendre compte de la qualité de la prédiction sur la classe 1, on définit :

— **Précision** : Proportion d'éléments bien classés pour une classe donnée :

$$Precision_{delaclasse1} = VP/(VP+FP)$$

— **Rappel** : Proportion d'éléments bien classés par rapport au nombre d'éléments de la classe à prédire :

$$Rappel_{delaclasse1} = VP/(VP+FN)$$

— **Taux de reconnaissance "Accuracy"** : Proportion d'éléments bien classés par rapport à la totalité des instances :

$$Accuracy_{delaclasse1} = (VP+VN)/(VP+VN+FP+FN)$$

— **F-mesure** : Mesure de compromis entre précision et rappel :

$$F-mesure_{delaclasse1} = 2 \times (Precision \times Rappel)/(Precision+Rappel)$$

Annexe B

Annexe : Evaluation détaillée des agents de l'approche DCR et de l'approche DCR-OL

Dans cet annexe, nous présentons les différents tableaux d'évaluations pour les sous jeux de données Aruba assignés aux agents dans les systèmes DCR et DCR-OL. Les Tableaux B.1, B.2, B.3, B.4, B.5, B.6, B.7, B.8, B.9 détaillent ces évaluations.

TABLE B.1. Agent A_{bed1} : évaluation du jeu de données *bedroom1* D_{bed1}

D_{bed_1}		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	73.89%	80.59%	79.54%	84.5%	82.03%	86.46%
	F-meas.	74.58%	76.14%	77.63%	77.39%	78.12%	83.07%
2-fold	Acc.	61.63%	57.09%	60.3%	64.07%	65.01%	77.33%
	F-meas.	60.02%	47.71%	56.51%	52.61%	60.71%	74.76%
3-fold	Acc.	66.81%	67.52%	67.94%	66.86%	69.13%	81.42%
	F-meas.	66.20%	60.0%	64.94%	57.61%	66.19%	79.09%
4-fold	Acc.	68.63%	77.93%	76.61%	78.06%	78.18%	85.52%
	F-meas.	72.95%	75.31%	77.18%	70.22%	76.33%	84.73%
5-fold	Acc.	58.94%	54.46%	59.55%	57.59%	57.0%	64.84%
	F-meas.	57.73%	49.97%	58.02%	55.03%	53.66%	63.44%
6-fold	Acc.	72.01%	77.8%	76.19%	78.05%	78.85%	88.07%
	F-meas.	74.26%	73.30%	76.87%	69.40%	75.51%	87.43%
7-fold	Acc.	56.45%	51.99%	51.8%	68.33%	52.24%	57.7%
	F-meas.	55.11%	39.86%	44.63%	47.65%	43.13%	65.62%
8-fold	Acc.	63.76%	54.57%	62.1%	61.63%	57.75%	70.21%
	F-meas.	60.72%	44.63%	58.15%	55.62%	51.31%	67.21%
9-fold	Acc.	69.16%	67.75%	69.52%	68.57%	67.97%	83.08%
	F-meas.	70.84%	62.78%	69.88%	58.39%	63.15%	83.26%
10-fold	Acc.	66.87%	61.95%	66.57%	67.06%	61.48%	74.5%
	F-meas.	65.18%	54.02%	63.27%	60.95%	52.75%	71.87%
Global	Acc.	65.81%	65.16%	67.01%	68.40%	66.96%	77.97%
	F-meas.	65.75%	58.37%	64.7%	60.48%	61.98%	76.04%

TABLE B.2. Agent A_d : évaluation du jeu de données *dining* D_d

D_d		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	62.08%	61.72%	61.6%	62.46%	61.81%	64.21%
	F-meas.	59.66%	57.38%	57.45%	57.52%	57.74%	61.08%
2-fold	Acc.	68.68%	70.75%	71.11%	71.28%	71.23%	72.94%
	F-meas.	67.53%	67.05%	70.45%	66.74%	67.72%	70.45%
3-fold	Acc.	66.16%	71.08%	70.75%	72.37%	71.23%	73.98%
	F-meas.	65.59%	67.25%	67.29%	67.22%	67.85%	71.84%
4-fold	Acc.	69.03%	70.72%	70.9%	71.04%	72.41%	74.45%
	F-meas.	68.70%	67.42%	68.35%	66.75%	70.08%	73.30%
5-fold	Acc.	65.66%	68.77%	68.44%	74.43%	69.3%	71.94%
	F-meas.	65.11%	64.89%	64.99%	63.71%	66.08%	70.11%
6-fold	Acc.	67.58%	66.46%	66.64%	66.82%	67.53%	71.05%
	F-meas.	66.02%	60.75%	62.17%	61.05%	63.26%	69.08%
7-fold	Acc.	57.64%	56.93%	57.34%	55.97%	57.96%	61.07%
	F-meas.	54.37%	49.95%	51.73%	48.63%	51.98%	57.72%
8-fold	Acc.	69.54%	71.43%	71.34%	71.87%	72.17%	74.78%
	F-meas.	69.11%	67.95%	68.13%	67.90%	70.09%	73.51%
9-fold	Acc.	67.41%	69.18%	69.51%	70.33%	69.39%	72.44%
	F-meas.	65.67%	64.59%	65.18%	64.44%	64.91%	69.82%
10-fold	Acc.	64.36%	65.52%	66.81%	63.56%	66.96%	69.18%
	F-meas.	60.65%	58.98%	65.63%	55.13%	62.43%	65.63%
Global	Acc.	65.81%	67.25%	68.43%	68.01%	67.99%	70.60%
	F-meas.	64.24%	62.62%	64.13%	61.9%	64.21%	68.25%

TABLE B.3. Agent A_c : évaluation du jeu de données *corridor* D_c

D_c		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	77.11%	78.16%	78.31%	78.5%	78.08%	79.14%
	F-meas.	69.87%	69.32%	70.09%	69.14%	69.41%	71.34%
2-fold	Acc.	84.96%	86.67%	86.73%	87.09%	86.79%	87.78%
	F-meas.	81.66%	81.33%	82.02%	81.14%	81.52%	83.40%
3-fold	Acc.	88.48%	89.83%	90.17%	90.71%	89.81%	91.2%
	F-meas.	87.02%	86.33%	87.09%	86.59%	86.35%	88.69%
4-fold	Acc.	87.64%	88.69%	88.54%	89.37%	88.73%	90.87%
	F-meas.	86.15%	84.64%	85.40%	84.63%	84.91%	88.61%
5-fold	Acc.	81.77%	82.32%	82.45%	81.6%	83.08%	84.83%
	F-meas.	78.64%	76.39%	76.79%	75.31%	77.97%	80.24%
6-fold	Acc.	86.69%	88.27%	88.29%	88.86%	88.56%	89.85%
	F-meas.	84.72%	83.92%	84.55%	83.63%	84.41%	86.56%
7-fold	Acc.	79.64%	79.83%	79.96%	80.23%	80.86%	82.74%
	F-meas.	74.24%	72.17%	73.36%	71.85%	74.59%	77.17%
8-fold	Acc.	88.75%	89.33%	89.39%	89.34%	89.32%	90.36%
	F-meas.	86.11%	85.0%	85.67%	84.87%	84.97%	87.08%
9-fold	Acc.	86.12%	87.78%	87.74%	88.75%	87.76%	88.69%
	F-meas.	83.32%	83.36%	83.47%	83.47%	83.39%	84.89%
10-fold	Acc.	80.17%	81.31%	80.82%	82.07%	81.62%	83.48%
	F-meas.	75.94%	74.68%	74.64%	74.24%	75.45%	78.72%
Global	Acc.	84.13%	85.21%	85.24%	85.65%	85.46%	86.89%
	F-meas.	80.76%	79.71%	80.3%	79.48%	80.29%	82.67%

TABLE B.4. Agent A_e : évaluation du jeu de données *exit* D_e

D_e		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	73.71%	72.46%	74.01%	72.69%	72.46%	75.92%
	F-meas.	72.84%	68.0%	72.66%	65.91%	68.01%	74.54%
2-fold	Acc.	70.62%	68.92%	70.54%	70.01%	69.07%	72.39%
	F-meas.	67.95%	63.44%	67.77%	67.06%	63.53%	69.62%
3-fold	Acc.	69.96%	67.16%	69.88%	73.74%	67.38%	72.02%
	F-meas.	68.65%	62.10%	68.55%	65.93%	62.29%	70.67%
4-fold	Acc.	73.42%	72.02%	73.93%	73.64%	72.31%	75.92%
	F-meas.	71.85%	66.47%	71.67%	71.14%	66.93%	73.56%
5-fold	Acc.	77.84%	78.65%	78.5%	78.61%	78.94%	79.97%
	F-meas.	76.57%	75.84%	76.94%	75.88%	76.22%	78.25%
6-fold	Acc.	78.28%	78.06%	78.57%	77.18%	77.84%	82.84%
	F-meas.	78.10%	73.81%	77.60%	75.67%	73.60%	81.45%
7-fold	Acc.	74.01%	74.59%	74.23%	75.19%	74.96%	78.35%
	F-meas.	72.60%	70.67%	73.13%	71.05%	71.33%	77.13%
8-fold	Acc.	75.55%	75.48%	76.22%	75.23%	75.92%	80.19%
	F-meas.	74.49%	71.23%	74.89%	73.53%	71.74%	78.74%
9-fold	Acc.	77.84%	75.99%	78.06%	78.39%	75.92%	80.56%
	F-meas.	77.62%	72.19%	76.77%	76.70%	72.13%	79.43%
10-fold	Acc.	67.43%	68.46%	68.53%	68.0%	68.31%	71.62%
	F-meas.	62.75%	61.50%	63.54%	60.84%	61.34%	66.41%
Global	Acc.	73.86%	73.17%	74.24%	74.26%	73.31%	76.97%
	F-meas.	72.34%	68.52%	72.35%	70.37%	68.71%	74.98%

TABLE B.5. Agent A_{bath2} : évaluation du jeu de données *bathroom2* D_{bath2}

D_{bath_2}		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	78.95%	79.7%	79.7%	79.5%	79.7%	80.08%
	F-meas.	71.75%	72.80%	72.12%	71.94%	72.12%	73.01%
2-fold	Acc.	88.35%	89.47%	89.85%	90.21%	89.85%	89.85%
	F-meas.	86.20%	86.89%	87.08%	87.27%	87.08%	87.08%
3-fold	Acc.	92.86%	93.23%	93.61%	91.07%	93.61%	93.61%
	F-meas.	91.92%	92.11%	92.31%	89.20%	92.31%	92.31%
4-fold	Acc.	91.73%	92.86%	92.86%	91.75%	92.86%	92.86%
	F-meas.	90.93%	91.51%	91.51%	89.99%	91.51%	91.51%
5-fold	Acc.	93.98%	96.62%	97.74%	96.36%	97.94%	97.74%
	F-meas.	95.07%	96.43%	97.0%	96.24%	97.0%	97.0%
6-fold	Acc.	95.11%	95.86%	96.62%	97.85%	96.62%	96.62%
	F-meas.	95.41%	95.79%	96.18%	96.62%	96.18%	96.18%
7-fold	Acc.	98.12%	98.5%	99.25%	99.25%	99.25%	99.25%
	F-meas.	99.30%	99.11%	99.49%	99.49%	99.49%	99.49%
8-fold	Acc.	91.35%	93.98%	96.24%	98.93%	96.24%	96.62%
	F-meas.	94.40%	96.11%	97.29%	98.31%	97.29%	97.48%
9-fold	Acc.	96.99%	96.99%	97.37%	97.17%	97.37%	97.37%
	F-meas.	96.07%	95.88%	96.07%	95.51%	96.07%	96.07%
10-fold	Acc.	95.99%	96.72%	96.72%	94.9%	96.72%	96.72%
	F-meas.	96.09%	96.53%	96.53%	93.82%	96.53%	96.53%
Global	Acc.	92.34%	93.39%	93.99%	93.69%	93.99%	94.07%
	F-meas.	91.71%	92.31%	92.55%	91.38%	92.55%	92.66%

TABLE B.6. Agent A_{bath1} : évaluation du jeu de données *bathroom1* D_{bath1}

D_{bath_1}		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	59.6%	61.62%	59.6%	62.49%	61.62%	63.64%
	F-meas.	45.53%	51.14%	45.53%	58.43%	49.77%	53.71%
2-fold	Acc.	64.65%	65.66%	79.8%	66.12%	73.74%	79.8%
	F-meas.	62.58%	60.05%	75.02%	61.44%	64.30%	75.02%
3-fold	Acc.	81.82%	77.78%	85.86%	80.41%	87.88%	91.92%
	F-meas.	78.55%	76.89%	81.19%	78.00%	82.20%	90.07%
4-fold	Acc.	71.72%	64.65%	71.72%	70.23%	70.71%	76.77%
	F-meas.	63.65%	57.51%	60.60%	64.41%	63.65%	70.91%
5-fold	Acc.	78.79%	77.78%	79.8%	78.67%	80.81%	80.81%
	F-meas.	70.33%	73.18%	70.83%	73.34%	73.56%	74.80%
6-fold	Acc.	86.87%	89.9%	88.89%	88.36%	90.91%	91.92%
	F-meas.	84.75%	86.49%	84.61%	84.09%	86.91%	90.11%
7-fold	Acc.	95.96%	94.95%	96.97%	97.27%	96.97%	96.97%
	F-meas.	94.97%	94.45%	95.47%	95.47%	95.47%	95.47%
8-fold	Acc.	92.93%	79.8%	94.95%	92.7%	94.95%	94.95%
	F-meas.	93.41%	87.05%	94.45%	92.88%	94.45%	95.44%
9-fold	Acc.	82.83%	75.76%	86.87%	84.44%	90.91%	91.92%
	F-meas.	80.54%	77.44%	82.64%	81.07%	87.87%	89.34%
10-fold	Acc.	85.86%	89.9%	90.91%	97.18%	94.95%	94.95%
	F-meas.	90.52%	92.76%	93.31%	96.47%	95.44%	95.44%
Global	Acc.	80.10%	77.78%	83.53%	81.78%	84.34%	86.36%
	F-meas.	76.48%	75.69%	78.36%	78.56%	79.36%	83.03%

TABLE B.7. Agent A_{bed2} : évaluation du jeu de données *bedroom2* D_{bed2}

D_{bed_2}		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	86.32%	86.62%	86.65%	86.53%	86.73%	86.92%
	F-meas.	82.55%	82.6%	82.84%	82.27%	82.64%	83.09%
2-fold	Acc.	88.63%	88.95%	89.01%	88.99%	88.97%	89.24%
	F-meas.	87.60%	87.34%	87.63%	87.42%	87.39%	87.90%
3-fold	Acc.	91.07%	91.81%	91.82%	91.71%	91.86%	92.06%
	F-meas.	89.95%	90.0%	90.19%	89.79%	90.11%	90.49%
4-fold	Acc.	93.79%	94.42%	94.38%	94.3%	94.43%	94.53%
	F-meas.	93.44%	93.73%	93.81%	93.43%	93.74%	93.93%
5-fold	Acc.	92.62%	93.21%	93.18%	93.19%	93.27%	93.42%
	F-meas.	92.21%	92.24%	92.33%	92.11%	92.31%	92.53%
6-fold	Acc.	94.08%	94.84%	94.74%	94.73%	94.87%	95.02%
	F-meas.	94.32%	94.35%	94.42%	94.12%	94.45%	94.74%
7-fold	Acc.	91.9%	92.45%	92.34%	92.19%	92.49%	92.68%
	F-meas.	91.01%	91.0%	91.07%	90.53%	91.12%	91.45%
8-fold	Acc.	95.06%	95.73%	95.55%	95.72%	95.73%	95.82%
	F-meas.	95.40%	95.67%	95.64%	95.47%	95.67%	95.84%
9-fold	Acc.	92.91%	93.7%	93.67%	93.58%	93.73%	93.88%
	F-meas.	92.23%	92.40%	92.44%	92.06%	92.46%	92.69%
10-fold	Acc.	93.53%	94.01%	93.9%	93.94%	93.99%	94.13%
	F-meas.	93.14%	93.17%	93.14%	92.85%	93.16%	93.40%
Global	Acc.	91.99%	92.57%	92.52%	92.48%	92.60%	92.77%
	F-meas.	91.18%	91.25%	91.35%	91.0%	91.30%	91.60%

TABLE B.8. Agent A_l : évaluation du jeu de données *livingroom* D_l

D_l		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	74.04%	73.91%	74.01%	73.93%	74.03%	74.51%
	F-meas.	71.09%	70.61%	70.70%	70.48%	70.71%	71.40%
2-fold	Acc.	77.43%	78.94%	78.9%	79.08%	79.02%	79.27%
	F-meas.	76.31%	77.29%	77.33%	77.36%	77.37%	77.80%
3-fold	Acc.	81.12%	81.44%	81.38%	81.47%	81.42%	81.84%
	F-meas.	80.87%	80.41%	80.55%	80.44%	80.47%	81.18%
4-fold	Acc.	77.46%	78.24%	78.21%	77.96%	78.18%	78.72%
	F-meas.	76.85%	77.11%	77.13%	76.72%	77.14%	77.91%
5-fold	Acc.	74.32%	75.61%	75.08%	75.43%	75.48%	76.3%
	F-meas.	74.24%	74.82%	74.79%	74.33%	74.94%	75.97%
6-fold	Acc.	74.77%	74.46%	74.8%	74.53%	75.0%	75.43%
	F-meas.	74.04%	74.67%	75.01%	73.09%	73.73%	76.1%
7-fold	Acc.	74.25%	72.55%	72.91%	72.7%	73.01%	73.44%
	F-meas.	73.41%	70.81%	71.66%	71.55%	71.98%	72.57%
8-fold	Acc.	76.66%	76.7%	76.99%	76.76%	76.98%	77.53%
	F-meas.	76.54%	75.67%	76.44%	76.10%	76.38%	77.23%
9-fold	Acc.	72.4%	73.26%	73.08%	73.14%	73.28%	73.68%
	F-meas.	72.62%	72.89%	72.79%	72.68%	72.98%	73.59%
10-fold	Acc.	77.98%	78.74%	78.63%	78.65%	78.75%	79.06%
	F-meas.	78.35%	79.6%	79.77%	78.33%	78.51%	80.04%
Global	Acc.	76.04%	76.38%	76.4%	76.36%	76.51%	76.97%
	F-meas.	75.43%	75.38%	75.61%	75.1%	75.42%	76.37%

TABLE B.9. Agent A_k : évaluation du jeu de données *kitchen* D_k

D_k		W-DCR	DCR			DCR-OL	upper bound
			max-trust	max-freq.	stacking	max-wPerf	
1-fold	Acc.	58.27%	41.28%	46.93%	48.65%	51.63%	81.19%
	F-meas.	56.59%	39.8%	40.62%	43.26%	46.06%	79.54%
2-fold	Acc.	55.16%	46.38%	48.36%	66.56%	64.18%	87.95%
	F-meas.	56.38%	47.25%	49.42%	53.19%	60.45%	88.03%
3-fold	Acc.	54.62%	51.83%	55.95%	56.16%	65.71%	87.28%
	F-meas.	53.98%	50.80%	55.01%	45.29%	62.78%	86.91%
4-fold	Acc.	62.45%	50.02%	44.6%	55.81%	54.29%	85.29%
	F-meas.	62.68%	48.04%	44.52%	41.0%	53.87%	85.94%
5-fold	Acc.	60.14%	59.6%	56.6%	67.34%	64.13%	85.37%
	F-meas.	60.58%	59.23%	56.99%	65.29%	63.29%	85.64%
6-fold	Acc.	59.82%	54.68%	59.43%	62.8%	61.56%	78.21%
	F-meas.	59.43%	52.54%	58.60%	60.17%	60.50%	77.79%
7-fold	Acc.	55.22%	51.67%	48.67%	68.91%	58.53%	82.28%
	F-meas.	57.42%	52.57%	50.51%	56.27%	57.11%	83.40%
8-fold	Acc.	61.28%	47.85%	51.1%	64.2%	57.69%	77.82%
	F-meas.	61.29%	47.97%	51.42%	53.12%	57.54%	78.09%
9-fold	Acc.	59.23%	45.86%	45.38%	59.46%	46.59%	73.28%
	F-meas.	58.71%	45.34%	45.11%	44.34%	46.54%	74.24%
10-fold	Acc.	61.95%	61.98%	62.57%	63.27%	63.58%	83.78%
	F-meas.	62.68%	61.74%	62.56%	60.52%	64.11%	84.84%
Global	Acc.	58.81%	50.41%	51.45%	61.31%	58.82%	83.17%
	F-meas.	58.97%	49.77%	50.88%	52.24%	57.22%	83.16%

Annexe C

Annexe : Structure de l'ontologie

Nous présentons une partie de l'ontologie de contexte que nous avons utilisée pour la représentation de l'activité humaine [Rodriguez et al., 2014]. Nous nous sommes concentrés sur trois concepts qui sont la localisation, les capteurs et les activités. Les figures suivantes sont obtenues auprès de l'outil Protégé.

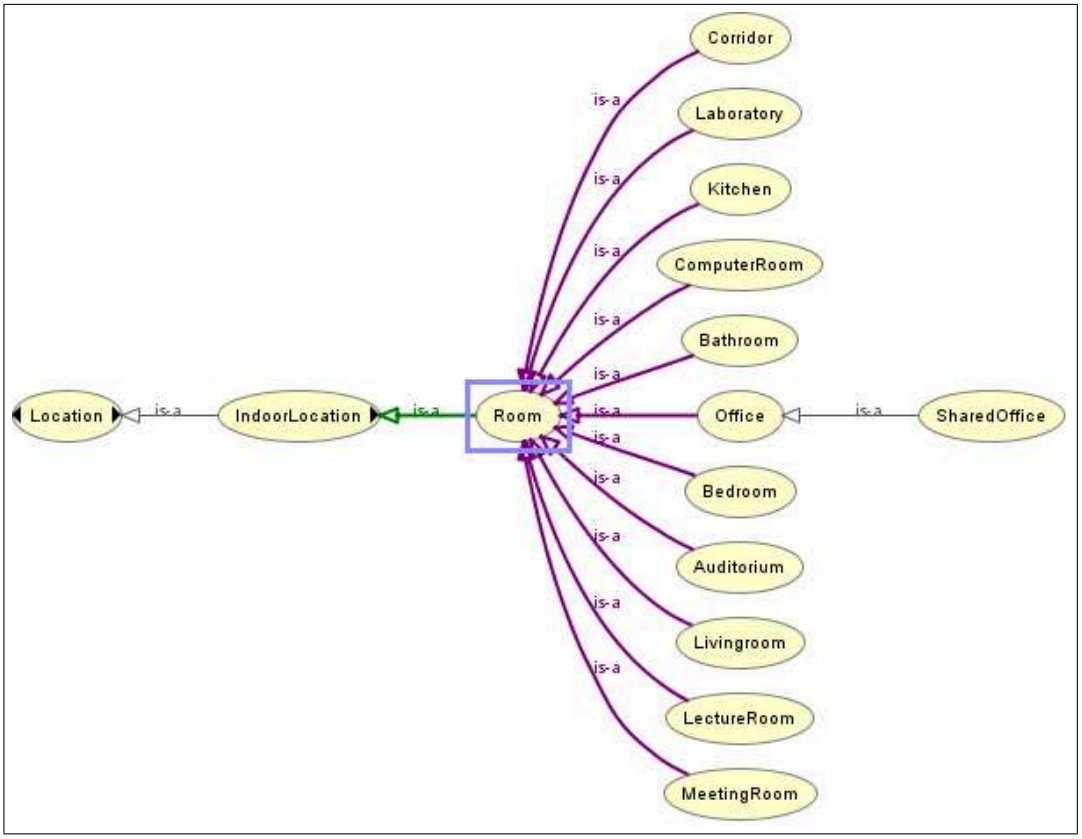


FIGURE C.1. Les concepts associés à la localisation

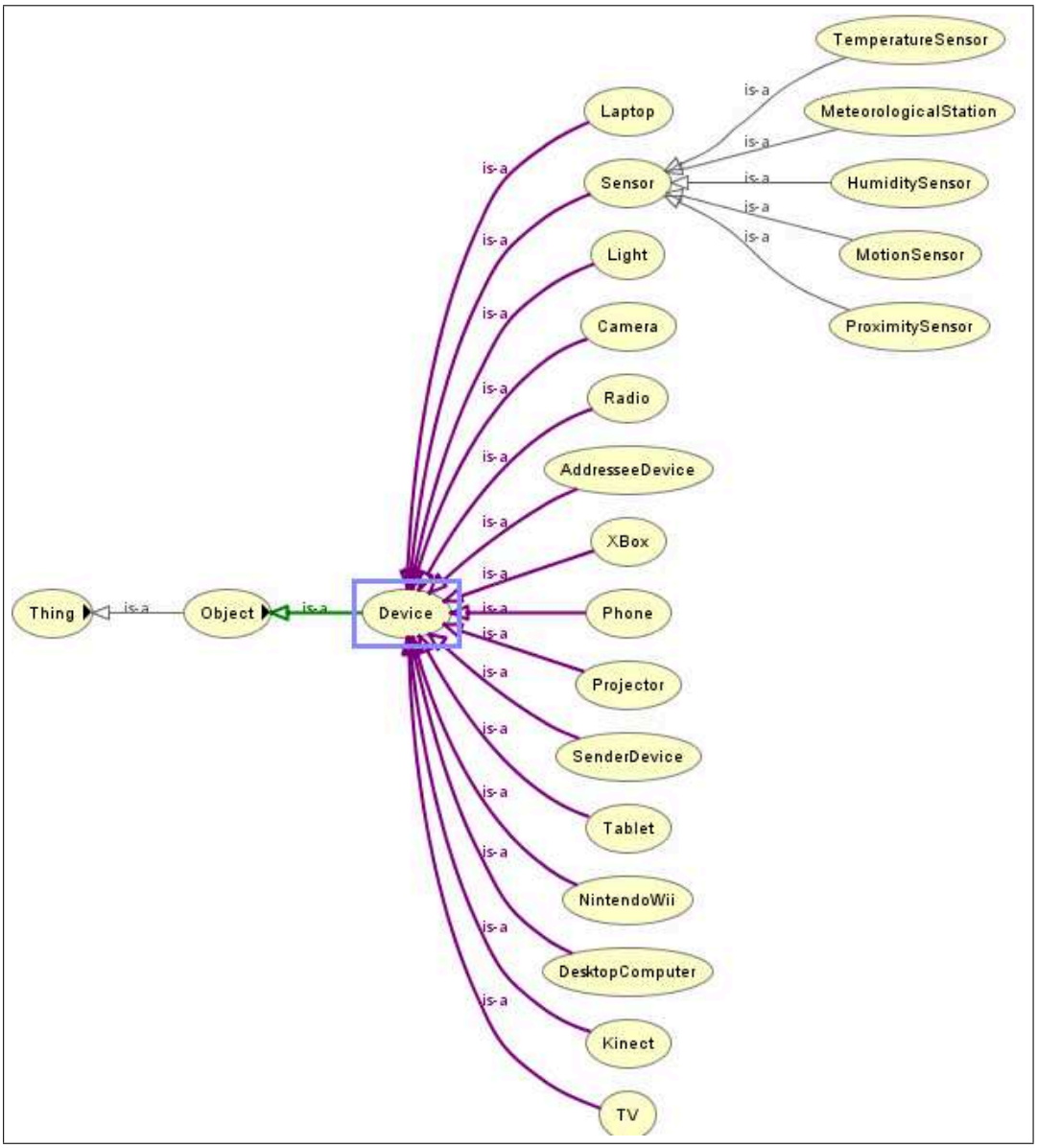


FIGURE C.2. Les concepts associés aux capteurs

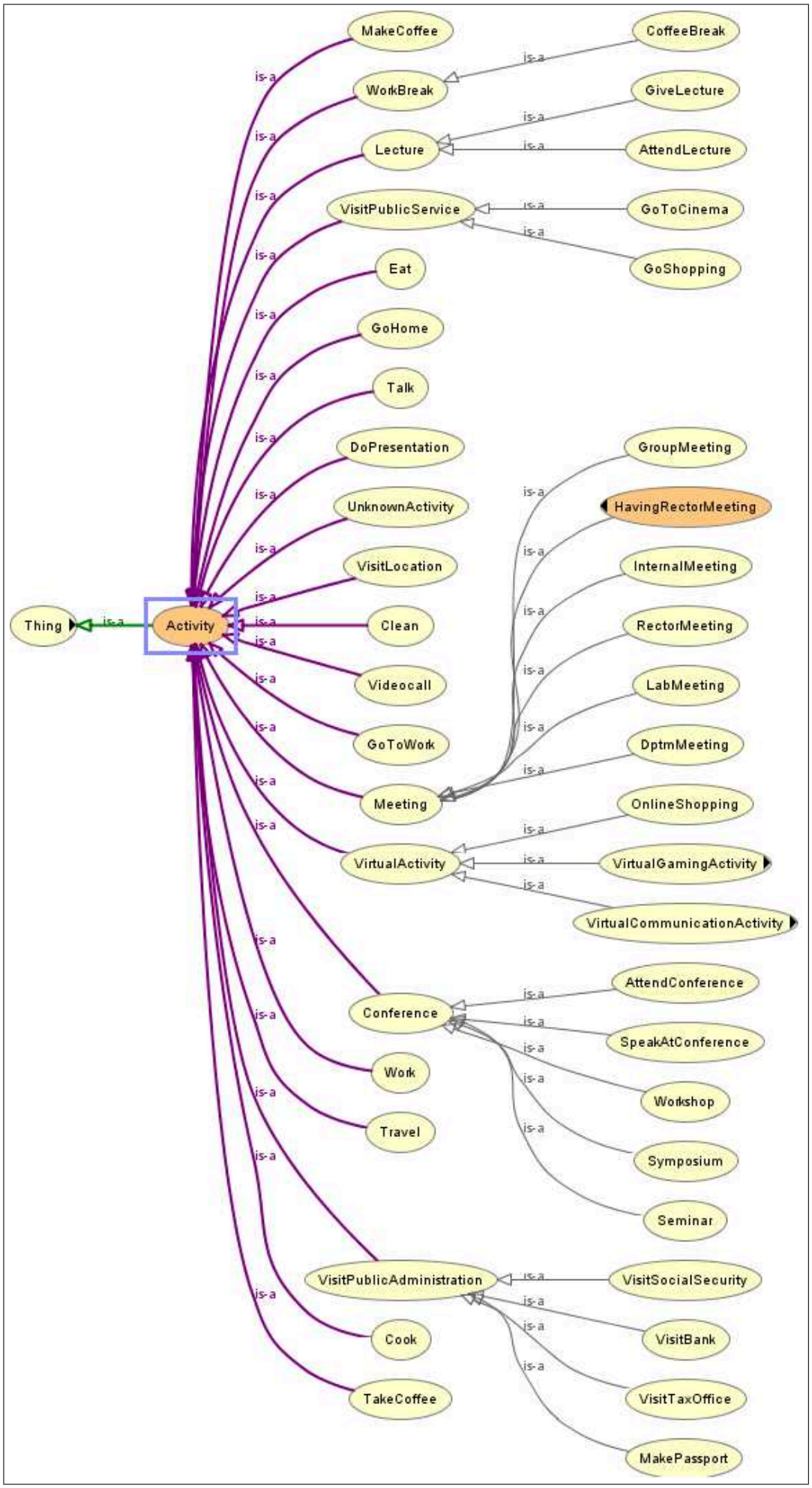


FIGURE C.3. Les concepts associés aux activités